Repulsion Region in a Simple Polygon*

Arthur van Goethem[1], Irina Kostitsyna[1], Kevin Verbeek[1], and Jules Wulms[1]

1 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**
{a.i.v.goethem|i.kostitsyna|k.a.b.verbeek|j.j.h.m.wulms}@tue.nl

Abstract

We study a motion planning problem for point objects controlled by an external repulsive force. Given a point object ζ inside a simple polygon P, a *repulsor* r (also represented by a point in P) moves ζ by always pushing it away from r. When ζ hits the boundary of P, then ζ slides along the boundary continuously increasing the distance to r until this is no longer possible. For a fixed polygon P and starting position s of ζ inside P, we define the *repulsion region* as the set of points in P that can be reached by ζ by placing a repulsor r anywhere inside P (and keeping the position of r fixed throughout the motion). In this paper we show that, for a specific class of polygons, the worst case complexity of the repulsion region is $\Theta(n^2)$ and the repulsion region can be computed in $O(n^2 \log n)$ time, where n is the complexity of the polygon.

1 Introduction

Algorithmic path planning and motion control questions are usually studied in the context of the actuating abilities of (autonomous or externally controlled) mobile objects. The object is put in motion by the forces produced from within, by its actuators. A very different approach is to consider a static object (or a set of objects), with no means of producing any kind of motion itself, controlled by external global forces. There has been a number of papers exploring motion planning of point objects by external control. Becker et al. [3] study the question of controlling a swarm of particles in a geometric environment by applying a sequence of global forces, such as gravity or a homogeneous magnetic field. Aloupis et al. [2] consider the problem of rolling a ball out of a polygon (or equivalently, draining a polygon filled with water) by tilting it. Akitaya et al. [1] consider the trash compaction problem, where a set of trash particles are pushed together with a sweep line to form a compact shape.

One commonality among these papers is that, at any given moment, the force applied to an object at any possible location has the same direction. In contrast, in the *beacon attraction* model [4, 5], a point magnet, called beacon, is placed inside the polygon which exerts a magnetic pull towards itself on all the points of the polygon. The forces applied to the objects at different locations are no longer parallel, which leads to rather particular geometric properties. Specifically, a *beacon attraction region*, i.e., all the points of the polygon that eventually reach a given beacon and not get stuck along the way, has very different properties than an *inverse beacon attraction region*, i.e., all the beacon locations that a given point can eventually reach and not get stuck along the way [4, 5, 8, 9].

Inspired by the beacon attraction model, Bose and Shermer [6] introduce the model of repulsion, where a repulsion actuator exerts a magnetic repulsive force on the points of a

* Kevin Verbeek is supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.021.541. Jules Wulms is supported by the Netherlands eScience Center (NLeSC) under project no. 027.015.G02. Research on the topic of this abstract was initiated at the 4th Workshop on Applied Geometric Algorithms (AGA 2018) in Langbroek, The Netherlands, supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.023.208.

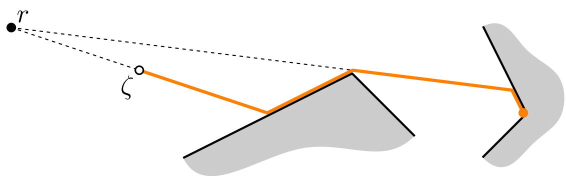

Figure 1 Repulsor r acts on the particle ζ. If the direction away from the repulsor is interior to the polygon, ζ moves away from the repulsor along a straight line. Otherwise, if possible, ζ slides along a boundary segment in the direction of increasing Euclidean distance.

polygon. They consider the question whether it is possible to gather all the points of a convex polygon in one point with one repulsion actuator. Mozafari and Shermer [10] study the problem of finding a set of *acceptable* points for a given target location t, i.e., the set of points that can be pushed to t with one repulsion actuator.

We continue to explore the repulsion model and study the complementary problem: given a starting position s, find all the points *reachable* from s with one repulsion actuator.

Problem statement and contributions. Consider a repulsion activator (or *repulsor*) r and a particle ζ inside a polygon P. Under the repulsion of r, the particle ζ moves away from r along a straight line as long as ζ is interior to P, even if r is not visible to ζ (refer to Figure 1). When ζ reaches the boundary of P, it slides along it in the direction of increasing Euclidean distance from r, if such a direction exists. Otherwise, the motion of the particle terminates.

Given a polygon P with n vertices and a point s in it, we define the *repulsion region* $R(s)$ as the set of points $p \in P$ for which there exists a repulsor $r \in P$ such that a particle placed at s will eventually reach the point p under the influence of the repulsor r. In this abstract we restrict the attention to a specific class of polygons, which we define in Section 2, along with some basic notation and properties. In Section 3 we show that the repulsion region has worst-case complexity $\Theta(n^2)$. Finally, we show in Section 4 that the repulsion region can be computed in $O(n^2 \log n)$ time. Although we believe these results hold for all simple polygons, proving so is significantly harder, and therefore left for the full version of this paper.

2 Preliminaries

Consider a particle ζ, currently at position p_ζ, that is moved by a repulsor r inside a simple polygon P. By definition, the Euclidean distance between r and ζ is monotonically increasing throughout the motion. For an edge e of P not containing p_ζ, consider the line ℓ perpendicular to e passing through p_ζ. If this line intersects e in a point h, then we call h the *split point* of e with respect to the current position of ζ (see Fig. 2 (left)). Note that, if r is strictly on one side of ℓ and pushes ζ onto e, then ζ will hit e on the opposite side of ℓ and continue moving away from h.

Furthermore, consider a reflex vertex v of P with two incident edges e_1 and e_2. Extend the edges e_1 and e_2 beyond v. When ζ lies inside the resulting cone, then a repulsor may push it towards either of the edges. If the segment $\overline{p_\zeta v}$ splits the interior angle at v into two obtuse angles, then we call v a *split vertex* with respect to the current position of ζ.

Now consider a particle ζ sliding along an edge e_1 towards a convex vertex v (see Fig. 2 (right)). Let e_2 be the other edge incident to v, and let ℓ be the line perpendicular to e_2 going through v. Let H_1 and H_2 be the half planes defined by ℓ containing e_1 and e_2, respectively.

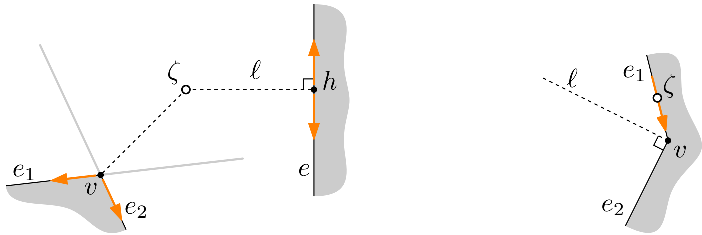

Figure 2 Left: point h is the split point of e, and reflex vertex v is a split vertex, with respect to the current location of particle ζ. Right: ζ is sliding along e_1 towards v, ℓ is perpendicular to e_2. If the repulsor is above ℓ, then ζ will continue sliding along e_2, otherwise it will remain in v.

If the repulsor pushing ζ towards v lies in H_1 then ζ will continue sliding along e_2 after it passes v. However, if the repulsor is in H_2 then the motion of ζ will terminate at v.

In the (degenerate) case that a repulsor is located exactly on the line through ζ and a split vertex/point with respect to the current position of ζ, the particle moves onto either of the two incident edges. This choice does not significantly impact the repulsion region.

Finally, we denote a path followed by the particle ζ (starting at s) repulsed by a repulsor r by π_r. Observe that π_r must be simple. Let π_1 and π_2 be two paths generated by repulsors r_1 and r_2, respectively. We say that π_1 and π_2 have a *convergence point* c, if $c \in \pi_1$ and $c \in \pi_2$. Note that the starting position s is a convergence point of all paths. If two paths have a convergence point other than s, then we say that they are *converging*. Furthermore, we say that π_1 and π_2 *properly* intersect in convergence point c if c is interior to P.

Regular polygons. We call a polygon *regular* if (1) looking at the split points/vertices with respect to s, the angle between consecutive split points/vertices around s is at most π, and (2) there does not exist a repulsor r such that we can identify two points p_1, p_2 on π_r for which $r \in \overline{p_1p_2}$ and $\overline{p_1p_2}$ is completely contained in P; otherwise the polygon is *irregular* (see Fig. 3(a) and (b)). Intuitively, the path π_r of a repulsor r cannot spiral around r in regular polygons. Contrary to regular polygons (see Section 3), paths of repulsors may properly intersect in irregular polygons, making the analysis significantly harder. Furthermore, we obtain the following useful property in regular polygons, which again, does not hold generally for irregular polygons.

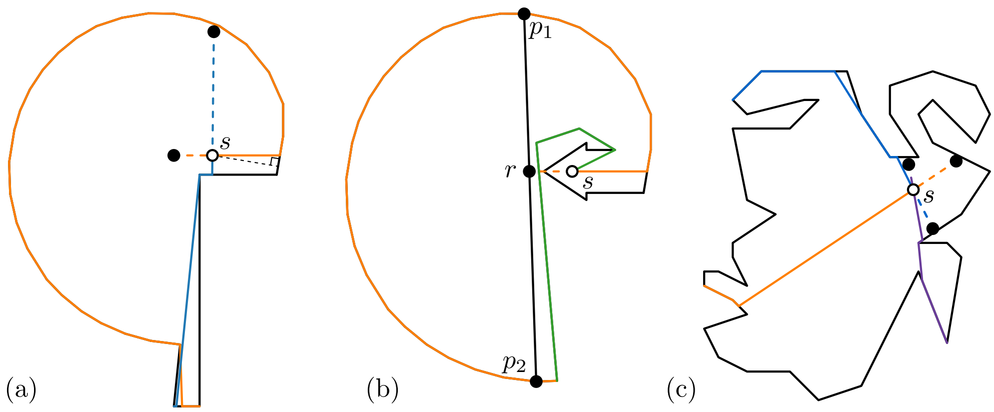

Figure 3 Regular vs irregular polygons: (a) condition (1) is violated and paths cross; (b) condition (2) is violated and geodesic distance to s decreases; (c) a regular polygon.

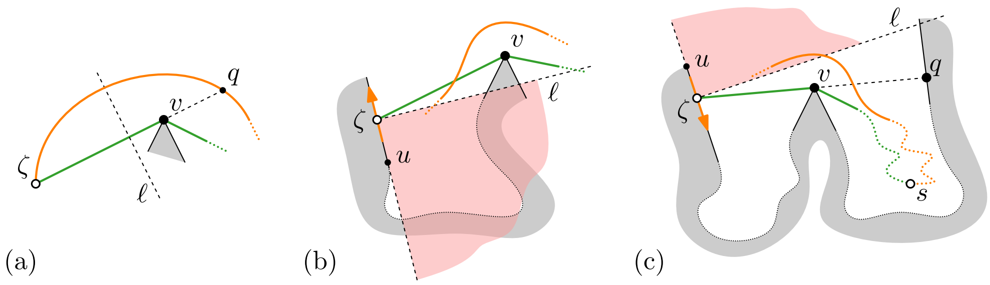

Figure 4 Illustration of Lemma 1. In a regular polygon the geodesic distance from ζ to s is monotonically increasing. Geodesic path is shown in green, path π_r in orange.

▶ **Lemma 1.** *In a regular polygon P, the geodesic distance inside P from a particle ζ to its starting position s is monotonically increasing.*

Proof sketch. Consider the geodesic path γ from s to a point p_ζ, the current position of ζ. Let v be the last vertex on γ before p_ζ. We consider several cases (see Fig. 4). If p_ζ is interior to P, then the result follows from the fact that the distance from ζ to r is monotonically increasing. If p_ζ is on the boundary of P, then let $\overline{vp_\zeta}$ split the polygon into two subpolygons P_1 and P_2, where P_1 contains s. If ζ is pushed into P_1, then the geodesic distance between ζ and s cannot decrease. Otherwise, the geodesic distance between ζ and s may only decrease if π_r spirals around r, but then P is irregular. ◀

3 Complexity of the repulsion region

To prove an upper bound on the complexity of the repulsion region, we first show that the paths of two repulsors cannot properly intersect in a regular polygon P.

▶ **Lemma 2.** *In a regular polygon P no two repulsion paths have a proper intersection.*

Proof sketch. Let π_1 and π_2 be the paths generated by two repulsors r_1 and r_2, and let ℓ be the line through r_1 and r_2. Assume that c and d are two consecutive convergence points of π_1 and π_2. We can show that c and d must be on the same side of ℓ, for otherwise there must exist another convergence point between c and d at ℓ. Additionally, π_1 and π_2 cannot cross ℓ between c and d. Then, due to the relative angles from d to r_1 and r_2, there cannot be a proper intersection at d. By repeating this argument, we conclude that π_1 and π_2 do not have a proper intersection. ◀

Now let v be a (reflex) vertex of P and consider two repulsors r_1 and r_2 for which the generated paths π_1 and π_2 contain v. Consider a repulsor r_3 that lies on the segment $\overline{r_1 r_2}$ (potentially outside P) and its repulsion path π_3. Since two repulsion paths cannot properly intersect, π_3 is essentially contained between π_1 and π_2, and hence must also contain v. We obtain the following result.

▶ **Corollary 3.** *The set of repulsors $S(v)$ (possibly outside of the polygon) that push the particle ζ to a reflex vertex v is convex.*

Consequently, the set of repulsors $S'(v)$ inside P that push the particle to a (reflex) vertex v consists of at most $O(n)$ connected components. To construct part of the repulsion region starting from v, we can simply combine the at most $O(n)$ cones emanating from v coming

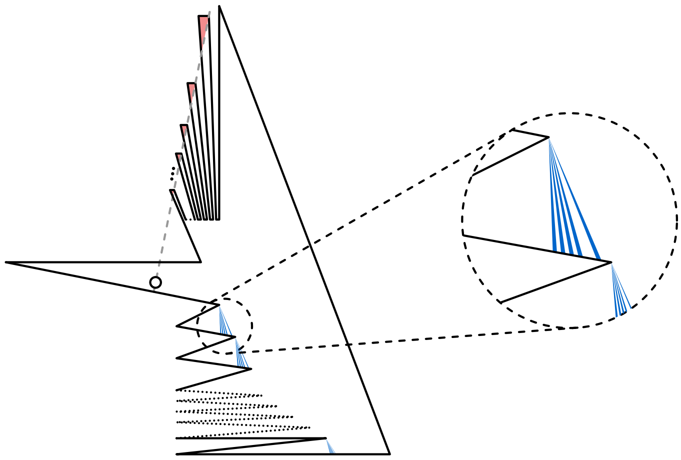

Figure 5 Polygon with $O(n)$ towers and $O(n)$ cliffs. The repulsion region we get by placing repulsors in the pink areas at the top, results in blue $O(n)$ cones per reflex vertex at the bottom.

from the $O(n)$ connected components of $S'(v)$. The complete repulsion region is then the union of all cones over all reflex vertices of P, as the path of the particle either follows the boundary, or moves internally through P starting from a reflex vertex. Since the cones cannot intersect, we get the following upper bound. This bound is tight, as shown by the example in Fig. 5.

▶ **Theorem 4.** *The repulsion region of a regular polygon P has worst-case complexity $\Theta(n^2)$, where n is the number of vertices of P.*

4 Computing the repulsion region

To compute the repulsion region, we must explicitly compute the corresponding convex sets for each (reflex) vertex of the regular polygon P. To compute the convex sets in the right order, we heavily rely on Lemma 1. Specifically, we construct the *shortest path map* (SPM) in P with source s (following the approach of [8]). This SPM is a subdivision of P which captures all possible shortest paths starting from s. For every region R_i of the SPM, there is a reflex vertex v_i (or s) that is the last vertex on the shortest path from s to a point $p \in R_i$. We call v_i the base of R_i. We now add the following vertices to P. For every reflex vertex v_i, we extend the last segment of the shortest path from s to v_i until it hits the boundary of P at w_i. The segment $\overline{v_i w_i}$ is called the *window* of R_i and lies on the boundary of R_i. We add the window vertex w_i to P. Furthermore, for every edge e in R_i, we add the split point of e with respect to v_i to P, if it exists. Finally, we construct the directed *repulsion graph* $G = (V, E)$, where V consists of all vertices, window vertices, and split points of P. The edges of G include all edges of the shortest path tree in P with source s, and the edges on the boundary of P directed towards larger geodesic distance from s (which is well defined, since we split P at split points). See Figure 6 for an example.

We can argue the following properties for any path π_r of a repulsor r: (1) π_r can enter a region R_i only via its base v_i or its window vertex w_i , and (2) if π_r reaches the base v_i of a region R_i, then it must proceed in R_i.As a result, the repulsion graph G represents the combinatorial structures of all possible paths π_r from s through P. Now, to construct the convex set for a particular vertex v of P, we can simply consider the incoming edges in G.

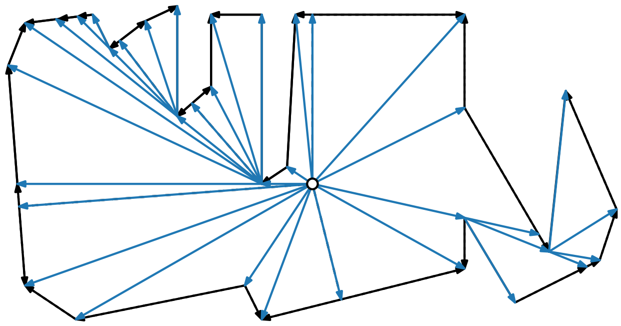

Figure 6 Repulsion graph G consisting of black edges along the boundary of P, and blue shortest path tree edges.

Typically, sliding along some incident edge of P towards v induces an additional half-plane constraint, as described in Section 2. Finally, we may need to combine the resulting convex sets from various incoming edges. However, since the union of these convex sets must again be convex, this simply means that some half-plane constraints can be eliminated, which is easy to compute.We can then obtain the following result.

▶ **Theorem 5.** *The repulsion region $R(s)$ of a regular polygon P can be computed in $O(n^2 \log n)$ time, where n is the number of vertices of P.*

Proof sketch. We first compute the SPM with source s in $O(n)$ time [7]. From the SPM we can easily compute the repulsion graph G in $O(n)$ time. Note that G has $O(n)$ complexity.

We then compute the convex set of repulsors for each vertex v of G, in order of increasing geodesic distance from s. For each incoming edge of v in G, we add the corresponding half-plane constraint to the convex set of repulsors reaching that edge and ending in v. Next, we combine the resulting convex sets for all incoming edges, thus constructing the convex set for v. Since each convex set may have $O(n)$ complexity, computing the convex set for v takes $O(nd_v)$ time, where d_v is the in-degree of v in G. Thus, we can compute the convex sets for all vertices in $\sum_v O(nd_v) = O(n^2)$ time.

Finally, we intersect the convex set of each vertex v with P in $O(n \log n)$ time using the algorithm in [11]. We can then easily compute the repulsion region $R(s)$ by extending the corresponding cones from each reflex vertex v. Thus, the repulsion region $R(s)$ can be computed in $O(n^2 \log n)$ time. ◀

References

1 Hugo Akitaya, Greg Aloupis, Maarten Löffler, and Anika Rounds. Trash compaction. In *Proc. 32nd European Workshop on Computational Geometry*, 2016.

2 Greg Aloupis, Jean Cardinal, Sébastien Collette, Ferran Hurtado, Stefan Langerman, and Joseph O'Rourke. Draining a polygon—or—rolling a ball out of a polygon. *Computational Geometry*, 47(2):316–328, 2014.

3 Aaron Becker, Erik Demaine, Sándor Fekete, Jarrett Lonsford, and Rose Morris-Wright. Particle computation: complexity, algorithms, and logic. *Natural Computing*, 18(1):181–201, 2019.

4 Michael Biro. *Beacon-based routing and guarding.* PhD thesis, Stony Brook University, 2013.

5 Michael Biro, Justin Iwerks, Irina Kostitsyna, and Joseph Mitchell. Beacon-based algorithms for geometric routing. In *Proc. 13th Algorithms and Data Structures Symposium (WADS)*, 2013.

6 Prosenjit Bose and Thomas Shermer. Gathering by repulsion. In *Proc. 16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 13:1–13:12, 2018.

7 Leonidas Guibas, John Hershberger, Daniel Leven, Micha Sharir, and Robert Endre Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987.

8 Irina Kostitsyna, Bahram Kouhestani, Stefan Langerman, and David Rappaport. An optimal algorithm to compute the inverse beacon attraction region. In *Proc. 34th International Symposium on Computational Geometry (SoCG)*, pages 55:1–55:14, 2018.

9 Bahram Kouhestani, David Rappaport, and Kai Salomaa. On the inverse beacon attraction region of a point. In *Proc. 27th Canadian Conference on Computational Geometry (CCCG)*, 2015.

10 Amirhossein Mozafari and Thomas Shermer. Transmitting particles in a polygonal domain by repulsion. In *Proc. 12th International Conference Combinatorial Optimization and Applications (COCOA)*, pages 495–508, 2018.

11 Ari Rappoport. An efficient algorithm for line and polygon clipping. *The Visual Computer*, 7(1):19–28, 1991.