Caging Polygons by a Finger and a Wall

Bahareh Banyassady* Mansoor Davoodi† Ali Mohades*

Abstract

This paper addresses the problem of caging an arbitrary polygon in the plane by a finger-wall caging grasp, which consists of a finger of a robot arm and a wall. An object is caged by a finger-wall grasp, when it is impossible for the object to move to an arbitrary placement far from its initial placement without penetrating the finger or the wall. We present an algorithm in $O(n^2 \log n)$ time for computing all configurations of the finger-wall grasp which cage a given polygon with n edges. In addition, the output set of all caging grasps can be queried in $O(\log n)$ time to check whether a given arbitrary finger-wall grasp cages the polygon.

1 Introduction

Robotic manipulators are designed to perform a wide variety of tasks in production lines of diverse industrial sectors, such as assembly or part orienting. To perform these tasks the robot arm has to first grasp the object in a proper way. Human being, whom robotic researchers have tried to mimic in many areas, to perform pick and place and also transportation, often grasp the object in a way that it can move among the grasping fingers, but cannot escape through them [1]. The caging problem (or capturing problem) was posed by Kuperberg [2] as a problem of finding the set of all placements of fingers which prevent an object from moving arbitrarily far from its given position.

It is generally assumed in the literature that objects are grasped at some point contacts and idealizations such as a line or surface contact can be approximated by two or more point contacts [3]. Fingertip grasps (point contacts) enable precise control of the object and adroit object manipulation, but limit the amount of force which can be exerted on the work piece. Inner-Link grasps where contacts occur at more than one point along a link are more stable in the face of environmental disturbances and can exert higher forces on a grasped object. However, they are geometrically more complicated and consequently more difficult to analyze [4]. Hence, it is important to look for the other possible practical grasps in order to reduce the number of point contacts. In everyday life, we frequently lean an object against a flat surface, such as a table or a wall, to constrain its motions. In the planar world, the analog of a wall is a supporting line [5]. Thus, we consider finger-wall grasps in which the finger is represented by a point, and the wall is represented by a line.

There are also some researches on caging polygons by three-finger and two-finger grasps. Pipattanasomporn and Sudsang [6], and Vahedi and van der Stappen [7] independently have solved the problem for two-finger grasps in $O(n^2 \log n)$ time, and also constructed a data structure capable of answering queries in $O(\log n)$ time. The running time of both solutions is independent of the complexity of the reported caging grasps. Whereas the number of elements in finger-wall caging grasps are equal to the number of elements in two-finger caging grasps, there are some similarities between these problems solving methods.

In this paper we present an algorithm to find the set of all finger-wall caging grasps of a given polygon in $O(nm \log n)$ where m is the size of the polygon's convex hull. In section 2 some basic definitions and notations are introduced. In section 3 the space configuration is partitioned in order to reduce the search space and in section 4 a graph which represents the search space is constructed. Finally, in section 5 we conclude the paper.

2 Preliminaries and Notations

This section addresses the problem of caging a polygon P by a point and a line. Formally, P is caged by a point and a line when its placement lies in a compact valid region of its free configuration space regarding the point and the line as obstacles. Informally, polygon P is caged by a point and a line when it is impossible to take the object to arbitrary placement far from its initial placement without penetrating the point and/or the line. A placement of the point and the line is a caging configuration if the object is caged by that placement of the point and the line. In general it is easier for the explanation to consider the polygon fixed and to move the point and the line instead while keeping their mutual distances fixed. Therefore, P is caged when it is impossible to rigidly move (translate and/or rotate) the point and the line to infinity without penetrating P.

*Department of Mathematics and Computer Science, Amirkabir University of Technology, Iran. {bahareh.banyasady, mohades}@aut.ac.ir

†Department of Computer Science and Information Technology, Institute for Advanced Studies in Basic Sciences, Iran. mdmonfared@iasbs.ac.ir

The given simple polygon P in the plane is bounded by n edges and is assumed to lie inside ω_1 (which ω_i is a circle of radius i centered at O). Finding the set of all valid configurations of the point and the line that cage P is the target in this paper. The bounded configuration space $\mathcal{F} \subset \mathbb{R}^4$ represents the set of all possible configurations of the point and the line. Formally $\mathcal{F} = \mathcal{F}_p \times \mathcal{F}_l$ in which $\mathcal{F}_p$ is the set of all points which lie outside the polygon and inside the ω_1 and $\mathcal{F}_l$ is the set of points which are obtained from a map of all lines which do not intersect the polygon and do intersect the ω_3. It can be proven that any point which lies outside the ω_1 is not the point of any caging configuration, also any line which does not intersect the ω_3 is not the line of any caging configuration.

A unit trajectory of a point is a continuous function $Tp : [0,1] \to \mathcal{F}_p$ that starts at $Tp(0) \in \mathcal{F}_p$ and terminates at $Tp(1) \in \mathcal{F}_p$, where $Tp(t)$ denotes the position of the point on the plane at a normalized time $t \in [0,1]$. A unit trajectory of a line is a continuous function $Tl : [0,1] \to \mathcal{F}_l$ that starts at $Tl(0) \in \mathcal{F}_l$ and terminates at $Tl(1) \in \mathcal{F}_l$, where $Tl(t)$ denotes the position of the line on the plane at a normalized time $t \in [0,1]$. A synchronized trajectory pair (Tp, Tl) represents the movement of the point and the line, such that $(Tp(t), Tl(t))$ denotes the system's configuration at a normalized time $t \in [0,1]$.

For $x = (p,l) \in \mathcal{F}$, we refer to the segment which starts at p and ends at its perpendicular intersection point with l (and does not contain the intersection point) as $h[x)$ or $h[p,l)$. x is a free configuration if $h[x)$ does not intersect the interior of the polygon. A synchronized trajectory pair which starts at $x \in \mathcal{F}$ and terminates at a free configuration is an escape trajectory of x. The Euclidean distance from p to l is represented by $d(x)$. Separation distance of the synchronized trajectory pair (Tp, Tl) is the maximum distance from $Tp(t)$ to $Tl(t)$ during the trajectory, and is represented by $Sd(Tp,Tl) = \max\limits_{0 \le t \le 1} d(Tp(t), Tl(t))$. The critical distance of x is the minimum separation distance of all escape trajectories of x and is represented by $cd(x) = \min\limits_{\forall (Tp,Tl)} Sd(Tp,Tl)$ where (Tp,Tl) is an escape trajectory of x. Obviously, for any $x \in \mathcal{F}$ there is $cd(x) \ge d(x)$.

Proposition 1 *$x \in \mathcal{F}$ is a caging configuration if and only if $cd(x) > d(x)$.*

Caging configurations x and x' are connected if there is a synchronized trajectory pair from x to x' in which all configurations are caging. We refer to a maximal connected set of caging configurations as a maximal caging set.

Lemma 2 *If x is a caging configuration and there is a synchronized trajectory pair (Tp,Tl) from x to x' that $Sd(Tp,Tl) < cd(x)$ then x and x' are connected.*

Lemma 3 *If x and x' are connected, $cd(x) = cd(x')$.*

Regarding the proposition 1 computing the critical distance of all configurations is sufficient to find the caging configurations. According to Lemma 3 the critical distance of all members of a maximal caging set are equal. Hence, in the following, the critical distance of maximal caging sets are computed by partitioning the configuration space and creating a corresponding graph.

3 Configutation Space Partitioning

The vertices of the polygon P in counterclockwise order are called $v_1, v_2, \ldots, v_n$ and its edges are called $e_i = v_i v_{i+1}$ ($1 \le i \le n$ and $v_{n+1} = v_1$). The vertices of the convex hull of P in counterclockwise order are called $V_1, V_2, \ldots, V_m$ in which $m = |CH(P)|$ and its edges are called $E_j = V_j V_{j+1}$ ($1 \le j \le m$ and $V_{m+1} = V_1$). We represent a partitioning for the configuration space, such that the resulting cells of the partitioning do not have intersections with each other unless in their boundaries. $x = (p,l) \in \mathcal{F}$ is in cell $F_{ij} \subset \mathcal{F}$, if e_i is the first edge of P that intersects with segment $h[p,l)$ and V_j is the nearest vertex of $CH(P)$ to line l. And $x = (p,l) \in \mathcal{F}$ is in cell $F_{free} \subset \mathcal{F}$ if $h[p,l)$ does not intersect P. Furthermore, we split some cells into two or three cells in order to make cells reaching some properties. In the following, the two types of split functions are introduced.

For $F_{ij} \subset \mathcal{F}$ we call the line which is along E_{j-1} as the left boundary line or *Lb* for short, and the line which is along E_j as the right boundary line or *Rb* for short. Directions of *Lb* and *Rb* show the range of directions for the line of F_{ij}'s configurations.

Type 1: If there is a configuration $(p_0, l_0) \in F_{ij}$ in which l_0 passes through V_j and directions of l_0 and e_i are equal, then split the F_{ij} into two cells of F^l_{ij} and F^r_{ij}. Cell F^l_{ij} contains $(p,l) \in F_{ij}$ in which the direction of l is between directions of *Lb* and l_0, and cell F^r_{ij} contains $(p,l) \in F_{ij}$ in which the direction of l is between directions of l_0 and *Rb*. By this split the line direction of a cell's configurations are all greater or all smaller than the direction of e_i.

Type 2: For cell F^k_{ij} ($k \in \{null, r, l\}$) and its boundary lines *Lb* and *Rb*, we call the endpoint of e_i, which has the smaller summation of distances from *Lb* and *Rb*, as nearest endpoint or N for short. If there is a configuration $(p_0, l_0) \in F^k_{ij}$ in which l_0 passes through V_j and is perpendicular to NV_j, then split the F^k_{ij} into two cells of $F^{l'}_{ij}$ and $F^{r'}_{ij}$. Cell $F^{l'}_{ij}$ contains $(p,l) \in F^k_{ij}$ in which the direction of l is between directions of *Lb* and l_0, and cell $F^{r'}_{ij}$ contains $(p,l) \in F^k_{ij}$ in which the direction of l is between directions of l_0 and *Rb*. By this split the line direction of a cell's configurations are all greater or all smaller than the direction of the perpendicular line to NV_j.

Finally the number of obtained cells is of the order of $O(nm)$. Regarding the properties of cells it can be observed that, for $(p_0, l_0) \in F_{ij}^k$ $(k \in \{null, r, l, r', l'\})$ in which $p_0 \in e_i$, if p_0 goes to N along e_i while the line is static, the distance of the point and the line decreases in the trajectory. Also for $(N, l_0) \in F_{ij}^k$ in which l_0 passes through V_j, if l_0 rotates around V_j to Lb or Rb while the point is static at N, the distance of the point and the line change strictly monotonic in any of both trajectories. Configuration $x_{min} \in F_{ij}^k$ is a local minima of F_{ij}^k if for all $x \in F_{ij}^k$ there is $d(x_{min}) \leq d(x)$. Therefore, $x_{min} = (N, B)$ where B is either Lb or Rb which is the nearest one to N.

Lemma 4 *If x_{min} is the local minima of F_{ij}^k, for any $x \in F_{ij}^k$ there is a trajectory that starts at x and ends at x_{min} in which the distance of the point and the line decreases.*

Proof. Construct the trajectory which starts at $x = (p, l)$ and continues by getting p and l close to each other along $h[x]$, and terminates at $x' = (p', l')$ in which p' is the intersection of $h[x]$ with e_i and l' is the parallel line to l passing through V_j. The trajectory continues by moving p' along e_i to N while the line l' is static, and then continues by rotating l' to B while the point is static at N. Obviously the final configuration is $x_{min} = (N, B)$ and the distance between the point and the line decreases during the trajectory. □

Lemma 5 *Any cell which is obtained from the configuration space partitioning has intersection with at most one maximal caging set.*

Proof. If x_{min} is the local minima of F_{ij}^k and $x_0 \in F_{ij}^k$ is a caging configuration, there is a trajectory with separation distance of $d(x_0)$ which starts at x_0 and terminates at x_{min} (Lemma 4); thus, x_0 and x_{min} are connected (Lemma 2). In results x_{min}, x and similarly all the other caging configurations of F_{ij}^k are members of the same maximal caging set. □

Conclusion: F_{ij}^k has intersection with a maximal caging set if and only if the local minima of F_{ij}^k is a caging configuration.

Lemma 6 *If x_{min} is the local minima of F_{ij}^k then $\min\limits_{x \in F_{ij}^k} cd(x) = cd(x_{min})$.*

Proof. If x_{min} is not a caging configuration then there is not any caging configuration in F_{ij}^k (Lemma 5) and $cd(x) = d(x)$ for all $x \in F_{ij}^k$. In results $\min cd(x) = \min d(x) = d(x_{min}) = cd(x_{min})$. If x_{min} is a caging configuration, assume to the contrary that there is $x_0 \in F_{ij}^k$ for which $cd(x_0) < cd(x_{min})$. According to Lemma 4 there is a trajectory from x_0 to x_{min} with separation distance of $d(x_0)$ which is $d(x_0) \leq cd(x_0) < cd(x_{min})$. So on the reverse trajectory is the one from the caging configuration x_{min} to x with separation distance of less than $cd(x_{min})$. So $cd(x) = cd(x_{min})$ (Lemma 2), which contradicts our assumption. Thus, $cd(x) \geq cd(x_{min})$ for all $x \in F_{ij}^k$ and it results $\min cd(x) = cd(x_{min})$. □

Theorem 7 *Any $x \in F_{ij}^k$ is a caging configuration if and only if $d(x) < cd(x_{min})$.*

Proof. If x is a caging configuration we have $d(x) < cd(x)$, also x and x_{min} are connected (Lemma 2); and $cd(x) = cd(x_{min})$ (Lemma 3); so, $d(x) < cd(x_{min})$. If $d(x) < cd(x_{min})$ it is equal to $d(x) < \min\limits_{x \in F_{ij}^k} cd(x)$ (Lemma 6) ; hence, $d(x) < cd(x)$ and it means that x is a caging configuration. □

Regarding the Theorem 7, in order to find the set of caging configurations of a cell, it is sufficient to compute the critical distance of the cell's local minima -called critical distance of the cell. In the following, computing the critical distance of cells is discussed.

4 Finding Critical Distance of Cells

Considering $x_b = (p_b, l_b) \in \mathcal{F}$ such that p_b is on the boundary of the polygon and l_b is a tangent line to the polygon. For escape trajectory (Tp, Tl) of x_b, the corresponding squeezed escape trajectory (Tp_s, Tl_s) with smaller separation distance is constructed in 3 steps. In the first step, for $t \in [0, t']$ when t' is the smallest value that $(Tp(t'), Tl(t'))$ is a free configuration, call the first intersection point of $h[Tp(t), Tl(t))$ with the polygon as $Tp'(t)$, and from the two lines which are parallel to $Tl(t)$ and tangent to the polygon call the nearest one to $Tl(t)$ as $Tl'(t)$. Obviously the separation distance of (Tp', Tl') is smaller than or equal to the separation distance of (Tp, Tl). In the second step, if Tp' is discontinuous, add segments between each two consecutive discontinuous points to make the continuous trajectory Tp'', so the separation distance of (Tp'', Tl') and (Tp', Tl') in $[0, t']$ are equal. In the third step, scale t' in order to make a synchronized trajectory pair in $[0, 1]$. We refer to the obtained trajectory as squeezed escape trajectory and represent it with (Tp_s, Tl_s). Obviously $Sd(Tp_s, Tl_s) \leq Sd(Tp, Tl)$. Thus, to find the critical distance of x_b computing the separation distance of all different squeezed escape trajectories is sufficient. Since the local minima of a cell, which was represented by $x_{min} = (N, B)$, has the properties of x_b, in order to find the critical distance of a cell, we will compute the separation distance of all squeezed escape trajectories of the cell's local minima. In the following, the relationship among critical distances of adjacent cells have been formalized.

Consider $B(F_i, F_j)$ represents the set of configurations which are in common boundaries of F_i and F_j. If $x_1 \in B(F_i, F_j)$ and $x_2 \in B(F_j, F_k)$ there are two trajectories, one from x_1 and the other from x_2 to local minima of F_j with separation distances of respectively $d(x_1)$ and $d(x_2)$. Hence, there is a trajectory from x_1 to x_2 with separation distance of $\max\{d(x_1), d(x_2)\}$. It concludes that for a squeezed escape trajectory which starts at x_0 and passes through $F_1, F_2, \ldots, F_f, F_{free}$ the separation distance is $\max\limits_{1\leq i\leq f}\{d(x_i)\}$ where $x_i \in B(F_i, F_{i+1})$. Thus, among all squeezed escape trajectories passing through $F_1, \ldots, F_f, F_{free}$ the one which passes through the local minima of common boundaries' configurations has the minimum separation distance.

There is a transition between each two cells F_i and F_j if $B(F_i, F_j) \neq \emptyset$, and the cost of the transition is equal to the minimum of $d(x)$ for all $x \in B(F_i, F_j)$. In the worst case situation, the number of transitions that a cell is associated with can be as high as $O(n)$. Fortunately, we found out there are only $O(1)$ transitions associating with a cell, which are called basic transitions, that have effects on the computation of separation distance of any squeezed escape trajectory.

Theorem 8 *If there is a transition between F_1 and F_2 with the cost of $d(x_{12})$, there is a sequence of basic transitions that starts at F_1 and ends at F_2, in which the maximum cost of the basic transitions are less than or equal to $d(x_{12})$.*

For $x_{min} \in F_i$, any squeezed escape trajectory is corresponding to a sequence of cells starting at F_i and ending at F_{free}. Thus in order to compute the $cd(x_{min})$ which is the minimum separation distance of all squeezed escape trajectories for x_{min}, we should find the minimum of maximum of basic transitions' cost in any sequence of cells starting at F_i and ending at F_{free}. In this paper the basic transitions are not described; however, finding the basic transitions and their cost is of order $O(nm\log n)$ by running $O(nm)$ times of Ray shooting algorithm. From insights we have obtained so far, we are now ready to concrete the definition of our search space.

Connectivity graph $G = (V_G, E_G)$ is a weighted graph that any vertex $vg_i \in V_G$ corresponds to cell $F_i \subset \mathcal{F}$ and any edge $e_{ij} \in E_G$ corresponds to a basic transition between two cells F_i and F_j, and its weight is equal to the cost of the basic transition which is called w_{ij}. Also critical distance of $vg_i \in V_G$ is equal to the critical distance of F_i which is called $D(vg_i)$.

Theorem 9 *For any $vg_i \in V_G$ if $N(vg_i) \subset V_G$ is the set of all vertices which are adjacent to vg_i, then $D(vg_i) = \min\limits_{vg_j\in N(vg_i)}(\max(D(vg_j), w_{ij}))$.*

According to the Theorem 9 if vg_i and vg_j are adjacent vertices of G then $D(vg_i) \leq \max\{(D(vg_i), w_{ij})\}$; To compute the critical distances of vertices we present a slightly modified Dijkstra algorithm. The first step of the algorithm is to initialize the critical distance of all vertices as infinite except the critical distance of vg_{free}, which corresponds to cell $F_{free} \subset \mathcal{F}$, as zero. In each step a vertex vg_i, which has the minimum critical distance between all the not selected vertices, is selected and the critical distance of its adjacent vertices are updated; If the present critical distance of $vg_j \in N(vg_i)$ is greater than $\max\{(D(vg_i), w_{ij})\}$ set $\max\{(D(vg_i), w_{ij})\}$ as the updated critical distance of vg_j. Then add the vg_i to the set of selected vertices. Do the same procedure until all the vertices have been selected. Finally, by having critical distance of vertices, the set of all caging configurations of any cell F_i, which is represented by $C_i = \{x \in F_i | d(x) < D(F_i) = D(vg_i)\}$, can be computed. To decide whether a given configuration x is a caging configuration or not, find cell F_i that $x \in F_i$ by running the Ray shooting algorithm in $O(\log n)$, then $d(x)$ and $D(F_i)$ should be compared.

5 Conclusion

It was discussed that computing the critical distance ofl maximal caging sets is sufficient to find all the caging configurations. Thus, the configuration space was partitioned and the critical distance of cells was computed by searching in a graph. Computing the vertices and the edges of the connectivity graph are respectively of the order of $O(nm)$ and $O(nm\log n)$, searching the graph to compute the critical distance of cells is also $O(nm\log n)$. Therefore, the running time of the algorithm is $O(nm\log n)$ and the query can be answered in $O(\log n)$ time.

References

[1] M. Vahedi. *Caging Polygons with Two and Three Fingers.* Phd thesis, Department of Information and Computing Science, Utrecht University, 2009.

[2] W. Kuperberg. *Problems on Polytopes and convex sets.* DIMACS Workshop on Polytopes, 1990.

[3] A. Bicchi, V. Kumar. *Robotic Grasping and Contact: A Review.* In ICRA, IEEE, 2000.

[4] B. S. Eberman. *Whole Arm Manipulation: Kinematics and Control.* Thesis, Department of Mechanical Engineering, Massachusetts Institute of Tech, 1989.

[5] C. Wentink, A. F. van der Stappen, Mark Overmars. *Algorithms for Fixture Design.* UU-CS, 1996.

[6] P. Pipattanasomporn, A. Sudsang. *Two-finger caging of nonconvex polytopes.* IEEE Transaction on Robotics, 2011.

[7] M. Vahedi, A. F. van der Stappen. *Towards Output-Sensitive Computation of Two-Finger Caging Grasps.* In CASE, 2008.