Reactive Algorithms for Grasping using a modified parallel jaw gripper

Marek Teichmann and Bud Mishra

Abstract

We consider the problem of grasping an unknown polygonal flat object using a parallel jaw gripper. Here we propose to equip a standard gripper with several light-beam sensors (close to each jaw) and describe a reactive grasping algorithm. This is done by probing the object to locate a good grasp position, and then grasping, without moving the object significantly. The goal is to do as little motion as possible to find a grasp. This algorithm can be viewed in a competitive framework, where our algorithm is competing against any algorithm which already knows the object.

1 Introduction

In the past, the usual approach to grasping has been to assume an accurate model of the object to be grasped and from such a model, an offline geometric algorithm determines a set of grip points, where the fingers are then placed. (See Mishra, Schwartz and Sharir [MSS87]). Once the grip points have been determined, the geometry of the object is deemed irrelevant and the grasp is determined and maintained by only controlling the magnitudes of the forces at the grip points. However, such a picture has not proven very useful, as in practice, these algorithms have turned out to be extremely non-robust. The only success in dexterous manipulation seems to have come from two directions: (1) telemanipulation, where a human in the loop uses much more sensory information than what is assumed theoretically necessary, and (2) simple parallel jaw grippers, where grasping algorithms can be made immune to lack of any sensory information (for instance, by removing friction in the transversal direction between a jaw of the gripper and the grasped object; see Goldberg [G90]). Here, we take an approach, where the sensory information is selectively used up to determine movement of the robot ("state-transitions") so that ultimately a grasp will be achieved independent of the quality of the sensory information or the reliability of the actuator movements.

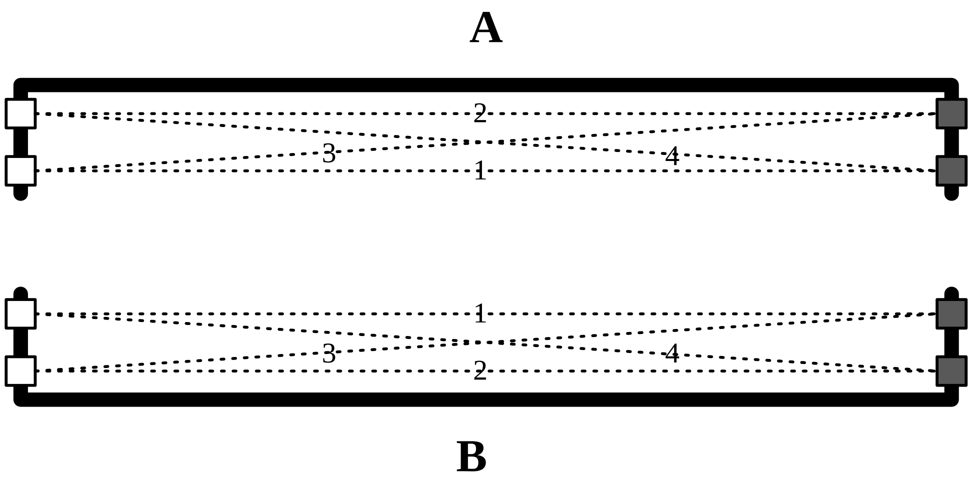

Figure 1: Parallel jaw gripper with sensors

Furthermore, unlike Goldberg's algorithm, our algorithm is non-disturbing in the sense that it does not cause any movement of the objects except for the movements desired.

However, our "algorithm" is rather different from what is usually understood by algorithms: the robots (i.e. actuators and sensors) themselves are active components of the algorithm, performing "computations" much like an analog computer in conjuncts with a digital computer. For this reason, the "algorithm design" also implies a careful choice of the participating actuators and sensors and their placement. For this reason, the algorithm does not separate in any meaningful way phases of "digital planning" interrupted by "analog actuation and sensing". Thus our reactive algorithm is much smoother in operation.

However this approach complicates the question of the "complexity" of the algorithm as individual steps of the algorithm are not clealy discernible, nor is there a clear-cut understanding of input size. We resolve this by proposing to use the notion of competitiveness (as in on-line algorithms, or in economics) in a novel way. We show that our algorithm is competitive.

Our algorithm can be described roughly as follows. We consider a standard parallel jaw gripper and attach to each jaw two infrared emitting diodes (LEDs) and two infrared detectors as depicted in figure 1.

This allows us to have in fact 4 light beams which we can test for being interrupted or not by alternately illuminating the IR LEDs.

The general scheme will be to close the gripper while constantly checking the status of the sensors, and performing actions according to that status in a table driven manner.

These algorithms can be used as low level primitives for grasping applications, for example in automated manufacturing. With minor modifications, the algorithms can also be used for determining the shape of (the convex hull of) a planar part, again without disturbing the part.

2 Related Literature

Our work shares some similarities with Goldberg's gripper [G90] and is motivated by the same paradigm as Canny and Goldberg's RISC robotics [CG93].

In [G90], Ken Goldberg has modified the basic gripper to reduce the friction between the gripper and the object being grasped. This friction occurs when the gripper is being closed on an object, and this object is forced to rotate. Several results exist on finding grasps for such a parallel jaw gripper, when the object is known [RG92a, RG92b]. Recent results consider closing the gripper on an unknown object, and measuring the jaw distance for various orientations [R93]. This permits the determination of the object, but also subjects it to many motions. Another method involves grasping in random orientations to distinguish planar parts.

In the Computational Geometry literature, there are several results on probing. Probing using finger probes [CY], and line and other probes [DEY] is considered. We are mainly interested in line probes: a line is swept (either in parallel or rotating) until the object is hit. The probe returns the position of the line where it hit the object first.

In [T93], Teichmann has proposed to add probing capabilities to a basic parallel jaw gripper and to combine probing techniques with grasping using light beam sensors and proximity sensors. The goal is to produce algorithms for obtaining grasps with as little knowledge as possible about the object, and without touching the object. Here we present fast grasping algorithms for such grippers equipped with light beam sensors as described below.

We also present a competitive framework in which these algorithms are analyzed.

Algorithm 1

State	A_2	A_1	B_1	B_2	*Action*
0	.	.	.	.	close the gripper
1	.	x	.	.	move up
2	.	.	x	.	move down
3	.	x	x	.	rotate in current direction
4	x	x	x	.	move up (parallel to jaw before previous rotation)
5	.	x	x	x	move down (parallel to jaw before previous rotation)
6	x	x	.	.	move up
7	.	.	x	x	move down
8	x	x	x	x	On first entry: reverse rotation direction, rotate Otherwise: stop

3 The Algorithm

3.1 The Diameter Function

Consider a fixed polygonal object P, and define the function $d(\theta)$ to be the diameter of P in the direction θ. This is defined as the length of the projection of P (a line segment) on a line that forms an angle of θ with the x-axis. Antipodal points pairs correspond to local maxima, and local minima correspond to possible grasps.

The current configuration of the gripper consists of

- the gripper orientation,
- the inter-jaw distance,
- the position of (the center of) the gripper.

The position of each pair of parallel beams can be represented by a point in the graph of the diameter function. Rotations of the gripper correspond to moving the point parallel to the θ axis (horizontally), and opening and closing of the gripper to moving vertically.

The four pairs of beams can be represented by four points in a '+' configuration. This last representation is not precise however since the translational component of the beam positions is not modeled.

3.2 First Algorithm

We assume that the initial orientation of the gripper is at angle 0 (with the x-axis, say), and that the gripper rotates around its center of symmetry.

Let A_1, A_2, A_3, A_4 be the state of the light beams on jaw A, and B_1, B_2, B_3, B_4 be the same for jaw B. In the following, *up*-ward translation means translation perpendicular to the gripper jaws and towards jaw A, *down* is towards jaw B. This also defines naturally *left* and *right* translation. Also we have a *current direction*, which is clockwise initially, and the initial position is such that the object is somewhere between the gripper jaws.

The algorithm checks the current state and performs an action according to the table above, where an 'x' indicates that the corresponding beam is interrupted. The action is performed until the state changes. This is performed in a tight control loop, and hence the motions are very small. We assume that initially the object being grasped in somewhere between the parallel jaws.

For example, at the beginning we are in state 0 and the gripper closes until either state 1 or 2 is reached. Then a small translation occurs and immediately we are back in state 0. An initial direction of rotation is chosen arbitrarily, which might be reversed in state 8.

Let D_{12} be the sum of the distances between beams 1 and 2, for each jaw.

Let θ be a local minimum of the diameter function d, and consider the plane containing the graph of the diameter function. Define a *valley of height h around θ* as the set of points (x, y) that are above the graph of d satisfying:

$$\max(d(x), d(\theta)) \leq y \leq d(\theta) + h$$

$$\max_{x<\theta}\{d(x) \leq d(\theta)\} \leq x \leq \min_{x>\theta}\{d(x) \leq d(\theta)\}.$$

The constraint on x says that the valley stops after d dips below $d(\theta)$. See figure 2. Then we have the following

Theorem 1. *For a flat polygonal object that has significantly smaller diameter than the length of the jaws, Algorithm 1 will terminate with a gripper orientation which is in the valley of height D_{12} that is closest to the initial orientation. We say it finds the closest local minimum up to its resolution.*

Proof: At each step, we either close the gripper or translate/rotate so that the gripper can close further. We can think of this algorithm as tracking the diameter curve using two vertically positioned test point, at a distance of D_{12} apart. The bottom point corresponds to the inner pair of beams, the top point to the outer pair, and we want to keep the curve between the two points (state 3). When the bottom point gets above the curve we are in state

0, and we close (go down in the diameter function graph). When the top point moves below the curve we are in state 8. States 1,2,4,5,6,7 take care of translating the gripper. This is not modeled by the diameter graph.

To show that the algorithm converges to a local minimum, we show that at any step we make progress. In state 0 we advance down in the diameter graph, and in state 3 we advance in the chosen direction. This direction can changes only once, when the initial direction increases the diameter graph. The other states, except state 8, reduce the number of interrupted beams by translating the gripper in an appropriate direction and hence eventually allow us to get to state 0 or 3. States 4 and 5 need some explanation. These are reached after a rotation (we assume that the gripper closes with sufficient precision so that beam 1 is always interrupted before beam 2 and we detect both events separately.) It can be seen that if we translate parallel to the jaws as they were before the previous rotation, we bring the object closer to the center of the gripper.

State 0 and 3 may increase the number of interrupted beams but we make progress by closing the gripper of rotating further, respectively. When state 8 is reached for the first time, it might be because we have chosen a wrong initial direction, and the diameter increases during that rotation. This is the only case where we rotate with all 4 beams interrupted. State 8 can only be entered from states 3, 4 and 5 and a detailed geometric analysis shows that the beam numbered 2 that is broken will be cleared. Step 8 thus reverses the search direction and continues. Upon second entry to state 8 we are certain of having crossed a local minimum, and the diameter function increases by at least D_{12}, and we are done. ∎

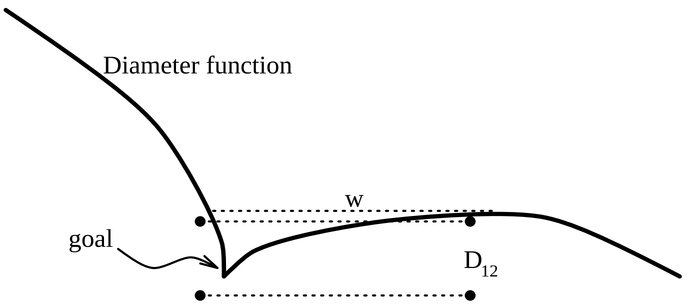

Figure 2: Angular error due to distance between beams

This algorithm will miss any "small" local minima whose height is smaller than D_{12}. This is the limit of the *resolution* of the gripper. As we decrease

D_{12} to zero, the algorithm misses smaller and smaller local minima, and ultimatly tracks the diameter curve with total precision. This is physically impossible, so we accept a small error.

This error implies that we don't have any maximum angle error guarantee since we can overshoot by a large angle if the diameter function increases slowly, see figure 2. For example for an oval-like polygonal shape, if the difference between the minimum and the maximum diameter is too small, we might miss the minimum. The algorithm in the next section finds a local minimum up to the resolution of the actuators and sensors.

3.2.1 Modification to get an Angular Error Guarantee

Once we have reached such a valley in the diameter function, we need to search it for the actual minimum if more precision is required. Once we are in state 8 above however, we cannot perform large rotations, since all the beams are interrupted, and rotation requires the outer beams for translational position adjustments. We hence depart from our table driven model and define a new primitive, which allows us to get out of this situation:

Back-up will move the gripper to the position it was before leaving the previous state.

This will be a very small motion and it can be done by saving the current position every time rotation or gripper closing is performed.

So in state 8, we Back-up and are free to rotate. Now the search will essentially be a binary search. We reverse the current direction, rotate (while adjusting position) to an angle halfway between the two previously saved positions, and close the gripper (again adjusting for position) until all beams are interrupted. After closing the gripper until beams number 2 are both interrupted, we measure the diameter, and Back-up (open the gripper). We need to decide in which direction to continue searching. We just choose an arbitrary direction, save the current angle and rotate. If we can reach the midpoint between the saved angle and the next furthest saved angle in the direction we are rotating, we continue by closing. Otherwise we have chosen the wrong direction and we Back-up, reverse and restart at the last saved angle. We can continue this procedure until the angles change by some previously established error bound.

Algorithm 2

State	A_3	A_4	B_4	B_3	*Action*
0	.	.	.	.	close the gripper
1	x	.	.	.	translate left and up
2	.	x	.	.	translate right and up
3	.	.	x	.	translate left and down
4	.	.	.	x	translate right and down
5	x	x	.	.	translate up
6	.	.	x	x	translate down
7	x	.	.	x	rotate clockwise
8	.	x	x	.	rotate counter–clockwise
9	x	.	x	.	translate left
10	.	x	.	x	translate right
11	x	x	.	x	translate parallel A_4 to right
12	x	x	x	.	translate parallel A_3 to left
13	x	.	x	x	translate parallel B_3 to right
14	.	x	x	x	translate parallel B_4 to left
15	x	x	x	x	check if done (see text)

Oscillations

It is conceivable that at some point due to inaccuracies or time delays, the algorithm would oscillate between a pair of "opposite" state (ex. 1 and 2, or 4 and 5). This could also happen if the beam distance is exactly the diameter of the polygon in the current orientation. This can be avoided by checking if no gripper closure has occurred between those states and forcing the algorithm to go to state 0 (close).

3.3 Competitiveness Analysis

Ideally, we would like to compare this algorithm with the best possible abstract algorithm that knows the object to be grasped. Unfortunately, it is difficult to evaluate the translational cost, and we only consider rotations here. We find the closest valley containing a local minimum in the direction of decreasing slope at the initial orientation, so this cost is inevitable. There is only a small overhead (depending on the slope of diameter function) initially

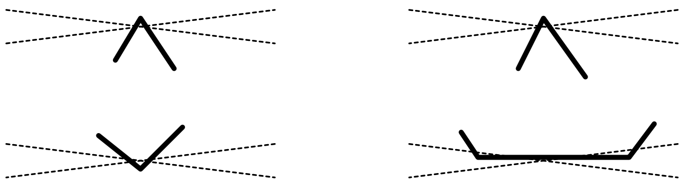

Figure 3: Two possible situations in state 15.

to find the right direction, hence our algorithm is competitive up to its resolution. If we wish to go beyond this resolution, we must consider the cost of the binary search at the second stage. Let w be the length of the valley reached by the algorithm. Then the cost of the binary search is bounded by $3w$ plus the overhead at each end of each rotation, which determines the direction of the next step. Unfortunately, this overhead depends on the polygon and can be large if the diameter function has nearly flat sections.

4 Second Algorithm using Crossed Beams

In the table above we present a different algorithm that performs not only rotations but also translation according to the beam states. It uses the angled beams 3 and 4.

The translations in states 1 to 4 are done perpendicular to the corresponding interrupted beams. The last state is more complex. We must discriminate between the two situations depicted in figure 3. This could be easily done by adding another beam (beam 5), parallel to beams 1 and 2 and passing through the intersection point of beams A_3 and A_4. Similarly we add beam 6, to the other jaw. Let C_A be the point where beams A_3 and A_4 intersect, define C_B similarly. Such a beam permits us to determine whether there is a vertex at C_A or C_B. We want to avoid having a vertex at both. In such a case, we are stuck at a local maximum, and we escape by performing a small rotation in the same direction as the previous rotation. If these additional beams are available, then the algorithm can be made entirely table driven.

In fact, by moving this beam slightly further from its jaw, we can eliminate cases when there is a vertex on each side such that the line joining them is almost orthogonal to the jaws. This situation corresponds to low stability grasps.

We can simulate beam 5 using either beam 1 or 2, and repositioning the gripper, assuming we have precise measurement data for the current gripper width.

For this algorithm we also need to assume that there are no two edges forming a very small angle. Also it seems that this algorithm cannot be generalized to objects with rounded boundaries, since it depends on the presence of vertices.

5 Conclusion

We have presented simple algorithms for finding a grasp of a thick planar object using a modified parallel jaw gripper. These modifications are simple and obviate the need for complex vision systems. This is one of the first attempts at producing robot manipulators that know how to grasp by themselves, feeling the object first, but only sufficiently in order to grasp it. We are currently in the process of constructing such a gripper, and we expect the algorithms to perform very smoothly, since the motions involved between steps will be very small.

References

[CG93] J.F. Canny, K.Y. Goldberg, *A "RISC" Paradigm for Industrial Robotics*, 1993 IEEE Int'l Conference in Robotics and Automation, Atlanta, May 1993.

[CY] R. Cole, C.K. Yap, *Shape from Probing*, J. Algorithms, Vol. 8, No 1, pages 11–38, 1987.

[DEY] D. Dobkin, H. Edelsbrunner, C. K. Yap, *Probing Convex Polytopes*, Proceedings of the 18th STOC, 1986, Pages 424–432.

[G90] K. Y. Goldberg, *Stochastic Plans for Robotic Manipulation*, Ph.D. Thesis, School of Computer Science, Carnegie Mellon University, Pittsburg, PA, 1990.

[MSS87] B. Mishra, J.T. Schwartz, and M. Sharir. On the Existence and Synthesis of Multifinger Positive Grips. *Algorithmica*, 2:541–558, 1987.

[R93] A. S. Rao, *Algorithmic Plans for Robotic Manipulation*, PhD. Thesis, University of Southern California, Los Angeles, May 1993.

[RG92a] A. S. Rao, K. Y. Goldberg, *Shape from Diameter: Strategies for Recognizing Polygonal Parts.* Technical Report #292, University of Southern California, Inst. of Robotics and Intelligent Systems (IRIS), Los Angeles, California, 1992.

[RG92b] A. S. Rao, K. Y. Goldberg, *Grasping Curved Planar Parts with a parallel jaw Gripper*, Technical Report #299, IRIS, August 1992.

[T93] M. Teichmann, Unpublished Notes, May 1993.