

Grasp Synthesis in Cluttered Environments for Dexterous Hands

Dmitry Berenson

The Robotics Institute, Carnegie Mellon University
Pittsburgh, PA, 15213 dberenso@cs.cmu.edu

Siddhartha S. Srinivasa

Intel Research Pittsburgh
Pittsburgh, PA, 15213 siddh@cmu.edu

Abstract—We present an algorithm for efficiently generating collision-free force-closure grasps for dexterous hands in cluttered environments. Computing a grasp is complicated by the high dimensionality of the hand configuration space, and the high cost of validating a candidate grasp by collision-checking and testing for force-closure. When an object is placed in a new scene, we use a novel cost function to focus our search to good regions of hand pose space for a given preshape. The proposed cost function is fast to compute and encapsulates aspects of the object, the scene, and the force-closure of the ensuing grasp. The low cost candidate grasps produced by the search are then validated. We demonstrate the generality of our approach by testing on the 3-fingered 4DOF Barrett hand and the anthropomorphic 22DOF Shadow hand. Our results show that the candidate grasps generated by our algorithm consistently have high probability of being valid for various hands, objects and scenes. Finally, we describe an implementation on a WAM arm with a Barrett Hand.

I. INTRODUCTION

Human environments are filled with clutter. When we reach to pick up a mug on a dish rack or on a table full of mugs (Fig.1), the direction we approach, the shape of our hand, and the grasp that we choose are acutely affected by the surrounding clutter. It is often necessary to choose grasps which would rarely have been chosen in the absence of clutter.

In this paper, we present an algorithm, illustrated in Fig.1, that enables a dexterous hand to reach in and pick up an object in a cluttered scene without touching the surrounding clutter. Our algorithm is based on three observations. First, we realize that while there might be an ideal grasp for the mug, we must allow the algorithm to adapt that grasp to fit the environment. Second, to reach in and grasp the object, the hand must have a clear approach direction to the object. Third, there must be enough clearance around each contact point to allow the fingers to curl in and make contact.

We compute a novel cost function that encapsulates all three observations. The cost function guides a search in the pose space of the hand and produces a set of grasp candidates that have a high probability of being easy to approach, collision-free as the fingers curl in, and fit the preshape well. Most importantly, the entire process takes, on average, as much time as testing about 8 grasps. Furthermore, the resulting grasp set has a success rate of over 80% in most tested scenes.

As an example, an evaluation of the predictive power of the Conical Clearance Map (ConCM), which encapsulates finger clearance, is shown in Fig.1. Here, green and black points correspond to predicted good and bad clearance, respectively, and red points correspond to the contact points of 1000 random successful grasps. The ConCM accurately predicts which parts

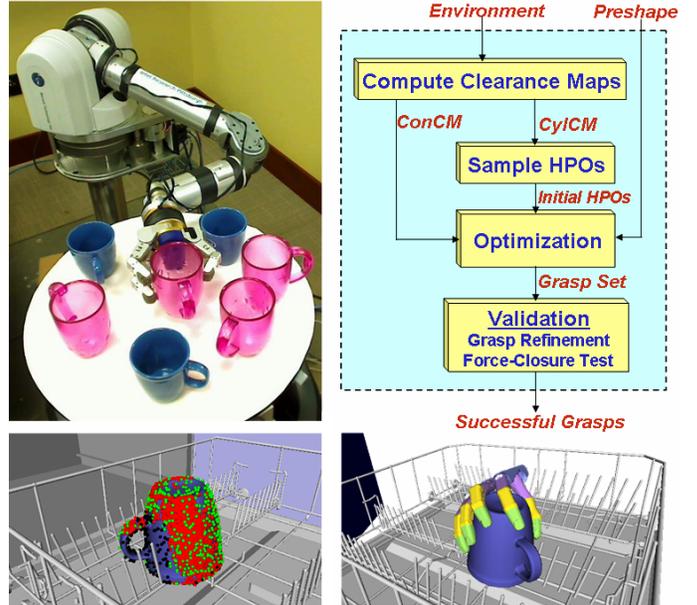


Fig. 1. Outline of the algorithm and implementation on the Barrett and Shadow hands. Bottom Left: ConCM predicting good finger placements

of the object are likely to be contacted by successful grasps and is also very quick to compute.

For best performance our algorithm requires the complete geometry of the scene but such information is rarely available. Instead, the algorithm can be given rough models of obstacle regions in the scene derived from vision or laser data without a significant change in performance. However, the algorithm is sensitive to the geometry of the object being grasped so its model should be fairly accurate.

II. BACKGROUND AND TERMINOLOGY

Autonomous manipulation has been a major goal in robotics for many years and has spawned numerous platforms such as the ARMAR [1], Dexter [2], Domo [3], HRP2 [4], Justin [5], the NASA Robonaut [6], STAIR[7], and UMan[8]. There is renewed interest in moving away from tele-operation, manually-scripted grasps, and simple scenes, to grasping with dexterous hands in unstructured human environments.

Early research on grasp synthesis focused on finding placements of contact points on an object's surface to optimize a given grasp metric [9, 10]. However, the applicability of these techniques to dexterous hands with complex kinematics and geometry is an open problem. Recently, [7, 11] have applied machine learning techniques to find grasps of novel objects using information about grasps of already-known objects.

Regardless of the method used for grasp selection, much previous research has focused on finding grasps for the object when it is isolated in the environment or in simple environments. Grasps are often generated as if the object is alone in the environment and then collision-checked, as in [12]. While this approach works well in simple or carefully constructed environments, it is understandably limited as it does not adapt to the environment. As a result, in environments with a great deal of clutter (see Figure 1) this method may take a very long time to find a successful grasp if one can be found at all. This highlights the problem that validating a grasp in a cluttered scene is expensive: with a state-of-the-art implementation, we can evaluate only about 2 grasps per second.

In previous work [13], we proposed an algorithm where a large number of stable grasps for an object were generated offline, ranked based on the environment, and then evaluated in order of rank. While successful for many objects and environments, this method proved problematic for certain environment-object combinations because the algorithm is “locked in” to pre-computed grasps; it cannot generate new grasps even if all that is required is to move the wrist slightly to avoid an obstacle.

We define the configuration of the hand by its internal shape, which we term *preshape*, and its pose. For our algorithm, a *preshape* is the ideal set of joint values of the hand for grasping a particular object. Pose is described as a 6D Hand Position and Orientation (HPO), comprising of position $HPO_p \in \mathbb{R}^3$, and orientation $HPO_o \in \mathbb{H}$ represented as a quaternion. A *Grasp* consists of two parts: A *preshape* and a HPO. We also define a *Directed Point*, which consists of a 3D position (in meters) and a 3D unit vector representing orientation.

We implemented a grasping controller, in simulation and on the real Barrett hand, which allows the fingers to wrap around objects. The hand starts at a certain set of joint values and each finger is curled in until it collides with any obstacle or reaches a joint limit. If a finger is controlled by more than one joint, the distal joints follow the motion of the proximal joint of the finger. If the proximal link collides with an obstacle, the distal joints continue to curl in.

III. PRESAPES

Methods for determining a *preshape* for a given object have been intensely studied in robotics and neuroscience literature for many years and are outside the scope of this paper. *Preshapes* can be selected from a *preshape* set based on nearest-neighbor algorithms [14] or through analysis of the affordances of an object [15]. Also, rule-based [16] and heuristic methods [12] can be used to determine a class of grasp to use, which can then be translated into a *preshape*. In practice, we use the technique of [14] combined with a manual selection of *preshapes* for more difficult objects.

For a given *preshape*, the contacting surface of the hand is sampled using a set of directed points, see Figure 2(a) for an example of such a sampling. This is necessary as subsequent steps of the framework will rely on these points to quickly match shapes and evaluate potential grasps. We also impose

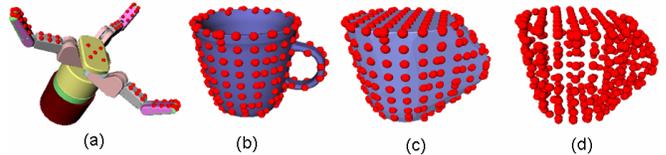


Fig. 2. (a) Sampling on the contacting surface of the Barrett Hand in a certain preshape. (b) The sampled surface of a mug. (c) The sampled surface of that mug’s convex hull. (d) The combined mug samples.

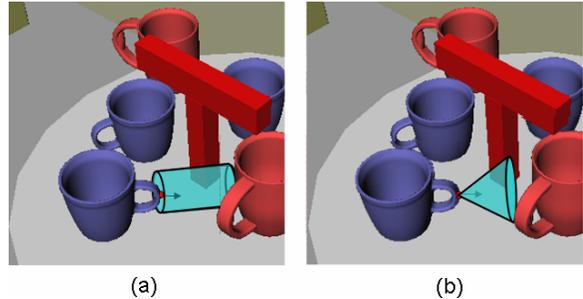


Fig. 3. Depiction of computing clearance for one point on an object using cones and cylinders. The cone/cylinder is oriented along the outward-facing surface normal of a point on the surface of the object. The length of the longest cone/cylinder that is collision-free is the clearance score assigned to that point.

the constraint that the directed points on the fingertips must be in force-closure for the *preshape* to be admissible. The *preshape* is meant to be a rough guess of the joint values of the desired grasp based only on the properties of the object and the hand. However, to determine the HPO of the grasp, we must take into account the object’s local environment.

IV. FINDING A SET OF HPOS

This section describes how to compute a set of valid HPOs given a *preshape* and the environment geometry. For an HPO to be valid: 1) The hand must not collide with the scene during execution of the grasping controller. 2) The ensuing grasp must be in force-closure. The computation necessary to validate an HPO consists of running the grasp controller, collision checking with the scene at every step, and evaluating force-closure.

In our experiments with the Barrett Hand, this process took roughly 0.45 seconds for a single grasp using PQP[17], a state of the art collision checker, and MATLAB’s `linprog`, a state of the art linear program solver used to evaluate force-closure. Given these run times, only about 2 HPOs can be validated in one second, necessitating techniques to focus search in the 6 dimensional HPO space.

Thus we present the following methods to quickly find likely HPOs which are then passed on to the expensive validation step. We sample promising HPOs using information about the object’s local environment and seed an optimizer with this initial sampling. The optimizer uses a novel cost function which attempts to predict the validity of a given HPO. Optimized HPOs are then validated. See Figure Fig.1 for an outline.

A. Generating an Initial Seed

The Cylindrical Clearance Map(CylCM) scores the likelihood of the hand being in collision with the scene at a given

HPO using inexpensive ray-collision checking.

To compute the CylCM, we use samples of the surface of the object being grasped as well as its convex hull (see Figure 2). Since they are only dependent on the object being grasped, the samples are computed offline. Let these samples be called O_{dp} . For each sampled point, we compute the longest cylinder that can be placed at the point and oriented along the outward surface normal without colliding with the scene (see Figure 3a). The length of the cylinder is assigned as the point’s *clearance score*. The cylinder approximates the fixed part of the hand. This fixed part, termed so because it is not controlled by any joints in the hand, usually consists of the palm and portions of the wrist. To generate this approximation, we sweep the fixed part around its central axis and use the largest radius for the CylCM. By construction, the fixed part will not collide with the scene when placed at a sample point if the clearance score of that point is greater than the length of the fixed part. Note that this is a sufficient but not necessary condition for the fixed part to be collision-free.

After generating the CylCM, we extract those sample points \mathbf{p} that are guaranteed to be collision-free for the fixed part. Given a certain number of desired seeds, N , we sample points from \mathbf{p} proportional to their clearance scores. Eqn.1 is used to generate HPO_p from these points.

$$\text{HPO}_p = \mathbf{p} + hl_{\max}\hat{\mathbf{n}} \quad (1)$$

where h is chosen uniformly from $[0, 1]$, l_{\max} is the length of the fingers when they are fully extended, and $\hat{\mathbf{n}}$ is the outward-facing surface normal at \mathbf{p} . At each \mathbf{p} , Eqn.1 produces HPOs that range uniformly from the hand’s palm being flush in contact with the object ($h = 0$) to being barely able to touch it with the fingertips ($h = 1$).

To generate HPO_o , we point the hand along $-\hat{\mathbf{n}}$ and add a random rotation about $\hat{\mathbf{n}}$ to randomize the roll of the hand. The hand’s origin is assumed to be the center of the palm and the hand’s orientation is the hand’s typical direction of approach when grasping. For the Barrett Hand, this direction is normal to the palm surface, while for the Shadow hand this direction is the palm normal offset by 45° toward the fingers.

B. Cost Function

An optimizer takes as input the initial seed and outputs a set of HPOs that is likely to be collision free and result in force-closure grasps. The cost function used by the optimizer must accurately predict hand-scene collisions during the execution of the grasp controller as well as the force-closure of the ensuing grasp. We define the cost of an HPO for object O in environment E using the following terms:

- *Approximate Collision* - $X(\text{HPO}, E)$ - Whether the fixed part of the hand will be in collision.
- *Fit Cost* - $F(\text{HPO}, O)$ - The error of the fit between the preshape and the object at this HPO.
- *Contact Safety Cost* - $S(\text{HPO}, O, E)$ - The likelihood of the fingers being able to reach the desired contact points without collision.

The individual costs are combined in Eqn.2 to produce the cost of the HPO $C(\text{HPO}, O, E)$.

$$C(\text{HPO}, O, E) = \frac{F(\text{HPO}, O) + \zeta S(\text{HPO}, O)}{X(\text{HPO}, E)} \quad (2)$$

where ζ is the trade off between fit and contact safety costs.

To compute $X(\text{HPO}, E)$, we place the approximating cylinder for the fixed part of the hand at the position and orientation given by the HPO and check collision with environment obstacles, note that we do not check collision with O . If the approximating cylinder is not in collision $X(\text{HPO}, E) = 1$, otherwise $X(\text{HPO}, E) = 0$ so that $C(\text{HPO}, O, E) = \infty$ when the approximating cylinder is in collision.

To compute $F(\text{HPO}, O)$, we transform the directed points on the surface of the hand corresponding to the given preshape to the position and orientation given by the HPO. Let these transformed preshape points be P_{dp} . We then perform a nearest-neighbors¹ query to find which directed points in O_{dp} are closest² to those in P_{dp} . The distances to the closest points are averaged as $F(\text{HPO}, O)$, which measures the appropriateness of the HPO when using the given preshape. If the preshape and HPO are not compatible, the grasp controller is not likely to end up in a configuration similar to the preshape (as it curls in the fingers until all have collided) and thus we cannot predict if this grasp will be in force-closure. The lower $F(\text{HPO}, O)$ is, the more likely the hand will be able to achieve the desired preshape after running the grasp controller.

In our experiments, we found that requiring all points on the hand surface to match to points on the surface of the object produced undesirable results. The hand was unable to grasp objects in many scenes because the entire hand was required to be close to the object, which requires a lot of clearance around the object. Furthermore, we found that fingertip contact was often sufficient for force-closure. Thus, we compute $F(\text{HPO}, O)$ for the set of surface points on the distal links of the fingers as well as for the entire set of surface points of the hand and choose the minimum.

To motivate our approximation of $S(\text{HPO}, O, E)$, recall that no part of the hand is allowed to collide with obstacles in the environment during the grasping process. To prevent the fingers from colliding with the object prematurely while approaching it, they must first be spread out from their target preshape. Once the HPO is reached, the grasp controller closes the fingers, making contact. If the fingers are to be opened to some degree and the position of the hand in the preshape is to be reached by a planner, it is clear that there must be free space around where the object is to be contacted. Thus it is more likely that the hand will be able to safely contact an object at a point surrounded by free space than it is to contact the object at a point close to other obstacles.

¹We use the OpenTSTool nearest-neighbor library which uses KD-Trees to efficiently find nearest neighbors along with their distances to the queries.

²We use an artificial discount factor α for directions, performing all nearest-neighbor queries on sets of directed points using the Euclidean metric on $(p, \alpha d)$, with $\alpha = 0.01$.

To compute the cost of contacting the object at each of the sample points, we use a procedure similar to that used to compute the CylCM. At each of the sample points of the object, we compute the Conical Clearance Map (ConCM), which uses the same procedure as the CylCM except with cones instead of cylinders. The height of the longest collision-free cone directed along the outward-facing surface normal at a point on the surface of the object becomes that point’s score (see Figure 3b). Again, ray-collision checking is used to compute this score efficiently. The angle of the cone(ϕ) is chosen experimentally.

The choice of a cone is motivated by the grasp controller. Imagine fingers curling in toward a particular contact point from many HPOs. As they curl in, they will arc toward their final destinations and a set of arcs starting above the plane of a point (as defined by the surface normal) and terminating at that point can be enveloped by a cone. The larger the cone, the more arcs are feasible for that contact point and thus the higher the probability that a grasp will be able to contact the object at the given point without colliding with obstacles.

Once the ConCM score is computed for every sample on the surface of the object, the scores are thresholded, giving points lower than the threshold a cost of 1 and points higher than the threshold a cost of 0. We term these costs as *point safety costs*. The threshold(β) regulates how much free space is desired around a contact point. Note that point safety costs are computed once per *scene* (as opposed to once per HPO).

We reuse the $F(\text{HPO}, O)$ nearest-neighbors query and sum the point safety costs of the samples in O_{dp} nearer than a distance γ to those in P_{dp} . $S(\text{HPO}, O, E)$ is set to this sum. γ defines how far apart a preshape point and a hand point can be while still being considered matched.

C. Optimization

Once we have generated an initial seed of HPOs, we need to decide how to use it to generate grasps for validation. One approach is to generate a large number of initial samples, determine their cost using the cost function described above, and pass some number of the top HPOs on to the validation step. However, most of the HPOs in the initial sampling will not be useful and an optimizer is needed to focus on and explore good regions of HPO space.

We use a nonlinear optimizer to refine the samples prior to the expensive validation step. We use a Genetic Algorithm(GA), which starts with a small initial sample as the seed population and runs until convergence. The top HPOs of the final population are then passed on to the validation step.

Specifically, for each generation of the GA, the top 50% of individuals in the population are selected as parents and the rest are discarded. The GA uses two operators, crossover and mutation, to generate new individuals from pairs of parents. For crossover, two random parents are combined to create two children. The first child takes the HPO_o of the first parent and the HPO_p of the second parent and vice versa for the second child. The children are then mutated by randomly perturbing individual values in HPO_o and HPO_p . Each value has a 25%

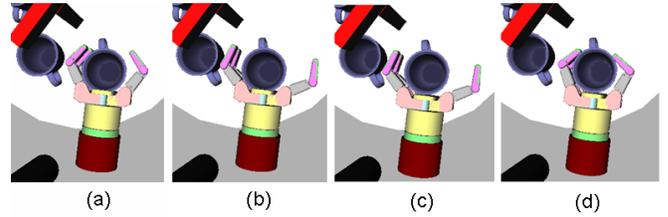


Fig. 4. Grasp refinement process for an example grasp. (a) An example grasp, as passed to the validation step. Note the interpenetration of the palm. (b) First, the fingers are uncurled until they reach collision or a joint limit and then curled until they are halfway between their starting position and the obstacle with which they collided. (c) If the hand is not in collision with the object at this step, this step is skipped. Otherwise, the hand is moved backward along the line defined by HPO_o until the hand is no longer in collision with the object and then moved forward slightly so that the hand is barely colliding with the object. This is done mainly to preserve palm contact with minimal interpenetration. (d) The fingers are curled in until each finger collides or joint limit.

chance of being perturbed, the magnitude of the perturbations is uniformly random between ± 0.1 for values in HPO_o and $\pm 3\text{cm}$. for values in HPO_p . To preserve quaternion validity, HPO_o is re-normalized after mutation. Once generated, children are added to the population and their costs are evaluated. This process iterates until the cost of the best individual does not change significantly for four generations.

The goal of both of the optimizer is to find a set of HPOs which move the hand into acceptable neighborhoods, *i.e.*, where the points of the preshape are close to a set of points on the object with low contact-safety cost and low fit cost. To refine the HPOs further, before being validated an HPO is “snapped” into place by aligning the preshape points with their nearest-neighbors on the object. To do this, we use the first technique discussed in [18]. This technique uses Singular Value Decomposition (SVD) to find the least-squares-best transform matrix to align two sets of points. This transform matrix is then converted into an HPO and passed to the validation step.

V. VALIDATION

There are two parts to the validation of grasps returned by the optimizer: grasp refinement and force-closure testing.

Because the hand, when placed at the HPO of the grasp with the corresponding preshape, may be interpenetrating with the object, we must determine where the fingers would actually collide with the object when running the grasping controller before evaluating force closure and checking collision with environment obstacles. Interpenetration is dangerous, especially at the palm, because it can cause the collision checker to give spurious contacts which will disrupt the force-closure test. The refinement process is described in Figure 4.

A force-closure grasp is able to resist an arbitrary disturbance wrench. We implemented the state of the art technique presented in [19] to evaluate force-closure. The test takes as input a set of contact points and normals, a coefficient of friction (μ), and the number of segments in a linear approximation of the friction cone(ρ) and states if the grasp is in force-closure. The contribution of [19] was to formulate the test as a linear program, resulting in faster runtime.

A refined grasp that is collision-free and in force-closure is considered valid.

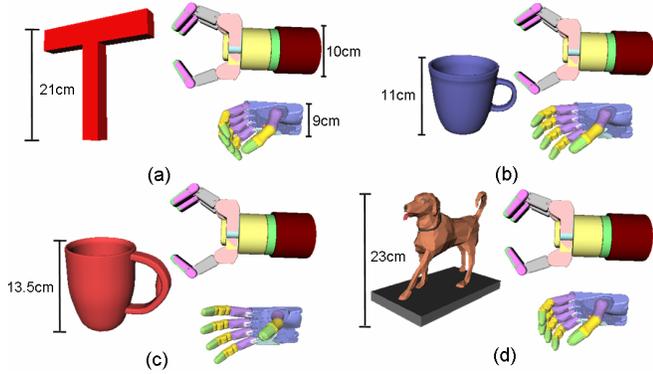


Fig. 5. Four objects used in the experiments and the preshapes used for these objects for both the Barrett and Shadow hands.

	1	2	3	4A	4B	4C	4D
Barrett Hand	96.3	97.5	83.3	84.4	88.7	80.3	54.7
Shadow Hand	82.9	95.0	96.3	91.9	94.0	90.0	43.5

TABLE I: PERCENT SUCCESS IN TEST SCENES

VI. RESULTS

We tested our algorithm on two types of hands: a three-fingered 7DOF Barrett hand and an anthropomorphic 22DOF Shadow hand. Figure 5 shows the four test objects and their respective preshapes. The objects chosen are meant to represent various levels and types of difficulties for grasping. Object A (the red T) is larger than both hands and contains large concavities, however its surface geometry is very regular and simple. Object B (the blue mug) has smaller concavities but its geometry is significantly more complicated. Object C (the red mug) is similar to Object B, but it is significantly larger, making it more difficult to grasp in tight spaces. Object D (the dog statue) is the most difficult because it contains sizable concavities and its surface geometry is very erratic. The parameter values used in all experiments were: $\phi = 45^\circ$, $\beta = 5\text{cm}$, $\gamma = 2\text{cm}$, $\zeta = 0.02$, $\mu = 0.75$, and $\rho = 8$.

A. Cost Function Evaluation

We compare the overall cost assigned by the cost function to the probability of success, see Figure 6. To do this, we generated 20,000 grasps for each of the three objects in the scene shown in Figure 6. Each grasp is validated, and the comparison between scores given by the cost function and probability of success is shown. The trend in each graph is clear, the lower the cost of a grasp, the more probable it is to succeed, thus the cost corresponds well to probability of success for the objects tested in this scene.

B. Trials in Simulated Scenes

We tested our algorithm on the test objects in several representative scenes. The algorithm was run 30 times in each scene with an initial sampling of 160 HPOs and the top 10% of HPOs in each final population were validated. The percent success of the generated grasps is shown in Table I. Examples of successful grasps in the test scenes are shown in Figure 7.

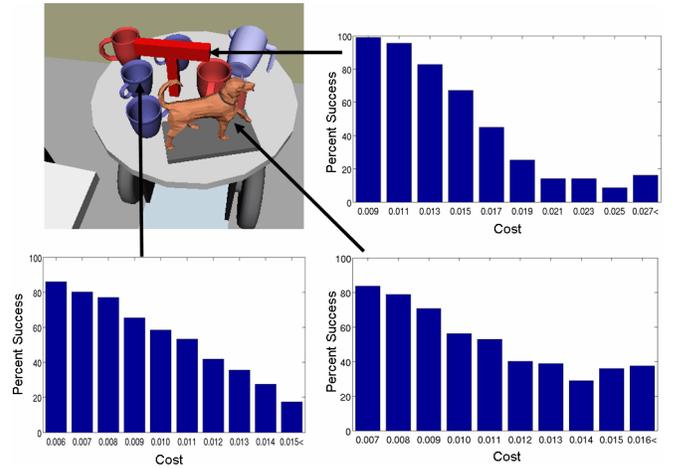


Fig. 6. Comparison of scores to percent success for the objects in the scene shown when using the Barrett Hand.

	CylCM	ConCM	Cost Fn	Total	Validation (per gras)
Barrett Hand					
Object A	0.164	1.15	0.994	2.31	0.42
Object B	0.475	1.90	1.01	3.38	0.48
Object C	0.552	2.79	1.13	4.47	0.50
Object D	0.413	3.84	1.31	5.57	0.42
Shadow Hand					
Object A	0.169	1.21	1.28	2.67	0.78
Object B	0.475	1.97	1.47	3.91	1.20
Object C	0.576	2.88	1.52	4.98	1.22
Object D	0.428	4.03	1.72	6.18	0.88

TABLE II: AVERAGE RUN TIMES(S)

Scenes 4A, 4B, 4C, and 4D were randomly generated by placing the object to be grasped at the origin and dispersing obstacles (2 blue mugs, 2 red mugs, 2 big boxes, a pitcher, a ketchup bottle, and a dog statue) around it. Obstacles were placed around the object at random poses within a cube of 50cm . No collisions were allowed. For each test object we generated 100 random scenes and ran the algorithm 30 times in each scene. The percent success in Table I is averaged over the scenes. As expected, object D is the most difficult to grasp with this algorithm, receiving the lowest success rate. Objects A and B turned out to have very similar success rates, illustrating that algorithm compensates for different geometries very well.

C. Run Times

Table II shows the run times of various components of our algorithm averaged over 30 runs in each of the 100 randomly-generated scenes. These results were obtained on an Intel Dual-Core 2.4GHz PC with 4 GB of RAM. CylCM and ConCM values are average times needed to compute clearance maps. Cost Fn values are the average sum of all cost function evaluations per run. Total values are the sums of all previous values. Validation times for a single grasp averaged over all runs for each object in each scene are also shown.

Note that the time needed to construct the CylCM and the ConCM varies with the surface area of the object because points are sampled on the surface at the same resolution, thus larger objects will have more surface points.

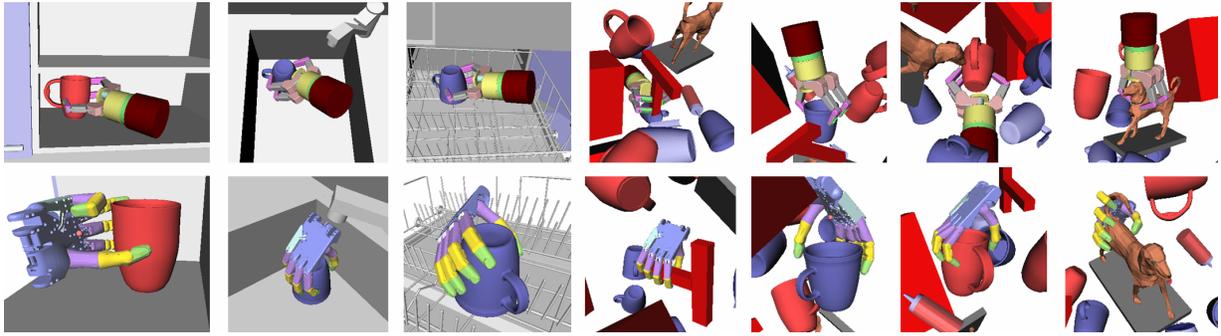


Fig. 7. The Barrett and Shadow hands successfully grasping objects in scenes 1, 2, 3, 4A, 4B, 4C, and 4D, respectively from left to right.



Fig. 8. The robot grasping blue and red mugs in various scenes. In the three right-most pictures an artificial obstacle was added above the mugs to increase difficulty.

D. Experiments on Robot

We implemented our algorithm on a robot consisting of a 7DOF WAM arm and a Barrett Hand. The task for the robot was to pick up two different kinds of mugs arranged arbitrarily on a table. The system uses an overhead camera to identify the mugs and obtain their 3D transformations. The grasp set generated by our algorithm is passed to a planner that uses inverse kinematics and BiDirectional RRTs to plan an arm trajectory to the HPO. Once the arm is in position, the fingers are curled in, squeezing the mug. The mug is then lifted up by $3cm$. Several snapshots of objects B and C being lifted in several scenes are shown in Figure 8.

To demonstrate that our algorithm can work in more confined spaces than our vision system can handle, we placed an artificial obstacle above the mug to prevent it from being grasped from the top in the three right-most scenes in Figure 8. No artificial obstacles were used in the two left-most scenes.

VII. CONCLUSION

We have presented an efficient and general algorithm for grasp synthesis in cluttered environments. We have demonstrated the ability of our algorithm to consistently generate force-closure grasps for a wide range of objects and scenes in a few seconds. We have also demonstrated the generality of our algorithm across manipulators of varying complexity and structure. Furthermore, we have demonstrated an implementation of the algorithm on a physical robotic system.

REFERENCES

- [1] A. Morales, T. Asfour, P. Azad, S. Knoop, and R. Dillmann, "Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands," in *IROS*, 2006.
- [2] R. Platt, "Learning and generalizing control based grasping and manipulation skills," Ph.D. dissertation, University of Massachusetts at Amherst, Department of Computer Science, 2006.
- [3] A. Edsinger-Gonzalez and J. Webber, "Domo: a force sensing humanoid robot for manipulation research," in *Humanoids*, 2004.
- [4] K. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, and H. Hirukawa, "A humanoid robot carrying a heavy object," in *ICRA*, 2005.
- [5] T. Wimbock, C. Ott, Hirzinger, and Gerd, "Impedance behaviors for two-handed manipulation: Design and experiments," in *ICRA*, 2007.
- [6] T. Martin, R. Ambrose, M. Diftler, R. Platt, and M. Butzer, "Tactile gloves for autonomous grasping with the nasa/darpa robonaut," in *ICRA*, 2004.
- [7] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, "Robotic grasping of novel objects," in *NIPS*, 2007.
- [8] D. Katz, E. Horrell, Y. Yang, B. Burns, T. Buckley, A. Grishkan, V. Zhytkovskyy, O. Brock, and E. Learned-Miller, "The umass mobile manipulator uman: An experimental platform for autonomous mobile manipulation," in *IEEE Work. on Manip. for Human Environments*, 2006.
- [9] Z. Li and S. Sastry, "Task oriented optimal grasping by multifingered robot hands," *ITRA*, 1988.
- [10] X. Zhu and J. Wang, "Synthesis of force-closure grasps on 3-d objects based on the q distance," *ITRA*, 2003.
- [11] K. Hsiao and T. Lozano-Perez, "Imitation learning of whole-body grasps," in *RSS Wkshop: Manipulation for Human Environments*, 2006.
- [12] A. Miller, S. Knoop, P. Allen, and H. Christensen, "Automatic grasp planning using shape primitives," in *ICRA*, 2003, pp. 1824–1829.
- [13] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Humanoids*, 2007.
- [14] Y. Li, J. Fu, and N. Pollard, "Data driven grasp synthesis using shape matching and task-based pruning," in *IEEE Trans. Vis. and Comp. Graph.*, 2007.
- [15] E. Oztop and M. Arbib, "Schema design and implementation of the grasp-related mirror neuron system," *Biological Cybernetics*, vol. 87, pp. 116–140, 2002.
- [16] S. Stansfield, "Robotic grasping of unknown objects: A knowledge-based approach," *IJRR*, vol. 10, pp. 314–326, 1991.
- [17] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," in *ICRA*, 2000.
- [18] D. Eggert, A. Lorusso, and R. Fisher, "Estimating 3-d rigid body transformations: a comparison of four major algorithms," *Machine Vision and Applications*, vol. 9, pp. 272–290, 1997.
- [19] Q. Zheng and W.-H. Qian, "An enhanced ray-shooting approach to force-closure problems," *Jour. Manufac. Sci. and Eng.*, vol. 128, no. 4, pp. 960–968, 2006.