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. We also propose an extension of the algorithm for two-handed grasps and demonstrate it on the HRP3 hands. 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

A prime application for humanoid robots is to perform a variety of manipulation tasks in the home. However, household environments are often 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 chose 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 or hands 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.

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 and pose of the object being grasped so its model and pose should be fairly accurate.

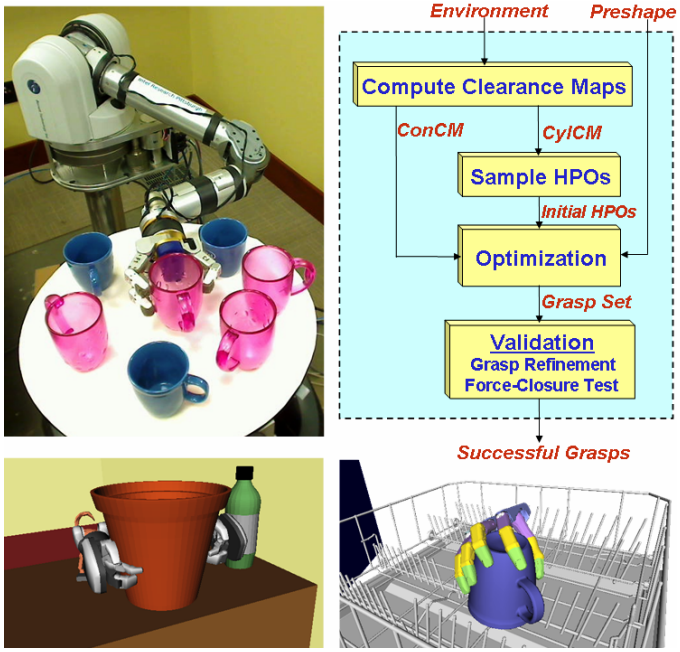

Fig. 1. Outline of the algorithm and implementation on the Barrett, Shadow, and HRP3 hands.

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], El-E [4], HRP2 [5], Justin [6], the NASA Robonaut [7], STAIR [8], and UMan [9]. There is renewed interest in moving away from tele-operation, manually-scripted grasps, and simple scenes, to autonomous 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 [10, 11]. However, the applicability of these techniques to dexterous hands with complex kinematics and geometry is an open problem. Recently, [8, 12] 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 environ-

ments. Grasps are often generated as if the object is alone in the environment and then collision-checked, as in [13]. 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 [14], we proposed an algorithm where a large number of force-closure grasps for an object were generated offline. When placed in a scene, the grasps were ranked online 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 [15], 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 $\text{HPO}_p \in \mathbb{R}^3$, and orientation $\text{HPO}_o \in \mathbb{H}$ 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.

The act of grasping is completely described by the preshape and HPO of the hand, and the action of the grasping controller. We define a grasp to be *valid* if

1) The hand does not collide with the scene during execution of the grasping controller.
2) The ensuing grasp is in force-closure.

III. Preshapes

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 [16] or through analysis of the affordances of an object [17]. Also, rule-based [18] and heuristic methods [13] 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 [16] combined with a manual selection of preshapes for more difficult objects.

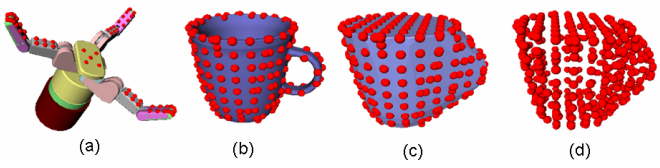

Fig. 2. (a) Sampling on the contacting surface of the Barrett Hand in a certain preshape. (b) The sampled surface of a mug. (c) The sampled surface of that mug's convex hull. (d) The combined mug samples.

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 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 the computation of a set of valid HPOs given a preshape and the environment geometry. We define a HPO to be valid if it produces a valid grasp. The computation necessary to validate an HPO consists of running the grasp controller, collision checking with the scene at every time 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[19], 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.

Our algorithm quickly finds likely HPOs which are then passed on to the expensive validation step. The algorithm has two main parts. First, we sample promising HPOs using information about the object's local environment and seed an optimizer with this initial sampling. Second, the optimizer uses a novel cost function which attempts to *predict* the validity of a given HPO, thereby bypassing the expensive validation step. It is only the optimized HPOs that are then validated. Fig.1 provides an outline of the algorithm.

A. Generating an Initial Seed

We use the Cylindrical Clearance Map(CylCM) to generate good initial seeds for the optimizer. The 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 pre-computed samples of the surface of the object and its convex hull (Fig.2). We denote the set of sample points by O_{dp}. The CylCM at each point in O_{dp} is defined as the length of the longest cylinder that can be placed at the point and oriented along the outward surface normal, without colliding with the scene(Fig.3a).

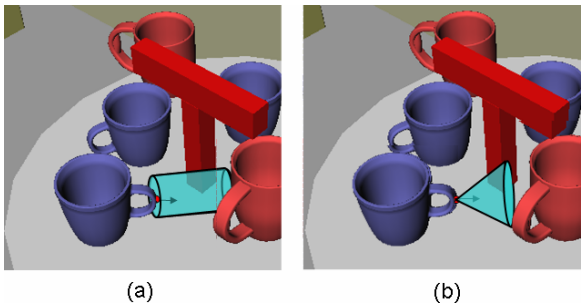

Fig. 3. Depiction of computing clearance for one point on an object using cones and cylinders. The cone/cylinder is oriented along the outward-facing surface normal of a point on the surface of the object. The length of the longest cone/cylinder that is collision-free is the clearance score assigned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

The radius of the cylinder is the radius of the bounding cylinder of 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. The fixed part is guaranteed to be collision free if the CylCM score is larger than the length of the bounding cylinder. Note that this is a conservative estimate.

Using the above check, we extract those sample points $\boldsymbol{p}$ that are guaranteed to be collision-free for the fixed part. Given a certain number of desired seeds, N, we sample points from $\boldsymbol{p}$ proportional to their CylCM score. Eqn.1 is used to generate HPO_p from these points.

$$\mathrm{HPO}_p = \boldsymbol{p} + hl_{\max}\hat{\boldsymbol{n}} \tag{1}$$

where h is chosen uniformly from $[0, 1]$, $l_{\max}$ is the length of the fingers when they are fully extended, and $\hat{\boldsymbol{n}}$ is the outward-facing surface normal at $\boldsymbol{p}$. At each $\boldsymbol{p}$, 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 $-\hat{\boldsymbol{n}}$ and add a random rotation about $\hat{\boldsymbol{n}}$ 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 valid. 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 combine three individual cost functions to produce the cost of an HPO for object O in environment E:

1) *Approximate Collision* - $X(\mathrm{HPO}, E)$ - measures the likelihood of the fixed part being in collision.
2) *Fit Cost* - $F(\mathrm{HPO}, O)$ - measures the error of the fit between the preshape and the object. The larger the error, the larger the likelihood of the grasp not being in force-closure.

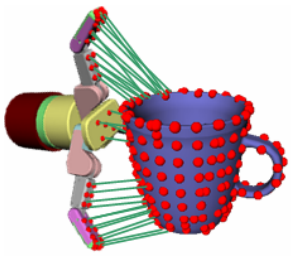

Fig. 4. Matching preshape points to their nearest neighbors on the object. The green lines represent nearest neighbor pairings and the average length of these lines is $F(\mathrm{HPO}, O)$. Samples on the convex hull of the mug are not shown.

3) *Contact Safety Cost* - $S(\mathrm{HPO}, O, E)$ - measures the likelihood of collision when the grasp controller curls in the fingers.

The combined cost function $C(\mathrm{HPO}, O, E)$ is given by

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

where ζ is the tradeoff between fit and contact safety costs.

To compute $X(\mathrm{HPO}, E)$, we place the bounding cylinder for the fixed part of the hand at the HPO and check collision with environment obstacles. If the approximating cylinder is not in collision $X(\mathrm{HPO}, E) = 1$, otherwise $X(\mathrm{HPO}, E) = 0$ making $C(\mathrm{HPO}, O, E) = \infty$. We do not check collision with O, allowing the optimizer to utilize the fixed part of the hand for the grasp.

To compute $F(\mathrm{HPO}, O)$, we transform the directed points on the surface of the hand to the HPO. We denote the transformed points as P_{dp}. We then perform a nearest-neighbors[1] query to find directed points in O_{dp} that are closest[2] to the directed points in P_{dp}. The distances to the closest points are averaged as $F(\mathrm{HPO}, O)$, which is a measure of how well the preshape fits the object at that HPO, see Figure 4. If the preshape and HPO are not compatible, the grasp controller is unlikely to end up in a configuration similar to the preshape (as it curls in the fingers until all have collided), thus making it hard to predict if the ensuing grasp will be in force-closure.

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 $F(\mathrm{HPO}, O)$ 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 $S(\mathrm{HPO}, O, E)$, recall that no part of the hand is allowed to collide with obstacles in the environment during the grasping process. To prevent

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

[2] We 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 $\alpha = 0.01$

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 by 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.

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 $F(\mathrm{HPO}, O)$ 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}. $S(\mathrm{HPO}, O, E)$ 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.

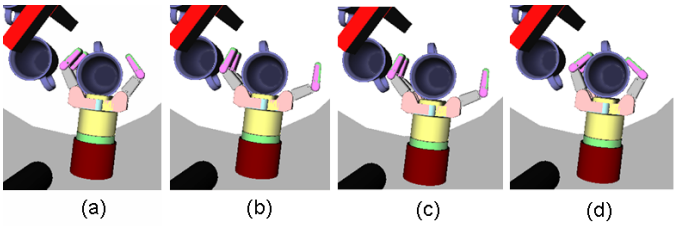

Fig. 5. Grasp refinement process for an example grasp. (a) An example grasp, as passed to the validation step. Note the interpentration of the palm. (b) First, the fingers are uncurled until they reach collision or a joint limit and then curled until they are halfway between their starting position and 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.

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% chance of being perturbed, the magnitude of the perturbations is uniformly random between ± 0.1 for values in HPO_o and $\pm$3cm. 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 [20]. 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

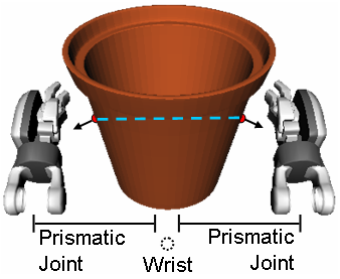

Fig. 6. Depiction of two-handed grasping extension. A pair of sampled directed points (red) is shown along with the distance between them (blue). Virtual prismatic joints and the virtual wrist are shown for the HRP3 hands.

refinement process is described in Figure 5.

A force-closure grasp is able to resist an arbitrary disturbance wrench. We implemented the state of the art technique presented in [21] 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 [21] 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.

VI. Extension to Two-Handed Grasping

Our algorithm can also be applied to two-handed grasps by treating the two hands as fingers connected by virtual joints to a virtual wrist (see Figure 6). By arranging the hands in this way, we turn the problem of two-handed grasping into the problem of grasping with a large gripper that has no wrist, which can be handled by our algorithm. The fingers of both hands are locked into an initial position and preshapes are generated by placing the hands opposite to each other and spreading the hands apart. The grasping controller moves the hands toward each other along the virtual prismatic joints. Once both hands make contact their fingers are curled in as usual.

While it is possible to simply connect the hands as above and run the algorithm, far better results can be achieved with the following extensions.

First, since there is no wrist for the opposing hands, the strategy for picking the initial seed described in Section IV-A is no longer applicable. For two-handed grasping, instead of sampling directly from the CylCM for the wrist position, we sample from *pairs* of points in the CylCM[3] to find suitable positions for the opposing hands. In order to be considered for sampling, the points' normals must be roughly opposing each other and they must be separated by roughly the width of the preshape. Once all pairs of such points are identified, we sample from this set proportional to the minimum of the CylCM scores for each pair of points. Once the pairs are sampled, we generate the seed HPOs by aligning the hands with the line between each pair of points and rotating the hands randomly about that line. Pairing hands (left or right) with points is also done randomly for each HPO.

[3]constructed using the approximating cylinder of one of the wrists

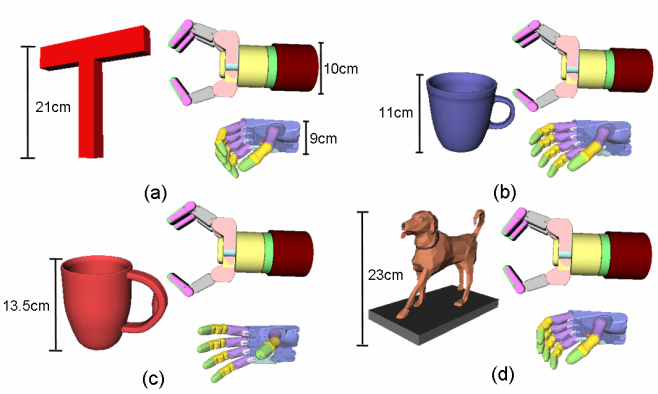

Fig. 7. Four objects used in the experiments and the preshapes used for these objects for both the Barrett and Shadow hands.

Second, the approximating cylinders for both wrists are used in the optimization. They are spread apart to match the preshape width and collision is checked with both of them when evaluating $X(\text{HPO}, E)$.

Third, fingertip grasps are not considered in the optimization to ensure that the entire hand touches the surface of the object being grasped if possible. This is done because holding large and possibly heavy objects with the fingertips alone may not be desirable because of torque limits on the finger motors.

VII. Results

We tested our algorithm on two types of hands: a three-fingered 7DOF Barrett hand and an anthropomorphic 22DOF Shadow hand. Figure 7 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: $\phi = 45°$, $\beta = 5$cm, $\gamma = 2$cm, $\zeta = 0.02$, $\mu = 0.75$, and $\rho = 8$.

A. Cost Function Evaluation

To gauge the effectiveness of the contact safety cost portion of the cost function, we compare the contact points of successful grasps to points classified as safe or unsafe by our cost function. The results are displayed in Figure 8. This figure shows that successful grasps rarely make contact near points deemed to be unsafe. Thus it is correct to assign higher costs to those points because contacts near them are rarely a part of successful grasps.

We also compare the overall cost assigned by the cost function to the probability of success, see Figure 9. To do

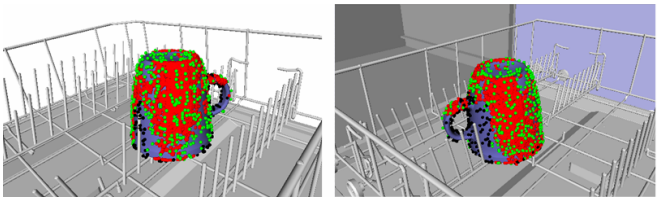

Fig. 8. Comparison of contact safety costs with contact points from 1000 successful grasps (generated using random sampling). The red points are the contact points of the grasps, the green points are surface samples of the object deemed to be safe, and the black points are surface samples deemed unsafe. Two views of the same example are shown.

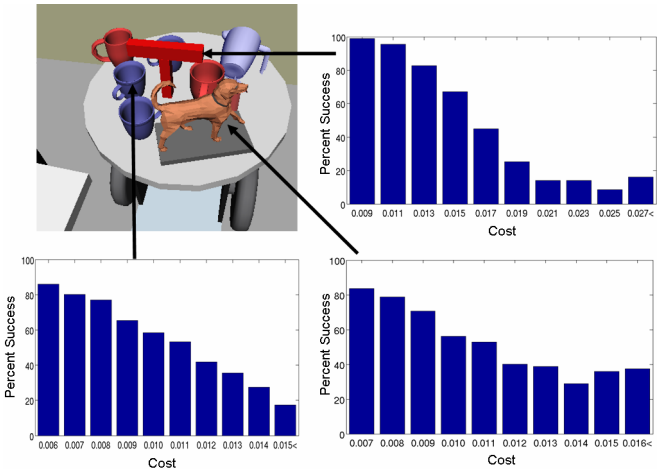

Fig. 9. Comparison of scores to percent success for the objects in the scene shown when using the Barrett Hand.

this, we generated 20,000 grasps for each of the three objects in the scene shown in Figure 9. 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. Optimization Comparison

To gauge the effectiveness of our optimization algorithm, we compare our GA to a Random Sampling method. In the Random Sampling method, we sample a much larger number of initial seed HPOs and rank them by the scoring function instead of running the GA. To compare these two methods, we again use the objects and scene shown in Figure 9. The GA method was run 63 times with an initial sampling size of 160 HPOs. The top 10% of HPOs in each of the final populations are validated, generating a total of 1008 HPOs along with their success or failure for each object. The Random Sampling method is run once, with a sampling size of 10,080 HPOs for each object. The top 10% of HPOs are validated, generating 1008 HPOs along with their success or failure for each object. The percent success (number of successful HPOs/1008) for each object is shown in Table I.

The GA method clearly outperforms Random Sampling because it is far more likely to generate successful grasps, even though it starts with a far smaller initial sampling. This occurs because the GA focuses its search on good solutions, generating new HPOs near low-cost HPOs, which are also likely to have a low cost. The GA also explores the space at the same time, so new HPOs farther from the initial sampling can be found. By contrast, the Random Sampling method is locked in to the initial seed and has no way to focus on good HPOs.

	Object A	Object B	Object C	Object D
Barrett Hand				
Genetic Algorithm	98.8	78.6	72.2	60.1
Random Sampling	73.2	31.9	60.0	39.3
Shadow Hand				
Genetic Algorithm	91.1	94.8	96.9	68.3
Random Sampling	81.3	66.4	81.4	70.8

TABLE 1

TABLE I: PERCENT SUCCESS FOR GA AND RANDOM SAMPLING

	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 II: PERCENT SUCCESS IN TEST SCENES

C. 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 10.

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 II 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.

D. Comparison to Previous Work

To benchmark our algorithm, we show the success rates of two previously-proposed algorithms for the Barrett Hand in Table III. The *Primitives* algorithm [13] works by first defining preshapes and approach directions to grasp primitive objects such as cylinders, boxes, and cones. An approximation of the geometry of the object to be grasped is then generated using these primitive shapes. The grasps corresponding to the approximating primitives are tested in the given scene using a grasp controller and force-closure test that is similar to ours. To generate success rates, we tested all grasps in the grasp set

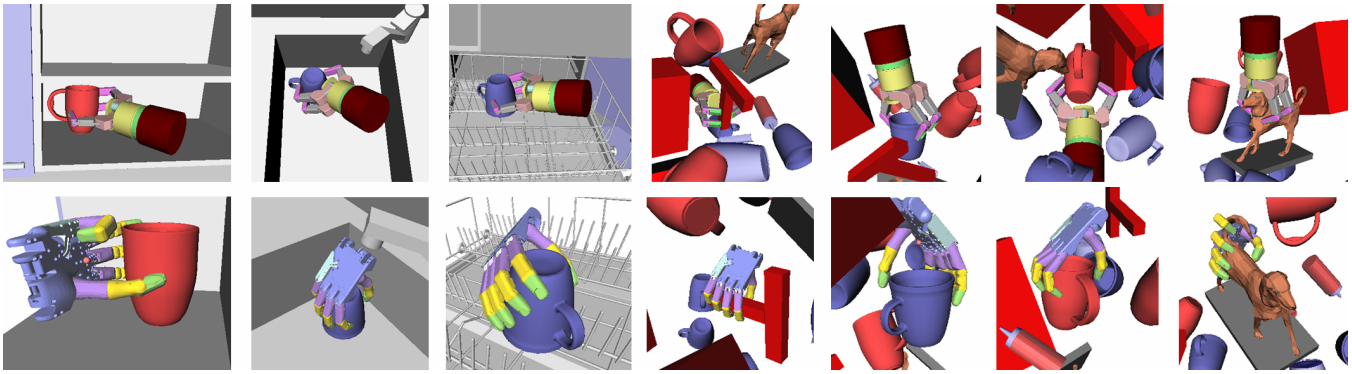

Fig. 10. The Barrett and Shadow hands successfully grasping objects in scenes 1, 2, 3, 4A, 4B, 4C, and 4D, respectively from left to right.

Fig. 11. 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.

	1	2	3	4A	4B	4C	4D
Proposed Algorithm	96.3	97.5	83.3	84.4	88.7	80.3	54.7
Primitives	0	0	0.95	12.8	12.2	12.4	10.2
Grasp Tables	0.68	0	2.33	44.9	51.7	48.3	53.2

TABLE III: COMPARISON OF PERCENT SUCCESS FOR BARRETT HAND

	CylCM	ConCM	Cost Fn	Total	Validation (per grasp)
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 IV: AVERAGE RUN TIMES(S)

(roughly 300 for each object) in the given scene and recorded the percent that were valid.

The *Grasp Tables* algorithm [14] is the one previously proposed by the authors. This algorithm generates a large table (roughly 600) of force-closure grasps offline for an object to be grasped by sampling the preshape and pose parameters of the hand and running the grasp controller. When the object is placed a new scene, the CylCM is constructed as in our proposed algorithm. The grasps in the table are then sorted by their CylCM scores and the top sixteen[4] are tested for collision. No force-closure test is necessary since all grasps in the table are known to be in force-closure.

Because both algorithms are determininstic they were run once in each test scene including each of the four-hundred randomly-generated scenes. We found that most of the cases where these methods failed were due to wrist collisions or finger collisions when executing the grasping controller.

E. Run Times

Table IV 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. The cost evaluation times also increase with the number of surface points because the Nearest-Neighbor query needs to evaluated more points, but this increase is moderate.

F. Two-Handed Grasping

We also ran the two-handed extension of our algorithm for the left and right hands of the HRP3. Each hand has 6 degrees of freedom in the fingers. The algorithm was run 30 times

[4]this is the same number of grasps validated by our proposed algorithm

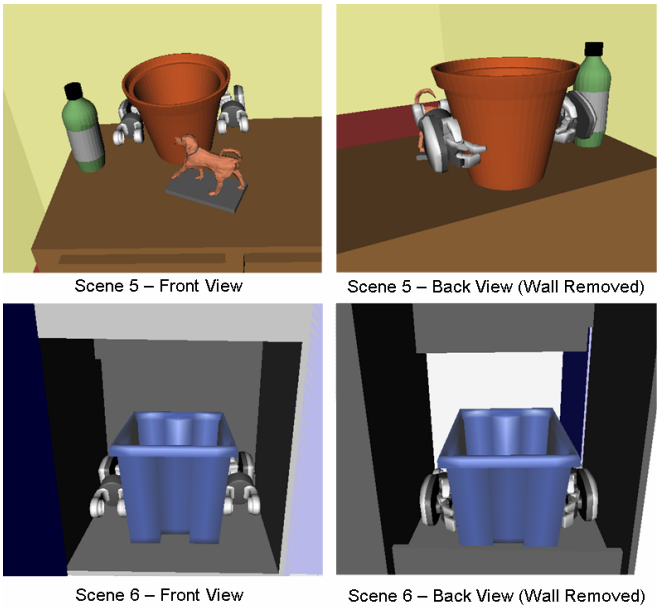

Fig. 12. Two test scenes used to evaluate the two-handed grasping extension. Two grasps generated by our algorithm are shown.

in the scenes shown in Figure 12. The success rates were 94.8% and 95.0% on average for scenes 5 and 6, respectively. Since the objects were quite large, computing the CylCM and ConCM took an average of 22.5 seconds. This run time can be improved by using a lower-resolution sampling of the object, however doing so may cause greater inaccuracy in fitting preshapes.

G. 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 11.

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 11. No artificial obstacles were used in the two left-most scenes. Several experiments are shown in our our video at:

`http://www.cs.cmu.edu/~dberenso/intelgrasping.mp4`

VIII. 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.

IX. Acknowledgements

We gratefully acknowledge the Digital Humanoid Research Center (AIST) for the use of the HRP3 hand models as well as the students and faculty involved with the ARMAR humanoid platform at the University of Karlsruhe for the use of the kitchen environment models. We would also like to thank Dave Ferguson for his advice and valuable discussions. This research is supported by the NSF Quality of Life Technology (QOLT) Center and Intel Research Pittsburgh.

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] H. Nguyen, C. Anderson, A. Trevor, A. Jain, Z. Xu, and C. C. Kemp., "El-e: An assistive robot that fetches objects from flat surfaces," in *"Robotic Helpers: User Interaction, Interfaces and Companions in Assistive and Therapy Robotics" an HRI Workshop*, 2008.
[5] 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.
[6] T. Wimbock, C. Ott, Hirzinger, and Gerd, "Impedance behaviors for two-handed manipulation: Design and experiments," in *ICRA*, 2007.
[7] T. Martin, R. Ambrose, M. Diftler, R. Platt, and M. Butzer, "Tactile gloves for autonomous grasping with the nasa/darpa robonaut," in *ICRA*, 2004.
[8] A. Saxena, J. Driemeyer, J. Kearns, and A. Ng, "Robotic grasping of novel objects," in *NIPS*, 2007.
[9] D. Katz, E. Horrell, Y. Yang, B. Burns, T. Buckley, A. Grishkan, V. Zhylkovskyy, 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.
[10] Z. Li and S. Sastry, "Task oriented optimal grasping by multifingered robot hands," *ITRA*, 1988.
[11] X. Zhu and J. Wang, "Synthesis of force-closure grasps on 3-d objects based on the q distance," *ITRA*, 2003.
[12] K. Hsiao and T. Lozano-Perez, "Imitation learning of whole-body grasps," in *RSS Wkshop: Manipulation for Human Environments*, 2006.
[13] A. Miller, S. Knoop, P. Allen, and H. Christensen, "Automatic grasp planning using shape primitives," in *ICRA*, 2003, pp. 1824–1829.
[14] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *Humanoids*, 2007.
[15] D. Berenson and S. Srinivasa, "Grasp synthesis in cluttered environments for dexterous hands," in *RSS Workshop on Robot Manipulation*, June 2008.
[16] 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.
[17] E. Oztop and M. Arbib, "Schema design and implementation of the grasp-related mirror neuron system," *Biological Cybernetics*, vol. 87, pp. 116–140, 2002.
[18] S. Stansfield, "Robotic grasping of unknown objects: A knowledge-based approach," *IJRR*, vol. 10, pp. 314–326, 1991.
[19] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," in *ICRA*, 2000.
[20] 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.
[21] 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.