Collaborative Grasp Planning with Multiple Object Representations

Peter Brook[†*], Matei Ciocarlie[†] and Kaijen Hsiao[†]

Abstract— **Grasp planning based on perceived sensor data of an object can be performed in different ways, depending on the chosen semantic interpretation of the sensed data. For example, if the object can be recognized and a complete 3D model is available, a different planning tool can be selected compared to the situation in which only the raw sensed data, such as a single point cloud, is available. Instead of choosing between these options, we present a framework that combines them, aiming to find consensus on how the object should be grasped by using the information from each object representation according to their confidence levels. We show that this method is robust to common errors in perception, such as incorrect object recognition, while also taking into account potential grasp execution errors due to imperfect robot calibration. We illustrate this method on the PR2 robot by grasping objects common in human environments.**

I. Introduction and Related Work

Robots operating in human settings must often make sense of an uncertain environment. In particular, grasping and manipulation tasks can be affected by both perception uncertainty (such as incomplete data due to occlusions or incorrect object segmentation and recognition) and execution errors (such as imperfect trajectory following during arm motion). Grasp planning algorithms aim to increase reliability in unknown scenarios by producing grasps able to resist a wide range of disturbances, for example, using well-established quality metrics based on either the Grasp or Task Wrench Space [10], [4]. However, this type of analysis does not handle potential errors such as the ones mentioned above, that affect the grasp planning and execution process itself.

The uncertainty inherent in unstructured environments also means that different perception algorithms can provide different interpretations of the same scene. For example, object recognition algorithms often return a list of possible results, each associated with a numerical measure of confidence, rather than a single result with 100% certainty. Different grasp planning algorithms, using different data representations, will have their own view of how an object can be grasped.

By combining this data in a common framework, we can take advantage of multiple sources of information, and produce algorithms better suited for handling uncertainty. In this paper, we use the results from multiple grasp planning approaches, running on different types of input data, in a way that is agnostic to the inner workings of each algorithm. In order to allow this exchange of information between algorithms that are intrinsically different, we use experimental data to map raw results from each algorithm to success probabilities, which allows us to combine different components in a single framework. We also propose a method of using pre-computed grasp information to replace expensive run-time computations when estimating how likely a grasp is to succeed, applicable to planners that can run off-line on a database on known models.

In addition to the uncertainty associated with sensor data, grasp execution is also affected by potential calibration errors between the sensors and the end-effector. This type of error can be mitigated, for example, through extensive calibration or controller tuning, but rarely eliminated altogether. We attempt to handle this case by extending our measure of confidence in the success of each grasp to also take into account the range of possible outcomes for the respective execution commands.

There is a long history of work that deals with the problem of planning robot motions and manipulation tasks under uncertainty, starting with preimage backchaining [18]. More recently, Berenson *et al.* [2] used Task Space Regions (TSRs) to generate manipulator trajectories that satisfy the requirements of a task despite robot pose uncertainty. Hsiao *et al.* [13] used a belief-based representation of object pose to plan actions to robustly execute specific grasps of known object shapes. Saxena *et al.* [23] used a probabilistic classifier to generate grasps and to predict their probability of success given features of an image and a point cloud of the object. Glover *et al.* used a probabilistic representation of 2-D shape to recognize and complete object outlines in an image for grasping. Finally, Balasubramanian *et al.* [1] examined how to improve the robustness of grasps using grasp measures derived from human-guided grasp demonstrations.

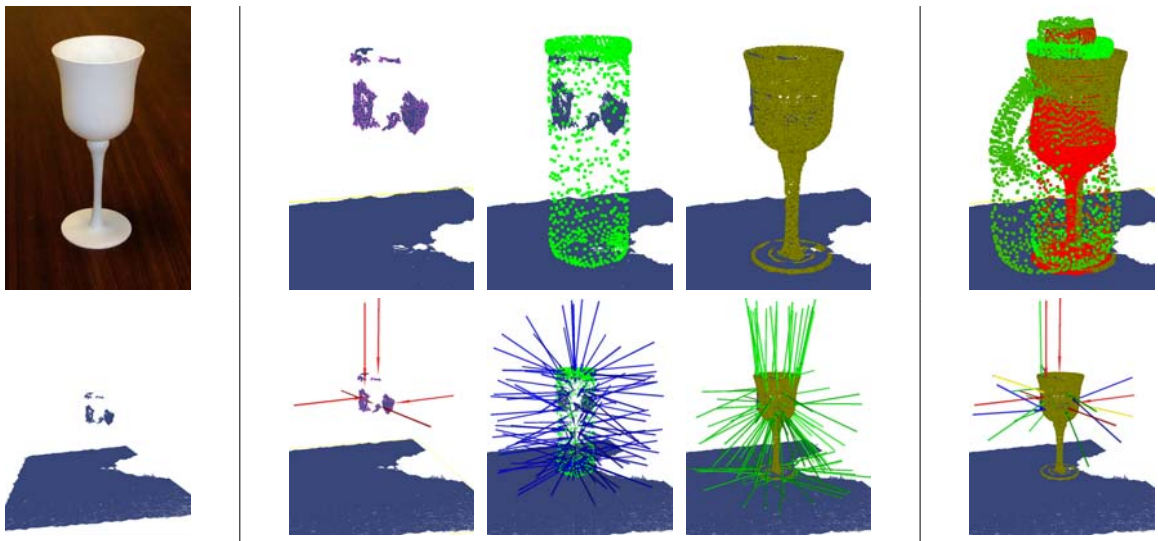
A database-driven grasp planning approach, including grasp evaluation across multiple objects of similar shapes, was recently discussed by Goldfeder *et al.* [11]. De Granville *et. al.* [8] also used mixtures of Gaussians over grasp position and orientation to represent functionally different grasp affordances, based on a database of human-demonstrated grasps; Detry *et al.* [9] represented grasp densities using a nonparametric kernel representation over grasp position and orientation, refined through robot grasp experiments.

II. Object Representations and Collaborative Grasp Planners

Consider the problem of a robot attempting to execute a grasp based on a perceived sensor image of the target object. In this study, we use a stereo camera equipped with a textured light projector to provide point clouds of the environment, although the framework that we will present is naturally extendable to other types of sensors. We also assume here

[†]Willow Garage Inc., Menlo Park, CA. Email: {pbrook, matei, hsiao}@willowgarage.com

[*]University of Washington, Seattle, WA.



(a) (b) (c) (d) (e)

Fig. 1: Challenges in sensing for manipulation. **(a)** An object to be grasped, as well the recorded point cloud. **(b)-(d)** Different object representations for grasping, and their corresponding planned grasps. These include a planner based only on the point cluster (b), and two results of the recognition algorithm, one incorrect (c) and one correct (d). **(e)** All representations superimposed, along with a set of grasps combining information from all of them.

that the segmentation problem is solved (*i.e.* the object is separated from the background). As in previous work [12], we use planar segmentation and Euclidian clustering to separate the objects from a planar support surface, as well as each other. Fig. 1(a) shows an example of the raw perceived data of an object, as well as the segmentation result. Notice that due to (self-) occlusions, noise, and other imperfections of the point cloud, the original object (in this case a wine glass) is barely recognizable to a human operator. This point cloud is the starting point of the grasp planning algorithm.

A. Object Representations

One possible method for performing the grasp is to attempt to recognize the object from a database of known models, and simply use a pre-computed grasp for the recognized model. This approach is discussed in [11]. In this study, we use the implementation presented in [7], relying on the model database described in [6]. This method, which we refer to as *database-driven planning*, has the advantage of reasoning about the complete object shape and choosing grasps accordingly. In addition, planning can be performed off-line on all models in the database, saving significant computational cost at run-time. However, database-driven planning only works on objects that can at least be reasonably approximated by models in the database, and is critically affected by the quality of the model recognition algorithm.

To illustrate the importance of correct object recognition, Fig. 1(c)(d) shows part of the result of our algorithm applied to the point cloud of Fig. 1(a). Note that the first result (a tennis ball can) is incorrect, even though it matches the observed part of the glass quite well. The correct model, and other similar objects, can be found further down in the list of recognition results. Intuitively, it seems that the results of the recognition algorithm contain at least some useful data, but a naive planning algorithm using only the first result (the tennis ball can) has a significant chance of failure.

If object recognition can not be fully trusted, a natural alternative is to select a grasp based strictly on the segmented point cloud, without attempting to identify the object, as shown in Fig. 1(b). Note that, in the rest of the paper, we will refer to the segmented point cloud of an object as a *point cluster*, or simply a *cluster*, in order to distinguish it from complete point clouds of an entire scene. Examples of *cluster-based planning* approaches can be found in [15] and also in [12]; the latter implementation is also used in this study. This approach has the advantage of being applicable in a wide range of situations, as it only requires segmentation of the target object (as described above). However, it is naturally limited by only having a partial view of the object to work with.

Generalizing from the two concrete cases presented above, we notice that different grasp planners often use different representations of the target object. For database-driven grasping, any model returned by a recognition algorithm can be considered a representation of the underlying sensor data; for cluster-based planning, the representation is the point cluster itself. Based on how well the given representation fits the object, and also on the intrinsic planning algorithm, the resulting grasps will be more or less likely to succeed when executed on the real object. In our formulation, each object representation is associated with a grasp planning algorithm that uses that particular method of interpreting the data.

In this study, we rely on the two representations discussed above for our implementations and results. We also note that

other object representations for grasping have been proposed in the literature. One example is to represent the object data using primitive shapes (such as boxes, spheres or cylinders) as in [14], [22], and attempt to plan grasps based on the primitives [19], [14] . Representations can also use different sensors, as in [23] where grasping is based directly on a monocular image of a scene. While these methods are not (yet) used in our framework, some of them are natural extensions and their inclusion is the subject of future work.

B. Multi-planner Framework

For a given sensor view of an object, we refer to our set of possible representations as $\boldsymbol{R}$, with an individual representation denoted as $r \in \boldsymbol{R}$. Furthermore, for any representation we define a *confidence function* $C(r|o)$, which encodes how confident the representation r is in its ability to predict whether grasps will succeed, based on the observed data o (we will expand on this topic in later sections).

We refer to a grasp that can be executed by the robot as g. In our study, we apply this formulation to a robot equipped with a simple parallel gripper. In this case $g \in \mathcal{R}^6$, simply encodes the pose (position and orientation) of the gripper relative to the target object. As in the case of object recognition, we define an estimate on the confidence that we will succeed at grasping (denoted s), given a particular grasp g and observed data o, as:

$$C(s|g,o) = \sum_{r \in \boldsymbol{R}} C(r|o)C(s|g,r) \tag{1}$$

Although our confidence values are not intended to accurately reflect real-life probabilities, they are values between 0 and 1 that behave in a similar manner to probabilistic estimates. Thus, to clarify the exposition, we use probabilistic notation, using for example $C(s|g,r)$ to denote the confidence in successful execution for grasp g given that the object representation r accurately represents the true geometry of the object (this quantity is independent of the data o, given the representation r).

The intuition behind this formulation is that a grasp is more likely to succeed in the real world if multiple object representations that are trying to explain the sensor data agree that the grasp would succeed if their representation accurately represented the true object shape. Furthermore, object representations that better fit the observed data should have more influence in deciding which grasps should be executed.

In order for a set of representations $\boldsymbol{R}$ to be used in this framework, the following requirements have to be met:

- at least one object representation in $\boldsymbol{R}$ must be able to propose a set of grasps to be tested, in order to form a pool of grasps g to be tested according to (1);
- each object representation must be able to test a possible grasp, *i.e.*, for each representation $r \in \boldsymbol{R}$, we must have a way of computing $C(s|g,r)$ for any given g. In the next section we will also propose a method for fast approximation of this term using off-line precomputation, applicable for database-driven planning.

The aim of this framework is to allow different object representations (and the associated planning algorithms) to reinforce each other, mitigating the risk of failure due to incorrect scene interpretation, as illustrated in Fig. 1(e). When using database-driven grasping (where we each possible recognition result is considered an independent representation), our framework encourages the selection of a grasp that would work well on a family of objects. The goal is to abstract away from the particular features of one object, unless the recognition algorithm is extremely confident in its result.

III. Data-driven Regression

In this section, we propose a method for evaluating a grasp g on a given object representation r indirectly, based on how well it matches other grasps that are available for r. If we assume that $g_r \in G_r$ form a set of grasps for which $C(s|g_r,r)$ is known, we would like to evaluate g based on how well it matches the grasps in G_r.

Recall that, in our implementation, the definition g of a grasp encodes the position and orientation of the gripper. Intuitively, two grasps are similar if they define two gripper poses that are close to each other in a Euclidian sense. We use the function $N^{g,\sigma_b}(g_r)$ to denote the similarity between two grasps, g and g_r. We assume that the position and orientation of the grasp are independent, and so N is the product of a normal distribution applied to the Euclidean distance between the position components of the grasps, and a Dimroth-Watson distribution applied to the shortest rotation angle between the angular components, as in [8], [9]. σ_b, which we will refer to as the *grasp bandwidth*, is a tunable parameter, defining the size of a region around itself that a given grasp informs. In this study, the positional component of the bandwidth has a standard deviation of 0.01 m, and the orientation component has a Dimroth-Watson concentration parameter corresponding to a standard deviation of 26°.

Using this framework, we compute a Locally Weighted Linear Regression $L(g,r)$ over the confidence level of the grasps in G_r as

$$L(g,r) = \frac{\sum_{g_r \in G_r} C(s|g_r,r)N^{g,\sigma_b}(g_r)}{\sum_{g_r \in G_r} N^{g,\sigma_b}(g_r)} \tag{2}$$

and use it to estimate the confidence level of a new grasp g:

$$C(s|g,r) = \min \begin{pmatrix} L(g,r), \\ C(s|g_r^*,r)N^{g,\sigma_b}(g_r^*) \end{pmatrix} \tag{3}$$

where g_r^* is the grasp in G_r that is most similar to the evaluated grasp g.

The purpose of this evaluation is to combine the information from multiple grasps in G_r in the same region of space, while limiting the region of influence of each grasp to an area defined by the bandwidth parameter σ_b. Other approaches also exist for the problem of learning continuous grasp quality functions from a discrete number of samples [17], [21]. However, we assume that regions with no grasps encode negative information; that is, a grasp in an empty region

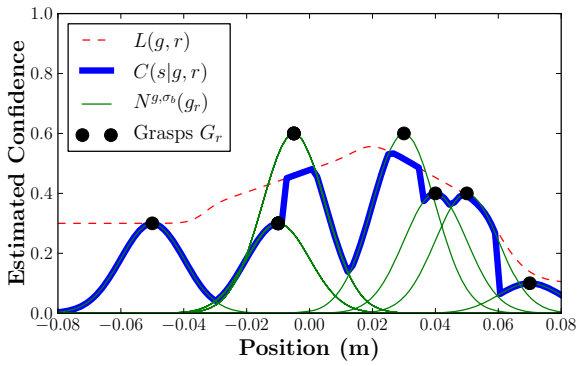


Fig. 2: Illustration of the data-driven regression function. The black circles represent known grasps from G_r, while the green bell curves show the value of a similarity function N centered at each grasp. The regression function (thick blue line) is informed by (the weighted average of) the closest grasp(s).

is likely to fail, and thus as we move away from known grasps, our confidence function drops off with distance. For illustration, a 1D toy example is presented in Fig. 2.

This function can be thought of as defining a continuous measure of grasp success over a region of space using a set of discrete samples. It is important to note, however, that this version closely approximates explicit grasp evaluation only if the set G_r fully samples the space of good grasps for r, with a density comparable to the value of the bandwidth parameter. It is well-suited for cases where an extensive set G_r can be computed off-line, as in the case of database-driven grasping.

IV. Implementation

In this section, we describe the implementation of the framework presented above using two object representations, one of them relying on recognized object models and the other using the object point cluster. The hardware used for implementation is the PR2 personal robot. The features of the PR2 most relevant for this study include two 7-DOF arms, allowing multiple ways of achieving a desired grasp pose, and a narrow-field-of-view stereo camera equipped with a texture projector, providing dense point clouds of the robot's workspace (as illustrated in Fig. 1). Each arm is equipped with a parallel jaw gripper, which is the end-effector used for the implementation of the methods presented in this study.

A. Object Recognition

Object recognition is an extremely active field of research in its own right; the question of which algorithm (or even which type of sensor data) is currently best suited for grasping applications is beyond the scope of this study. In order to provide an implementation of our framework, we used a simple matching algorithm to seed the database-driven component. Once an object point cluster is segmented, an iterative technique similar to ICP [3] attempts to match the cluster against each 3D model in our database.

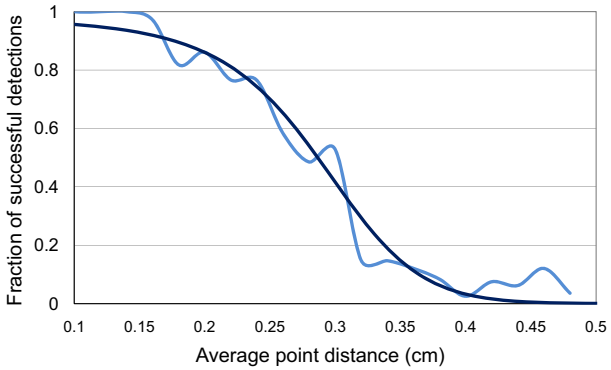


Fig. 3: Conversion from raw object recognition scores to the estimated confidence in the recognition result. Blue line shows ground truth data while the black line shown the analytical model used to approximate it.

The current matching algorithm only operates in 2 dimensions, and thus can only recognize objects sitting upright on a tabletop, either rotationally symmetrical or sitting in a known orientation. Furthermore, it only evaluates how well the points in a cluster match a given model, without reasoning about negative (or the absence of) sensed data. As a result, small point clusters will be considered excellent matches for large database objects, even if they only explain a small part of the mesh. We believe that such limitations underscore the importance of robust grasp planning. One of the strengths of the framework presented here is also its ability to integrate data from multiple sources. As more general and more reliable object recognition algorithms become available, they should directly benefit this approach to grasp planning.

One aspect which requires further attention is the output of a recognition algorithm, or its *confidence level*. In general, different recognition algorithms use different internal quality metrics; our approach returns the quality of the match between the point cluster and a 3D model as the average distance between the points in the cluster and their closest counterpart on the mesh. In order to combine such information from multiple algorithms, these raw results must be translated into correct-detection probabilities.

To perform this translation for our detector, we employed a data-driven method that has the advantage of being agnostic to the inner workings of the recognition algorithm. We collected raw recognition results on a set of 892 point clouds from 44 objects. For a raw score falling in each of 25 discrete intervals, we computed the ratio of correct results (object model correctly identified; we assume that the pose is approximately correct if the model is correct for this detector) vs. total recognition attempts. Fig. 3 shows this data, superimposed with the analytical model derived from it. This model is used in the rest of the paper to map raw recognition scores to probabilities of correct detection.

B. Database-driven Planning

For computing and evaluating grasps on 3D meshes contained in our database of models, we used the publicly

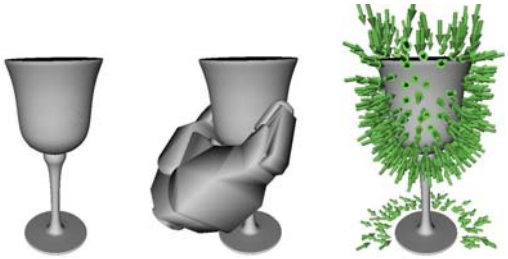


Fig. 4: Database-driven grasp planning: the object model, an example grasp and the complete set of computed grasps.

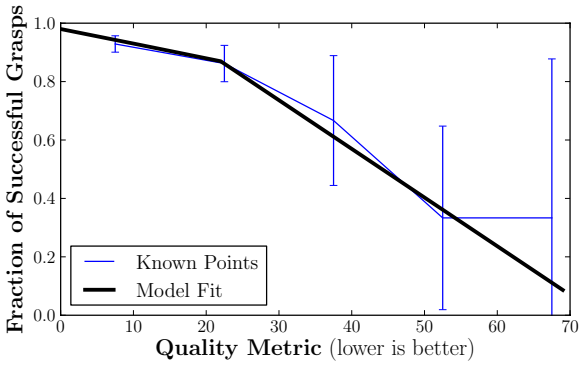


Fig. 5: Conversion from *GraspIt!*'s grasp quality metric to experimental grasp success percentages. Blue lines show ground truth data; the black line shows the analytical model used to approximate it. Blue error bars indicate 95% confidence on the mean, computed using bootstrap sampling.

available *GraspIt!* [20] simulator. To evaluate the quality of a given grasp, and to plan new grasps for an object, we used the energy metric and simulated annealing search described in [5]. An example is shown in Fig. 4.

As previously noted, an object representation should ideally both generate its own list of grasps and test grasps generated using other representations. Testing a novel grasp on a known 3D model can be performed explicitly inside the simulator. However, this test, performed at run-time, is computationally expensive (on the order of 100 ms per grasp). The alternative is to use the regression method presented in Sec. III, where the set of known grasps G_r can be precomputed off-line. Fig. 4 also shows a complete set of reference grasps pre-computed for a given object. The precomputation time for generating grasps was 4 hours per object, and resulted in an average of 539 grasps for each model in the database.

As in the case of object recognition, in order to combine multiple grasp planners under a single framework, we must make the conversion from individual grasp quality metrics to a value comparable among different grasp evaluators–in this case, the probability of grasp success. Again, to provide an analytical model for this conversion, we used a data-driven approach, where 490 grasps generated using *GraspIt!* were executed on the real robot, in order to compare the planner's quality metric against real-life execution results. Fig. 5 shows the resulting data, as well the analytical model used to map a raw score to a grasp success confidence value ($C(s|g,r)$).

C. Robustness to Execution Errors

To account for possible errors in grasp execution (due to imperfect robot calibration or trajectory following), we have extended the quality metric presented above for database-driven planning to also consider a range of possible outcomes for a commanded grasp. Specifically, the quality of a grasp g is determined by evaluating the quality metric not only for g, but also for a set of grasps $g_p \in G_p(g)$ that we refer to as the *perturbations* of g, or grasps that are possible outcomes when the command for executing g is sent to the robot.

In general, an estimate on the confidence in the success of a grasp g can be computed, taking into account possible execution errors, as

$$C(s|g,r) = \sum_{g_p \in G_p(g)} C(s|g_p,r)P(g_p|g) \qquad (4)$$

where $P(g_p|g)$, an error model encoding the probability of actually executing grasp g_p when g is intended (normalized over our samples to sum to 1), depends on the calibration of the robot. In this study, we built a simple error model by independently sampling one perturbation each along the positive and negative directions of the X, Y and Z axes for the position component of a grasp g. Each of the 6 perturbations was assigned a weight $P(g_p|g) = 1/9$, with $P(g|g) = 3/9$ (correctly executing grasp g when g was intended) completing the model. This model is equivalent to performing an unscented transformation over (X, Y, Z) with a center point weight $W^{(0)} = 1/3$ and standard deviation $\sigma = 0.47$cm [16].

D. Cluster-based Planning

The second object representation used in this paper relies solely on the observed point cloud of an object. As a result, it is applicable in a wide range of scenarios, as it does not depend on other components such as a model database, recognition algorithm, or primitive fitter.

For this object representation, we used the grasp evaluation and planning algorithms presented in detail in [12]. This planner uses a set of heuristics, such as hand alignment with the cluster bounding box, size of the bounding box compared to gripper aperture, and number of observed points that fit inside the gripper, to assess the quality of grasp, and also to populate a list of grasp candidates. As in the previous case, grasps planned using other representations can be evaluated by the cluster-based planner either explicitly or by using the regression method of Sec. III.

E. Combining Object Representations

As the cluster-based representation is used in the same framework as the object-recognition-based approach, the relative *confidence levels* used for both representations requires further attention. We recall from Eq. 1 that the information from each object representation r (and associated grasp

Fig. 6: The set of test objects used in this study.

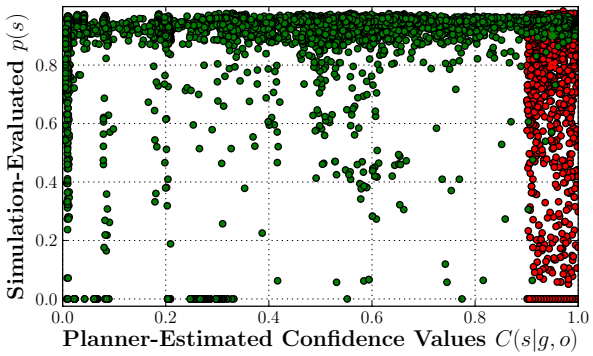


Fig. 7: Comparison of collaborative (green markers) and naive (red markers) grasp planners.

planner) is weighted by the confidence function $C(r|o)$ for that representation, based on the observed data o.

In this study, we are combining the cluster representation, which we will denote by r_c, with n model detection results, which we will denote by r_d^i for $i \in 1,..n$. Each recognition result r_d^i has an associated probability of correct model detection (computed from the raw detection score as in Sec. IV-A), which we will refer to as $P(r_d^i|o)$. An additional normalization step is used to ensure that the absolute number of recognition results does not affect the final result, and only their relative confidence levels do. We use the following methods for computing the associated confidence levels:

$$C(r_c|o) = 1 - \max_k \left(P(r_d^k|o)\right) \quad (5)$$

$$C(r_d^i|o) = \frac{P(r_d^i|o)}{\sum_{j=1..n} P(r_d^j|o)} \max_k \left(P(r_d^k|o)\right) \quad (6)$$

The best recognition result, $\max_k \left(P(r_d^k|o)\right)$, is thus used as an overall measure for the relative weights of the cluster-based and recognition-based representations. This formulation allows us to combine grasp evaluations from both types of representations in a continuous fashion, with object representations ranging from extremely confident in their representation of the actual object geometry (and thus in their grasp predictions) to only informative and finally to contributing no useful information. We note that, while we have found this formulation to be well suited in practice for this particular case, it is arbitrary in nature, and it might not be directly compatible with other potential object detection results or object representations. Such extensions are the focus of current work.

V. Experimental Results and Analysis

We now have implementations available for all the components of the framework of Eq. 1, with the additional option of using the regression-based evaluator of Eq. 3. Our test set, shown in Fig. 6, consisted of 25 objects common in human environments, such as dinnerware and containers of various shapes. The models of these objects were all part of the database used for object detection. The database also contained 45 additional models that, while not part of our real-life test set, were included in the detection results and used by the database-driven grasp planning component.

In order to analyze robustness to detection errors, we also performed tests where each object was temporarily removed from our model database, to simulate grasping non-database objects while retaining knowledge of actual object pose and shape for ground-truth evaluation. We refer to this condition as *novel-object* testing. We note that for objects belonging to a class well-represented in our set (such as cylinders), a similar but non-identical object was often available.

In addition to real-life testing on the PR2 robot, we tested the results of the planner in simulation, using *GraspIt!* to evaluate the planned grasps. All the tests were run on real-life point clouds of the objects acquired with the PR2 robot's stereo cameras. For the simulated tests we used recorded real-life sensor data as input, manually annotated with ground truth information for object identity and pose.

For both object representations discussed in this study, we had the option of using explicit computation of the grasp confidence term, or the data-driven regression of Sec. III. The following options were tested:

- for cluster-based planning, we used explicit grasp evaluation for the simulated results and regression-based evaluation on the real robot.
- for database-driven planning, we used regression-based evaluation based on the pre-computed set of grasps for each object stored in our database.

The collaborative grasp planner requires an average of 3.2s run-time per object on a standard desktop computer, with an additional 2.3s when using explicit cluster-based evaluation.

A. Simulation-based Planner Evaluation

A typical test of the grasp planner on one of the objects in our set used a recorded point cloud of the object sitting alone on a table as input to the planner, then used the simulation engine to test all the returned grasps. Each grasp was evaluated in simulation on the ground-truth object model and location, and its quality metric and ground-truth success probability were evaluated as described in Sec. IV-B. The ground-truth success probability was then compared against the confidence estimated by the planner. It is important to note that, while the planner could only see a single point cloud of the object (similar to run-time conditions on the robot), the final ground-truth testing of the planned grasps

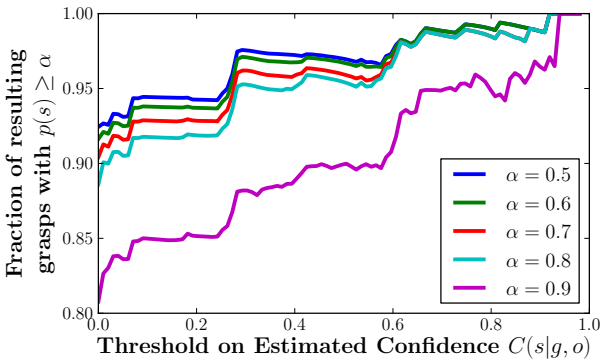


Fig. 8: Choosing a confidence threshold. For each value of the planner-estimated confidence used as a threshold, the plot shows percentage of resulting grasps that exceed a value t for the ground-truth evaluated probability.

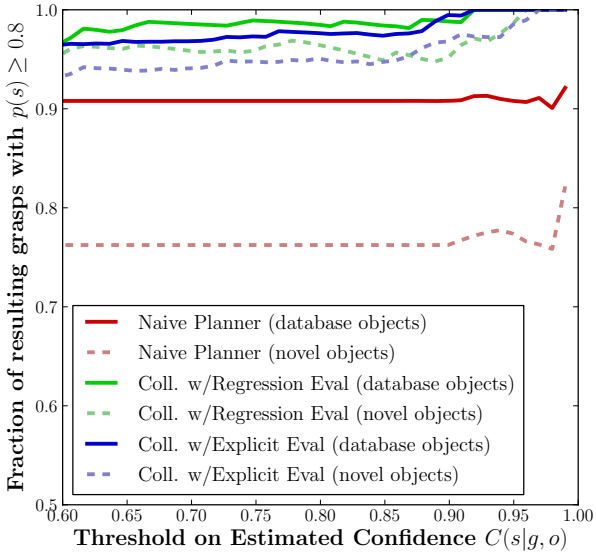


Fig. 9: Comparison of different grasp planners based on percentage of returned grasps that exceed an evaluated success probability of 0.8.

in simulation was performed against a complete object model placed at the correct location in the world.

As a first benchmark, we compared the collaborative planner against a version which assumes that the best result from the recognition algorithm is correct, and simply returns all the grasps pre-computed for that particular model whose quality metrics indicate a probability of success greater than 90%. We refer to this version as the *naive planner*. Fig. 7 shows the result of this comparison by plotting, for all the grasps returned by the two planners, the confidence level estimated by the planner vs. the results of ground-truth simulation testing, for both the novel-object case as well as the case where the tested objects are in the database.

The first thing we note is that all the grasps returned by the naive planner have very high estimated confidence. Indeed, since the naive planner is 100% confident in the result of the detection, and only returns grasps that have more than 90% success on the detected model, it will be highly confident in its results. However, due to errors in object detection, a significant number of those grasps will fail ground-truth testing. In contrast, the collaborative planner is more cautious in its returned results, and succeeds in eliminating a large part of the grasps that would ultimately fail.

It is important to note that the confidence level estimated by the collaborative planner can at best be considered a conservative estimate of the probability of success of a grasp, a fact also reflected in the results shown in Fig. 7. In practice, we have found that a common use case is where a user asks the following question: if I want to choose a grasp with a high probability of success, is there a threshold on the planner-estimated confidence level that I can use? Fig. 8 shows how choosing a threshold for the planner-estimated confidence level (horizontal axis) can insure that a high percentage of the resulting grasps (vertical axis) exceeds a desired probability of success in execution, as evaluated in simulation testing based on ground-truth object identity and pose.

Fig. 9 uses the same format for a more exhaustive comparison of the collaborative and naive planners, shown with a cutoff threshold of 0.6. We performed the following tests:

- Database vs. novel-object detection. As expected, the naive planner behaves significantly worse under the novel-object condition. However, the collaborative planner is able to use information from the cluster planner and similar, but non-identical objects in the database, and so the drop in performance is minimal.
- In order to test the data-driven regression of Sec. III, we also ran the collaborative planner using the explicit *GraspIt!*-backed grasp evaluator throughout its operation. This test required significant computational effort, with the planner spending approximately 10 minutes per object. The results obtained when using data-driven regression are quite similar (as desired), but require far less run-time computation.

B. Grasp Results on the PR2 Robot

Real-life testing of the collaborative framework presented here was carried out using the PR2 robot on 25 objects, shown in Fig. 6. Each object was placed alone on a table within reach of the PR2, and object detection based on the stereo point cloud was done as in Section IV-A. The grasps returned by the planner were subjected to additional testing for situation-specific constraints (such as a feasible Inverse Kinematics solution and a collision-free arm path for placing the gripper in the desired grasp position). The grasps were tested in decreasing order of estimated confidence; once a grasp was deemed feasible, it was executed by the robot. Success was defined as successfully grasping the object, lifting it off the table and moving it to the side of the robot without dropping it. The same object and pose test set were used for both naive and collaborative planners.

As before, tests were performed with both database and novel-object detection. For both cases, we compared the performance of the collaborative planner against the naive planner, with the results shown in Table I. We notice that the collaborative planner has the most significant impact in the case of novel-object testing (improving from 72% to 88% success rate), while also showing a smaller improvement in the case of database objects (from 84% to 88%).

	coll. planner	naive planner
novel-object	22/25	18/25
database object	22/25	21/25

TABLE I: Number of objects successfully grasped and lifted on the PR2 robot.

VI. Conclusions and Future Work

We have presented a collaborative approach to grasp planning based on the observation that real-life sensed data of an object is often incomplete and noisy, allowing for multiple interpretations of a scene. The different ways to interpret object data suggest different approaches to the grasp planning problem. Our framework attempts to combine these approaches and look for consensus among them, thus choosing final solutions (grasps) that are robust to errors in perception and sensor data processing.

In particular, the implementation that we have presented uses information from multiple results from an object recognition algorithm, attempting to find grasps that work on several of them, weighted by their relative confidences. This component is complemented by a grasp planner using an altogether different object representation, consisting only of the observed point cluster. The overall system can handle situations ranging from completely unknown objects to confidently recognized models in a consistent way.

For the case of database-driven grasp planning (or any object representation where grasp evaluation is computationally expensive, but can be performed off-line), we have also used a data-driven regression method that evaluates the confidence in the success of a grasp based on pre-computed data. Our results show that this method can draw on data generated off-line to enable fast on-line execution, without a noticeable decrease in performance.

One of the main advantages of the framework we propose is robustness to errors in the underlying object analysis. The results shown here were obtained with a simple approach to object recognition and using a limited database of models. In future work, we would like to investigate the performance levels that can be achieved using state-of-the-art recognition algorithms, as well as an extensive model set.

It is interesting to consider the underlying reasons that a grasp has a low estimated confidence level. While these do not make an immediate difference (a bad grasp should not be executed regardless of why it is bad), they can help inform a higher-level behavioral planner. If the robot is not confident in its ability to grasp an object because of incomplete data, it can attempt to collect more sensor data. If, however, the object itself affords no good grasps with our current algorithms, then a different task should be attempted. Making this distinction will require more in-depth analysis of the interplay between detection confidence, object representations, and grasp quality. Finally, the current implementation is targeted for a simple robotic hand, which adds no intrinsic degrees of freedom to the grasp planning problem. More dexterous hands will require a better notion of what it means for two grasps to be "similar". Both of these topics are the subject of future work.

References

[1] R. Balasubramanian, L. Xu, P. Brook, J. Smith, and Matsuoka Y. Human-guided grasp measures improve grasp robustness on physical robot. In *ICRA*, 2010.

[2] D. Berenson, S.S. Srinivasa, and J.J. Kuffner. Addressing pose uncertainty in manipulation planning using task space regions. In *Intl. Conf. on Intelligent Robots and Systems*, 2009.

[3] P. J. Besl and M. I. Warren. A method for registration of 3-d shapes. *IEEE Trans. on Pattern Analysis*, 14(2):239–256, 1992.

[4] C. Borst, M. Fischer, and G. Hirzinger. Grasp planning: How to choose a suitable task wrench space. In *IEEE Intl. Conf. on Robotics and Automation*, pages 319–325, 2004.

[5] M. Ciocarlie and P. Allen. Hand posture subspaces for dexterous robotic grasping. *Intl. Journal of Rob. Research*, 28:851–867, 2009.

[6] M. Ciocarlie, G. Bradski, K. Hsiao, and P. Brook. A dataset for grasping and manipulation using ros. In *Workshop: Towards a World Wide Web for Robots, IROS*, 2010.

[7] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Sucan. Towards reliable grasping and manipulation in household environments. In *Intl. Symp. on Exp. Robotics*, 2010.

[8] C. de Granville, J. Southerland, and A.H. Fagg. Learning grasp affordances through human demonstration. In *Intl. Conf. on Development and Learning*, 2006.

[9] R. Detry, E. Baseski, M. Popovic, Y. Touati, N. Krueger, O. Kroemer, J. Peters, and J. Piater. Learning object-specific grasp affordance densities. In *Intl. Conf. on Development and Learning (ICDL)*, 2009.

[10] C. Ferrari and J. Canny. Planning optimal grasps. In *IEEE Intl. Conf. on Robotics and Automation*, pages 2290–2295, 1992.

[11] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. Allen. Data-driven grasping with partial sensor data. In *IROS*, 2009.

[12] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones. Contact-reactive grasping of objects with partial shape information. In *Workshop on Mobile Manipulation, ICRA*, 2010.

[13] K. Hsiao, L.P. Kaelbling, and T. Lozano-Pérez. Task-driven tactile exploration. In *RSS*, 2010.

[14] K. Huebner, S. Ruthotto, and D. Kragic. Minimum volume bounding box decomposition for shape approximation in robot grasping. In *Intl. Conf. on Robotics and Automation*, Pasadena, USA, 2008.

[15] A. Jain and C.C. Kemp. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonom. Rob.*, 2010.

[16] S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401 – 422, mar. 2004.

[17] I. Kamon, T. Flash, and S. Edelman. Learning to grasp using visual information. In *Intl. Conf. on Robotics and Automation*, 1996.

[18] T. Lozano-Perez, M.T. Mason, and R.H. Taylor. Automatic synthesis of fine-motion strategies for robots. *IJRR*, 1984.

[19] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen. Automatic grasp planning using shape primitives. In *Intl. Conf. on Robotics and Automation*, pages 1824–1829, 2003.

[20] Andrew Miller and Peter K. Allen. GraspIt!: a versatile simulator for robotic grasping. *IEEE Rob. and Autom. Mag.*, 11(4), 2004.

[21] R. Pelossof, A. Miller, P. Allen, and T. Jebara. An SVM learning approach to robotic grasping. In *Intl. Conf. on Robotics and Automation*, 2004.

[22] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in human environments. In *Intl. Conf. on Intelligent Robots and Systems*, St. Louis, USA, October 11-15 2009.

[23] A. Saxena, L. Wong, and A.Y. Ng. Learning grasp strategies with partial shape information. In *AAAI*, 2008.