

# Grasp Planning Using Low Dimensional Subspaces

Peter K. Allen, Matei Ciocarlie and Corey Goldfeder

## I. INTRODUCTION

In this chapter we explore the concept of low-dimensional posture subspaces for artificial hands. Recent advances in neuroscience research have shown that control of the human hand during grasping is dominated by movement in a configuration space of highly reduced dimensionality. This has led our group to explore how artificial hands may take advantage of similar subspaces to perform complex grasping tasks.

Subspaces are important not only because they are biologically motivated but because they allow us to create computational frameworks that are tractable for difficult problems, characterized by a large number of degrees of freedom, such as dexterous grasping. The work described in this chapter allows us to compute metrics on thousands of grasps quite efficiently, leading to a sampling-based approach that can adequately cover the space of grasps for a number of robotic hands as well as a human hand model.

This approach is based on a hand posture subspace defined by a small number of basis vectors which we call *eigengrasps*. The implied dimensionality reduction has allowed us to perform *online dexterous grasp planning* both for robots needing to find a correct grasp for an object and for prosthetic devices in which the human provide a subset of the necessary Degrees of Freedom (DOFs), allowing the planner to work in real-time to find a stable grasp. Further, the ability to pre-compute thousands and thousands of stable grasps for dexterous hands over a large class of objects has motivated a new direction in grasp synthesis, which we call *data driven grasping*. If the number of objects to be grasped in the database is very large and comprehensive then robotic grasping becomes a pre-computed indexed database lookup which is extensible to grasping novel objects not in the database.

## II. LOW-DIMENSIONAL KINEMATICS

As the result of complex, longterm evolutionary adaptations at the structural and neural level [41], biological versatility is challenging to replicate in robotics. By using experimental approaches to study human hand control and generation of hand postures we can however gain new insights into how humans so effortlessly grasp objects.

A common thread in the study of human grasping and its applications to robotics builds on the observation that humans simplify the huge space of possible grasps through learning and experience, enabling them to quickly choose good grasps for a wide variety of objects. The attempt to simplify the configuration space of dexterous artificial hands has resulted in the concept of “grasp taxonomy,” widely discussed in the robotics literature (for reviews, see [12], [23]). When focusing on continuous subspaces rather than discrete taxonomies, previous studies have also shown that control of grasping in humans occurs in a smaller space of possible hand configurations [38], [36], [39], [37]. Simultaneous motion of the fingers is characterized by coordination patterns that reduce the number of independent degrees of freedom to be controlled: despite the large number of DOFs of the hand, *humans utilize linear combinations of relatively few eigenvectors of the DOF space to match hand shape to object geometry*. This behavior results from biomechanically constrained interactions among joints of the digits as well as complex coordination patterns of hand muscle activity [41].

Numerical analysis of human hand postures can thus not only reveal the intrinsic low-dimensional nature of the data, but also suggest new directions for robotic hands. Based on inter-digit coordination caused by mechanical constraints in the anatomy of the hand, robotic hands can be built with highly interconnected finger actuation mechanisms [5], [21]. The assumption that motor control synergies also take place at a higher level in the Central Nervous System [28], [8] implies the use of low-dimensional control algorithms for dexterous robotic hands, such as the ones presented in this chapter.

Another important aspect of grasping concerns the relationship between low dimensional subspaces and the task being performed. Previous work [47] has shown that the execution of different manipulation tasks (such as flipping pages or crumpling paper) are characterized by different linear combinations. Interestingly, a posture subspace can be found even in the less constrained setting of haptic exploration [46]. Hand posture during the reach phase of a complete reach-to-grasp action is also described by a different (and lower-dimensional) principal component spectrum than the grasp phase [28], [37]. These results show that, when using a low-dimensional control space for robotic hands, the choice of the subspace has to be correlated with the proposed task. Finally, all the studies discussed so far have used Principal Component Analysis, and thus have addressed only linear subspaces that can be extracted from hand posture data. Linear decomposition has been successfully used in the past on different types of biometric data, ranging from face appearance [49] to dynamics of arm motion [15]. However, non-linear dimensionality reduction methods can potentially reveal different manifold structures of the same data [48].

### A. Eigengrasps

When performing human user studies, the usefulness of a hand posture subspace can be quantified by how well it approximates a given set of input data. In this chapter, we present a constructive approach, oriented towards application for artificial hands: given a hand posture subspace, we will use it to synthesize new hand postures for accomplishing a particular grasping task.

Any hand posture is fully specified by its joint values, and can therefore be thought of as a point in a high-dimensional joint space. If  $d$  is the number of DOFs of the hand, then a posture  $\mathbf{p}$  can be defined as

$$\mathbf{p} = [\theta_1 \ \theta_2 \ \dots \ \theta_d] \in \mathcal{R}^d \quad (1)$$

where  $\theta_i$  is the value of  $i$ -th degree of freedom.

As we have discussed above, previous research suggests that most human grasping postures derive from a relatively small set of discrete pregrasp shapes. In particular, Santello *et al.* [36] collected and analyzed a large set of data containing grasping poses from subjects that were asked to shape their hands as if they were grasping a familiar object. Principal Component Analysis of this data revealed that *the first two principal components account for more than 80% of the variance*, suggesting that a very good characterization of the recorded data can be obtained using a much lower dimensionality approximation of the joint space.

In our work, we will refer to the principal components of these postures as *eigengrasps*. The implication is that they form a **low-dimensionality basis for grasp postures**, and can be linearly combined to closely approximate most common grasping positions. By choosing a basis comprising  $b$  eigengrasps, a hand posture placed in the subspace defined by this basis can be expressed as a function of the amplitudes  $a_i$  along each eigengrasp direction  $\mathbf{e}_i$ :

$$\mathbf{p} = \mathbf{p}_m + \sum_{i=1}^b a_i \mathbf{e}_i \quad (2)$$

$$\mathbf{e}_i = [e_{i,1} \ e_{i,2} \ \dots \ e_{i,d}] \quad (3)$$

where  $\mathbf{p}_m$  is the “mean” posture that describes the origin of the eigengrasp subspace. When computing eigengrasps based on recorded human data,  $\mathbf{p}_m$  is simply the average of all recorded hand postures. Each eigengrasp  $\mathbf{e}_i$  is a  $d$ -dimensional vector and can also be thought of as direction of motion in joint space. Motion along one eigengrasp direction will usually imply motion along all (or most) degrees of freedom of the hand. We note that the hand posture is now defined by the amplitudes vector  $\mathbf{a} = [a_1 \ \dots \ a_b] \in \mathcal{R}^b$ .

### B. Extending Eigengrasps to Robotic Hands

Although the work of Santello *et al.* is centered on the study of the human hand, we have found this approach to be extremely useful for robotic hands as well. In our study, we have applied the eigengrasp concept to a total of 4 hand models: the BarrettHand™, the DLR hand [27], the Robonaut hand [6] and finally a human hand model. All our hand models, as well as the eigengrasps used in each case, are presented in table I.

For the human hand we have chosen eigengrasp directions based on the published results in [36], taking advantage of the fact that they have been derived through rigorous study over a large number of recorded samples. Since such data is not available for robotic hand models, we have derived eigengrasps attempting to define grasp subspaces similar to the one obtained using human hand eigengrasps. In most cases, such decisions could be made based directly on the similarities with the human hand. For example, the human metacarpophalangeal (MCP) and interphalangeal (IP) joints can be mapped to the proximal and distal joints of robotic fingers. In the case of the BarrettHand, changes in the spread angle DOF were mapped to human finger abduction. All eigengrasps were defined such that they satisfy the orthogonality constraint that naturally occurs when using Principal Component Analysis. While we found our choices to produce good results, the optimal choice of eigengrasps for non-human hands, as well as the choice of which eigengrasps to use for a particular task, are open questions and interesting directions for future research.

The eigengrasp concept allows us to design flexible control algorithms that operate identically across all the presented hand models. The key to our approach is that the eigengrasps encapsulate the kinematic characteristics of each hand design. This enables control algorithms that operate on eigengrasp amplitudes to ignore low-level operations and concentrate on the high-level task. Another advantage is the significant dimensionality reduction (by as much as a factor of 10 for complex hands) obtained by operating in the reduced basis eigengrasp space as opposed to the full joint space. In the next section we will derive a grasp planning algorithm that makes use of both these concepts.

## III. GRASP PLANNING USING EIGENGRASPS

In essence, the grasp planning task can be thought of as an optimization problem in a high-dimensional space that describes both hand posture (intrinsic DOFs) and position (extrinsic DOFs). Consider the goal of maximizing a quality function of the form:

$$Q = f(\mathbf{p}, \mathbf{w}) \quad (4)$$

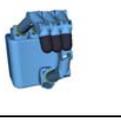
Model	DOFs	Eigengrasp 1			Eigengrasp 2		
		Description	min	max	Description	min	max
Barrett	4	Spread angle opening			Finger flexion		
DLR	12	Prox. joints flexion Finger abduction Thumb flexion			Dist. joints flexion Prox. joints extension Thumb flexion		
Robonaut	14	Thumb flexion MCP flexion Index abduction			Thumb flexion MCP extension PIP flexion		
Human	20	Thumb rotation Thumb flexion MCP flexion Index abduction			Thumb flexion MCP extension PIP flexion		

TABLE I  
EIGENGRASPS DEFINED FOR THE ROBOTIC HAND MODELS USED IN THIS CHAPTER.

If  $d$  is the number of intrinsic hand DOFs then  $p \in \mathcal{R}^d$  represents the hand posture and  $w \in \mathcal{R}^6$  contains the position and orientation of the wrist.

Intuitively, this function has to be related to the quality of the grasp. However, most formulations pose a number of problems. First, it is very difficult, or even impossible, to compute an analytical gradient. Second, such functions are highly non-linear, as small changes in both finger posture and wrist position can drastically alter the quality of the resulting grasp. Finally, the legal parameter space is complex, having to satisfy multiple constraints: prevent inter-penetration with the object to be grasped as well as potential obstacles, and maintain joint values within their acceptable ranges.

#### A. Optimization Algorithm

We directly address all of these problems by using **simulated annealing** as the preferred optimization method (for a general review of the simulated annealing algorithm we refer the reader to [24]). Its stochastic nature makes it a particularly good choice for our task: since new states are generated as random neighbors of the current state, computation of the energy gradient is not necessary, and the algorithm works well on non-linear functions. Furthermore, the possibility of an “uphill move” to a state of higher energy allows it to escape local minima which can trap greedier methods such as gradient descent. However, the random exploration of the input domain means that high dimensionality of the parameter space will severely affect the computational efficiency of this algorithm.

We therefore propose performing the optimization **in eigengrasp space**, as opposed to DOF space. The energy function takes the form

$$Q = f(\mathbf{a}, \mathbf{w}) \quad (5)$$

where  $\mathbf{a} \in \mathcal{R}^2$  is the vector of eigengrasp amplitudes. This effectively reduces the parameter space to 8 dimensions (2 eigengrasp amplitudes plus 6 extrinsic DOFs) from as high as 26 dimensions in the case of the human hand.

#### B. Energy Function

Most grasp quality metrics that have been proposed in the literature are based on the locations of the contacts between the hand and the target object. Our context is somewhat different: we need a quality metric that can also assess the quality of a *pre-grasp*, where the hand is very close, but not in contact with the target. For each hand model, we pre-define a number of *expected* contact locations by sampling each link of the fingers as well as the palm, as shown in Fig. 1a. The value of the quality function is maximized for those hand postures that bring each expected contact location as close as possible to the target object. We are therefore searching for postures where the hand is wrapped around the object, generating a large contact area using all the fingers as well as the palm. As shown in Fig. 1b, for each desired contact location on the hand, identified by the index  $i$ , we define the local surface normal  $\hat{\mathbf{n}}_i$  as well as the distance  $o_i$  between the desired contact location and the target

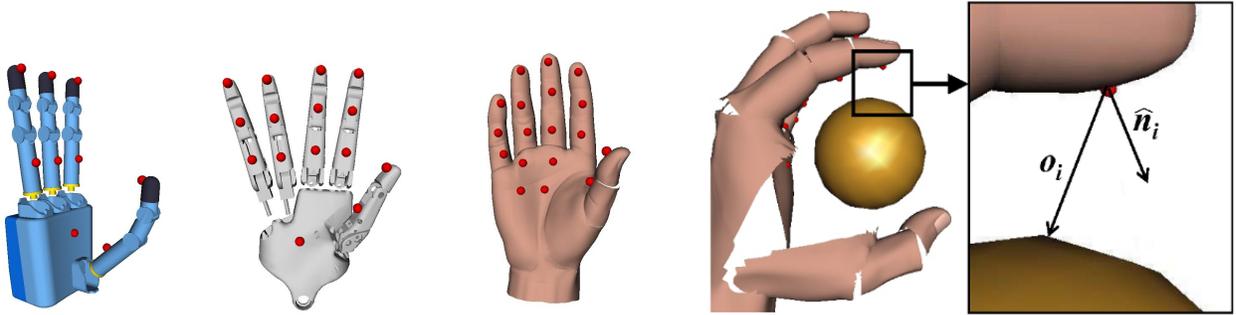


Fig. 1. Left: complete set of pre-defined desired contact locations for the DLR, Robonaut and Human hands. Right: for a desired contact with index  $i$ , we define the surface normal  $\hat{n}_i$  and the current distance to the target object  $\mathbf{o}_i$ .

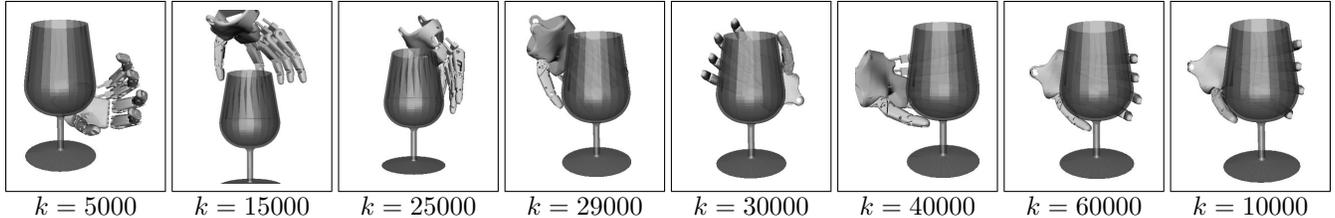


Fig. 2. Simulated annealing example over 100,000 iterations. Each image shows the best state found until iteration  $k$ .

object. We then compute a measure  $\delta_i$  of the distance (both linear and angular) between the desired contact and the surface of the object:

$$\delta_i = \frac{|\mathbf{o}_i|}{\alpha} + \left(1 - \frac{\hat{n}_i \cdot \mathbf{o}_i}{|\mathbf{o}_i|}\right) \quad (6)$$

where  $\alpha$  is a scaling parameter required to bring the range of useful linear distances (measured in mm) in the same range as the normalized dot product between  $\hat{n}_i$  and  $\mathbf{o}_i$  (in our study we use a value of  $\alpha = 50$ ). For a given hand posture, the total value of the quality function is then computed as:

$$Q = \sum_{\text{all desired contacts}} (1 - \delta_i) \quad (7)$$

In most cases, the hand postures that maximize the value of  $Q$  create an enveloping grasp of the object, especially for complex models grasping objects similar in size to the hand. The optimized value of this function can be seen as a measure of how well the hand shape can be set in order to match a given object while operating in a low-dimensional subspace. In Section IV we will also present an alternative quality function formulation that includes a built-in notion of grasp wrench space analysis.

### C. Simulated Annealing example

We have implemented the simulated annealing approach using our publicly available *GraspIt!* simulation engine [31]. For each state generated during the annealing schedule, *GraspIt!* uses forward kinematics to place the robotic hand model in the correct posture and checks for collisions against the object to be grasped as well as other obstacles. If the state is found to be legal, the corresponding energy function is computed and the annealing algorithm continues. This process can be repeated until a satisfactory energy level has been reached, or a pre-specified number of iterations has been exceeded.

Before presenting an extensive test of this optimization method, involving multiple hand models as well as different objects, we will discuss a typical example, in order to analyze the behavior of the simulated annealing algorithm in more detail. This example involves the Robonaut hand grasping a glass. The optimization was performed over 100,000 iterations. Fig. 2 shows the temporary solution (best state found so far) at various points during the optimization. We can observe what is considered typical behavior for a simulated annealing implementation: at first, the search goes through random states, accepting bad positions as well as good positions. As the annealing schedule progresses, the search space is sampled more often in the vicinity of the good states, while bad states are no longer accepted. Finally, in the later stages, the search is confined in a small neighborhood around the best state, which is progressively refined. The total time required for the optimization was 173 seconds, or 1.73 milliseconds / iteration. The most significant amount of computation was spent checking the feasibility of each generated state (*i.e.* checking for collisions and inter-penetrations).

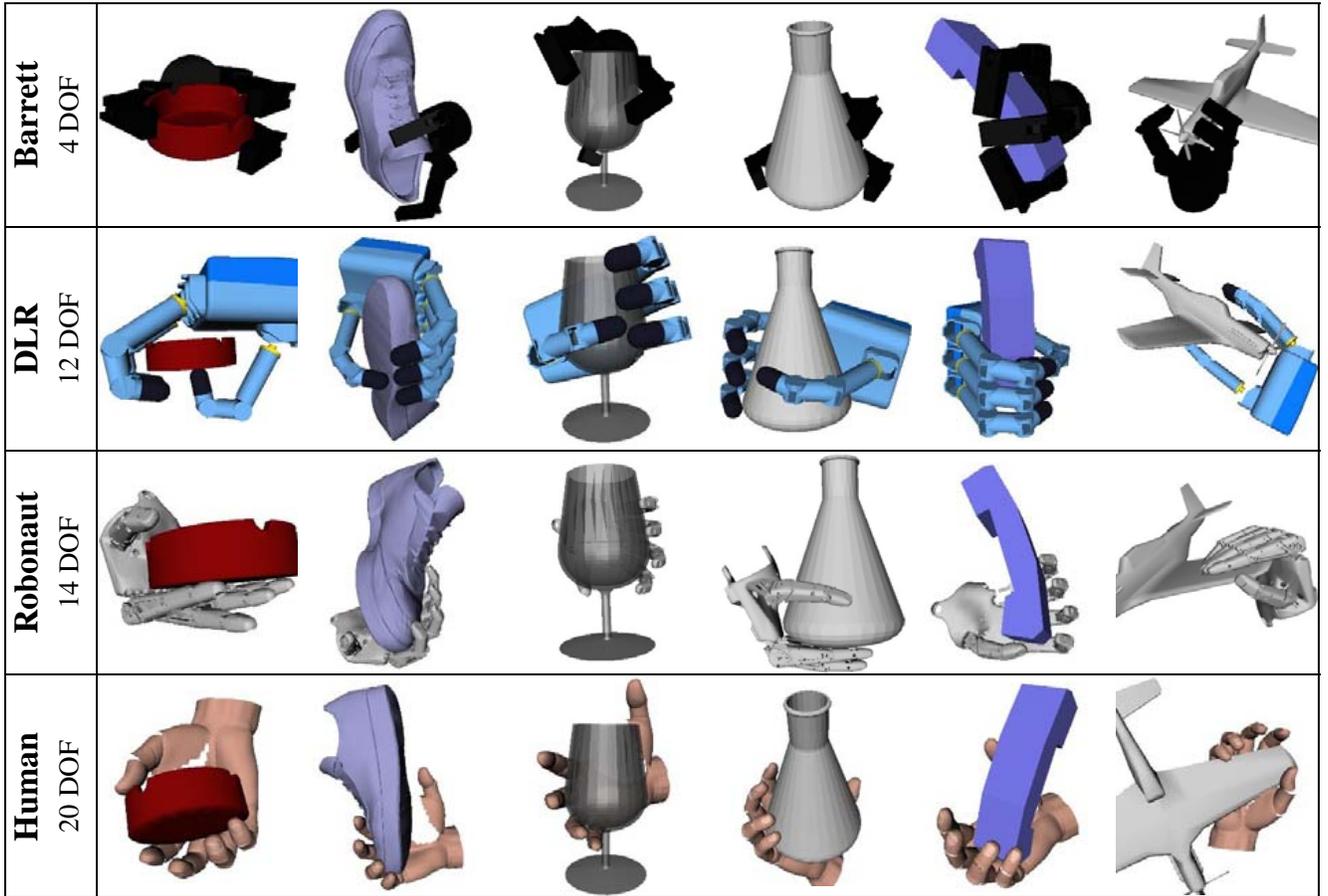


Fig. 3. Best hand postures found in a 2-dimensional eigengrasp space using simulated annealing optimization.

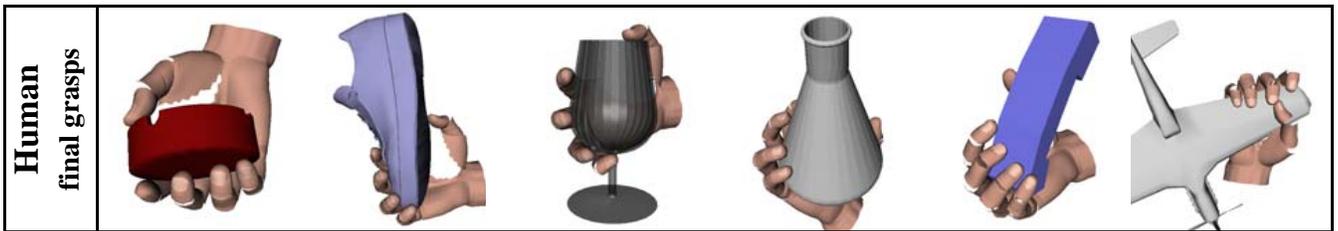


Fig. 4. Examples of final grasps obtained from optimized postures by closing each finger until motion is stopped by contact with the object.

#### D. Grasp Planning Examples

In order to test the effectiveness of our framework for the task of dexterous grasp planning, we have applied the 2-dimensional eigengrasp optimization using all four previously discussed robotic hand models on a set of six objects. Fig. 3 shows the result of the annealing search for each hand-object combination. Our focus in this section is to evaluate the best hand postures that can be found in eigengrasp space. Therefore, Fig. 3 presents the best hand posture found by the optimization algorithm without any additional refinements, allowing a direct assessment of the optimization method through visual inspection of its output.

These results show that, when the search is confined to a low-dimensional eigengrasp space, it does not reach a global optimum of the quality function where all the desired contact location touch the target object. However, the local optimum found in eigengrasp space can be used as a *pre-grasp* by performing an additional adjustment where the hand leaves the planning subspace in order to conform to the surface of the object: execution of a binary “close all fingers” command, that simply closes all fingers until contact with either the object or another nger prevents further motion (Fig. 4). We use form-closure as the analysis criterion for the resulting grasps, as our goal is the synthesis of stable grasps with no weak points. Following [29], we label a grasp as form-closed if and only if it can resist an infinitesimal disturbance regardless of its direction.

In order to perform a quantitative analysis of the pre-grasps obtained through posture optimization, we have applied this adjustment to the 20 distinct postures with the highest quality values found by one execution of the optimization algorithm for

	Ashtray		Shoe		Glass		Flask		Phone		Plane	
	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD	Avg.	SD
BarrettHand	2.8	2.2	1.0	1.0	5.8	5.3	3.4	1.7	1.4	2.1	4.0	3.1
DLR	11.0	3.6	6.0	3.4	0.8	0.8	3.2	2.7	2.8	3.6	1.8	0.8
Robonaut	7.0	2.3	9.0	1.7	14.4	3.7	14.4	3.7	10.0	2.8	3.6	2.3
Human	14.6	2.3	11.0	2.5	11.2	1.6	13.4	3.5	8.4	2.3	1.8	1.3

TABLE II

NUMBER OF FORM-CLOSED GRASPS OBTAINED FROM 20 PRE-GRASPS FOUND IN A 2D EIGENGRASP SPACE (AVERAGE AND STANDARD DEVIATION OVER 5 EXECUTIONS FOR EACH HAND AND OBJECT).

each hand-object combination. On average, we obtained 6 form-closure grasps for each case; the detailed results for all the hand models and objects are presented in Table II. Each optimization was performed over 70,000 iterations, with an average running time of 162 seconds. In the case of the human hand, Fig. 4 also shows all the final grasps obtained when using as pregrasps the corresponding postures from Fig. 3.

These findings confirm our expectations of eigengrasp space as a *pre-grasp* or *grasp planning space*: in general, closing the fingers of a dexterous hand starting from a random configuration is not enough to obtain a stable grasp. However, the presented results show that if the starting position is the result of the eigengrasp optimization algorithm, we can obtain multiple form-closure grasps even when using highly dexterous hands.

#### IV. ONLINE INTERACTIVE DEXTEROUS GRASPING

In the previous section we have presented an optimization algorithm that uses a low-dimensional subspace when searching for hand postures that match the shape of a grasped object. However, a significant amount of the computational effort was dedicated to optimizing extrinsic DOFs (wrist position and orientation, 6 variables) versus intrinsic DOFs (eigengrasp amplitudes, 2 variables). As the focus has been on dimensionality reduction for the intrinsic DOFs domain, no attempt has been made to simplify the extrinsic DOFs search domain. For fully autonomous grasp synthesis this is a necessary component: a correct finger posture is only relevant when combined with an appropriate wrist position relative to the target object.

An important category of grasp planning applications that do not require complete autonomy stems from the field of neural hand prosthetics. Such devices combine a degree of human control with artificial hardware and algorithms. Formalizing this concept using our framework means that an external operator can specify desired values for some, but not all of, the variables that define a grasp. For example, previous research [45] has enabled a primate to directly control the linear velocity of the endpoint of a robot arm through 3 DOFs in real time. This control was achieved by measuring the activity of individual cortical neurons that correspond to individual preferred directions of each neuron in space. The vector sum of preferred directions of a population of neurons, each scaled by their individual unit activity, provided the velocity of robotic end-effector movement.

In contrast, controlling finger posture has proven to be significantly more difficult. A number of possible approaches are described in the literature, including electromyography [50] and cortical implants [44]. These studies have shown success in decoding a limited number of information channels, therefore controlling a highly dexterous hand for interactive grasping remains an open and challenging problem. In this study we propose a grasp planning method that combines the eigengrasp framework for reducing the dimensionality of the hand configuration space with real-time operator input simplifying the spatial components of the search. Our goal is to enable an operator to complete dexterous grasping tasks with limited direct control over finger posture.

##### A. System Overview

In our current implementation, the user provides on-line information on the position and orientation of the wrist. This data is currently provided using a 6-DOF magnetic tracker. While we have not yet integrated this component in a real prosthetic system, it is our directional goal; we envision that hand position information will be extracted from cortical activity in a similar fashion to the primate study described above. In contrast, finger posture is entirely under control of the automatic component, which selects an appropriate hand shape by combining information about the geometry and pose of the target object with the position input provided by the operator. The only additional information needed from the user is a binary “click” command for completing a grasp, which we will describe below.

It is important to note that our approach must compensate for the lack of complete user grasp data by using knowledge of the target object geometry, as well as its initial position relative to the hand. In previous work [25], we have shown that it is possible to perform grasp planning using a vision-based system for object recognition and localization. Compared to the optimization method presented in the previous section, this system also has to satisfy two important criteria: first, the output has to be in the form of explicit form-closure grasps rather than optimized pre-grasps; second, solution grasps must be found at a fast enough rate to enable on-line interaction with the operator and usage of real-time input. The execution of a grasping

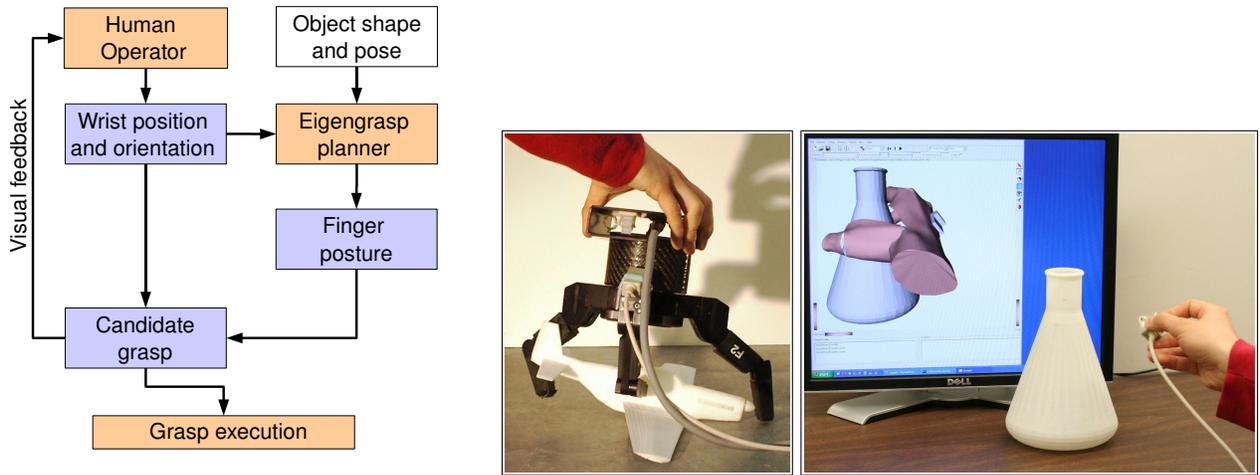


Fig. 5. Interactive grasp planning using wrist position input from a human operator. Left: system overview; Right: applied examples using a real BarrettHand and a dexterous hand in a simulated environment.

task must therefore be performed at a speed that approaches natural human behavior, on the order of seconds as opposed to minutes.

A high-level overview of the complete system and the interaction with the operator is provided in Fig. 5. The planning algorithm runs on the *GraspIt!* simulator platform. Even though the grasp planner runs in a simulated environment, the results can be applied to a real robotic hand, allowing the user to interact with the hand directly and use it to pick up surrounding objects. The simulator receives user input and sends it to the grasp planner which processes it and outputs potential grasps, which are in turn used to generate commands for the robotic hand. The operator can hold the hand and approach the target object; the position of the hand relative to the target is tracked using a Flock of Birds (Ascension Corp., VA) magnetic tracker. We have also applied our method on a range of more complex hand designs (including the DLR hand, the Robonaut hand as well as the human hand model) using the virtual environment in *GraspIt!*; the operator can change the position of the virtual wrist by directly manipulating the magnetic tracker. In both cases, wrist position is supplied as input to the grasp planner, but the operator has no direct control over finger posture.

### B. User Interaction with the Grasp Planner

In general, in order to uniquely identify a grasp, three variables are needed to specify the position of the wrist and three more for its orientation. In the context of our application, we expect the user to specify a desired approach direction to the target; however, the presence of such external input does not fully eliminate the spatial component of the grasp planning search. First, it is not practical to wait until the user has brought the wrist into a final grasping position before starting the search for an appropriate finger posture, as this behavior would decrease the interactivity of the system. Rather, it is preferable to start the search early, and attempt to predict where the user intends to place the wrist. Second, this prediction allows the system to offer feedback to the user: as soon as an anticipated grasp is found, the grasp planner can shape the fingers accordingly. The user can then decide if the grasp is satisfactory and either continue towards the target or choose another approach direction if the system is unable to find an acceptable solution.

This behavior can be implemented efficiently by re-parameterizing the spatial component of the grasp planner as shown in Fig. 6. For each hand model, we define a preferred search direction  $\mathbf{d}$  based on the kinematics of the hand, usually normal to the palm. Then, starting from a hand position specified by the operator, we search for good grasps in a conical region around the search direction using 3 variables: the distance  $|\mathbf{d}|$  along the approach direction, as well as two angular variables,  $\theta$  and  $\phi$ . The operator is instructed to approach the object along a direction that is generally similar to the search cone; however, the search directions are defined in order to make this a natural choice. In the examples in Fig. 6 this means that the user is asked to keep the palm approximately facing the target, as opposed to other possibilities like a sideways or backwards approach.

The role of this parameterization is to reduce the number of extrinsic DOFs that are used for grasp planning, focusing on areas where good grasps are most likely to be found. Using this heuristic, the search will automatically ignore states where, for example, the hand is facing away from the target object. However, the user is not expected to specify an exact wrist position for a good grasp; by searching along the approach direction  $\mathbf{d}$  the planner attempts to anticipate the intended final grasp. The angular variables  $\theta$  and  $\phi$  allow the planner to compensate for noisy measurements in the intended wrist position, and allow for more flexibility in the search for solution grasps. By adding these 3 variables to the eigengrasp amplitudes describing hand posture, we obtain a low-dimensional domain that can be searched fast enough to respond to on-line changes in the wrist position input provided by the human operator.

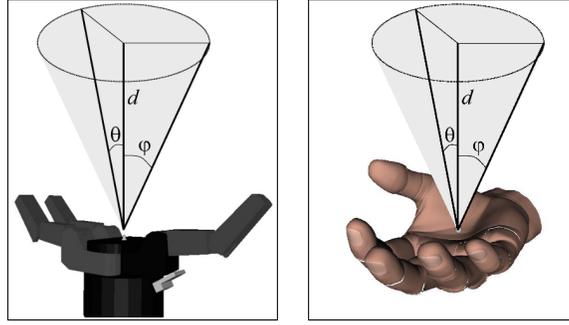


Fig. 6. Search directions defined for the BarrettHand and human hand models. The direction of the vector  $d$  is predefined relative to the palm. Its magnitude, as well as the values of the angles  $\theta$  and  $\phi$  are variables defining a conical search area.

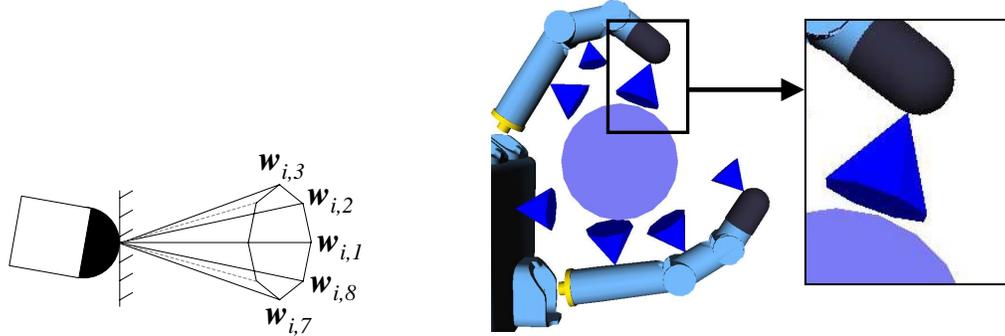


Fig. 7. Left: contact wrench space example using a Coulomb friction cone; Right: Multiple contact wrench spaces, each scaled based on the contact distance metric  $\delta_i$ .

### C. Quality Function Formulation Using Scaled Contact Wrench Spaces

When the posture optimization algorithm is used for on-line grasping tasks, we use a formulation of the quality function  $Q$  that is better adapted for interactive operation. Recall that, in the form presented in section III, our formulation rewards hand postures that bring all the fingers, as well as the palm, as close to the surface of the object as possible. For the application presented here, it is necessary to also reward hand postures that create stable, but not necessarily enveloping grasps (consider as an example the case of a fingertip pinch grasp applied on a thin object). We therefore propose an alternative quality function which is fast to compute and can assess the *potential* quality of a pre-grasp posture using a modified version of the Grasp Wrench Space (GWS)  $\epsilon$ -metric [14]. A detailed description of this metric in its original form is beyond the scope of this paper; we provide a brief overview below, and for further details we also refer the reader to [30].

For each contact  $i$ , we assume that the space of forces and torques that can be transmitted is bounded by the convex hull of a finite set of 6D wrenches  $w_{i,j}$  where  $1 \leq j \leq k$ . The convex hull of these wrenches forms the Contact Wrench Space (CWS). We note that this approach imposes a linear form for what are normally quadratic friction constraints; for example, in the case of Coulomb friction, the torque components of all  $w_{i,j}$  are null, and the force components sample a polygonal approximation to the contact friction cone. In order to define the GWS, the contact wrenches from all contacts are first expressed relative to a common coordinate system. This coordinate system is usually anchored at the center of mass of the object and the choice of axes directions is arbitrary. We denote the matrix that transforms a wrench from the local coordinate system of contact  $i$  to the global object coordinate system by  $R_i \in \mathcal{R}^{6 \times 6}$ .

In our implementation, we are usually assessing the quality of a pre-grasp shape where the fingers are not in contact with the target. Therefore, we assume that the hand can apply *potential* contact wrenches using the desired contact locations shown in Fig. 1. When computing the GWS, we scale the potential wrenches at each desired contact proportional to the inverse of the distance metric  $\delta_i$  computed as in (6):

$$\text{GWS} = \text{ConvexHull} \left\{ \bigcup_{\text{all desired contacts}} (1 - \delta_i) R_i \bigcup_{j=1}^k w_{i,j} \right\} \quad (8)$$

Thus, if the value of  $\delta_i$  is small, the contact will have a significant contribution to the GWS, and states that bring it closer to the object surface will be rewarded by a higher quality value. If, on the contrary, the desired contact is far from the object, it will not significantly affect the grasp quality measurement. If the contact is far enough from the object so that its corresponding weight of  $1 - \delta_i$  is negative, it is completely excluded from the computation.

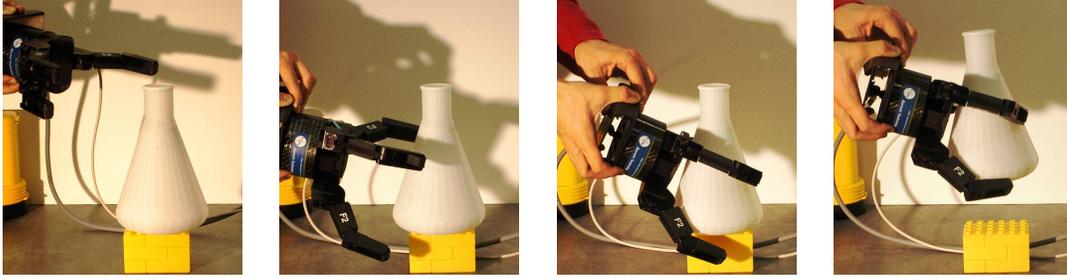


Fig. 8. Example of a complete grasping task: initial approach, finger-preshaping using grasp planning result, continued approach and final grasp execution.

After building the GWS, we compute the  $\epsilon$ -quality measure as described in [14], [30]. The quality of the grasp is considered equal to the radius  $\epsilon$  of the largest 6D ball, centered at the wrench space origin, that can be enclosed within the GWS. If  $\epsilon = 0$  then the origin itself is not contained in the hull, and the grasp does not have form closure. For  $\epsilon > 0$ , the grasp can resist any disturbance, and the maximum magnitude of the contact forces needed to resist a disturbance is inversely proportional with  $\epsilon$ .

The process is illustrated in Fig. 7 for the DLR hand grasping a disc. In this example, each contact is modeled by a friction cone, approximating Coulomb friction for rigid bodies. However, other local contact models can also be used. For example, the ability to create stable, encompassing grasps with subsets of fingers is increased by using soft fingertips that deform during contact. In addition to tangential friction, such contacts can also apply frictional torque. The friction cone is thus replaced by a four-dimensional “friction ellipsoid” which constrains the relationship between tangential force and frictional torque [22]. This effect can be captured by linearizing the friction ellipsoid and using the appropriate contact wrenches  $w_{i,j}$ , as shown in [9]. This method enables the use of rubber-coated fingertips for our robotic hands, without compromising the accuracy of the grasp quality computations.

#### D. Computation of Form-closure Grasps

The automated grasp planner searches for solution grasps in two stages. The first stage is the posture-optimization algorithm presented in Section III, using the quality function formulation described above. For interactive tasks, each run of the simulated annealing algorithm is performed over 2000 iterations, taking advantage of the fact that the search domain is 5-dimensional (2 eigengrasp amplitudes and 3 wrist position/orientation variables), as opposed to the 8-dimensional domain used for fully autonomous searches. After reaching this number of iterations, the search is restarted by resetting the annealing temperature. As a result, the planner does not get stuck if one particular search fails; rather, the search is restarted and takes advantage of any changes in the approach direction provided by the operator.

The user-specified reference wrist position is updated continuously during the search. The results of the optimization are therefore always relative to the current position of the wrist. However, we recall that the low-dimensional optimization procedure can still only produce *pre-grasp* shapes; in order for the system to allow successful completion of the task, *final* grasping postures satisfying the form-closure requirement are necessary. In order to achieve interactive rates, this expensive computation is only performed using the best pre-grasps found during each run of the annealing optimization, which are queued and sent to the second stage of the planning process.

For each candidate pre-grasp resulting from the first stage, we use the contact detection engine within *GraspIt!* to compute the final grasp that results by closing the fingers on the object. Once the contacts between the hand and the object have been determined, we compute the exact quality value of the final grasp by applying the Ferrari-Canny metric in the original form presented in [14]. If the grasp is found to have form-closure, it is saved, along with its associated quality value, as a potential solution, and used by the next component of the system, which is responsible for interaction with the human user.

When computing the final grasping posture resulting from a candidate pre-grasp, we take into account specific mechanical properties of the hand, such as passive mechanical adaptation to the shape of the target. A number of robotic hands, such as the BarrettHand, the SDM Hand [13] and the CyberHand [7] rely on passive mechanical adaptation, as it significantly increases grasp stability without increasing the complexity of the control mechanisms. All of the results involving the BarrettHand presented in this paper take into account its adaptive actuation mechanism which allows distal joints to close even when proximal joints controlled by the same motor have been stopped due to contact.

In our implementation, the two planning phases described in this section (simulated annealing search for pre-grasps and final grasp testing for form-closure) run in separate threads. As soon as a candidate pre-grasp is found, it is queued for testing, but the search for new candidates continues independently of the testing phase. Also, candidate pre-grasps are independent of each other, and can be tested simultaneously. This parallelism allows us to take advantage of the current evolution in multi-core architectures, largely available on standard desktop computers.

We can now provide a complete step-by-step walk-through of a grasping task that combines user input and automated grasp planning. Fig. 8 shows the execution of a grasp proceeding through the following stages:

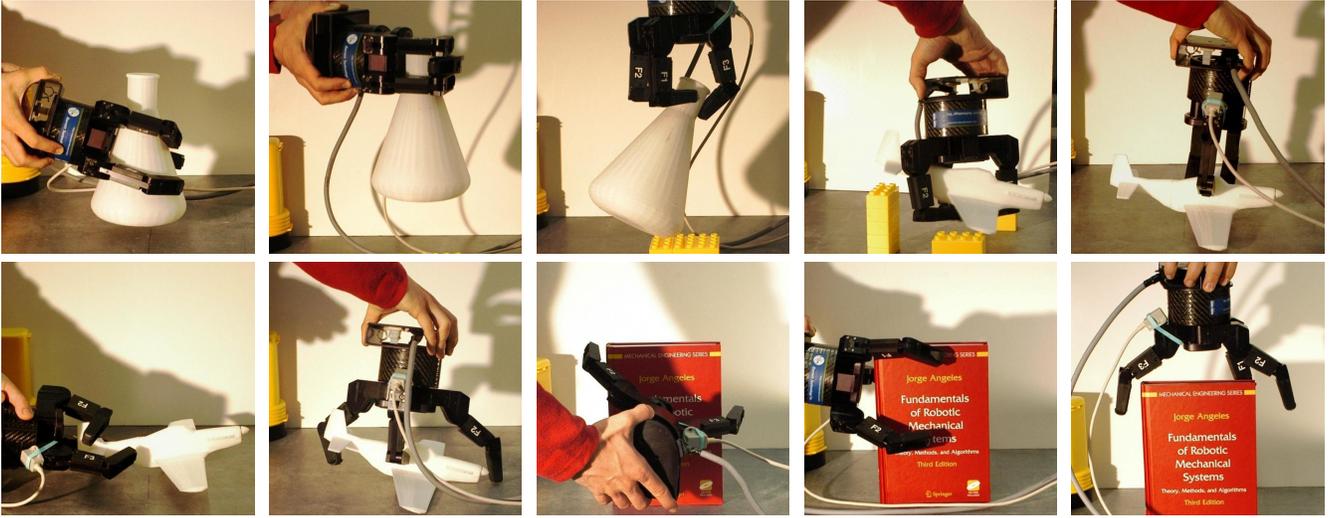


Fig. 9. Examples of interactive grasping tasks; each image shows the grasp found for a different approach direction or target object. In all cases the object was successfully grasped and lifted off the table.

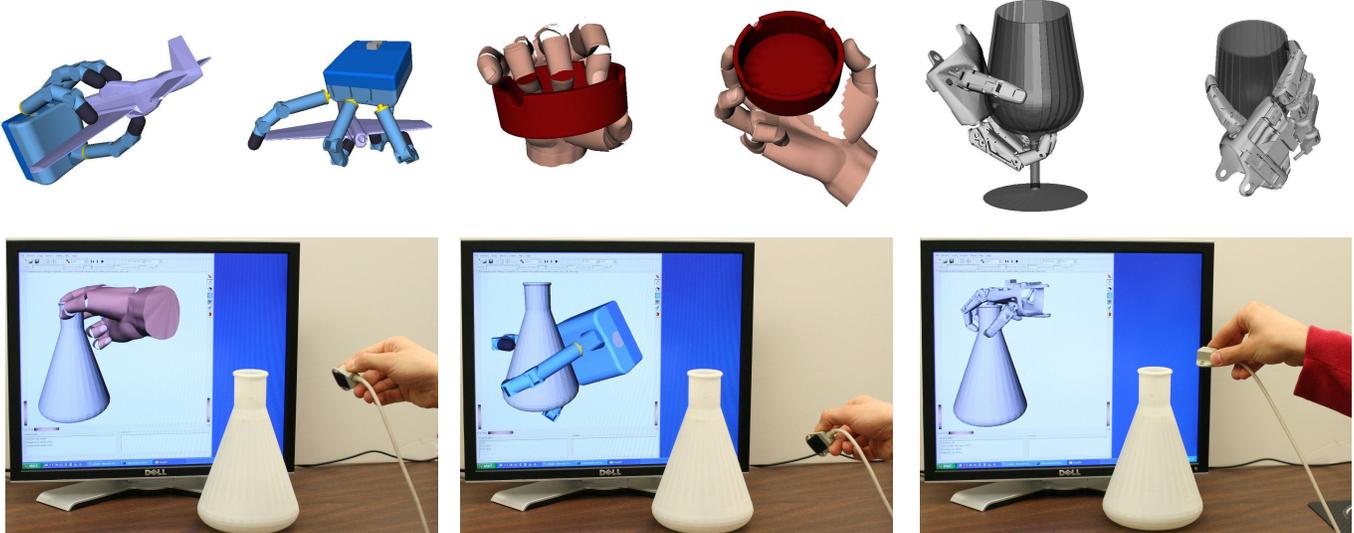


Fig. 10. Examples of grasping tasks executed in simulated environments. Bottom row images also show the user providing the approach direction via a magnetic tracker. All the presented grasps have form-closure.

- as the user approaches the target object, the grasp planner searches for a good grasp in a cone-shaped area around the given approach direction; when a solution is found, it is used to set the hand posture, allowing the user to react. If multiple solutions are found, the one that is closest to the current user approach direction currently is chosen for presentation (*i.e.*, the solution with the lowest values for the angular variables  $\theta$  and  $\phi$ ).
- the planner continuously attempts to improve the current result, by finding new grasps that either have a better quality metric or are closer to the current position established by the user.
- if the planner is unable to find a grasp in the current search area, or the user is not satisfied with the resulting hand posture, the user can reposition the hand and attempt to grasp a different part of the target object.
- if the user is satisfied with the hand posture, he/she continues along the current approach direction. As the real hand position approaches the target grasp, the fingers are gradually closed around the object. The user can therefore predict where the object will be touched and finally issue a "close all fingers" command which completes the grasping task.

### E. Results

Fig. 9 presents the application of our method using the BarrettHand in a real environment, while Fig. 10 shows interactive grasps performed in a simulated environment using the DLR hand, the Robonaut hand and the human hand model. In most cases, the images show only the final grasp applied by the user; due to space constraints we are unable to include images showing the evolution of the grasping task from approach direction, pre-grasp and final grasp. In order to better evaluate the

interactive nature of our application, a video showing a number of complete examples is available at <http://www.cs.columbia.edu/~cmatei/interactive>.

For any given grasping task, the exact computational effort required to find a stable grasp depends on the complexity of the hand and target object, as well as the approach direction chosen by the user. On average, the first stage of the grasp planning algorithm processes approx. 1000 hand postures per second, while the second testing phase, running in parallel, can evaluate approx. 20 candidate pre-grasps per second. In most cases, solution grasps are found at interactive rates: in the example presented in Fig. 8, the grasp planner found 8 stable grasps in 13.6 seconds of computation. These are representative numbers for the behavior of the system, which generally requires less than 2 seconds to find a solution grasp for a new approach direction. All of our tests were performed using a commodity desktop computer equipped with a 2.13GHz Intel Core2 CPU.

The ability of the system to allow for successful task completion in a short time is more difficult to quantify, as it also depends on how well the user reacts to the behavior of the automated components. All the results presented in Figs. 9 and 10, as well as in the video presented in Extension 1, were obtained at interactive rates, usually requiring between 5 and 15 seconds from first approach to final grasp execution. For the more difficult tasks, taking up to 30 seconds to complete, we found two main reasons that led to the increased execution time: either the planner repeatedly failed to find form-closure grasps for selected approach directions, or the human user could not interpret some of the finger postures selected by the planner and had to attempt different grasps. These cases represent a small minority of our tests and examples; however, the tests were performed by well-trained users familiar with the inner workings of the planning algorithm.

In other work, we have used this online grasp planner for biomimetic grasp planning. Our idea is a shared control paradigm where incomplete reach and grasp control information from the cortex is supplemented with control synthesized by the automatic grasp planning system described above. A key aspect of this task is the presence of on-line user input for hand posture, which will ultimately be obtained by identifying and extracting the relevant signals from brain activity. Our grasping system can combine partial or noisy user input and autonomous planning to enable the robot to perform stable grasping tasks. We have tested two applications of this algorithm, using data collected from both primate and human subjects during grasping, to demonstrate its ability to synthesize stable grasps using partial control input in real or near-real time [11].

## V. DATA DRIVEN GRASP PLANNING

The last application of hand posture subspaces that we explore is the idea of a data driven grasp planner. Dimensionality reduction allows pre-computation of many thousands of stable grasps on thousands of everyday objects. Using this data, we can create a database of known grasps. To grasp a novel object, we can index into a database of known 3D models and use a pre-computed grasp for a similarly shaped object to suggest a new grasp. We refer to this idea as data driven grasping. To support this, we have built the Columbia Grasp Data Base (CGDB) [19], a database of pre-computed grasps, which is described below. The database is freely available at [www.grasping.cs.columbia.edu](http://www.grasping.cs.columbia.edu) both as a PostgreSQL database and as flat text files.

Our primary interest is in using an object's 3D geometry as an index into the database. Given a new 3D object, we can find geometric neighbors in the database, and the accompanying stable grasps for these similar objects. If the number of objects to be grasped in the database is very large and comprehensive then robotic grasping becomes a pre-computed database lookup.

We are not aware of any previous attempt to construct a large scale grasp database, or of any commonly used benchmarks for evaluating robotic grasping. However, researchers have investigated grasp planning approaches that assume such a database already exists. Bowers and Lumia [4] collected grasps for a small number of planar objects and used fuzzy logic to extrapolate grasping rules. Morales *et al.* [33] used *GraspIt!* to compute offline grasps for a small database of graspable objects, and successfully identified and executed those grasps on real objects in a complex environment. Unlike the planner we present in this chapter, their approach requires an exact model of every possible graspable object.

Other researchers have experimented with different forms of precomputed grasp knowledge. Li and Pollard [26] collected a database of 17 hand poses, and used shape matching to match poses to graspable objects. Their work highlighted the difficulty of automatically generating grasps that are both stable and plausibly humanlike. Aleotti and Caselli demonstrated grasp synthesis using examples acquired with a dataglove [1]. Saxena *et al.* [40] generated 2D renderings of a large set of example objects, and learned a model-free mapping from images to graspable features. Their work uses a two fingered gripper and does not have an obvious extension to dexterous hands.

### A. Building a Grasp Database

Although the approach of building a large grasp database by direct data collection from human users has difficulty scaling to thousands of objects, the basic strategy of performing many grasps and recording their poses is still valid. However, it requires replacing human input with an automated procedure that does not need user attention. In our own previous work [32], [18] we have demonstrated the use of a grasp planner running in a simulation engine. Grasp planning can be considered an optimization task that can be performed in simulation without user supervision. However, even when simulations are performed on a powerful computer, the space of possible grasps is too large to sample directly. This is particularly true in the case of dexterous hands with many intrinsic degrees of freedom (DOFs).

### B. Enhanced Eigengrasp planner

In section II we presented a grasp planning algorithm that optimizes hand posture in a low dimensional eigengrasp space to find effective pre-grasp postures. These pre-grasps are then heuristically developed into stable, form closure grasps. To build our database of grasps, we apply this planner to a very large set of objects, and obtain multiple results for each object. We use a slightly modified multi-threaded version which takes advantage of multi-core architectures widely available today on commodity computers. A single parent thread searches the eigengrasp space for likely pre-grasps; for each pre-grasp position that crosses a quality threshold, a separate child thread is created to refine it. The child thread performs a quick local simulated annealing search with finer step values, attempting to develop the pre-grasp into form closure. If the resulting grasp indeed has form closure, the pre-grasp and grasp are both saved in the result list.

After creating a child thread for a pre-grasp state, the parent thread’s state generation function rejects states close to the child thread’s search area, forcing it to look elsewhere in the state space for new grasps. The planner can be restarted at any point by resetting the annealing temperature. The process continues until either the desired number of grasps are found, or a pre-set time limit is exceeded.

### C. Constructing the Database

A crucial aspect of the database construction is the running time required for each hand-object combination. As the planner has no intrinsic time limit, one has to be imposed based on the size of the object set and the computing power available. When building the database described in the next section, each execution of the planner ran until we found 15 form closure grasps of the target object. In general, this required about 10 minutes of run time. When dealing with large datasets and different hand models, a subset of objects will inevitably prove difficult to grasp using our algorithm. To prevent these from dominating the computation time we also set an upper time limit of 20 minutes per model.

The total number of planner executions used to build the database was approximately 22,000, as described in following section. The total computation time was approximately 1 month on 6 multi-core computers.

### D. 3D Models and Scaling

The first requirement for a grasp database is a set of objects to grasp. Rather than distributing a new set of 3D models, we chose to reuse the models from the Princeton Shape Benchmark (PSB) [42], which is already in common use in the shape matching community. We hope that a shared dataset will encourage increased collaboration between shape researchers and roboticists.

The disadvantage of this choice is that the PSB models were not originally selected with an eye towards robotic grasping, and so some of the models are not obvious choices for grasping experiments. For example, the database contains both airplanes and insects, all of which are outside the normal grasping range of a human-sized hand. We chose to treat all such “ungraspable” objects as toys, and rescaled them accordingly. The rescaling factor for each model, as compared with the “original” scale in the standard PSB, is included in the database.

Even with all of the models at graspable size, the issue of scale required further attention. Grasping is inherently scale-dependent, but most of the models in the PSB might plausibly exist anywhere within a small range of scales. This was particularly true for the models rescaled to “toy” size. To soften the impact of scale, we cloned each object at four distinct scales, 0.75, 1.0, 1.25 and 1.5, where 1.0 represents the rescaled size from above. As the PSB contains 1,814 distinct models, this left us with a total of 7,256 models to grasp.

For each model we store the radius of a ball that approximately contains the model. We use an approximately enclosing ball rather than an absolutely enclosing ball to avoid outlier sensitivity. We assume that all of the points on a model’s surface are normally distributed around some fixed radius from the center of mass. The fixed radius is simply the mean distance from the surface to the center of mass, and the “approximate radius” stored in the database is the mean distance to the surface plus two standard deviations of the distance to the center once the mean has been subtracted.

Along with shape and scale, the space of possible grasps is influenced by the frictional properties and deformability of both the hand and the object. Both in the construction the database and in our experiments we treated all models as being made of rigid plastic. The properties of the different hands are described below.

### E. Robotic Hands

Grasping is strongly hand-dependent, and so we need to specify the hands used in our database. For the first version of the database, we chose to focus on two hands; a human hand model in order to emphasize the “humanlike” nature of the grasp selection, and the three-fingered BarrettHand, which is ubiquitous in robotics research due to its durability and relatively low cost. The human hand model has 20 degrees of freedom. The BarrettHand has 4 degrees of freedom, plus a disengaging clutch mechanism which allows conformance even when the proximal link of a finger is blocked. Both models are available for download with *GraspIt!*.

Metal Barrett	25,585
Rubber Barrett	132,421
Human	80,731
<b>Total</b>	<b>238,737</b>

TABLE III  
PRECOMPUTED GRASPS FOR EACH HAND OVER 7,256 SCALED MODELS.

Frictional forces play an important role in grasping, and so we must specify the materials for each of our hands. There is no exact consensus on the friction coefficient of human skin and so we chose  $\mu = 1.0$  as a plausible value for the friction between the human hand and plastic [43]. The ability to create stable, encompassing grasps with subsets of fingers is also increased by using soft fingertips that deform during contact and apply a larger space of frictional forces and moments than their rigid counterparts. In order to take into account such effects, we use a fast analytical model for soft finger contacts that we have introduced in previous work [9]. The BarrettHand is made of metal, but can be coated with a higher friction material. We created two versions of the BarrettHand, one uncoated and one with rubberized fingers, and computed grasps for then independently, in effect giving us three hand models. For the metal BarrettHand we used  $\mu = 0.4$  and for the rubber coated version we used  $\mu = 1.0$ . As the kinematic models are identical, grasps computed for either BarrettHand model can be executed using the other, making it possible to evaluate the advantage afforded by using the higher friction material. We note that grasps from the regular hand can be assumed to be form closure for the rubberized hand as well, but that this guarantee does not hold in reverse.

#### F. Grasps and Pre-Grasps

Our grasp database is intended to be used in conjunction with *GraspIt!* or a similar grasp simulation tool. As such, we provide the necessary data to recreate each grasp, in the form of joint angles and hand position, and the contact points between hand and object, which can be used as a check to ensure that the grasp was simulated correctly. We also provide the two measures of grasp quality mentioned in Section V-C.

Each grasp entry consists of a *pre-grasp* and the final grasp pose. A pre-grasp is a pose from the instance before the hand contacts the object; it represents “pure” grasp information untainted by conformance to an exact object. Each of our pre-grasps lies within the two dimensional Eigengrasp subspace, as described in [10]. In contrast, the grasp poses represent final positions for form closure. This first version of our database contains 238,737 distinct form closure grasps, each with an associated pre-grasp. The breakdown of these grasps is shown in Table III.

#### G. Caveats

Since the grasps in the database were found using an automated planner, not all of the grasps are truly humanlike or reliable. There can be cases where a grasp satisfies our quality metrics, but would require a degree of precision that cannot be obtained in real-life execution. Recent work has underlined the fact that quality metrics used in simulation do not always translate to stable or human-like real-world grasps [2], [3]. Aside from the intrinsic limitations of grasp quality metrics, for which there is as of yet no firm consensus on which to use, our approach to grasp planning is purely geometric. This presents problems for objects that do not match our assumptions. For example, our assumption that all objects are rigid plastic results in geometrically correct but unrealistic grasps on objects such as flowers or leaves.

Furthermore, the lack of domain-specific knowledge means that some of our grasps are semantically incorrect, such as a mug grasped by placing the fingers inside, although they are still form closed.

Finally, all of our grasps were obtained from pre-grasps that sample a low-dimensional subspace of the hand DOF space. This is for the moment a necessary simplification, without which the planning problem for dexterous hands is intractable at this scale. While our choice of subspace is theoretically justified and shown to be effective [10], we cannot reasonably claim that the database covers the entire space of possible grasps. The choice of optimal subspace is one of our directions for future research.

## VI. DATABASED-BACKED GRASP PLANNING

One of our primary motivations of building a grasp database was to collect enough grasping data to build new grasp planners based on learning. In this section we present a grasp planner that uses a k-Nearest-Neighbors approach to find candidate grasps for a model not in the database. In general, the relation between hand pose and grasp quality for a given object is both nonlinear and discontinuous, and more sophisticated learning methods such as SVMs have so far been shown to work only for simple objects [35], [4]. We hope that the data we have collected will facilitate further research in this direction.

### A. Algorithm

Our grasp planning algorithm is based on the intuition that similar objects are likely to have similar grasps. Therefore, if we wish to grasp an object not in our database, it makes sense to look at the most similar objects that *are* in the database and to attempt to grasp the new object in the same way.

Given a model to grasp  $\alpha$ , we use a shape matching algorithm to find  $N = \{n_1 \dots n_k\}$ , the  $k$  models in the database most similar to  $\alpha$  under some shape similarity metric. In this study we use  $L^2$  distances between Zernike descriptors [34], which we have previously shown to be scalable to very large libraries of 3D models [16]. Zernike matching is scale-normalized, but as detailed in Section V-D, each PSB model exists in our database at 4 distinct scales. For each  $n_i$  we consider up to 2 models,  $n_i^<$ , the largest neighbor smaller than  $\alpha$  and  $n_i^>$ , the smallest neighbor larger than  $\alpha$ , using the scaled approximate radius described in Section V-D. In the case of  $\alpha$  smaller or larger than all 4 versions of the neighbor we only used one model for  $n_i$ .

We present our grasp planning algorithm here. For simplicity, we have ignored the issue of scale and treated each  $n_i$  as a single model.

---

#### Algorithm 1 DATABASE-BACKED GRASP PLANNING

---

**Require:** Model  $\alpha$  to grasp with hand  $H$ , using  $k$  neighbors and quality threshold  $\tau$ .

```

 $N \leftarrow \text{NEARESTNEIGHBORS}(\alpha, k)$ 
 $R \leftarrow \{\}$  {The resulting form closure grasps.}
for all  $n_i \in N$  do
  Align  $\alpha$  and  $n_i$  using PCA.
  Co-locate the centers of mass of  $\alpha$  and  $n_i$ .
   $G \leftarrow$  the precomputed pre-grasps on  $n_i$ .
  for all  $g \in G$  do
    Transform  $g$  to  $\alpha$ 's coordinate system.
    Place  $H$  in the position and configuration of  $g$ .
    repeat
      Move  $H$  backwards.
    until  $\text{NOCOLLISIONS}(\text{PALM}(H), \alpha)$ 
    for all  $finger \in \text{FINGERS}(H)$  do
      repeat
        Open  $finger$ .
      until  $\text{NOCOLLISIONS}(finger, \alpha)$ 
    end for
  repeat
    Move  $H$  forwards.
  until  $\text{INCONTACT}(H, \alpha)$ 
   $\text{CLOSEFINGERS}(H)$ 
  if  $\text{EPSILONQUALITY}(H, \alpha) > \tau$  then
     $R \leftarrow R \cup g$ 
  end if
end for
end for
return  $R$ 

```

---

The entire process, from shape matching through final output, takes approximately 20 seconds. To illustrate the behavior of this algorithm, we provide a number of examples in Fig. 11.

### B. Experiments

For our experiments, we removed each PSB model at scale 1.0 from the database one at a time and attempted to grasp it using only the known grasps from the remaining models. To isolate the effects of shape matching, we used three methods of choosing similar models. In each case we used  $k = 5$  neighbors for every model.

Our first method used the ground-truth labels provided with the PSB. For each model, we chose neighbors within the same shape category, starting with the finest categorization and moving up to coarser categories if fewer than  $k$  neighbors were available. Within the same category the choice of neighbors was arbitrary. We designated the chosen models as the ‘*PSB classes*’ neighbors. This method of indexing, while not usable for arbitrary unclassified models, approximates the performance of a theoretical ideal<sup>1</sup> shape matching algorithm that has perfect precision and recall over the PSB.

<sup>1</sup>Even with perfect precision and recall, this theoretical algorithm may not truly be ‘ideal’, as the categories in the PSB are semantic rather than purely geometric. Nevertheless, since shape matching algorithms are regularly evaluated using these categories as a ground truth, we adopt the same convention.



Fig. 11. Three example models and their grasps, using the database-backed planner with Zernike neighbors. For each model  $\alpha$  (left), the top row of images shows a neighbor  $n_k$  from the database, the value of  $k$ , and a pre-computed grasp on that neighbor. Directly below each neighbor is the same grasp executed on  $\alpha$ , along with its GWS epsilon quality measure.

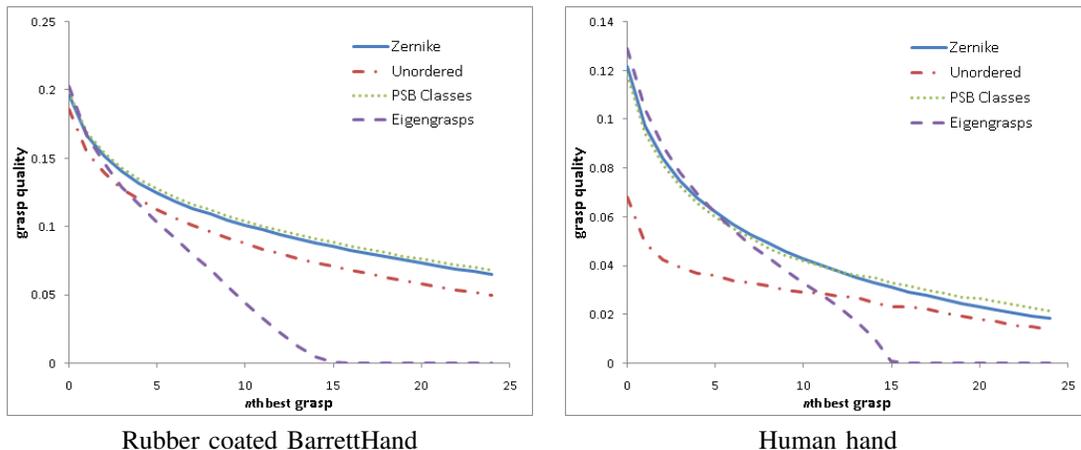


Fig. 12. The  $n$ th best grasp from database-backed grasping with 3 neighbor selection methods and from the eigengrasp planner, averaged over the 1,814 models in the database at scale 1.0. This figure is best viewed in color.

Our second method used  $L^2$  distances between Zernike descriptors [34]. For each model, we designated the  $k$  models with the smallest  $L^2$  distance in Zernike descriptor space as the ‘Zernike’ neighbors. These descriptors are computed on voxel grids and are quite robust, making them suitable for use in matching newly acquired objects into the database.

For our third method, we randomly selected  $k$  models from the database and designated them as ‘unordered’ neighbors. We emphasize that these are *not* random grasps in any sense;  $\alpha$  has been translated and axis-aligned with some model of a similar scale. Furthermore, the pre-grasps applied to it are pointed in the right direction, with joint angles drawn from a high quality eigengrasp subspace and known to produce form closure on another model with aligned principal axes. We therefore expect that some of the pregrasps taken from unordered neighbors will result in form closure grasps. Our aim in using the unordered neighbors is to isolate the performance gains based on shape matching while holding constant the performance due to the overall high quality of all of the grasps in the database.

We ran the experiment separately for each type of neighbor selection and averaged the grasp quality of the  $n$ th best grasp on each model over all 1,814 models in the database at scale 1.0. As mentioned in the introduction, we believe this to be one of the most comprehensive grasp planning experiments in the literature, as it consists of thousands of runs on a highly varying set of objects. We can analyze these results from a number of different perspectives: the absolute performance of database-backed grasp planning, the relative behavior of different neighbor selection methods and finally the performance compared to running the eigengrasp planner of Section V-B directly on the target object. Fig. 12 shows these results for the human hand and the BarrettHand.

Although Zernike descriptors do not have perfect precision and recall over the PSB, their performance for our grasp planner is nearly identical to the ground truth PSB classification. This is likely because the PSB classification is partially semantic, whereas our grasp quality measures are purely geometric; a ‘poor’ neighbor for semantic shape matching may still be geometrically close enough to share high quality grasps.

The performance of the ‘unordered’ neighbors is as expected; good, due to the general quality of grasps in the database and the axis-alignment between  $\alpha$  and the unordered neighbor, but not as good as the shape matching methods. Although the improvement due to shape matching is small for the BarrettHand, for the human hand the difference is quite significant. We attribute this difference to the many additional DOFs of the human hand, which creates a need for careful pre-grasping. The BarrettHand, with its 4 degrees of freedom, has a far simpler configuration space, and the importance of pre-grasping is correspondingly less.

Of special interest is the comparison between the database-backed methods and the eigengrasp planner. For the first few grasps, the performance of the shape matching methods is essentially identical to that of the eigengrasp planner. However, for subsequent grasps the quality quickly diverges, with the advantage going to the database-backed methods. This is even more impressive when we recall that the eigengrasp planner ran for approximately 10 minutes per model, whereas the database-backed planners ran for about 20 seconds. The database-backed approach can take advantage of pre-computed grasp data from multiple objects, essentially extracting the useful information obtained from several runs of the eigengrasp planner.

### C. Planning for Real Objects

The ultimate goal of our database-backed planner is to grasp new objects that are not in our database, using sensor data. Realistically, sensors will only be able to acquire partial models for novel objects, which presents a problem for many planning algorithms. A data-driven planner, however, will be able to operate even on such incomplete data as long as the underlying 3D model database has been indexed to allow for partial matching.

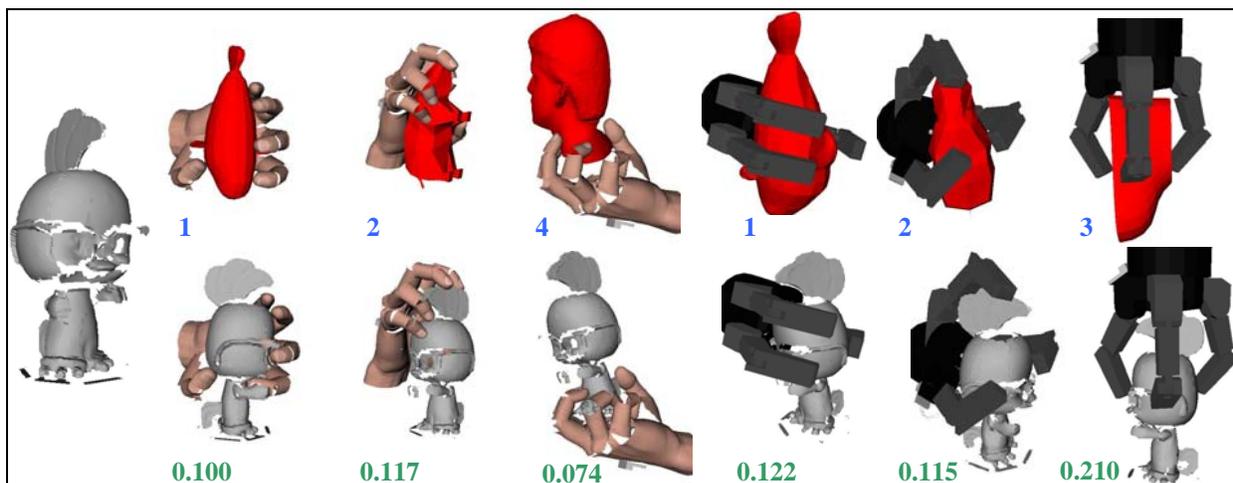


Fig. 13. Some of the grasps planned for an acquired object with holes and occlusions, using the database-backed planner with Zernike neighbors. In total 88 form closure grasps were found for the human hand and 112 for the BarrettHand.

While a comprehensive evaluation using such acquired data is beyond the scope of this chapter, we present here our preliminary results. Using a commodity desktop 3D scanner, we acquired a range image of a plastic toy. Due to the intrinsic limitations of the acquisition method, the range image was both noisy and incomplete, with several occlusions. We computed the Zernike descriptor of the scan, found the nearest neighbors in the database, and ran the planner as before for both the BarrettHand and the human hand. We found 88 form closure grasps using the human hand and 112 form closure grasps using the BarrettHand. Some of these results are shown in Fig. 13. For an in-depth treatment of data-driven grasping with partial sensor data, refer to [17].

## VII. CONCLUSION AND FUTURE WORK

Our online eigengrasp planner and database planner each have limitations. In both cases, full knowledge of object geometry is needed to either compute a stable grasp or to match an object's shape in the database. We have performed initial experiments with the database planner to use only partial shape information to find the best grasp [20], [17].

In the case of the eigengrasp planner, choosing the correct couplings between the joints is somewhat ad-hoc. For anthropomorphic hands such as the Robonaut hand, these couplings can be directly transferred from human data observations. In non-anthropomorphic hands, determining the correct set of eigenvectors is a subject of further research.

For the database planner, the approach to grasping is entirely geometric. It leaves out important considerations such as non-uniform mass distribution in computing a stable grasp. It may be possible to add this information into the database, but it would require a scale-up of the database in size.

Overall, we have found that using these low-dimensional subspace ideas has been extremely fruitful in making robotic grasping systems that actually work in the physical world. We hope to push these ideas beyond static grasping and into manipulation in the future.

## ACKNOWLEDGMENT

The authors would like to thank Hao Dang for his help in building the Columbia Grasp Database.

## REFERENCES

- [1] J. Aleotti and S. Caselli. Robot grasp synthesis from virtual demonstration and topology-preserving environment reconstruction. In *Intl. Conf. on Intelligent Robots and Systems*, 2007.
- [2] R. Balasubramanian, L. Xu, P. Brook, J. R. Smith, and Y. Matsuoka. Human-guided grasp measures improve grasp robustness on physical robot. In *IEEE Intl. Conf. on Robotics and Automation*, pages 2294–2301, 2010.
- [3] R. Balasubramanian, L. Xu, P. Brook, J. R. Smith, and Y. Matsuoka. Physical human interactive guidance: Identifying grasping principles from human-planned grasps. *IEEE Trans. on Robotics*, 2012 (in press).
- [4] David L. Bowers and Ron Lumia. Manipulation of unmodeled objects using intelligent grasping schemes. *Transactions on Fuzzy Systems*, 11(3), 2003.
- [5] C. Brown and H. Asada. Inter-finger coordination and postural synergies in robot hands via mechanical implementation of principal components analysis. In *IEEE-RAS Intl. Conf. on Intelligent Robots and Systems*, pages 2877–2882, 2007.
- [6] J. Butterfass, G. Hirzinger, S. Knoch, and H. Liu. DLR's articulated hand, part I: Hard- and software architecture. In *IEEE Intl. Conf. on Robotics and Automation*, pages 2081–2086, 1998.
- [7] M. C. Carrozza, G. Cappiello, S. Micera, B. B. Edin, L. Beccai, and C. Cipriani. Design of a cybernetic hand for perception and action. *Biol. Cybern.*, 95(6):629–644, 2006.

- [8] V. C. K. Cheung, A. d'Avella, M. C. Tresch, and E. Bizzi. Central and sensory contributions to the activation and organization of muscle synergies during natural motor behaviors. *Journal of Neuroscience*, 25(27):6419–6434, 2005.
- [9] M. Ciocarlie, C. Lackner, and P. Allen. Soft finger model with adaptive contact geometry for grasping and manipulation tasks. In *Joint Eurohaptics Conf. and IEEE Symp. on Haptic Interfaces*, 2007.
- [10] Matei Ciocarlie, Corey Goldfeder, and Peter K. Allen. Dimensionality reduction for hand-independent dexterous robotic grasping. In *International Conference on Intelligent Robots and Systems*, 2007.
- [11] Matei T. Ciocarlie, Samuel T. Clanton, M. Chance Spalding, and Peter K. Allen. Biomimetic grasp planning for cortical control of a robotic hand. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.
- [12] M. R. Cutkosky. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on Robotics and Automation*, 5:269–279, 1989.
- [13] A. Dollar and R. Howe. Simple, robust autonomous grasping in unstructured environments. In *IEEE Intl. Conf. on Robotics and Automation*, 2007.
- [14] C. Ferrari and J. Canny. Planning optimal grasps. In *IEEE Intl. Conf. on Robotics and Automation*, pages 2290–2295, 1992.
- [15] A. Fod, M.J. Mataric, and O.C. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12:39–54(16), 2002.
- [16] Corey Goldfeder and Peter K. Allen. Autotagging to improve text search for 3d models. In *Joint Conference on Digital Libraries*, 2008.
- [17] Corey Goldfeder and Peter K. Allen. Data-driven grasping. *Autonomous Robots*, 31(1), 2011.
- [18] Corey Goldfeder, Peter K. Allen, Claire Lackner, and Raphael Pelossof. Grasp planning via decomposition trees. In *Intl. Conf. on Robotics and Automation*, 2007.
- [19] Corey Goldfeder, Matei Ciocarlie, Hao Dang, and Peter K. Allen. The Columbia grasp database. In *IEEE International Conference on Robotics and Automation*, pages 1710–1716, May 2009.
- [20] Corey Goldfeder, Matei Ciocarlie, Jaime Peretzman, Hao Dang, and Peter K. Allen. Data-driven grasping with partial sensor data. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1278–1284, October 2009.
- [21] C. Gosselin, F. Pelletier, and T. Laliberte. An anthropomorphic underactuated robotic hand with 15 Dofs and a single actuator. *IEEE Intl. Conf. on Robotics and Automation*, 2008.
- [22] R.D. Howe and M.R. Cutkosky. Practical force-motion models for sliding manipulation. *Intl. J. of Robotics Research*, 15(6):557–572, 1996.
- [23] Thea Iberall. Human prehension and dexterous robot hands. *International Journal of Robotics Research*, 16:285–299, 1997.
- [24] L. Ingber. Very fast simulated re-annealing. *J. Mathl. Comput. Modelling*, 12(8):967–973, December 1989.
- [25] D. Kragic, A. Miller, and P. Allen. Real-time tracking meets online planning. In *IEEE Intl. Conf. on Robotics and Automation*, Seoul, 2001.
- [26] Ying Li and Nancy S. Pollard. A shape matching algorithm for synthesizing humanlike enveloping grasps. In *Humanoid Robots*, 2005.
- [27] C. S. Lovchik and M. A. Diftler. The robonaut hand: A dextrous robot hand for space. In *IEEE Intl. Conf. on Robotics and Automation*, pages 907–912, 1998.
- [28] C. R. Mason, J. E. Gomez, and T. J. Ebner. Hand synergies during reach-to-grasp. *Journal of Neurophysiology*, 86:2896–2910, 2001.
- [29] M. Mason and K. Salisbury. *Robot hands and the mechanics of manipulation*. MIT Press, 1985.
- [30] A. Miller and P. Allen. Examples of 3-d grasp quality computations. In *IEEE Intl. Conf. on Robotics and Automation*, pages 1240–1246, Detroit, MI, 1999.
- [31] Andrew Miller and Peter K. Allen. Graspit!: a versatile simulator for robotic grasping. *IEEE Robotics and Automation Magazine*, 11(4):110–122, 2004.
- [32] Andrew T. Miller, S. Knoop, H. I. Christensen, and Peter K. Allen. Automatic grasp planning using shape primitives. In *Intl. Conf. on Robotics and Automation*, 2003.
- [33] Antonio Morales, Tamim Asfour, Pedram Azad, Steffen Knoop, and Rudiger Dillmann. Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands. In *Intl. Conf. on Intelligent Robots and Systems*, 2006.
- [34] Marcin Novotni and Reinhard Klein. 3D zernike descriptors for content based shape retrieval. In *Solid Modeling and Applications*, June 2003.
- [35] Rafael Pelossof, Andrew Miller, Peter Allen, and Tony Jebara. An SVM learning approach to robotic grasping. In *Intl. Conf. on Robotics and Automation*, 2004.
- [36] M. Santello, M. Flanders, and J. F. Soechting. Postural hand synergies for tool use. *Journal of Neuroscience*, 18(23):10105–10115, 1998.
- [37] M. Santello, M. Flanders, and J. F. Soechting. Patterns of hand motion during grasping and the influence of sensory guidance. *J. of Neuroscience*, 22:1426–1435, 2002.
- [38] M. Santello and J. F. Soechting. Matching object size by controlling finger span and hand shape. *Somatosensory and Motor Research*, 14:203–212, 1997.
- [39] M. Santello and J. F. Soechting. Gradual molding of the hand to object contours. *J. of Neurophysiology*, 79:1307–1320, 1998.
- [40] Ashutosh Saxena, Justin Driemeyer, and Andrew. Robotic grasping of novel objects using vision. *Intl. Journal of Robotics Research*, 27(2), 2008.
- [41] M. H. Schieber and M. Santello. Hand function: Neural control and peripheral limits to performance. (invited review). *J. of Applied Physiology*, 96:2293–2300, 2004.
- [42] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Shape Modeling and Applications*, 2004.
- [43] Raja K. Sivamani, Jack Goodman, Norm V. Gitis, and Howard I. Maibach. Coefficient of friction: tribological studies in man an overview. *Skin Research and Technology*, 9(3), 2003.
- [44] D. Taylor, S. H. Tillery, and A. Schwartz. Information conveyed through brain control: Cursor versus robot. *IEEE Trans. Neural Systems Rehab Eng.*, 1(2):195–199, 2003.
- [45] D. M. Taylor, S. H. Tillery, and A. B. Schwartz. Direct cortical control of 3D neuroprosthetic devices. *Science*, 296(5574):1829–1832, 2002.
- [46] P. H. Thakur, A. J. Bastian, and S. Hsiao. Multidigit movement synergies of the human hand in an unconstrained haptic exploration task. *Journal of Neuroscience*, 28(6):1271–1281, 2008.
- [47] E. Todorov and Z. Ghahramani. Analysis of the synergies underlying complex hand manipulation. In *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4637–4640, 2004.
- [48] A. Tsoli and O. C. Jenkins. 2d subspaces for user-driven robot grasping. In *Robotics, Science and Systems Conference: Workshop on Robot Manipulation*, 2007.
- [49] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.
- [50] M. Zecca, S. Micera, M. C. Carrozza, and P. Dario. Control of multifunctional prosthetic hands by processing the electromyographic signal. *Critical Reviews in Biomedical Engineering*, 30:459–485, 2002.