A Framework for Optimal Grasp Contact Planning

Kaiyu Hang, Johannes A. Stork, Nancy S. Pollard, and Danica Kragic

***Abstract*—We consider the problem of finding grasp contacts that are *optimal* under a given grasp quality function on arbitrary objects. Our approach formulates a framework for contact-level grasping as a path finding problem in the space of *supercontact* grasps. The initial supercontact grasp contains all grasps and in each step along a path grasps are removed. For this, we introduce and formally characterize search space structure and cost functions under which minimal cost paths correspond to optimal grasps. Our formulation avoids expensive exhaustive search and reduces computational cost by several orders of magnitude. We present admissible heuristic functions and exploit approximate heuristic search to further reduce the computational cost while maintaining *bounded suboptimality* for resulting grasps. We exemplify our formulation with point-contact grasping for which we define domain specific heuristics and demonstrate optimality and bounded suboptimality by comparing against exhaustive and uniform cost search on example objects. Furthermore, we explain how to restrict the search graph to satisfy grasp constraints for modeling hand kinematics. We also analyze our algorithm empirically in terms of created and visited search states and resultant effective branching factor.**

***Index Terms*—Grasping, dexterous manipulation, multifingered hands, contact modeling.**

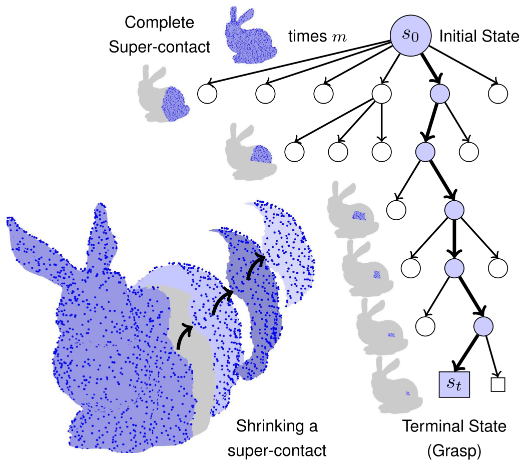


Fig. 1. Our approach is based on reducing optimal grasping to finding a minimal cost path. Starting with complete *super-contacts* that cover the whole object, we step-by-step remove contacts until we reach m-contact grasps. Each state is a combination of m super-contacts and each arc shrinks the included super-contacts. The minimal cost path ends at the optimal grasp.

I. Introduction

CONTACTS are the most fundamental building blocks of grasps and determine stability and utility of a grasp. Finding good grasp contacts is a key challenge in robotic grasping [1] and fixture layout design for industrial automation [2]. The main difficulty consists in deciding where on the object's surface contacts should be placed and which combinations of contacts result in a reliable grasp. Grasp reliability is commonly characterized by wrench space analysis [1], [3]–[5] and grasp contacts are found by some form of optimization [1], [6]–[8], heuristic [3], [9], or data-driven approach [10].

While analytical methods for optimal quality grasps exist for primitive objects [11], optimal grasping on arbitrary objects still requires enumerating all contact combinations exhaustively [12]. However, even for less than half of the contacts shown in Fig. 1, analyzing all three-finger grasps requires thousands of minutes on current non-specialized hardware. This effort increases exponentially with the number of fingers.

In this work, we offer an *general* formulation of the optimal grasping problem that allows *efficiently* identifying *optimal* and *bounded suboptimal* grasps while avoiding computation on each possible grasp. In practice our algorithm can reduce computational cost by several orders of magnitude. Intuitively, we start with the set of all possible grasps and step-by-step remove grasps until reaching the optimal grasp. For this, we represent a set of grasps by a combination of *super-contacts* which contain several contacts at once. Provided that the grasp quality function can be applied to super-contact grasps, we show that this approach corresponds to a *path finding problem*. As illustrated in Fig. 1, each state corresponds to a super-contact grasp and each arc removes contacts from a super-contact until terminal states are left with only one contact for each finger. We set the cost of an arc to represent the loss in grasp quality between connected states which means that minimal cost paths terminate in grasps with maximal quality.

We contribute by (1) formally characterizing search graph structure and grasp quality functions for reducing the optimal grasping problem to a path finding problem. Further, we (2) introduce a family of admissible heuristic functions for efficient and approximative heuristic search. For exemplary evaluation with point-contact grasping, we (3) prove that the popular Ferrari-Canny quality Q_1 [5] is compatible to our formulation, and (4) define a family of domain specific successor

Manuscript received September 10, 2016; accepted December 14, 2016. Date of publication January 10, 2017; date of current version February 2, 2017. This paper was recommended for publication by Associate Editor Y. Zheng and Editor H. Ding upon evaluation of the reviewers' comments. This work was supported in part by the ERC FLEXBOT, Knut and Alice Wallenberg IPSYS and in part by the NSF Award IIS-1637853. (*Kaiyu Hang and Johannes A. Stork contributed equally to this work.*)

K. Hang, J. A. Stork, and D. Kragic are with the Robotics, Perception, and Learning Lab, KTH Royal Institute of Technology, Stockholm 11428, Sweden (e-mail: kaiyuh@kth.se; jastork@kth.se; dani@kth.se).

N. S. Pollard is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA (e-mail: nsp@cs.cmu.edu).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LRA.2017.2651381

functions and heuristics. Finally, (5) we show incorporation of grasping-related constraints, e.g., hand kinematics, into the construction of the search graph. To the best of our knowledge, this is the first work that provides an efficient *complete and optimal* algorithm for finding *optimal* grasp contacts on arbitrary objects that allows choosing an ε *sub-optimality bound* for grasp quality.

II. Related Work

The problem of finding grasp contacts on an object's surface is addressed by a host of diverse approaches as described in overviews by Sahbani *et al.*. [8], Bohg *et al.* [10], Roa and Suárez [4], and most relevant to our work Bicchi and Kumar [1]. While most algorithms listed in the works above can result in high-quality grasps under favorable conditions, none guarantees optimality or bounded sub-optimality with reasonable computational effort. On the contrary, the most popular grasping approaches are based on *ex post* analysis of sampled contacts [3], [13], [14] and operate with simplified object models [11], [15]–[18]. Our only assumptions are a finite contact space and a quality function that allows sets of contacts per finger. The common quality function that models wrench resistance for frictional hard-finger contacts [3], [5] is compliant with our formulation and we use it in our experiments.

We argue that identifying optimal grasp contacts is useful not only for grasping, but also for benchmarking, analyzing objects, designing fixtures, and for providing training data for learning. Approaches that aim for optimal grasps often improve grasps iteratively and exploiting contact neighborhoods but fail to provide optimality guarantees [6], [17]–[19]. In contrast, we proceed in a top-down fashion refining super-contacts which initially representing all grasps—similar to the concept of *Object Wrench Space* [20].

In this, our search-based approach is conceptually related to the branch-and-bound algorithm for optimal grasping of Watanabe and Yoshikawa [12]. Both algorithms process a discrete set of contacts and repeatedly eliminate low-quality solutions. However, while Watanabe and Yoshikawa employ problem relaxation and exploit bounds on subproblem solutions to exclude suboptimal grasps from further consideration, we define admissible heuristics. Instead of comparing subproblem solutions for each candidate grasp, we construct grasps by reducing each finger's contact options step-by-step. In experiments, we aim for maximally resilient force-closure grasps while Watanabe and Yoshikawa want to satisfy an external force set. Both algorithms are complete and identify optimal grasps, but our algorithm allows trading efficiency for bounded sub-optimality and we show satisfaction of additional grasp constraints in form of hand kinematics.

III. Optimal Grasping as Path Finding

We formulate contact-based grasping as a path finding problem that can be solved using well-known heuristic search algorithms. For this, we formally define the *optimal grasping problem over a set of contacts* in Section III-A and present fundamentals and algorithms for optimal and bounded sub-optimal heuristic search in Section III-B. In Section III-C we introduce the concept of *super-contact* grasps which we use to reduce the optimal grasping problem to a *family of path finding problems*. By characterizing grasp quality functions and search graphs, we prove that minimal cost paths correspond to optimal grasps. The *family of consistent or admissible heuristic functions* that we introduce in Section III-D provides the basis for bounded sub-optimality results.

A. Optimal Grasping Problem and Grasp Quality

We consider grasping problems that are based on a finite set of suitable contacts on the object's surface $C = \{c_1, c_2, \dots c_k\}$ and a grasp quality function $q\colon 2^C \to \mathbb{R}$, where 2^C is the power set of C. For the set C we can imagine point contacts consisting of positions and surface normals for a hard finger model with Coulomb friction [6] or surface contacts defined by overlapping surface patches [21]. A m-contact grasp $\mathbf{g}$ consists of a tuple of m contacts from C, denoted as

$$\mathbf{g} = (c_1, c_2, \dots c_m), \tag{1}$$

where $c_i \in C$.

A grasp quality function determines how stable or reliable a grasp is and assigns higher quality values to better grasps. For our approach, we are only interested in grasp quality functions that do not increase when contacts are removed.

Definition 3.1: A grasp quality function $q\colon 2^C \to \mathbb{R}$ is monotone if grasp quality does not increase when the set of contacts is reduced,

$$\forall \mathbf{g}' \subseteq \mathbf{g} : q(\mathbf{g}') \leq q(\mathbf{g}), \tag{2}$$

where $\mathbf{g}, \mathbf{g}' \in 2^C$.

In the following, we consider all grasps over C of size m as possible solutions and compare them solely based on their q. In this context, we are interested in the best possible grasp which leads to the class of grasping problems that is formalized below.

Definition 3.2: An optimal grasping problem (OGP) is a tuple $\langle C, q, m\rangle$ where C and q as introduced above and $m > 1$ is the number of sought contacts. A solution is a grasp $\mathbf{g}^* \in C^m$ with maximum quality $q(\mathbf{g}^*)$ of all m-contact grasps over C.

An OGP is a difficult combinatorial optimization problem with a large and unstructured search space of $|C|^m$ states. For this type of problem stochastic optimization [6] and branch-and-bound [12] methods have been designed. The first type cannot guarantee quality bounds while the second has to solve relaxed problems for each possible grasp. In the following section, we introduce our novel top-down method for addressing OGPs which instead constructs optimal solutions by iteratively constraining contact options for each finger.

B. Heuristic Search With Bounded Sub-Optimality

When formulating a path finding problem (PFP), we provide a locally finite directed graph $G = (S, E)$ with a set of nodes or search states $S = \{s_1, s_2, \dots s_n\}$ and a set of edges or arcs $E = \{(s_i, s_j) \mid s_i, s_j \in S, s_j \in \Gamma(s_i)\}$. Arcs are defined by a successor mapping into the power set of states, $\Gamma : S \to 2^S$. While G specifies the problem domain, a PFP instance additionally consists of an initial state $s_0 \in S$, a set of goal or terminal states $S_T \subseteq S$, and a cost or distance function $d : E \to \mathbb{R}^+$. A solution to a PFP consists in a path from the initial state to a terminal state, written as $\pi = (s_1, s_2, \ldots, s_n)$, where $(s_i, s_{i+1}) \in E$ for all steps. The cost of a (partial) solution is computed along the path and referred to by the last state, $g(s_n) = \sum_{i=1}^{n-1} d(s_i, s_{i+1})$.

Definition 3.3: A path finding problem (PFP) is a tuple $\langle G, d, s_0, S_T \rangle$ with elements as introduced above. An optimal solution is a path $\pi^* = (s_0, \ldots, s_t)$ where $s_t \in S_T$ with minimal integral cost.

To construct a solution, a search algorithm begins with the initial state s_0 and applies the successor mapping to explore the graph until a terminal state $s \in S_T$ is encountered. Heuristic search algorithms attempt to improve *average case* efficiency by estimating the remaining cost to a terminal state by a heuristic function $h : S \to \mathbb{R}^+$. The A* SEARCH algorithm employs a combined cost function,

$$f(s) = g(s) + h(s), \tag{3}$$

and terminates with the optimal solution if the heuristic function is a lower bound of the true minimal cost [22]. A* SEARCH is also optimally efficient compared to other algorithms provided with the same heuristic function [23].

For optimality A* SEARCH must consider all equally optimal partial solutions. However, using bounded relaxation we can increase efficiency at the expense of optimality if we instead accept solutions with bounded sub-optimality. We can turn A* SEARCH into such an approximate algorithm by inflating the heuristic estimate (i.e. WA*)[24]–[26],

$$f(s) = g(s) + (1 + \varepsilon)\, h(s), \tag{4}$$

and guarantee that the found solution does not exceed the optimal cost by a factor larger than $(1 + \varepsilon)$ for $\varepsilon > 0$. This relaxation quickly directs the search into a more promising direction [26].

C. Path Finding Problems for Optimal Grasping

We reduce an optimal grasping problem $\langle C, q, m \rangle$ (see Def. 3.2) to a path finding problem $\langle G, d, s_0, S_T \rangle$ (see Def. 3.3) by defining a search graph G with cost function d and show that we can interpret the terminal state of a minimal cost path π^* as the sought optimal grasp $\mathbf{g}^*$. For this, we introduce the concepts of super-contacts and super-contact grasps.

Definition 3.4: If C is a set of contacts, then $\mathbf{c} \subseteq C$ with $|\mathbf{c}| > 0$ is a super-contact and $\mathbf{C} = 2^C \setminus \varnothing$ is the super-contact set of C. A m-contact super-contact grasp

$$\mathbf{s} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m) \tag{5}$$

consists of one super-contact $\mathbf{c}_i \in \mathbf{C}$ for each of the $m > 1$ contacts.

We define the state space S as all super-contact grasps based on the contact set C. The initial state s_0 has the maximal super-contact at each position $s_0 = C^m$, while each of the terminal states $s \in S_T$ consist of minimal super-contacts $S_T = \{(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m) : \mathbf{c}_i \in \mathbf{C}, |\mathbf{c}_i| = 1\}$. Consequently, a solution path $\pi = (s_0, \ldots, s_t)$ leads from m maximal super-contacts to

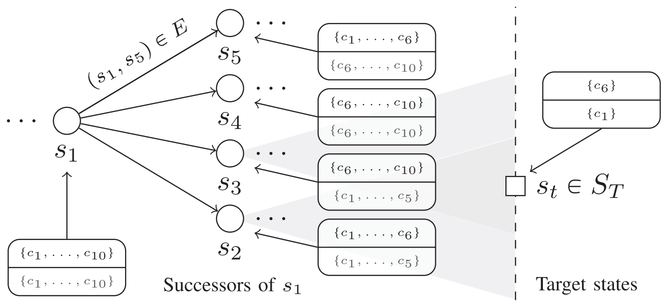


Fig. 2. A reducing and preserving successor function Γ defines a search graph based on super-contacts s_i. The grasp set $\mathbf{G}(s_1) = \{c_1, \ldots, c_{10}\}^2$ is represented by all states $s_i \in \Gamma(s_1)$ together where the grasp sets of successor states of s_1 are strictly smaller than $\mathbf{G}(s_1)$. Since the grasp (c_6, c_1) is represented by two successors, G is not necessarily a tree. However, G can be constructed as a tree if the partitions do not overlap. For this figure $m = 2$.

a state $s_t \in S_T$ that is equivalent to a grasp of m single contacts $\mathbf{g} = (c_1, c_2, \ldots, c_m)$ as introduced in Section III-A. In this context, we use the notation s and $\mathbf{s}$ interchangeably.

Each super-contact grasp $\mathbf{s} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m)$ represents a set of grasps $\mathbf{G}(\mathbf{s})$ where each contained grasp $\mathbf{g} = (c_1, c_2, \ldots, c_m) \in \mathbf{G}(\mathbf{s})$ draws one contact c_i from each super-contact $\mathbf{c}_i$. When defining our successor function Γ, we *preserve* the grasp set $\mathbf{G}(s)$ represented by a state while at the same time *reducing* at least one super-contact $\mathbf{c}_i$ along each arc in E. This means that the collection of all successors $\Gamma(s)$ represents the same grasp set as their single common predecessor s but each single successors s' represents strictly less grasps.

Definition 3.5: The successor function Γ reduces grasp sets if for all states $s \in S$ and $s' \in \Gamma(s)$,

$$\mathbf{G}(s') \subsetneq \mathbf{G}(s), \tag{6}$$

and preserves the grasp sets $\mathbf{G}(s)$ if for all states $s \in S$ and $s' \in \Gamma(s)$

$$\forall \mathbf{g} \in \mathbf{G}(s)\, \exists s' \in \Gamma(s) : \mathbf{g} \in \mathbf{G}(s'), \tag{7}$$

where $\mathbf{g}$ is a grasp as introduced in Section III-A.

As Fig. 2 shows, a reducing and preserving Γ does also not necessarily define G as a tree since two different parent states with overlapping super-contacts allow for identical successor states. Nevertheless, a reducing and preserving successor function Γ induces pairwise relationships for super-contacts $\mathbf{c}_i$ and $\mathbf{c}'_i$ of directly connected states $(s, s') \in E$,

$$\forall i : \mathbf{c}'_i \subseteq \mathbf{c}_i \qquad \text{and} \qquad \exists i : \mathbf{c}'_i \subsetneq \mathbf{c}_i, \tag{8}$$

where $i \in \{1, 2, \ldots, m\}$. This allows us to define the cost function d based on the grasp qualities $q(s)$ and $q(s')$ of the connected states $(s, s') \in E$. We lift the grasp quality function q to super-contacts,

$$q(s) = q(\mathbf{c}_1 \cup \mathbf{c}_2 \cup \ldots \cup \mathbf{c}_m) \tag{9}$$

where $s = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m)$ and we define the arc cost as

$$d(s, s') = q(s) - q(s'). \tag{10}$$

The expression in Eq. (10) is the loss in quality associated with reducing super-contacts along the arc. Consequently, the best successor state for s is the state with maximal grasp quality from $\Gamma(s)$ if q is monotone as defined in Def. 3.1.

Finally, given a minimal cost solution to the PFP $\pi^* = (s_0, \ldots, s_t)$ with $s_t \in S_T$, we interpret the last state's grasp as the final result $\mathbf{g}^*$. In the following, we show that this step yields an optimal solution to the original OGP.

Lemma 3.1: In the PFP $\langle G, d, s_0, S_T \rangle$ as introduced in this section, all grasps from the contact space $\mathbf{g} \in C^m$ are possible as final results.

Proof: All grasps from the contact space are contained in the grasp set of the initial node $\mathbf{G}(s_0)$ as well as in the set of terminal states S_T. Since the successor function is preserving, each grasp $\mathbf{g} \in \mathbf{G}(s_0)$ is represented by some state in each depth-level of the search graph. Since the successor function is reducing all nodes in S_T are finally reached. ■

This result shows that a search algorithm as described in Section III-B (e.g. A* SEARCH) can produce a solution path π^* with minimal cost that could terminate at any m-contact grasp over the contact set $\mathbf{g} \in C^m$. It remains to show that the solution path π^* provides the correct optimal grasp $\mathbf{g}^*$.

Lemma 3.2: Given a minimal cost solution $\pi^* = (s_0, \ldots, s_t)$, the grasp s_t is optimal.

Proof: If π^* is the minimal cost path to a state in S_T, then $g(s_t)$ is minimal. Therefore $q(s_0) - q(s_t)$ is minimal because all intermediate terms cancel. Since $q(s_0)$ is fixed and equal for all paths, $q(s_t)$ must be maximal. ■

Since the solution space of our PFP is complete and any minimal cost path represents an optimal grasp, we can state our main result.

Theorem 3.1: We can reduce the optimal grasping problem $\langle C, q, m \rangle$ to a path finding problem $\langle G, d, s_0, S_T \rangle$ as introduced in this section.

Proof: The result follows from Lemma 3.1 and 3.2 and the fact that C is finite and therefore G is finite. ■

D. Path Finding Heuristics for Optimal Grasping

The search graph $G = (S, E)$ introduced for the OGP in Section III-C has a large state space, $|S| \gg |C^m|$, that contains all possible super-contact grasps and we expect that many super-contact grasps have similar grasp quality as large contact sets provide for redundancy. This makes finding the minimal cost path π^* computationally expensive. However, often a sub-optimal grasp is sufficient and more desirable if it can be obtained at significantly lower computational cost and still exceeds a quality threshold. Approximate heuristic search as described in Section III-B allows for such a tradeoff.

For estimating the remaining cost to a terminal state, we introduce a family of admissible heuristic functions $h : S \to \mathbb{R}$ that are based on a reducing and preserving successor function Γ_h. We define the heuristic estimate $h(s)$ as the minimal quality difference between the state s and its children $s' \in \Gamma_h(s)$ with reduced super-contacts,

$$h(s) = \min_{s' \in \Gamma_h(s)} q(s) - q(s') = q(s) - \max_{s' \in \Gamma_h(s)} q(s'). \quad (11)$$

In Section IV-A, we define a *domain specific* Γ_h that generates a search tree, for which A* SEARCH with an *admissible* heuristic is optimal (i.e. expands states at most once) and result with the shortest path [25].

Theorem 3.2: If Γ_h is reducing and preserving, then h from Eq. (11) is (a) admissible, and (b) if $\Gamma = \Gamma_h$, consistent.

Proof: For (a), we show that $h(s) \leq h^*(s), \forall s \in S$,

$$q(s) - \max_{s' \in \Gamma_h(s)} q(s') \leq q(s) - q(s_t)$$

$$\max_{s' \in \Gamma_h(s)} q(s') \geq q(s_t), \quad (12)$$

where $h^*(s)$ is the actual cost to the optimal leaf $s_t \in S_T$ below s. For (b) with $\Gamma = \Gamma_h$, we show that $h(s) \leq d(s, s') + h(s')$, $\forall s \in S$ and $\forall s' \in \Gamma_h(s)$,

$$q(s) - \max_{s' \in \Gamma(s)} q(s') \leq q(s) - q(s') + q(s') - \max_{s'' \in \Gamma(s')} q(s'')$$

$$\max_{s' \in \Gamma(s)} q(s') \geq \max_{s'' \in \Gamma(s')} q(s''). \quad (13)$$

Eq. (12) and Eq. (13) hold since q is monotone (Def. 3.1) and both Γ and Γ_h are reducing. ■

The heuristics from Eq. (11) can be used for both A* SEARCH and WA* SEARCH which—to our knowledge—for the first time allows increasing efficiency for solving optimal grasping problems by relaxing optimality with a guaranteed sub-optimality bound.

E. Computational Complexity

If we use a successor function Γ with a bounded number of successor states $|\Gamma(s)| \leq b$ and the heuristic function h from Eq. (11), A* SEARCH and Dijkstra's search algorithm share the *worst case* time complexity $\mathcal{O}(|S| \log(|S|))$ [27] for solving a PFP for an OGP. This analysis assumes that the search graph is a tree and both G and the arc costs $d(s, s')$ are known in advance. In practice graph expansion and arc costs determination have additional computational cost which we detail in the Section IV where we define variants of Γ and Γ_h that induce search trees.

IV. Practical Example: Point Contacts

In this section, we apply results from Section III-C in a concrete example considering point contact grasping [1], [3], [5], [6], [10], [20], [28]. Here, the contact set C consists of point contacts $c_i = (\mathbf{p}_i, \mathbf{n}_i)$ with positions on the object's surface $\mathbf{p}_i \in \mathbb{R}^3$ and surface normals $\mathbf{n}_i \in \mathbb{R}^3$ [1]. The goal is to find a static hard-finger grasp that can withstand maximal external influences as characterized by the Ferrari-Canny grasp quality function $Q_1 : C \to \mathbb{R}$ [5]. With the object's center of mass as a reference point for wrenches and approximating friction cones by 8 vectors, we use Q_1 to map force-closure grasps to positive values. For reducing the OGP $\langle C, q, m \rangle$ with $C = \{(\mathbf{p}_i, \mathbf{n}_i)\}_i$ and $q = Q_1$, we define reducing and preserving successor functions Γ_b^{ru} and Γ_b^{fc} according to a partitioning scheme and show that Q_1 is monotone in Section IV-A. We generate optimal grasps satisfying grasp constraints in Section IV-C and in Section IV-D we identify equivalent solutions to improve search performance.

A. Search Space and Heuristics for Point Contact Grasping

Defining a successor function that allows the search algorithm to quickly eliminate low quality grasps from grasp sets $\mathbf{G}(s)$ improves search performance significantly. For comparison, we define one *random-based* and one *domain specific* successor function. Both partition each super-contacts $\mathbf{c}_i$ into $b > 1$ disjoint subsets $\mathcal{P}(\mathbf{c}_i) = \{\mathbf{c}_i^1, \mathbf{c}_i^2, \ldots, \mathbf{c}_i^b\}$ and assign different partition elements $\mathbf{c}_i^j$ to different successors,

$$\Gamma_b(s) = \{(\mathbf{c}_i^{j_1}, \mathbf{c}_i^{j_2}, \ldots, \mathbf{c}_i^{j_m}) \in \mathcal{P}(\mathbf{c}_i)^m\}. \qquad (14)$$

With such a successor function, G is a tree and each state has b^m successor states with disjoint grasp sets. The successor function Γ_b^{ru} partitions super-contacts *randomly* with most *uniformly* sized partition elements $\mathbf{c}_i^j$. This results in super-contacts with size $|\mathbf{c}_i| = \frac{|C|}{b^l}$ in tree level l. The successor function Γ_b^{fc} is domain specific and partitions super-contacts by *clustering* their contacts in a position and normal based *feature space* Ψ [21], [29] using t iterations of k-MEANS [30]. Therefore, Γ_b^{fc} groups contacts that result in grasps of similar quality but does not ensure equal super-contact size which can result in G becoming unbalanced.

Theorem 4.1: The successor function Γ_b (and therefore Γ_b^{ru} and Γ_b^{fc}) is reducing and preserving.

Proof: All super-contacts with $|\mathbf{c}_i| > 1$ are split up, ensuring that corresponding super-contacts in successor states are strictly smaller. Partitionings $\mathcal{P}(\mathbf{c}_i)$ are exhaustive and all combinations are maintained in successor states. ■

For using the heuristic defined in Section III-D, we set Γ_h to Γ_b^{ru} or Γ_b^{fc} for a fixed branching factor $b > 1$ and show that the grasp quality function Q_1 is monotone.

Theorem 4.2: The Ferrari-Canny [5] grasp quality function Q_1 is monotone.

Proof: Since the convex hull operator is nondecreasing [31], Q_1 does not increase when contacts and their wrenches are removed. ■

B. Computational Complexity With Point Contacts

The computational cost for expanding states in G is dominated by computing the partitioning $\mathcal{P}$. For Γ_b^{ru}, time complexity is $\mathcal{O}(|\mathbf{c}_i|)$ while for Γ_b^{fc} time complexity is $\mathcal{O}(t\,|\mathbf{c}_i|)$ for each super-contacts $\mathbf{c}_i$. The computational cost of determining the step cost $d(s, s')$ is dominated by computing the convex hull of polyhedral wenches cones having worst case time complexity quadratic in number of wenches [32]. If we for analysis assume that $|C| = b^l$ and that Γ_b produces a balanced search tree with l levels and branching factor b, then level i of G has b^{im} states which have m super-contacts of size b^{l-i}. The computational complexity of completely searching level i for the worst case scenario is therefore $\mathcal{O}(b^{im}\,(mb^{l-i})^2)$ with additional costs $\mathcal{O}(b_h^m b^{im}\,(mb^{l-i}/b_h)^2)$ for h excluding expansion using expansion factor b_h. However, the branching factor b is of little relevance in practice since an informative heuristic results in an effective branching factor $b^* \ll b$ for A* SEARCH as we show empirically in Section V.

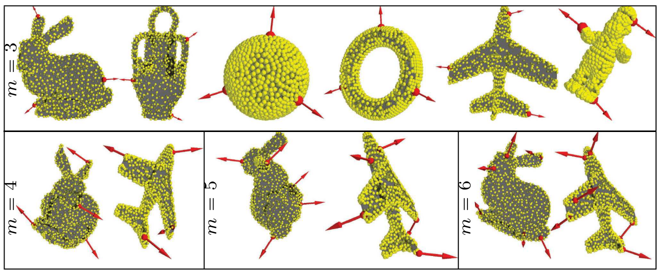


Fig. 3. Optimal grasps with $m = 3$ contacts found by our approach using A* SEARCH shown by arrows for the objects *bunny*, *amphora*, *sphere*, *donut*, *plane*, and *homer* and with $m = 4, 5$ and 6 for *bunny* and *plane*. The contact set C is indicated in yellow. The results obtained are identical to ES.

C. Grasp Constraints and Valid State

We model constraints on grasps $\mathbf{g}$ as predicates P that determine whether a grasp is *valid,* $P(\mathbf{g})$. If a super-contact's grasp set $\mathbf{G}(s)$ cannot provide any grasp that satisfies the constraint, we can prune the search tree below without risking incompleteness which models grasp constraints via the state space's structure. For this, the successor function Γ is restricted to valid states s' as in

$$\Gamma\restriction_P(s) = \{s' \in \Gamma(s) \mid \exists \mathbf{g} \in \mathbf{G}(s') : P(\mathbf{g})\}. \qquad (15)$$

Such constraints can model robot kinematics, tasks, and environments relevant to the gasp.

D. Search Space Symmetries

In the presence of symmetry, search algorithms evaluate many equivalent states and progress towards terminal states is slowed down. Several of our search states are identical or nearly identical grasps since we consider m ordered contacts and many objects are geometrically symmetric. In the first case, swapping contact indexes, in the second case rotating or translating the object results in grasps with similar quality and relative contacts arrangement. For removing symmetries, we define an (approximate) equivalence relation $\sim$ on search states and search the quotient space $S_{/\sim}$ instead of S. We define $\sim$ by mapping states into a discretized domain specific feature space $\Psi = \dot{\bigcup}_i \Psi_i$ with volumes Ψ_i and set

$$s \sim s' \iff \exists \Psi_i : \kappa(s) \in \Psi_i \text{ and } \kappa(s') \in \Psi_i \qquad (16)$$

where $\kappa : S \to \Psi$ is the feature space embedding [29], [33]. Firstly, the feature space contains the relative distances and normal differences for each of the m fingers' mean contact and is therefore invariant under translations and rotations. Secondly, it also contains the distance between the object's center of mass and the grasping center to account for moment arms of contacts. The feature space is discretized by a fixed distance threshold and class representative during state expansion. The effects of this symmetry removal are analyzed in Section V-B.

V. Experimental Evaluation

In this section, we evaluate and compare the performance of our approach from Section IV based on objects with complex asymmetric or symmetric shapes as shown in Fig. 3. For evaluation, we synthesize grasps with $m = 3$ contacts on the object

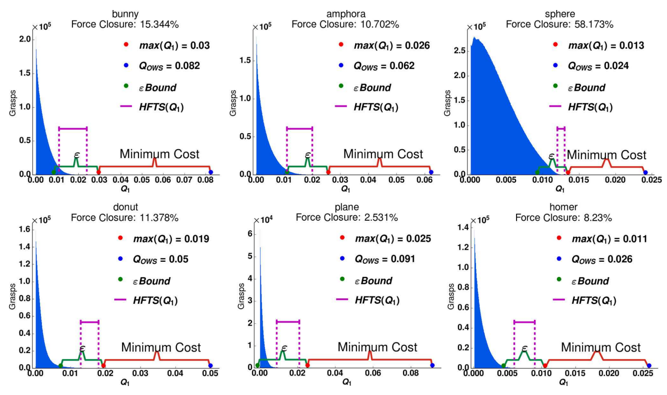


Fig. 4. Comparison between ES, HFTS, and our approach for m=3 Histograms (blue) show the Q_1 quality distributions for each object. Minimum path cost indicates the quality drop from $q(s_0)|\,Q_{OWS}$ to the optimal solution $g(s_t)$. Empirical standard deviation shows large quality variance for HFTS. A* SEARCH identifies the optimal solution (red) and ε=0.4 bounds (green) show ranges of bounded sub-optimality results.

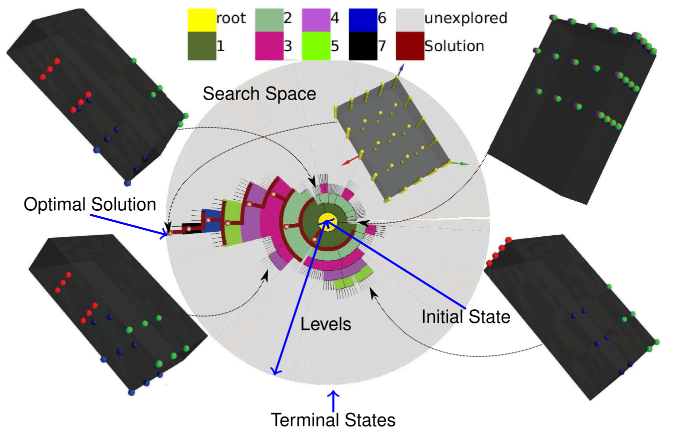


Fig. 5. Circular plot of the search tree G showing *created* and *visited* search states according to search depth for the *box* with 100 contacts. A* SEARCH creates 512 and visits 70 states. Colored sections indicate visited states and stems show created but unvisited states. The dark red lines show the optimal solution path. Only a small fraction of the search space is explored. For 4 states, we show the $m = 3$ super-contacts, which indicate the reason why the corresponding branches did not lead to the optimal solution.

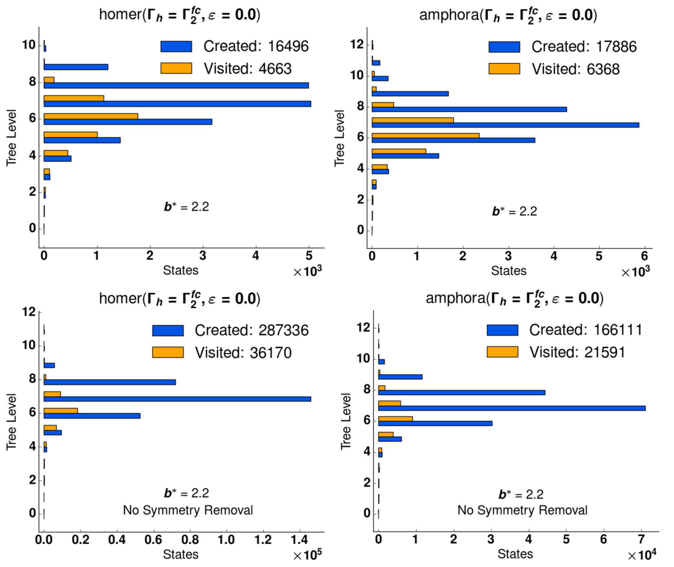


Fig. 6. Created and visited states for A* SEARCH with and without symmetry removal. Most states are considered on the middle levels in the search tree. Effective branching factors are shown as b^*. Note that the total number of possible states is approximately: $\sum_{i=0}^{\log_2 |C|}(b^m)^i \approx 1,227,133,513$.

surface and maximize the Ferrari-Canny grasp quality Q_1 [5]. The grasp quality Q_1 of the initial state s_0 is denoted as the quality of Object Wrench Space Q_{OWS} [20]. We set the inflation factor ε to the indicated values and employ different successor functions for search tree expansion and heuristic, Γ and Γ_h respectively. For tree expansion we set $\Gamma = \Gamma_2^{\text{fc}}$, and vary Γ_h for evaluation. Symmetries are removed as defined in Section IV. In Section V-A, we empirically verify optimality and compare to baseline approaches. In Sections V-B and V-C we analyze heuristic search in terms of state space exploration and number of search steps. Timing results are discussed in Section V-D. In Section V-E we generate optimal grasps for a specific robotic hand with kinematic grasp constraints.

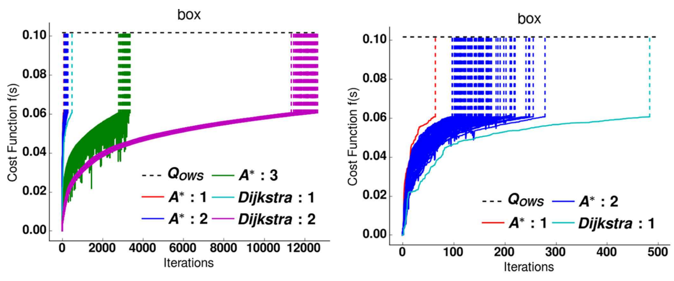
A. Optimality and Baseline Comparison

We compare our approach with $\Gamma_h = \Gamma_2^{\text{fc}}$ using both A* SEARCH and WA* SEARCH with $\varepsilon = 0.4$ to one exact and one approximate baseline: First, *exhaustive search* (ES), which considers each of the $\binom{1000}{3} = 166,167,000$ m-contact grasp over C and returns the optimal grasp. Second, *hierarchical fingertip space optimization* (HFTS) [21], which iteratively optimizes grasp in refined representations. To characterize the difficulty of the OGP for each of the objects, we plot histograms of grasp qualities in Fig. 4. The diagrams show that the percentage of *force-closure* grasps on each object is low while most force-closure grasps have low quality making it difficult to identify high quality grasps.

HFTS identifies high quality grasps from the tail of the distributions, but the variance in grasp quality is large. In no case the optimal grasp is returned and for all objects besides the simple *sphere* and *donut* a substantial margin is left to the optimal grasp. A* SEARCH identifies the same optimal grasps as ES (see Figs. 3 and 4) for $m = 3$ which confirms that our algorithm is complete and optimal. For $m > 3$ contacts, exhaustive search is prohibitively expensive, examples of optimal $m = 4, 5, 6$ contacts grasps found by our algorithm for bunny and plane are shown in Figs. 3. and 4 also shows that $\varepsilon = 0.4$ limits WA* SEARCH to high quality force-closure solutions for all objects but *plane* where the ε-bound includes low-quality force-closure grasps as indicated.

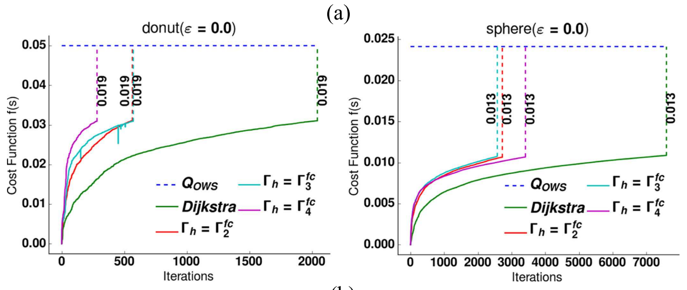
B. Heuristic Search

We qualitatively analyze the efficiency of A* SEARCH with $\Gamma_h = \Gamma_2^{\text{fc}}$ by recording the search tree for *box* in a circular plot. In Fig. 5 we see that only a small fraction of the search tree G is visited which indicates that the heuristic informs the search. The pruned tree branches represent grasps with overlapping or ill-placed contacts while the explored branches show spread out contact positions. Fig. 6 shows statistics of created and visited states per level for *homer* and *amphora* and allows a more detailed analysis. Most states are both created and visited in the middle levels but the visiting ration is highest for lower



Search	A*:1	A*:2	A*:3	Dijkstra's:1	Dijkstra's:2
Γ / Γ_h	Γ_2^{fc} / Γ_2^{fc}	Γ_2^{fc} / Γ_2^{ru}	Γ_2^{ru} / Γ_2^{fc}	Γ_2^{fc} / N.A.	Γ_2^{ru} / N.A.
b^*	1.9	2.4	4.7	2.4	6.1

(a)



(b)

Fig. 7. Efficiency analysis based on number of iterations for different random-based and domain specific expansion and heuristic functions for A* search and Dijkstra's search from 100 executions. (a) Overview and closeup plots of iterations against combined cost. Table: Legend and effective branching factor. (b) Different heuristic functions.

levels indicating that the heuristic is more precise for smaller super-contacts. The data also show that retaining symmetries leads to factor 17 and 9 more created and factor 7 and 3 more visited states for *homer* and *amphora* respectively.

Search efficiency can also be measured by number of search iterations as seen in Fig. 7 where the number of iterations is plotted against the current state's combined cost $f(s)$ given in Eq. (3). First we see in Fig. 7(a) that heuristic search with domain specific expansion and heuristic using Γ_2^{fc} requires the least amount of iterations. In this case, the effective branching factor b^* which characterizes a balanced tree with identical depth and number of explored states [25] is 1.9 as compared to $8 = b^m$ in the search tree. Using Γ_2^{ru} for heuristic or expansion instead requires more iterations where expansion is more sensitive leading to b^* of 2.4 and 4.7. Dijkstra's search has $b^* = 6.1$ when expanding with Γ_2^{ru}. The improvement from $b^* = 2.4$ for Dijkstra's expansion with Γ_2^{fc} to $b^* = 1.9$ for heuristic search is due to the domain specific heuristic. Different domain specific heuristics are analyzed in Fig. 7(b) where we see again that Dijkstra's is worse than heuristic search with $\Gamma_h = \Gamma_b^{\mathrm{fc}}$ while the largest value $b = 4$ corresponding to largest size reduction has the best result. In both Fig. 7(a) and (b), some curves show drops in combined cost, when $\Gamma \neq \Gamma_h$ i.e. the heuristic function is locally inconsistent. Theorem 3.2 shows that h is still admissible and solutions are optimal.

C. Approximative Heuristic Search

For WA* SEARCH, we analyze the tradeoff between sub-optimality and search steps using different values of ε for *bunny* and *plane* using $\Gamma_h = \Gamma_4^{\mathrm{fc}}$. The results in Fig. 8 show that increasing values of ε lead to earlier termination but that quality remains high when taking Fig. 4 as reference. Since inflation can result in an inadmissible heuristic, the current state's combined cost may drop during search as seen clearly for larger ε-values in Fig. 8.

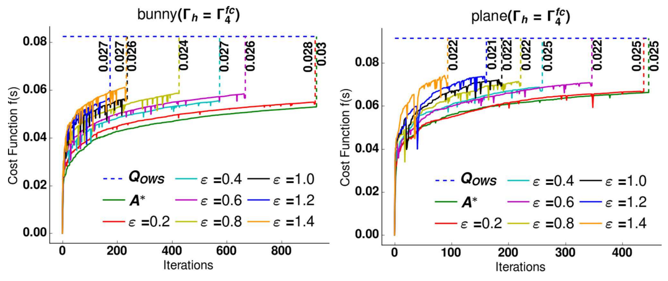


Fig. 8. Combined cost plotted against iterations WA* SEARCH using $\Gamma_h = \Gamma_4^{\mathrm{fc}}$. Larger inflation factors ε lead to less computation. Dashed lines show solution qualities.

(ε)	*bunny*	*amphora*	*sphere*	*donut*	*plane*	*homer*
A*	14.59	18.87	26.2	15.46	13.81	17.79
WA*						
(0.2)	9.99	14.31	17.07	8.98	11.47	17.21
(0.4)	7.42	7.83	20.39	9.05	6.8	13.05
(0.6)	8.08	10.42	17.22	8.01	9.21	9.98
(0.8)	7.28	16.19	17.29	7.64	6.98	13.50
(1.0)	4.29	4.51	8.85	4.69	4.39	10.43
(1.2)	4.61	7.33	16.26	5.43	5.53	8.92
(1.4)	4.81	6.26	11.44	6.35	2.94	7.63
ES	On average 2386.96 ± 6.73					

Fig. 9. CPU time in minutes using our method (with A* and WA* SEARCH) compared to exhaustive search (ES). Results are collected by setting different ε as indicated and $\Gamma_h = \Gamma_2^{\mathrm{fc}}$. Execution in Python with Ubuntu 12.04 running on an Intel Core i7-3770 @ 3.40 GHz × 8. with 32GB RAM and QHULL version 5.0 [32].

D. Computational Cost

The main computational load of our algorithm is caused by computing grasp quality for super-contact grasps which can contain many contacts. In Fig. 9 we compare timing results for finding optimal and bounded sub-optimal grasps using our algorithm against the computational cost of evaluation all grasps with ES. Comparing the first and last rows in Fig. 9 reveals that for optimal solutions, our approach is significantly more efficient than ES. The data also show that approximate search tends to be faster than A* SEARCH for larger ε. Due to the cost of computing Q_1, the algorithm does not perform for real-time application but it is faster than exhaustive search by several orders of magnitude. On average, our approach spends 87.3% of time on calculating convex hulls for grasp quality.

E. Hand Kinematics as Grasp Constraint

We constrain the successor function Γ as described in Section IV-C for generating optimal grasps for a specific robotic hand. The predicate P models fingertip reachability and is approximated by lookup in a pre-sampled feature space that contains all kinematically feasible fingertip configurations

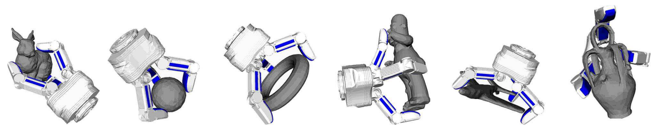


Fig. 10. Constraints on grasps: Optimal kinematically feasible grasps by Schunk-SDH hand for 3 fingertip contacts.

[21], [33]. Fig. 10 shows the optimal grasps for the hand as modeled by P.

VI. Conclusion

We addressed the problem of finding optimal grasp contacts on arbitrary objects with a novel complete and optimal algorithm by formulating a path finding problem. The advantage of our formulation is that it—for the first time—allows for efficient optimal and bounded sub-optimality solutions. This is important not only for grasping, but also for benchmarking other grasping approaches, for analyzing and characterizing objects, designing fixtures, and generating grasp databases for learning-based grasping. The formulation rendered optimal grasps as minimal cost paths and the search space consisted of super-contact grasps which represented sets of grasps. In each step along a path, the state's grasp set was reduced by removing contacts. We characterized grasp quality functions and state successor functions for reducing optimal grasping to a path finding problem and thereby defined a general framework for optimal grasping.

For the example of point-contact grasping, we presented a concrete search space constructed by a domain specific successor function and showed how a common grasp quality function can be used in our framework. Furthermore, we showed that a state validation mechanism can be employed to incorporate hand kinematic feasibility constraints, which ensures the produced optimal grasp contacts can be realized by a given robotic hand. In experiments, our approach required substantially less computational effort than exhaustive search to results with optimal grasps. Our evaluation was based on number of visited and created nodes and the development of the search fringe. The smallest effective branching factor was obtained with our domain specific heuristic function. Evaluation with approximate search showed significant cost reduction while maintaining high grasp quality.

References

[1] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2000, pp. 348–353.

[2] Y.-H. Liu, M.-L. Lam, and D. Ding, "A complete and efficient algorithm for searching 3-D form-closure grasps in the discrete domain," *IEEE Trans. Robot.*, vol. 20, no. 5, pp. 805–816, Oct. 2004.

[3] C. Borst, M. Fischer, and G. Hirzinger, "Grasping the dice by dicing the grasp," in *Proc. 2003 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2003, vol. 4, pp. 3692–3697.

[4] M. Roa and R. Suárez, "Grasp quality measures: Review and performance," *Auton. Robots*, vol. 38, no. 1, pp. 65–88, 2015.

[5] C. Ferrari and J. Canny, "Planning optimal grasps," in *Proc. 1992 IEEE Int. Conf. Robot. Autom.*, 1992, pp. 2290–2295.

[6] K. Hang, J. A. Stork, F. T. Pokorny, and D. Kragic, "Combinatorial optimization for hierarchical contact-level grasping," in *Proc. 2014 IEEE Int. Conf. Robot. Autom.*, 2014, pp. 381–388.

[7] M. T. Ciocarlie and P. K. Allen, "Hand posture subspaces for dexterous robotic grasping," *Int. J. Robot. Res.*, vol. 28, no. 7, pp. 851–867, 2009.

[8] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3D object grasp synthesis algorithms," *Robot. Auton. Syst.*, vol. 60, pp. 326–336, 2012.

[9] S. El Khoury, M. Li, and A. Billard, "Bridging the gap: One shot grasp synthesis approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2027–2034.

[10] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—A survey," *IEEE Trans. Robot.*, vol. 30, no. 2, pp. 289–309, Apr. 2014.

[11] B. Mirtich and J. Canny, "Easily computable optimum grasps in 2-D and 3-D," in *Proc. 1994 IEEE Int. Conf. Robot. Autom.*, 1994, pp. 739–747.

[12] T. Watanabe and T. Yoshikawa, "Grasping optimization using a required external force set," *IEEE Trans. Autom. Sci. Eng.*, vol. 4, no. 1, pp. 52–66, Jan. 2007.

[13] R. Diankov and J. Kuffner, "OpenRAVE: A planning architecture for autonomous robotics," Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-08-34, 2008, vol. 79.

[14] A. T. Miller and P. K. Allen, "Graspit! A versatile simulator for robotic grasping," *IEEE Robot. Autom. Mag.*, vol. 11, no. 4, pp. 110–122, Dec. 2004.

[15] R. Pelossof, A. Miller, P. Allen, and T. Jebara, "An SVM learning approach to robotic grasping," in *Proc. 2004 IEEE Int. Conf. Robot. Autom.*, 2004, vol. 4, pp. 3512–3518.

[16] J. Ponce and B. Faverjon, "On computing three-finger force-closure grasps of polygonal objects," *IEEE Trans. Robot. Autom.*, vol. 11, no. 6, pp. 868–881, Dec. 1995.

[17] S. Liu and S. Carpin, "Global grasp planning using triangular meshes," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 4904–4910.

[18] J. Cornella and R. Suárez, "Efficient determination of four-point form-closure optimal constraints of polygonal objects," *IEEE Trans. Autom. Sci. Eng.*, vol. 6, no. 1, pp. 121–130, Jan. 2009.

[19] M. A. Roa and R. Suárez, "Finding locally optimum force-closure grasps," *Robot. Comput.-Integr. Manuf.*, vol. 25, no. 3, pp. 536–544, 2009.

[20] N. S. Pollard, "Parallel methods for synthesizing whole-hand grasps from generalized prototypes" , MIT Artif. Intell. Lab., Cambridge, MA, USA, *Tech. Rep. AI-TR-1464*, 1994.

[21] K. Hang, J. A. Stork, and D. Kragic, "Hierarchical fingertip space for multi-fingered precision grasping," in *Proc. 2014 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 1641–1648.

[22] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[23] R. Dechter and J. Pearl, "Generalized best-first search strategies and the optimality of A," *J. ACM*, vol. 32, no. 3, pp. 505–536, 1985.

[24] I. Pohl, "Heuristic search viewed as path finding in a graph," *Artif. Intell.*, vol. 1, no. 3–4, pp. 193–204, 1970.

[25] J. Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Reading, MA, USA: Addison-Wesley, 1984.

[26] R. Ebendt and R. Drechsler, "Weighted A search—Unifying view and application," *Artif. Intell.*, vol. 173, no. 14, pp. 1310–1342, 2009.

[27] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *J. ACM*, vol. 34, no. 3, pp. 596–615, 1987.

[28] V.-D. Nguyen, "Constructing force-closure grasps," *Int. J. Robot. Res.*, vol. 7, no. 3, pp. 3–16, 1988.

[29] J. A. Stork, "Representation and learning for robotic grasping, caging, and planning," Ph.D. dissertation, School Comput. Sci. Commun., KTH Roy. Inst. Technol., Stockholm, Sweden, Jun. 2016.

[30] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *J. Roy. Statist. Soc. Ser. C (Appl. Statist.)*, vol. 28, no. 1, pp. 100–108, 1979.

[31] F. P. Preparata and M. Shamos, *Computational Geometry: An Introduction*. New York, NY, USA: Springer, 2012.

[32] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, 1996.

[33] K. Hang, J. A. Haustein, M. Li, A. Billard, C. Smith, and D. Kragic, "On the evolution of fingertip grasping manifolds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 2022–2029.