Patch Tree: Exploiting the Gauss Map and Principal Component Analysis for Robotic Grasping

Yan-Bin Jia[1], Yuechuan Xue[2] and Ling Tang[1]

***Abstract*— Grasp planning must consider an object's local geometry (at the finger contacts), for the range of applicable wrenches under friction, and its global geometry, for force closure and grasp quality. Most everyday objects have curved surfaces unamenable to a pure combinatorial approach but treatable with tools from differential geometry. Our idea is to "discretize" such a surface in a top-down fashion into elementary patches (e-patches), each consisting of points that would yield close enough wrenches. Preprocessing based on Gaussian curvature decomposes the surface into strictly convex, strictly concave, ruled, and saddle patches. The Gauss map guides the subdivision of any patch with a large variation in the contact force direction, with the aid of a Platonic solid. The principal component analysis (PCA) further subdivides any patch that has a large variation in torque. The final structure is called a *patch tree*, which stores e-patches at its leaves, and force or torque ranges at its internal nodes. Grasp synthesis and optimization operates on the patch tree with a stack to efficiently prune away non-promising finger placements. Simulation and experiment with a Shadow Hand have been conducted over everyday items. The patch tree exhibits different levels of surface granularity. It has a good promise for efficient planning of finger gaits to carry out grasping and tool manipulation.**

I. Introduction

Grasping [1], [2] and dexterous in-hand manipulation [3], [4] involve computation of finger contact locations on the surface of an object. A placement of m fingers can be tested for force closure [5], [6], and if verified, evaluated under various metrics [7]–[9], and optimized over finger forces [6], [10] via linear or nonlinear programming. As all the points on the surface constitute a two-dimensional (2-D) set, all the finger placements form a $2m$-D set. To deal with such a large search space, grasp analysis, synthesis, and optimization have traditionally focused on polygons [11]–[14] and polyhedra [15], [16], which are amenable to combinatorial approaches. Curved shapes (except for those with symmetries) have received much less attention, and if so, have been limited to 2-D [17]–[21] or subjected to discretization [22] in 3-D. This is in sharp contrast to the study of contact kinematics [23]–[25] and dexterous manipulation [26], [27] where curved 3-D shapes are often considered in their original geometry.

Advances in 3-D sensing and scanning have made it possible for grasping an unknown object based on matching its surface (as a point cloud or triangular mesh) with that of the hand to establish shape complementarity via discretization and optimization [28], [29] or with a probability-based measure [30]. Other approaches extracted "grasping points" via the use of supervised learning [31] or based on curvature, and then filtered them under quality measures [32].

Availability of 3D meshes has facilitated grasp synthesis, whether data-driven [33] or learning-based [34], for everyday items. Data-driven methods generate grasps on an input object by transferring those on similar objects, identified via model matching [35] or deep learning [36], [37], from a database of annotated grasps, and ranking them under a geometric or probabilistic quality measure [38], [39]. The "eigengrasp" approach [40] transferred the first two principal components of human hand configurations to an anthropomorphic hand through optimization. Other works on grasping examined how uncertainties in friction affected grasp qualities [41], and how state estimation and trajectory optimization were applied in the presence of occlusions [42].

Quality measures were proposed using probability or considering dynamics for evaluating a grasp in terms of its ability to handle uncertainties in the object's pose [43] or movements [44] during the action. Efforts [45]–[47] to benchmark robotic grasping have tried to restrict arm-hand platforms, designate grasp planners, standardize post-grasp actions for testing, classify failures, and employ common datasets with or without real experiments. However, these benchmarks have not reached a level of maturity or generality to be regarded as "universal" measures of grasp execution.

The hierarchical fingertip space [48] clustered "unit" mesh regions bottom-up into surface partitions at different levels, also taking into reachability and adaptability for grasp synthesis. Depending heavily on training data, the clustering criteria needed more clarity from both geometry and mechanics. In [49], the point cloud of an object was decomposed into boxes, each mapped to a hand pose and an interaction wrench under learning, so grasps could be predicted and then selected under a metric combining multiple cost indices.

With its continuously varying geometry, a curved surface is not directly tractable using a combinatorial approach. Our purpose, similar to [48], is to "discretize" the surface into elementary patches (e-patches) but in a top-down manner. Within each patch only a small variation exists in the wrench (force and torque) generated by the same level of normal contact force, making the patch adequate for contact with at most one finger. Using first the Gauss map from differential geometry and later the principal component analysis, we introduce a tree structure to represent different levels of subdivision.

[1]Yan-Bin Jia and Ling Tang are with the Department of Computer Science, Iowa State University, Ames, IA 50011, USA. jia, ling@iastate.edu

[2]Yuechuan Xue is with Amazon, Cambridge, MA 02142, USA. yuechxue@amazon.com

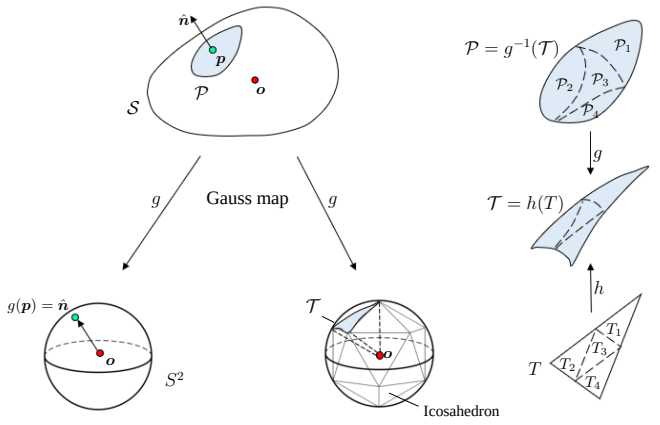


Fig. 1. Gauss map and subdivision (stage 1) based on variance of the surface normal. Surface $\mathcal{S}$ and patch $\mathcal{P}$ under the map onto the unit sphere S^2, in which an inscribed icosahedron is used for subdividing $\mathcal{P}$ and $\mathcal{S}$.

Section II presents the surface subdivision procedure, with computation treated in Section III. Section IV describes the patch tree. Section V shows how to use the tree effectively in grasp synthesis and optimization. Simulation and experimental results are presented in Section VI. Discussions and future extensions follow in Section VII.

II. Subdividing the Object's Surface

The closed surface $\mathcal{S}$ of an object is described in a body frame located at some point $\boldsymbol{o}$, say, the object's center of mass. The surface is positively oriented with unit outward normal vectors. For consistency with differential geometry and clarity of description, we treat all normal contact forces as tensile—that is, they point outward. This merely negates the wrench space but does not affect the force closure property of a grasp (which requires the wrench space to contain the origin in its interior). There will be also no change to the outcome of evaluation under many grasp metrics.

In the context of grasping, a finger placed at a point $\boldsymbol{p}$ on $\mathcal{S}$ exerts some contact force inside the friction cone centered about the outward normal, and subsequently, a torque about the reference point $\boldsymbol{o}$. Our objective is to "discretize" $\mathcal{S}$ into small patches, each to be contacted by at most one finger. Such a patch needs to have small variations in the contact normal as well as in the torque with respect to $\boldsymbol{o}$ under unit normal force. Small wrench variations under tangential friction are subsequently guaranteed as well.

A. Bounding Variation in the Contact Force Direction

We first consider that $\mathcal{S}$ is convex and smooth. The normal varies continuously on $\mathcal{S}$. At the point $\boldsymbol{p}$ with the unit outward normal $\hat{\boldsymbol{n}}$, the two principal curvatures κ_1 and κ_2 measure the two extreme rates of local bending of the surface, which are known to be along two orthogonal tangent directions [50, p. 133]. We have $\kappa_1, \kappa_2 \leq 0$ and the Gaussian curvature $K = \kappa_1\kappa_2 \geq 0$ at $\boldsymbol{p}$. The Gauss map g takes $\boldsymbol{p}$ to the point $\hat{\boldsymbol{n}}$ on the unit sphere S^2 at $\boldsymbol{o}$, as shown on the left in Fig. 1. The image $g(\mathcal{S})$ of $\mathcal{S}$ is S^2 because every point on the sphere is the normal of some point on $\mathcal{S}$.

Our subdivision algorithm makes use of a convex, regular polyhedron $\mathcal{D}$ (called a Platonic solid[1]) inscribed in S^2 with congruent triangular facets which tessellate and are geometrically sturdy and easy to be subdivided. Whether a tetrahedron, octahedron, or icosahedron is selected depends on the object's geometric complexity. Consider a projection h that maps every point $\boldsymbol{q}$ on $\mathcal{D}$ to where the ray $\overrightarrow{\boldsymbol{oq}}$ intersects S^2. On the far right in Fig. 1 shows that every triangular facet T on $\mathcal{D}$ (an icosahedron) is projected to a spherical triangle $\mathcal{T}$ on S^2, which is the image of some surface patch $\mathcal{P} \subseteq \mathcal{S}$ under the Gauss map g. Namely, $\mathcal{P} = g^{-1}(h(T))$.

The total Gaussian curvature $\gamma(\mathcal{P})$ of $\mathcal{P}$ is the integral of K over the patch; it is equal to the area of the image region $g(\mathcal{P})$ on the unit sphere. The term directly measures variation in the normal on $\mathcal{P}$. It holds that $\gamma(\mathcal{S}) = 4\pi$, which is the area of S^2.[2] If $\gamma(\mathcal{P})$ is not small enough, the patch is further subdivided. This is done through splitting the triangle T equally into four triangles T_i, $1 \leq i \leq 4$, as illustrated on the right in Fig. 1. Each triangle T_i induces a patch $\mathcal{P}_i = g^{-1}(\mathcal{T}_i)$, where $\mathcal{T}_i = h(T_i)$ is a spherical triangle. The Gaussian curvature $\gamma(\mathcal{P}_i)$ as the area of $\mathcal{T}_i$ is equal to the solid angle $\Omega(T_i)$ subtended by T_i on S^2. (The solid angle can be evaluated from the triangle's vertex locations using a formula derived in [51].) We call T_i the *subtending triangle* of the patch $\mathcal{P}_i$.

Subdivision continues until the total Gaussian curvature of every newly generated patch is below some threshold δ_K, so the patch is deemed "flat enough". The image of a non-smooth object such as a cylinder or cone under the Gauss map degenerates into one or more curves and isolated points. Subdivision of its surface is carried out using the spherical triangles intersected by such a curve or containing such a point. Their total area reflects variation of the normal proportionally.

B. Accounting for Torque Variations

Consider a patch $\mathcal{P}'$ generated during the subdivision described above. The unit normal force $\hat{\boldsymbol{n}}$ applied at $\boldsymbol{p} \in \mathcal{P}'$ yields a torque $\boldsymbol{\tau} = f(\boldsymbol{p}) \stackrel{\text{def}}{=} \boldsymbol{p} \times \hat{\boldsymbol{n}}$ about the origin $\boldsymbol{o}$. The patch $\mathcal{P}'$ is thus mapped to a patch $\mathcal{Q} = f(\mathcal{P}')$ of torques. Further subdivision of $\mathcal{P}'$ is needed if $\mathcal{Q}$ has wide variations.

Torque variations in $\mathcal{Q}$ is best measured via the principal component analysis [52], which yields the tightest bounding cuboid $\mathcal{C}$ for $\mathcal{Q}$ with its three orthogonal axes $\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2, \hat{\boldsymbol{u}}_3$, referred to as the *principal components*, indicating respectively the primary, secondary, and tertiary directions along which the torques on $\mathcal{Q}$ are distributed.

Let δ_1, δ_2, and δ_3 be the variations along the three axes, respectively. If $\sqrt{\delta_1^2 + \delta_2^2 + \delta_3^2} > \delta_\tau$, for some $\delta_\tau > 0$, we divide the cuboid at the centroid $\bar{\boldsymbol{\tau}}$ of $\mathcal{Q}$ along the $\hat{\boldsymbol{u}}_1$- and $\hat{\boldsymbol{u}}_2$-axes equally into four smaller cuboids, and subdivide the

[1]Only five convex Platonic solids exist: tetrahedron, cube, octahedron, dodecahedron, and icosahedron.

[2]by the Gauss-Bonnet Theorem [50, pp. 260–265] which states that this value for a compact (i.e., closed and bounded) surface equals $4\pi(1 - \zeta)$, where ζ is the genus (i.e., the number of holes).

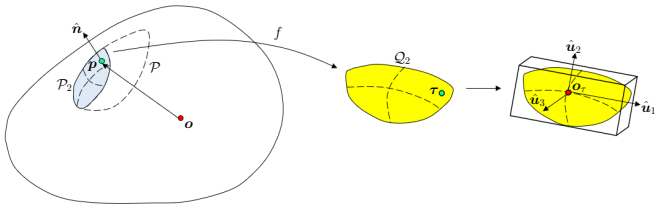


Fig. 2. Subdivision (stage 2) based on torque variance. (a) The patch $\mathcal{P}_2$, one of the four patches making up $\mathcal{P}$ in Fig. 1, is mapped to the torque patch $\mathcal{Q}_2$ in (b) under $f : \boldsymbol{p} \mapsto \boldsymbol{\tau} = \boldsymbol{p} \times \hat{\boldsymbol{n}}$. (c) The tightest cuboid of the torque patch is spanned by the set's three principal axes $\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2, \hat{\boldsymbol{u}}_3$, the first two of which can be utilized for subdividing $\mathcal{Q}_2$ and $\mathcal{P}_2$.

torque patch $\mathcal{Q}$ accordingly. This induces a subdivision of the surface patch $\mathcal{P}'$ (as illustrated in Fig. 2 on a patch generated after the first stage of subdivision). The *principal rectangle* of the torque patch $f(\mathcal{P}')$ is defined to be centered at $\bar{\boldsymbol{\tau}}$ with four vertices $\bar{\boldsymbol{\tau}} \pm \frac{\delta_1}{2}\hat{\boldsymbol{u}}_1 \pm \frac{\delta_2}{2}\hat{\boldsymbol{u}}_2$. It will be used for grasp synthesis in Section V.

The above subdivision step is repeated until the torque variation within every newly generated patch is within some threshold δ_τ. The generated patches, with small variations in normal force direction and torque, are called *elementary patches* (e-patches).

The subdivision strategy, in two stages based on surface normal and torque, respectively, does not take into account forces and torques generated due to friction. Grasp synthesis and optimization leveraging e-patches in Section V will take full account of friction, hence nothing is compromised.

C. The Case of a Non-Convex Solid

When the surface $\mathcal{S}$ is non-convex, it can be divided into patches that are convex (the Gaussian curvature $K > 0$ and the two principal curvatures $\kappa_1, \kappa_2 < 0$ everywhere), concave ($K, \kappa_1, \kappa_2 > 0$ everywhere), developable ($K = 0$ everywhere), or saddle-shaped ($K < 0$ everywhere). The resulting patches are referred to as *monotone patches*. Each is subjected to the two-stage subdivision described in Section II A and B.[3]

III. Computation Over a Triangular Mesh

The object is input as a triangular mesh. The normal, Gaussian curvature K, and mean curvature H at a vertex v of the mesh are estimated using a known method (supported by analyses) from [53].

The mesh is decomposed into regions representing monotonic surface patches of the object. This is carried out in one or more rounds of depth-first search (DFS). Each round starts at an unvisited vertex. The search backtracks every time the Gaussian curvature K changes its sign, whether from '+' or '−' to 0, or from 0 to '+' or '−'.[4] Even though a convex patch and a concave patch both satisfy $K > 0$, their mesh regions will always be traversed in different DFS rounds because K becomes zero at the their boundary, where the current traversal ends, if the two are adjacent.

During the DFS traversal of the mesh region $\mathcal{R}_M$ representing a monotonic patch $\mathcal{P}_M$, we also estimate the total Gaussian curvature $\gamma(\mathcal{P}_M)$ as follows. Consider a triangular facet T inside $\mathcal{R}_M$ with vertices, say, v_1, v_2, and v_3, and its represented patch element $\mathcal{E}$ on $\mathcal{S}$. The total Gaussian curvature $\gamma(\mathcal{E})$ of $\mathcal{E}$ is equal to the solid angle $\Omega(T)$ subtended by the triangle T with $\hat{\boldsymbol{n}}(v_1), \hat{\boldsymbol{n}}(v_2)$, and $\hat{\boldsymbol{n}}(v_3)$ as vertices. We have $\gamma(\mathcal{P}_M) \approx \sum_{T \subset \mathcal{R}_M} \Omega(T)$. Because the patch is monotonic and the triangular facets tessellate $\mathcal{R}_M$, the spherical triangles induced by them also tessellate on the unit sphere. The evaluation thus has high accuracy.

Consider a mesh region $\mathcal{R}$ representing the patch $\mathcal{P}$ with vertices, say, $v_1, v_2, \ldots, v_k$. The PCA for $\mathcal{R}$ is conducted over the torques $\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_k$ generated by unit normal force at these vertices. The principal components $\hat{\boldsymbol{u}}_1, \hat{\boldsymbol{u}}_2, \hat{\boldsymbol{u}}_3$ and variations $\delta_1, \delta_2, \delta_3$ are extracted from the matrix $L = (\boldsymbol{\tau}_1 - \overline{\boldsymbol{\tau}}, \boldsymbol{\tau}_2 - \overline{\boldsymbol{\tau}}, \ldots, \boldsymbol{\tau}_k - \overline{\boldsymbol{\tau}})$, where $\overline{\boldsymbol{\tau}}$ is the centroid of the torques.

IV. Patch Tree

Subdivision is conducted simultaneously with the construction of a *patch tree* Υ, in which every node N represents a patch, denoted by $\mathcal{P}_N$, and its corresponding mesh region, denoted by $\mathcal{R}_N$. The leaves $L_1, L_2, \ldots, L_h$ of Υ store the meshes $\breve{\mathcal{R}}_1, \breve{\mathcal{R}}_2, \ldots, \breve{\mathcal{R}}_h$, called *elementary regions* (e-regions), which correspond to the e-patches $\breve{\mathcal{P}}_1, \breve{\mathcal{P}}_2, \ldots, \breve{\mathcal{P}}_h$.

The root of Υ represents the object's surface $\mathcal{S}$. Every child node M of the root represents a monotonic patch $\mathcal{P}_M$, for which a separate Platonic solid $\mathcal{D}_M$ and a separate unit sphere S^2_M are devoted to its subdivision. The node M stores the triangular facets of $\mathcal{D}_M$ intersected by the normals of $\mathcal{P}_M$ on S^2_M.[5] Every child of M, at depth 2 and storing exactly one triangular facet, is the root of a quadtree Ψ [54, pp. 307–315] to be generated from the two-stage subdivision strategy described in Section II.[6]

An internal node N of the quadtree Ψ does not store the represented patch $\mathcal{P}_N$. If the node results from a normal-based subdivision, then it stores the subtending triangle T_N of $\mathcal{P}_N$ in its spatial location. If it results from a torque-based subdivision, then it stores the principal rectangle B_N (in the torque space) of the torque patch $f(\mathcal{P}_N)$. The smaller patches generated from splitting $\mathcal{P}_N$ are passed on to the children of N. In the tree Ψ, only the e-patches $\breve{\mathcal{P}}_1, \breve{\mathcal{P}}_2, \ldots, \breve{\mathcal{P}}_h$ are explicitly stored — at the leaves.[7] The overall storage is thus dominated by the size n of the object's mesh, that is, $O(n)$.

The first stage of subdivision is represented in the tree up to the depth $b_1 = \log_4(4\pi/\delta_K)$, where δ_K is the earlier introduced threshold on the Gaussian curvature. The effort of

[3] A point on the object's surface can be contacted by the fingertip only if it does not have a principal curvature greater than $-\kappa_{\max}$, where $\kappa_{\max} < 0$ is the maximum principal curvature anywhere on the (convex) fingertip. A point violating the above condition must appear on a concave, ruled, or saddle patch, and should be excluded from the patch.

[4] Numerically, we deem K as zero if its value is small enough.

[5] In the case of a cylinder, for example, these facets are intersected by the unit disk bounded by the equator.

[6] Every further subdivision step generates up to four smaller patches.

[7] To facilitate grasp synthesis, every leaf also stores the innermost vertex of its region.

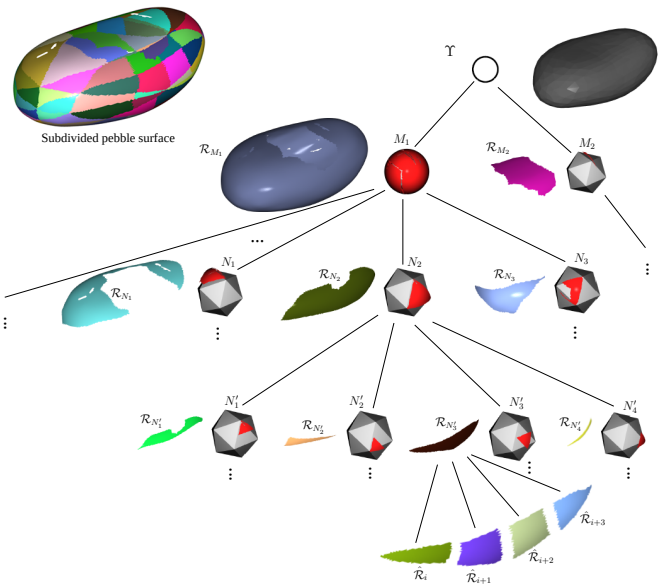


Fig. 3. Patch tree constructed for a pebble (with the subdivided surface shown on the upper left). Shown next to an internal node N generated from a normal-based subdivision is the represented patch, whose image (red) under the Gauss map is drawn at N together with an icosahedron (on which the patch's representing triangle(s) T_N are covered by the image). At the four leaves as children of N_3' generated from a torque-based subdivision, only their represented patches are displayed because the representing principal rectangles are in the torque space. The threshold values are $\delta_G = \pi/10$ on the total Gaussian curvature and $\delta_\tau = 0.008$ on torque variation.

computation at every depth is $O(n)$. Let b_2 be the maximum steps of torque-based subdivision in the second stage. The PCA on a torque patch takes time linear in its number of vertices. For all the nodes corresponding to the same step of subdivision in the second stage, the total amount of effort remains $O(n)$. Hence, construction of the entire patch tree is done in $O(nb)$ time, where $b = b_1 + b_2$ is the tree height.

Fig. 3 illustrates a patch tree Υ of height 4 and with 215 leaves as constructed over the triangular mesh of a pebble generated by a 3D scanner from Next Engine, Inc. The two nodes at depth 1 represent monotonic mesh regions $\mathcal{R}_{M_1}$ and $\mathcal{R}_{M_2}$. The convex region $\mathcal{R}_{M_1}$ is subdivided into 20 smaller regions at depth 2 based on the surface normal using an icosahedron. Only three of these regions are shown due to the limited space. The middle region $\mathcal{R}_{N_2}$ is further split into $\mathcal{R}'_{N_1}, \mathcal{R}'_{N_2}, \mathcal{R}'_{N_3}$, and $\mathcal{R}'_{N_4}$ at depth 3. Based on torque variation, one of them, say, $\mathcal{R}'_{N_3}$, is split into four smaller regions stored at leaves.[8] The concave monotonic region $\mathcal{R}_{M_2}$ has one child, four grandchildren, and five leaves at depth 4.

V. Grasp Synthesis and Optimization

Consider a placement $\mathcal{G}$ of m fingers at the locations $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m$ on the object's surface. At the contact point $\boldsymbol{p}_i$ with the i-th finger, we let $\hat{\boldsymbol{n}}_i$ be the unit outward normal, and $\hat{\boldsymbol{s}}_i$ and $\hat{\boldsymbol{t}}_i$ be two orthogonal tangent vectors. The finger exerts the force $\boldsymbol{f}_i$ inside the contact friction cone, and also a torque $\boldsymbol{p}_i \times \boldsymbol{f}_i$. To compare force and torque, we will divide the latter by the "average" radius of gyration $\rho = \sqrt{\bar{I}/m}$, where $\bar{I}$ is the average moment of inertia over all possible axes of rotation on the unit sphere S^2. (It can be shown that $\bar{I}$ equals the average of the moments of inertia about the three principal axes.)

[8]The principal rectangle in the torque space used in the partitioning is not shown to avoid a mixing of two different spaces.

A. Force Closure and Grasp Quality Evaluation

The contact map for the i-th finger,

$$G_i = \begin{pmatrix} \hat{\boldsymbol{n}}_i & \hat{\boldsymbol{s}}_i & \hat{\boldsymbol{t}}_i \\ \boldsymbol{p}_i \times \hat{\boldsymbol{n}}_i/\rho & \boldsymbol{p}_i \times \hat{\boldsymbol{s}}_i/\rho & \boldsymbol{p}_i \times \hat{\boldsymbol{t}}_i/\rho \end{pmatrix},$$

generates the *primitive wrench set* under unit normal force:

$$W_i = \{G_i(1, f_{is}, f_{it})^T \mid f_{is}^2 + f_{it}^2 \le \mu^2\}.$$

where μ is the coefficient of friction. The *grasp wrench space* [56] is the convex hull of $W = \cup_{i=1}^m W_i$, namely,

$$\mathrm{CH}(W) = \left\{ G\boldsymbol{f} \;\middle|\; f_{in} \ge 0, \sum_{i=1}^m f_{in} = 1, f_{is}^2 + f_{it}^2 \le \mu^2 f_{in}^2 \right\},$$

where $G = (G_1, G_2, \ldots, G_m) \in \mathbb{R}^{6\times 3m}$ is the *grasp matrix* [55, pp. 218–219] and $\boldsymbol{f} = (\boldsymbol{f}_1^\top, \boldsymbol{f}_2^\top, \ldots, \boldsymbol{f}_m^\top)^\top$. The space includes all the wrenches that can be generated by unit total normal force from the fingers, All the possible wrenches generated by the grasp form a set

$$C = \{\lambda \boldsymbol{w} \mid \lambda \ge 0 \text{ and } \boldsymbol{w} \in \mathrm{CH}(W)\}.$$

The grasp $\mathcal{G}$ achieves *force closure* if the origin is in the interior of $\mathrm{CH}(W)$. It is not difficult to show that force closure is equivalent to satisfaction of two conditions: i) $\mathrm{rank}(G) = 6$ and ii) $-\boldsymbol{w}_c \in C$, where

$$\boldsymbol{w}_c = \frac{1}{m} \sum_{i=1}^m \begin{pmatrix} \hat{\boldsymbol{n}}_i \\ \boldsymbol{p}_i \times \hat{\boldsymbol{n}}_i \end{pmatrix} \in \mathrm{CH}(W) \subset C.$$

A commonly used grasp quality [14] is the minimum distance from the origin to $\mathrm{CH}(W)$, i.e., $d(\mathcal{G}) = \min_{\boldsymbol{w}\in \mathrm{CH}(W)} \|\boldsymbol{w}\|$. Here, we use the algorithm from [9] which computes the following equivalent form:

$$d(\mathcal{G}) = \min_{\boldsymbol{u}\in\mathbb{R}^6, \|\boldsymbol{u}\|=1} \max_{\boldsymbol{w}\in W} \boldsymbol{u}^T \boldsymbol{w}. \tag{1}$$

The algorithm begins with a force closure test, which checks if W is full rank and, if so, employs a procedure from [57] to verify that $-\boldsymbol{w}_c$ is in the cone C. It then iteratively updates a convex hull in 6D that contains the origin in the interior, each time adding a vector from W which maximizes the dot product with the normal of the current closest facet to the origin. The algorithm can achieve a quality estimate less than $d(\mathcal{G})$ by any small $\epsilon > 0$.

B. Quick Rejection of a Finger Placement

For m fingers to reach force closure under contact friction, $m > 2$ is necessary. Given a total of t e-patches, a brute-force strategy would enumerate $\binom{t}{m}$ patch combinations (each determining a finger placement) until a grasp is found. With the patch tree Υ, we can prune away many combinations of

e-patches contained in one or multiple larger patches which together are impossible to yield a grasp.

To avoid duplicates for specifying a finger placement, we order all the nodes in Υ by the visit time during a postorder traversal. Two nodes N_1 and N_2 assume $N_1 \preceq N_2$ if either they are the same node or N_1 precedes N_2 in the traversal. For example, in Fig. 3, $N_1 \preceq N_1' \preceq N_4' \preceq N_2 \preceq N_3 \preceq M_1 \preceq M_2$.

We impose $N_i \preceq N_j$ for the patches $\mathcal{P}_{N_i}$ and $\mathcal{P}_{N_j}$ in respective contact with the fingers $\mathcal{F}_i$ and $\mathcal{F}_j$, $i < j$. An m-tuple $\langle N_1, N_2, \ldots, N_m \rangle$ of nodes in the patch tree Υ is said to be a *valid* finger placement if $N_1 \preceq N_2 \preceq \cdots \preceq N_m$.

The finger placement $\langle N_1, N_2, \ldots, N_m \rangle$ is subjected to a quick test over the triangles T_{N_i} or principal rectangles B_{N_i}, $1 \leq i \leq m$, stored at these nodes to represent the patches $\mathcal{P}_{N_1}, \mathcal{P}_{N_2}, \ldots, \mathcal{P}_{N_m}$. If the patches have all resulted from normal-based subdivision (or all from torque-based subdivision), the test fails if the triangles (or the principal rectangles) lies on or to one side of some plane through the origin. [9] If the patches are mixed. The same checking is performed after replacing every patch from torque-based subdivision with the triangle of its closest ancestor in the patch tree generated from normal-based subdivision.

C. Search for a Grasp

We start with all the children $M_1, M_2, \ldots, M_a$, from left to right, of the root of Υ. Enumerate, in the reverse lexicographic order, valid m-tuples $\langle M_{i_1}, M_{i_2}, \ldots, M_{i_m} \rangle$, $1 \leq i_1, i_2, \ldots, i_m \leq a$, with repeats of an index allowed since multiple fingers can be placed on the same patch out of $\mathcal{P}_{M_1}, \mathcal{P}_{M_2}, \ldots, \mathcal{P}_{M_a}$. Every enumerated m-tuple $\langle M_{i_1}, M_{i_2}, \ldots, M_{i_m} \rangle$ is subjected to the quick rejection test in Section V-B, and after passing it, pushed onto an initially empty stack Q as $(1, \langle M_{i_1}, M_{i_2}, \ldots, M_{i_m} \rangle)$.

In every iteration, the top element $(i, \langle N_1, N_2, \ldots, N_m \rangle)$ is popped out of the stack. The node N_i is up for expansion. If N_i is not a leaf node, set $k \leftarrow i$. If it is, we search for the first k, cyclically from $i+1$ to m to 1 and to $i-1$, such that N_k is not a leaf node. Replace N_k in the m-tuple with each of its children N' satisfying $N_{k-1} \prec N'$ if N_{k-1} is a leaf or $N_{k-1} \preceq N'$ otherwise. Every replacement results in a new m-tuple α which, after passing the test in Section V-B, is pushed onto the stack as an element $((k \bmod m) + 1, \alpha)$ in the reverse lexicographic order.

If no such N_k is found after k reaches $i-1$, the element has only distinct e-patches, say, $\breve{P}_1, \breve{P}_2, \ldots, \breve{P}_m$. We perform the force closure test on the innermost vertices of the corresponding mesh regions $\breve{R}_1, \breve{R}_2, \ldots, \breve{R}_m$. This finger placement, after passing the force closure test, is checked for kinematic feasibility to ensure that these innermost vertices can be simultaneously reached by the fingers of the robotic hand. This is an inverse kinematics problem solvable using, say, the Levenberg-Marquardt method [58].

[9] It suffices to check their vertices. The situation happens if and only if there exists two vertices whose cross product has dot products with the remaining vertices that have the same sign or are equal to zero.

(a)

	Bottle	Screwdriver Handle	Mug	Cup
δ_G	$\pi/25$	$\pi/5$	$\pi/5$	$\pi/5$
δ_τ	0.015	0.01	0.01	0.01
Height of Υ	8	6	6	7
# Leaves (t)	329	172	250	157

(b)

Fig. 4. (a) Subdivided surfaces of a bottle, screwdriver handle, mug, and cup. (b) Thresholds used by and statistics of the generated patch trees.

D. Grasp Optimization

Grasp synthesis may stop as soon as the kinematic feasibility test is passed. To find the optimal grasp under (1), we continue until the stack Q becomes empty. The grasp $\mathcal{G}_0$ with the highest score $d(\mathcal{G}_0)$ is selected.

Improvements over $\mathcal{G}_0$ can be done by locally moving the fingers $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_m$ within their respectively elementary regions. The optimization proceeds in multiple rounds. Within each round, it randomly orders the fingers and adjust their positions one by one. A finger under adjustment is relocated to a neighboring vertex within the same elementary region for the most grasp quality increase. No action is taken if such a neighboring vertex does not exist. The algorithm terminates after a round in which no finger gets repositioned. The grasp $\mathcal{G}^*$ with the fingers placed at their final locations $v_1^*, v_2^*, \ldots, v_m^*$ is returned.

VI. Simulation and Experiment

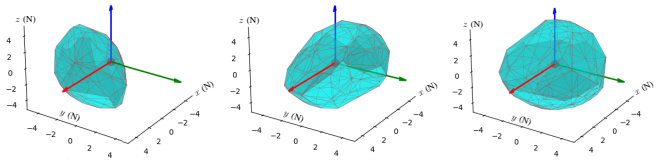
Fig. 4(a) shows the subdivided surfaces generated on the mesh models of a bottle, screwdriver handle, mug, and coffee cup. The first and third models are drawn from the datasets provided by the Dexterity Network [59], while the second and fourth ones were scanned. All objects are non-convex with holes in the last three. Part (b) of the figure lists the control parameter values used in the constructions as well as some statistics about the resulting patch trees.

The grasp optimization algorithm is simulated for three fingers on a second pebble (different than the one in Fig. 3), a potato, a mouse, and the mug in Fig. 4. As shown in Fig. 5(a), the algorithm has pruned away $80-90\%$ of triples of e-patches. Grasp executions with the Shadow Hand are simulated using the MuJoCo physics engine [60]. To test the robustness, the normal finger contact forces are initialized to counter the gravity of the object only. As the adversary force in one direction increases, impedance control adjusts the finger forces accordingly. The maximum force magnitude just before a contact slip is recorded for the direction. Part

	Pebble	Potato	Mouse	Cup
Triples evaluated	14.21%	10.24%	21.06%	18.41%
$d(\mathcal{G}_0)$	0.2720	0.3267	0.2302	0.2130
$d(\mathcal{G}^*)$	0.3127	0.3556	0.2476	0.2354

(a)

(b)



(c)

Fig. 5. (a) Qualities of optimal three-fingered grasps ($\mu = 0.5$ uniformly). (b) Grasps on the first three objects with qualities 0.199, 0.2024, 0.195, respectively, as executed by the Shadow Hand. (c) Polytopes of maximum resistible forces by these grasps under impedance control with the normal contact force initialized to balance gravity.

(c) of the figure plots a polytope that approximates maximum resistible forces in all directions for each of the first three objects.

The grasping algorithm was tested with a Shadow Dexterous Hand driven by a 4-DOF WAM Arm on the same potato, mouse, and cup used in the simulation. Each object is placed manually into the hand in a configuration computed from the grasp under test via inverse kinematics. The contact forces were estimated from the hand's strain gauge readings using a procedure from our previous work [61]. Slight position adjustments were made on the fingertips such that the sums of normal contact forces were kept at 3 N for all tested grasps.

The WAM Arm shook the grasped object of mass m along the y- and z-axes.[10] The acceleration $\boldsymbol{a}$ during a shaking action yielded the inertial force $-m\boldsymbol{a}$ that was identified with disturbance. The acceleration value was calculated from differentiating the velocity of the WAM Arm's open end. On each of the potato, mouse, and cup, the optimal grasp and an "ordinary" grasp (arbitrarily picked) were tested.

Fig. 6 displays all six grasps. The potato in the ordinary grasp was dropped during the shakes along the y- and z-axes (pointing rightward and upward, respectively) at the arm's accelerations $-3.0721\ \mathrm{m/s^2}$ and $-1.0863.3\ \mathrm{m/s^2}$ (see Fig. 6(e)), respectively. It was held firmly in the optimal grasp at the extreme accelerations $3.2369\ \mathrm{m/s^2}$ and $-4.3491\ \mathrm{m/s^2}$ along the y-axis, and $2.7841\ \mathrm{m/s^2}$ and $-4.7740\ \mathrm{m/s^2}$ along the z-axis. The mouse in the ordinary grasp was

[10]The WAM Arm's payload, though increased from the specified 4 kg after removal of its forearm, slightly exceeded the hand weight of 4.3 kg. As a result, only small accelerations could be generated. Shaking along the x-axis would demand joint torques approaching their limits, causing the Arm to vibrate constantly and its acceleration estimate not to be trustworthy.

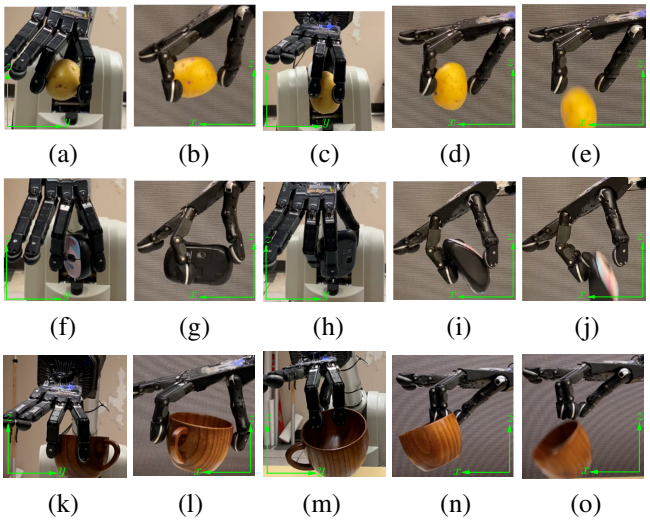


Fig. 6. Grasp tests. Optimal grasps, with qualities 0.3556, 0.2476, and 0.2354, respectively, on (a–b) the potato and (f–g) the mouse from Fig. 5(b), and (k–l) the cup from Fig. 4(a). Arbitrarily chosen grasps (c–e), (h–j), and (m–o) on them with qualities 0.1030, 0.0300, and 0.0467, respectively.

dropped along the same two axes at accelerations $-3.1268\ \mathrm{m/s^2}$ and $2.7251\ \mathrm{m/s^2}$ (see Fig. 6(j)), respectively, while no drop happened from the optimal grasp at the extreme accelerations $2.3406\ \mathrm{m/s^2}$ and $-2.9784\ \mathrm{m/s^2}$ along the y-axis, and $2.9042\ \mathrm{m/s^2}$ and $-5.4973\ \mathrm{m/s^2}$ along the z-axis. The cup was dropped in the ordinary grasp along the z-axis at acceleration $-1.3676\ \mathrm{m/s^2}$ (see Fig. 6(o)) but held firmly at the extreme accelerations $2.7364\ \mathrm{m/s^2}$ and $-3.8293\ \mathrm{m/s^2}$ along the z-axis, and at $3.0503\ \mathrm{m/s^2}$ and $-3.8418\ \mathrm{m/s^2}$ along the y-axis.

VII. Discussion and Future Work

The main contribution of this paper is introduction of the patch tree structure and algorithms operating on it to facilitate grasp analysis and synthesis. Simulation and experiment demonstrate the soundness of grasps on real objects obtained using an existing quality measure. The work does not develop a new grasp metric or a control strategy for grasp realization.

The patch tree does not just have a one-time usage for grasp synthesis. The full potential of its induced hierarchical partitioning, as we believe, will be in planning efficient finger gaits to form a grasp [61], [62], achieve it, and even to manipulate the object, especially when it is a tool. This planning will be at the top level, in conjunction with dynamics-based trajectory planning for gaits at the middle level and control-based execution at the bottom level.

The thresholds δ_G and δ_τ for e-patches are currently hand-tuned. We will look into automatic value setting from balancing the tree height, range of wrench variation, and grasp quality. Subdivision of a non-convex solid needs further investigation to exclude patches such as pockets that are impossible to reach.

VIII. Acknowledgment

We would like to thank Amazon Robotics AI for its donation of the Shadow Dexterous Hand used in this research, and the anonymous reviewers for their valuable feedback.

REFERENCES

[1] K. B. Shimoga. Robot grasp synthesis algorithms: a survey. *Int. J. Robot. Res.*, vol. 15, no. 3, pp. 230–266, 1996.

[2] A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2000, pp. 348–353.

[3] A. Bicchi. Hands for dexterous manipulation and robust grasping: a difficult road toward simplicity. *IEEE Trans. Robot. Automat.*, vol. 16, no. 6, pp. 652–662, 2000.

[4] A. M. Okamura, N. Smaby, and M. R. Cutkosky. An overview of dexterous manipulation. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2000, pp. 255–262.

[5] J. C. Trinkle. A quantitative test for form closure grasps. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1992, pp. 1670–1677.

[6] Y.-H. Liu. Qualitative test and force optimization of 3-D frictional form-closure grasps using linear programming. *IEEE Trans. Robot. Automat.*, vol. 15, no. 1, pp. 163–173, 1999.

[7] B. Mishra. Grasp metrics: optimality and complexity. In Ken Goldberg et al., ed., *Algorithmic Foundations of Robotics*, pp. 137–165. A. K. Peters, Boston, MA, 1995.

[8] M. A. Roa and R. Suárez. Grasp quality measures: review and performance. *Autonomous Robots*, vol. 38, no. 1, pp. 65–88, 2015.

[9] Y. Zheng. An efficient algorithm for a grasp quality measure. *IEEE Trans. Robot.*, vol. 29, no. 2, pp. 579–585, 2013.

[10] S. P. Boyd and B. Wegbreit. Fast computation of optimal contact forces. *IEEE Trans. Robot.*, vol. 23, no. 6, pp. 1117–1132, 2007.

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

[12] X. Markenscoff and C. H. Papadimitriou. Optimum grip of a polygon. *Int. J.Robot. Res.*, vol. 8, no. 2, pp. 17–29, 1989.

[13] Y. C. Park and G. P. Starr. Grasp synthesis of polygonal objects using a three-fingered robot hand. *Int. J.Robot. Res.*, vol. 11, no. 3, no. 163–184, 1992.

[14] C. Ferrari and J. Canny. Planning optimal grasps. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1992, pp. 2290–2295.

[15] J. Ponce, S. Sullivan, J.-D. Boissonnat, and J.-P. Merlet. On characterizing and computing three- and four-finger force-closure grasps of polyhedral objects. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1993, pp. 821–827.

[16] J. Ponce, S. Sullivan, A. Sudsang, J. Boissonnat, and J. P. Merlet. On computing four-finger equilibrium and force-closure grasps of polyhedral objects. *Int. J. Robot. Res.*, vol. 16, no. 1, pp. 11–35, 1997.

[17] J. Ponce, D. Stam, and B. Faverjon. On computing two-finger force-closure grasps of curved 2D objects. *Int. J. Robot. Res.*, vol. 12, no. 3, pp. 263–273, 1993.

[18] I-M. Chen and J. W. Burdick. Finding antipodal point grasps on irregularly shaped objects. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1992, pp. 2278–2283.

[19] E. Rimon and A. Blake. Caging planar bodies by one-parameter two-fingered gripping systems. *Int. J. Robot. Res.*, vol. 18, no. 299–318, 1999.

[20] Y.-B. Jia. Computation on parametric curves with application in grasping. *Int. J. Robot. Res.*, vol. 23, no. 7–8, pp. 825–855, 2004.

[21] A. Rodriguez and M. T. Mason. Grasp invariance. *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 236–248, 2012.

[22] D. Ding, Y.-H. Liu, and M. Y. Wang. On computing immobilizing grasps of 3-D curved objects. In *Proc. IEEE Int. Symp. Comput. Intell. Automat.*, 2001, pp. 11–16.

[23] C. Cai and B. Roth. On the spatial motion of a rigid body with point contact. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1987, pp. 686–695.

[24] D. J. Montana. The kinematics of contact and grasp. *Int. J. Robot. Res.*, vol. 7, no. 3, pp. 17–32, 1988.

[25] Z. Li and J. Canny. Motion of two rigid bodies with rolling constraint. *IEEE Trans. Robot. Automat.*, vol. 6, no. 1, pp. 62–72, 1990.

[26] A. A. Cole, P. Hsu, and S. Sastry. Kinematics and control of multifingered hands with rolling contact. *IEEE Trans. Autom. Control*, vol. 34, no. 4, pp. 398–404, 1989.

[27] N. Sarkar, X. Yun, and V. Kumar. Dynamic control of a 3-D rolling contacts in two-arm manipulation. *IEEE Trans. Robot. Automat.*, vol. 13, no. 3, pp. 364–376, November 1997.

[28] Y. Fan and M. Tomizuka. Efficient grasp planning and execution with multifingered hands by surface fitting. *IEEE Robot. Auto. Lett.*, 4(4):3995–4002, 2019.

[29] M. Kiatos, S. Malassiotis, and I. Sarantopoulos. A geometric approach for grasping unknown objects with multifingered hands. *IEEE Trans. Robot.*, vol. 37, no. 3, pp. 735–746, 2021.

[30] M. Adjigble, N. Marturi, V. Ortenzi, V. Rajasekaran, P. Corke, and R. Stolkin. Model-free and learning-free grasping by local contact moment matching. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Sys.*, 2018, pp. 2933–2940.

[31] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *Int. J. Robot. Res.*, vol. 27, no. 2, pp. 157–173, 2008.

[32] B. Calli, M. Wisse, and P. Jonker. Grasping of unknown objects via curvature maximization using active vision. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Sys.*, 2011, pp. 995–1001.

[33] 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, 2014.

[34] R. Newbsbury, M. Gu, L. Chumbley, A. Mousavian, C. E. J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic, D. Fox, and A. Cosgun. Deep learning approaches to grasp synthesis: A review. *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 3994–4015, 2023.

[35] C. Goldfeder and P. K. Allen. Data-driven grasping. *Autonomous Robots*, vol. 31, pp. 1–20, 2011.

[36] J. Mahler, F. T. Pokomy, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-Net 1.0: A cloud-based network of 3D objects for robust grasp planning using a a multi-armed bandit model with correlated rewards. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1957–1964.

[37] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robot.: Sci. Syst.*, 2017.

[38] R. Paolini, A. Rodriguez, S. S. Srinivasa, and M. T. Mason. A data-driven statistical framework for post-grasp manipulation. *Int. J. Robot. Res.*, vol. 33, no. 4, pp. 600–615, 2014.

[39] J. Mahler, M. Matl, X. Liu, A. Li, D. Gealy, and K. Goldberg. Dex-Net 3.0: Computing robust vacuum suction grasp targets in point clouds uing a new analytic model and deep learning. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 5620–5627.

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

[41] K. Hang, F. T. Pokorny, and D. Kragic. Friction coefficients and grasp synthesis. In *IEEE/RSJ Int. Conf. Intell. Robots Sys.*, 2013, pp. 3520–3526.

[42] G. Kahn, P. Sujan, S. Patil, S. Bopardikar, J. Ryde, K. Goldberg, and P. Abbeel. Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 4783–4790.

[43] J. Weisz and P. K. Allen. Pose error robust grasping from contact wrench space metrics. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 557–562.

[44] J. Kim, K. Iwamoto, J. J. Kuffner, Y. Ota, and N. S. Pollard. Physically based grasp quality evaluation under pose uncertainty. *IEEE Trans. Robot.*, vol. 29, no. 6, pp. 1424–1439, 2013.

[45] F. Bottarel, G. Vezzani, U. Pattacini, and L. Natale. GRASPA 1.0: GRASPA is a robot arm grasping performance benchmark. *IEEE Robot. Auto. Lett.*, vol. 5, no. 2, pp. 836–843, 2020.

[46] Y. Bekiroglu, N. Marturi, M. A. Roa, K. Jean Maxime Adjigble, T. Pardi, C. Grimm, R. Balasubramanian, K. Hang, and R. Stolkin. Benchmarking protocol for grasp planning algorithms. *IEEE Robot. Auto. Lett.*, vol. 5, no. 2, pp. 315–322, 2020.

[47] B. Denoun, M. Hansard, B. León, and L. Jamoone. Statistical stratification and benchmarking of robotic grasping performance. *IEEE Trans. Robot.*, vol. 39, no. 6, pp. 4539–4551, 2023.

[48] K. Hang, M. Li, J. A. Stork, Y. Bekiroglu, F. T. Pokorny, A. Billard, and D. Kragic. Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaption. *IEEE Trans. Robot.*, vol. 32, no. 4, pp. 960–972, 2016.

[49] A. Palleschi, F. Angelini, C. Gllieri, D. W. Park, L. Pallottino, A. Bicchi, and M. Garabini. Grasp It Like a Pro 2.0: A data-driven approach exploiting basic shape decomposition and human data for grasping unknown objects. *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 4016–4036, 2023.

[50] A. Pressley. *Elementary Differential Geometry*. Springer-Verlag, 2001.

[51] A. Van Oosterom and J. Strackee. The solid angle of a plane triangle. *IEEE Trans. Biomed. Eng.*, vol. BME-30, no. 2, pp. 125–126, 1983.

[52] I. T. Jolliffe. *Principal Component Analysis*. Springer, 2 ed., 2002.
[53] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, 2002, pp. 35–57,
[54] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry*. Springer-Verlag, 3 ed., 2008.
[55] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Boca Raton, FL, 1994.
[56] N. Pollard. Synthesizing grasps from generalized prototypes. In *Proc. IEEE Int. Conf. Robot. Automat.*, 1996, pp. 2124–2130.
[57] Y. Zheng and C.-M. Chew. Distance between a point and a convex cone in n-dimensional space: Computation and applications. *IEEE Trans. Robot.*, vol. 25, no. 16, pp. 1397–1412, 2009.
[58] T. Sugihara. Solvability-unconcerned inverse kinematics by the Levenberg-Marquardt method. IEEE Trans. Robot., vol. 27, no. 5, pp. 984–991, 2011.
[59] Berkeley AUTOLab. Dex-net. `https://berkeleyautomation.github.io/dex-net/`.
[60] E. Todorov. Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in Mujoco. In *IEEE Int. Conf. Robot. Autom.*, 2014, pp. 6054-–6061.
[61] Y. Xue, L. Tang, and Y.-B. Jia. Dynamic finger gaits via pivoting and adapting contact forces. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Sys.*, 2023, pp. 8784–8791.
[62] L. Tang, Y.-B. Jia, and Y. Xue. Robotic manipulation of hand tools: The case of screwdriving. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 13883–13890.