A gripper design algorithm for grasping a set of parts in manufacturing lines

Avishai Sintov[1]

Department of Mechanical Engineering, Ben-Gurion University of the Negev, Beer-Sheva.

Roland J. Menassa

Manufacturing Systems Research Lab., Global General Motors R&D, Warren, MI.

Amir Shapiro

Department of Mechanical Engineering, Ben-Gurion University of the Negev, Beer-Sheva.

Abstract

A simple gripper for a robotic arm capable of grasping various objects in manufacturing lines provides great benefits in terms of standardization of grippers, reducing engineering time and costs. This will provide the possibility to reuse the manufacturing line for several products of different geometry without significant changes. The goal is to make a gripper such that it will be a commodity similar to the robot arms. The algorithm, termed 3D-OCOG (3-Dimensional Objects COmmon Grasp search) and proposed in this paper, searches for a common grasp configuration over a set of spatial objects. It maps all possible grasps for each object that satisfy force closure and quality criteria so the grasps could counter-balance external wrenches (forces and torque) applied to the object. The mapped grasps are parameterized as feature vectors in a high-dimensional space. This feature vector describes the design of the gripper. A database of feature vectors is generated for all possible grasps for each object in the feature space. A similarity join based on nearest-neighbor search and classification algorithm are used for intersecting all possible feature vectors over all objects and finding common ones. Each feature vector found is a grasp configuration for the group of objects, which directly implies the gripper design. Simulations of a 3-finger grasp of four meshed objects resulted in several common grasp solutions. Therefore, a designated experimental setup was established composed of three robotic fingers, to simulate the grasp of the test objects. Results of the simulations and experiments validate the feasibility of the proposed algorithm.

Keywords:
Common grasp, Grasp search algorithm, gripper design, Manufacturing line robotics.

1. Introduction

Today, the mass production in assembly lines and manufacturing plants often relies on robotics and automation [1, 2, 3, 4]. Robot arms equipped with grippers are key components to carry out a myriad number of critical automation tasks such as material handling and assembly [5, 6]. Each robotic gripper is designed, built, and optimized for carrying out a specific task while handling a specific part. This makes them very inflexible in terms of variations of objects and tasks. Current automatic solutions to flexibility involve either adding active elements to the grippers in order to adjust to different part geometries or simply adding a variety of grippers adjacent to the robot that can be replaced via a robot-interface plate [7, 8]. Exchanging grippers requires more factory space and consumes time for connecting and calibrations, and thus is costly. Other solutions use highly dexterous grippers which are expensive and not feasible for large objects. The design, manufacture, and testing phase of a current typical gripper consumes a considerable amount of engineering time and adds extra cost to the final product.

The goal of this work is to develop an algorithm that will find a configuration of a simple gripper for grasping a given set of objects. The configuration of the gripper

[1]Corresponding author: Tel.: +972-54-5562555, e-mail: sintova@post.bgu.ac.il.

is defined to be the relative position between the contact points and the surface normals at each contact. A flowchart presenting the high level functionality of the algorithm is presented in Figure 1. Given a set of CAD models of the objects, the goal is to design a gripper that is universal, i.e., able to hold a wide set of components, rather than a single instance of it, for multiple tasks. We propose a novel solution for designing a simple gripper able to do so. Specifically, our aim is to develop an algorithm for computing the grasp configuration suitable for a set of objects. As we discuss, in the design of an industrial gripper, the final configuration has to be simple and low cost; thus, it has to be with minimal degrees of freedom (besides simple degrees of freedom for applying contact forces such as clamping) and qualitative.

This paper presents an algorithm for finding a common grasp for a set of objects. It presents parameterization of the grasps, which are force-closure, for each object and using it for classification of the objects with respect to these grasps. In the first stage of the algorithm a *Force Closure Grasp Set* (FCGS) is constructed for every object by sampling all possible grasps (up to mesh size), filtering out those with low grasp quality measure, and representing the possible grasps as feature vectors in the feature space. Each feature vector is constructed in a unique form to injectively define the grasp invariant to any reference frame. The feature vector implies the grippers' design based on the common grasp. The next step of the algorithm is nearest-neighbor based the similarity join, for finding pairs of common feature vectors in the FCGS of all objects. Finally, classification is done in order to find minimal feature vectors that cover the whole set of objects. In this work, the ability and kinematics of the gripper are not considered. We assume the ability of the fingers to reach every contact point with no geometry limitations (by design).

This paper extends our previous work on 2D parts [9, 10] for grasping 3D objects. In [11] we presented a general overview of the algorithm for 2D and 3D objects. In this paper we present an extended algorithm with test-runs, experiments and analysis. The main contribution in this paper is in the adaptation of the grasping model and algorithm to 3D objects, a novel algorithm for parameterization of an *n*-finger frictional grasp and a sufficient condition for a non-force-closure grasp which is used for filtering out non-feasible grasps.

The paper is organized as follows. The next section is a summary of related work. Section 3 gives an overview of grasping fundamentals used in this work. The structure of the grasp feature vector and the FCGS generation algorithm is described in section 4. Section 5 presents

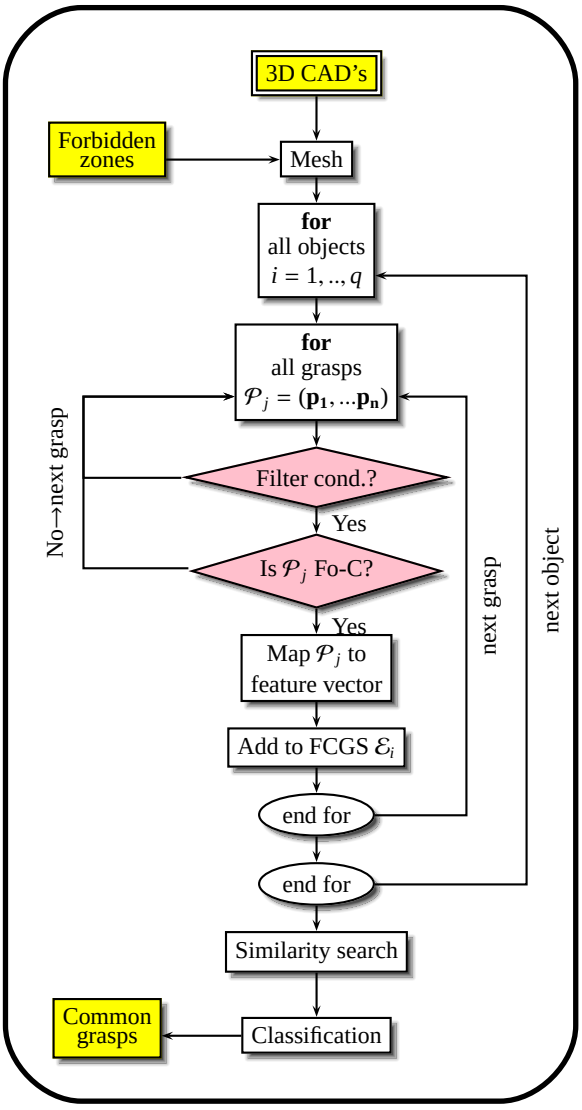

Figure 1: 3D-OCOG algorithm approach.

the main algorithm for similarity join and classification of the common feature vectors. Section 6 provides complexity analysis of the algorithm. Section 7 describes the implementing the proposed algorithms. Section 8 presents the experiments done to validate the concept and simulations. Finally, the last section contains a summary of the work and proposes future work.

2. Related Work

Synthesis and evaluation of robotic grasps have been widely surveyed in literature. The notions of force closure and grasp quality measure are the fundamentals of

a good grasp and there are various definitions for them. Using a generalized vector termed *wrench*, combining force and torque, and the wrench space derived from it, in [12] force-closure grasp criterion used the notion of convexity conditions for a defined grasp map. Mishra et al. [13] presented several conditions for a force-closure grasp for non-frictional grasps. Niparnan and Sudsang [14] used the notion of the wrench space as a 3 or 6 dimensional (for planar or spatial cases, respectively) convex-hull previously introduced by Ferrari and Canny [15] to define geometrical conditions for defining force closure grasp criteria. A force-closure grasp is defined to be capable of resisting external forces and torques applied to the object by forces applied at the contact points with the fingers.

Several algorithms for grasp synthesis have been presented [16]. Roa and Suarez [17] proposed a grasp optimization algorithm based on the convex hull criteria, meaning maximizing the largest external wrench a grasp can resist independent of the wrenches direction. Wang [18] introduced a greedy algorithm for fixture synthesis of frictionless grasps on a discrete point set, minimizing the workpiece positioning errors. Ponce and Faverjon [19] proposed a computation method for the three-finger grasps of 2D polygonal objects with frictional point contact, by using linear sufficient conditions for force closure. Liu [20] addressed the same problem; however, a new sufficient condition for force-closure was introduced, based on dimensionality reduction of the convex-hull. The work done in GraspIt [21] has similarities to this work in the analysis methods of force closure and quality measure. In [22] a randomized grasp generator was introduced. An algorithm has been shown to generate grasp candidate with a method to increase the ratio between force-closure and non-force closure grasps. Such notion has similarities to our work.

Several grasp optimization methods using different grasp quality measures have been presented. A summary of common grasp quality measures can be seen in [23]. Ferrari [15] and Li and Sastry [24] introduced a quality measure (used in this work) based on the external wrench to be resisted, where the first introduced a general measure based on the largest wrench magnitude that the grasp can resist; the second used a task-oriented quality measure defined by the specific wrenches applied to the object during execution of the task. Li and Sastry also introduced a quality that measures how far the grasp configuration is from reaching singularity. Most quality measures mentioned are based on the position of the fingers; other quality measures are based on geometric criteria where the distribution of the fingers on the objects is maximized [25, 26].

Some heuristic approaches for force closure and quality measure analysis were introduced. Niparnan et al. [27] proposed force closure criteria of an *n*-finger grasp, where a heuristic condition, based on vectorial theorems, was used for initial filtering of grasps, thus reducing running time. Prado and Suarez [28] introduced a heuristic approach for generating and measuring a 3-finger grasp, based on geometrical conditions considering the relative orientation and position of the three contact faces.

To the best of our knowledge, no similar work has been done searching for common grasps for the design of grippers for a set of objects. Balan and Bone [29] presented an automated gripper design for a set of objects, however, with limitation for a 3-finger jaw gripper. The work in [30] has similarities to ours. A finger design that can hold an object invariant of its scale or pose is searched. However, in our work we focus on grasping objects of various geometries but of the same scale. Detry et al. [31] provides analysis of strategies for grasping objects with correspond ace to the designated task. Further, by learning of these strategies, the known grasps are transferred to grasp new objects. Much work has been done in the area of 3D shape similarity comparison algorithms, such as the work described in [32], [33] and [34], presenting algorithms that are used for Internet and local database search, face recognition, image processing, or parts identification. The work of Ohbuchi et al. [35] on shape similarity search uses a generalized feature vector of a 3D polygonal mesh constructed from the moment of inertia, average distance of the surface from the model's axis, and its variance. However, such methods deal with mean parameterization of the geometry (such as volume, shape distribution, moment of inertia) of the objects and cannot be applied for grasping. The work in [36] is based on shape matching for finding the best grasp of a set of objects. The best grasp is found by matching hand poses from a database of objects and human grasp postures. This is done by using a predefined parameterization of the object surface and the hand poses. This shape matching method inspired this work.

Grasping is widely researched in a range of fields: synthesis, grasp quality measure and grippers design. Grasp synthesis usually deals with how to grasp a given object with a given gripper based on some quality measure. Gripper design deals with defining the configuration and kinematics of a gripper, usually a dexterous gripper with multiple degrees of freedom. However, there is no work on combining these notions for designing a minimal and simple gripper to qualitatively grasp multiple given objects, i.e., find a common grasp con-

figuration.

3. Fundamentals

In this section we elaborate grasping fundamentals that we use in our work. We present the grasp model we use and discuss the notions of force-closure and quality measure.

3.1. Grasping Model

Forces and torques can be represented as wrench vectors in the wrench space. A wrench is a 6-dimensional vector (in the case of 3D objects) and is denoted as $\mathbf{w} = \begin{pmatrix}\mathbf{f} & \tau\end{pmatrix}^T \in \mathbb{R}^6$ where $\mathbf{f} \in \mathbb{R}^3$ is the force vector and $\tau \in \mathbb{R}^3$ is the torque vector. Furthermore, a wrench applied at the contact point, $\mathbf{p}_i$, can be described as $\mathbf{w}_i = \begin{pmatrix}\mathbf{f} & \mathbf{p}_i \times \mathbf{f}_i\end{pmatrix}^T$, where $\mathbf{p}_i$ is represented in the object coordinate frame (Figure 2). Friction exists at the contacts between the fingertips of the gripper and the object's surface. Friction can be represented by the simple Coulomb friction model. In this model, forces exerted at the contact point must lie within a cone centered about the surface normal. This is known as the *Friction Cone* (FC)

$$FC = \left\{ \begin{pmatrix} \mathbf{f_{i,1}} \\ \mathbf{f_{i,2}} \\ \mathbf{f_{i,3}} \end{pmatrix} : \left| \sqrt{\mathbf{f_{i,2}}^2 + \mathbf{f_{i,3}}^2} \right| \leq \mu \mathbf{f_{i,1}},\ \forall \mathbf{f_{i,1}} > 0 \right\} , \tag{1}$$

where $\mathbf{f_{i,1}}$ is the normal component, $\mathbf{f_{i,2}}$ and $\mathbf{f_{i,3}}$ are the tangential components at the contact point, and μ is the coefficient of friction. The FC is non-linear and therefore can be approximated with an s-sided convex polytope and every force exerted within the FC can be represented by a linear combination of the unit vectors $\hat{\mathbf{f}}_{\mathbf{ik}} \in FC$ (primitive forces) constructing the linearized friction cone (LFC),

$$LFC = \left\{ \mathbf{f_i} : \mathbf{f_i} = \sum_{k=1}^{s} a_{ik} \hat{\mathbf{f}}_{\mathbf{ik}},\ a_{ik} \geq 0 \right\} \tag{2}$$

where $LFC \subset FC$ and a_{ik} are nonnegative coefficients [20]. The $\hat{}$ sign denotes a unit vector. The associated wrenches can be expressed by the primitive forces as

$$\mathbf{w_i} = \sum_{k=1}^{s} a_{ik} \hat{\mathbf{w}}_{\mathbf{ik}} = \sum_{k=1}^{s} a_{ik} \begin{pmatrix} \hat{\mathbf{f}}_{\mathbf{ik}} \\ \mathbf{p_i} \times \hat{\mathbf{f}}_{\mathbf{ik}} \end{pmatrix} \tag{3}$$

where $\hat{\mathbf{w}}_{\mathbf{ik}}$ are the primitive wrenches associated with the primitive forces.
An n-finger grasp can be represented by the location of all contact points $\mathcal{P} = \{\mathbf{p_1}, ..., \mathbf{p_n}\}$. Equivalently, we can represent the grasp using the matching wrenches applied at the contact points represented in the object coordinate frame $\tilde{\mathcal{W}} = \{\mathbf{w_1}, ..., \mathbf{w_n}\}$. If we consider the friction cones, the wrench set can be expressed by the primitive wrenches

$$\mathcal{W} = \{\mathbf{w_{11}}, \mathbf{w_{12}}, ...\mathbf{w_{1s}}, ..., \mathbf{w_{n1}}, \mathbf{w_{n2}}, ..., \mathbf{w_{ns}}\} \tag{4}$$

Based on the model of the grasp, we now want to define the feasibility of the grasp. Therefore, in the next subsection we present the notion of force closure, which defines whether the grasp is feasible or not.

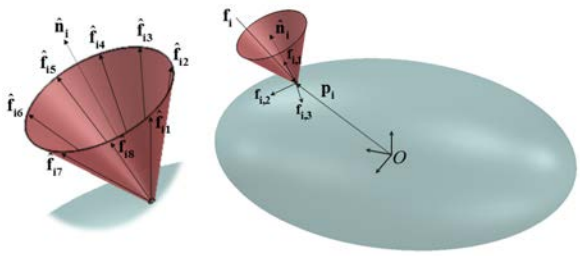

Figure 2: An object and a friction cone at the contact point.

3.2. Force Closure

A grasp is said to be force-closure if it is possible to apply wrenches at the contacts such that any external forces and torques acting on the object can be counter-balanced by the contact forces. A system of wrenches can achieve force-closure when they positively span the entire wrench space. Hence, any external load can be balanced by a non-negative combination of the wrenches [19].

Definition 1. *The Convex Hull (CH) of system $\mathcal{S}$ of vectors $\mathbf{s_1}, ..., \mathbf{s_n}$ is the set of all non-negative convex combinations of the subsets of vectors from $\mathcal{S}$. In other words, the CH is the minimal convex set containing $\mathcal{S}$ and is defined as*

$$CH(\mathcal{S}) = \left\{ \sum_{i=1}^{n} a_i \mathbf{s_i} : \mathbf{s_i} \in \mathcal{S}, \sum_{i=1}^{n} a_i = 1 \text{ and } a_i \geq 0 \right\} \tag{5}$$

where a_i is the convex combination coefficient bounded to be positive, to ensure positive grip (non-sticky fingers).

The CH is a mathematical tool to analyze the grasp and to determine whether it is force-closure.

Definition 2. *The convex hull of the system of contact wrenches is denoted as the Grasp Wrench Set (GWS).*

After the GWS is defined, we use it to check if the CH spans the entire wrench space, meaning whether the grasp is force-closure.

Theorem 1. *[12, 19, 13] A necessary and sufficient condition for a system of $n \times s$ wrenches*

$$\mathcal{W} = \{\mathbf{w_{11}}, \mathbf{w_{12}}, ...\mathbf{w_{1s}}, ..., \mathbf{w_{n1}}, \mathbf{w_{n2}}, ..., \mathbf{w_{ns}}\}$$

to be force-closure is that the origin of $\mathbb{R}^k$ lies in the interior of the convex hull of the contact primitive wrenches. Meaning,

$$O \in \ interior(CH(\mathcal{W})) \tag{6}$$

To practically check if a CH satisfies condition 6, we use theorem 2 for the implementation of it in our algorithm.

Theorem 2. *[37] Let G be a grasp with an associated set of wrenches $\mathcal{W}$, and H_k be a hyper-plane on the boundary of $CH(\mathcal{W})$. The origin O of the wrench space satisfies $O \in interior(CH(\mathcal{W}))$ if and only if $\forall k$ any vector $\mathbf{r} \in interior(CH(\mathcal{W}))$ and O lie in the same open half-space defined by H_k.*

By selecting **r** as a positive linear combination of the grasp wrenches, we can guarantee that it will constantly be in the interior of $CH(\mathcal{W})$. Thus, theorem 2 is used in this work for force closure verification by checking if the interior point **r** and the origin O are on the same side of each of the convex hulls facets.

3.3. Grasp quality measure

As mentioned above, a grasp that is Fo-C can resist external loads; we now need to quantify the quality of the grasp. That is, how much external load it can resist or, in other words, how many resources (in terms of contact force) it needs to apply in order to resist the external load. The quality measure quantifies how much a grasp can resist an external wrench without the fingers losing contact or starting to slip [38]. A higher quality measure reduces object deformations due to contact force and actuator resources. The quality measure will be used as a grasp criterion for the algorithm presented later in this paper and will provide a selection tool for grasps.

There are several known quality measures [23], most of them based on the *task wrench set* (TWS). The TWS is a wrench set of all external wrenches that needs to be applied to the object during execution of a prescribed task. In general, the quality measure is the relation between wrenches that need to be applied (denoted by the TWS) and wrenches that can be applied (denoted by the GWS). The most common quality measure is the largest ball criterion that will be used in this work. This measure is based on a general TWS and is used when there is no prior knowledge of the task forces. In this method, the grasp quality is equivalent to the radius of the largest ball centered at the origin of the GWS and fully contained in the $CH(\mathcal{W})$ [17]. In other words, the grasp quality measure is defined as the distance from the origin of the GWS to the closest facet of the $CH(\mathcal{W})$. Formally, we can say that the quality measure Q is defined as

$$Q = \min_{\mathbf{w} \in \partial CH(\mathcal{W})} \|\mathbf{w}\| \tag{7}$$

where $\partial CH(\mathcal{W})$ is the boundary of $CH(\mathcal{W})$ [15]. The quality measure in this method denotes the weakest net wrench that can be applied to counter-balance the external wrenches in its direction. Figure 3 illustrates the GWS and the largest ball contained in it of a 2D grasp (the GWS of a 3D grasp cannot be illustrated as it is 6-dimensional). The quality measure Q is the radius of the ball. This means that large contact forces would have to be applied when an external wrench is applied along the weakest direction, defined by the vector from the origin to the point where the Q sized ball is tangent to the boundary of $CH(\mathcal{W})$.

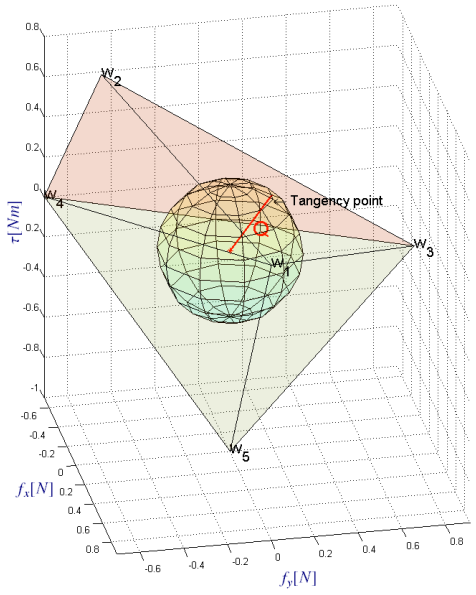

Figure 3: The grasp wrench set (GWS) and the largest ball (TWS).

3.4. Sufficient condition for a non-force-closure grasp

Due to the high complexity of the generation of a convex hull and computing the largest ball criteria, a condition is added in order to filter out non-force-closure grasps prior to the "expensive" computation of the convex-hull. This can prevent excessive convex-hull computation and reduce the required CPU time. The following lemma is a necessary condition for a set of vectors to positively span the entire space.

Lemma 1. *[27] A necessary condition for a set of vectors to positively span $\mathbb{R}^k$ is that the projection of the vectors on any subspace $\mathbb{R}^{t<k}$ must positively span the subspace.*

According to Lemma 1, a necessary condition for a set of vectors to span the wrench space $\mathbb{R}^6$ is for them to span the force space $\mathbb{R}^3$ as well. The following lemmas are necessary conditions for 3-contact-point and n contact-point grasps to positively span the force space $\mathbb{R}^3$. Lemma 2 is a necessary condition for the friction cones of a 3-contact points grasp to positively span the force space.

Lemma 2. *[28] A necessary and sufficient condition for a set of a 3-frictional contacts grasp to positively span the force space $\mathbb{R}^3$ is for the 3 normals (unit vectors) at the contact points $\hat{\mathbf{n}}_\mathbf{i}, \hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}$, to satisfy the condition*

$$\theta = cos^{-1}(\hat{\mathbf{n}}_\mathbf{h} \cdot \hat{\mathbf{n}}_\mathbf{l}) - \frac{\pi}{2} < \tan^{-1}\mu,\ h = i, j, k \qquad (8)$$

where $\hat{\mathbf{n}}_\mathbf{l}$ is the normal to hyper-plane H_l parallel to the plane formed by vectors $\hat{\mathbf{n}}_\mathbf{i}, \hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}$ (Figure 4).

Proof. Sufficient condition: Denote the angle between $\hat{\mathbf{n}}_\mathbf{h}$ for $h = i, j, k$ and plane H_l as θ (the angles between the normals and the base of the formed tetrahedron are equal in a tetrahedron with three equal edges from the same vertex) and the friction angle $\alpha = \tan^{-1}\mu$. H_l is a supporting plane separating $\mathbb{R}^3$ to two half spaces. If $\theta \geq \alpha$, then the three associated friction cones remain in the same half space and cannot positively span the other half space. If $\theta < \alpha$, then the friction cones of the three normals partly cross to the other half space and therefore may span the entire force space $\mathbb{R}^3$. This is equivalent to condition (8), as the left side of the inequality equals θ and the right side of the equation is the friction angle computation according to the coefficient of friction and is equal to α.

Necessary condition: Let $\theta \geq \alpha$ and therefore, the three friction cones of the three normals remain at the same half space formed by plane H_l. We will show that in such a case any point v at the opposite side of the plane H_l cannot be represented by a positive combination of the vectors in the friction cones of the normals $\hat{\mathbf{n}}_\mathbf{i}, \hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}$. The three normals $\hat{\mathbf{n}}_\mathbf{i}, \hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}$ and their friction cones $FC(\hat{\mathbf{n}}_\mathbf{i}), FC(\hat{\mathbf{n}}_\mathbf{j}), FC(\hat{\mathbf{n}}_\mathbf{k})$ span a cone that also lies at the same half space. That is, the convex-hull of the three friction cones, which is shown in Figure 5, does not contain the origin in its interior and therefore does not positively span the force space $\mathbb{R}^3$. In other words, we can say that

$$v \notin CH\left(FC(\hat{\mathbf{n}}_\mathbf{i}), FC(\hat{\mathbf{n}}_\mathbf{j}), FC(\hat{\mathbf{n}}_\mathbf{k})\right) \qquad (9)$$

and therefore cannot represent all vectors in the space with a positive combination of the friction cone. Thus, they cannot positively span the space.

If condition 8 is satisfied and the friction cones pass to the other half space, then the force space is spanned. If not, the force space cannot be spanned. □

The described Lemma is valid only for 3-finger grasp ($n = 3$). The following Lemma is for the case of $n = 4$ and is a sufficient and necessary condition for four normals to positively span the force space.

Lemma 3. *[39] A sufficient and necessary condition for a quartet of normals at the contact points $\hat{\mathbf{n}}_\mathbf{1}, ..., \hat{\mathbf{n}}_\mathbf{4}$, to positively span the force space $\mathbb{R}^3$ (without friction cones), is for the negative of any of these normals to lie strictly inside a cone formed by the other three normals.*

For a grasp with friction ($\mu > 0$), we can expand the condition and say that the negative of any of the normals must lie on or inside the cone formed by the other three normals in order to determine the 4 normals to span the force space. The next lemma is an auxiliary condition for constructing the final sufficient condition for n normals not to span the force space.

Lemma 4. *If there exist n normals $\hat{\mathbf{n}}_\mathbf{1}, ..., \hat{\mathbf{n}}_\mathbf{n} \in \mathbb{R}^3$ that lie at one side of a plane and there does not exist a triplet from the n normals that satisfies Lemma 2, then the friction cones of the normals do not positively span the force space $\mathbb{R}^3$.*

Proof. If n normals lie at one side of a plane H_b in $\mathbb{R}^3$, then there exists a cone where its apex is on the origin and contains all of the normals. In particular, there exists a minimal cone that contains at least three normals on its surface and all other normals are within the cone. This situation is illustrated in Figure 6, where the minimal cone containing all the normals is shown. Checking Lemma 2 for the three normals constructing the minimal cone is enough to determine if the friction cones of the normals positively span the force space $\mathbb{R}^3$, i.e., if some

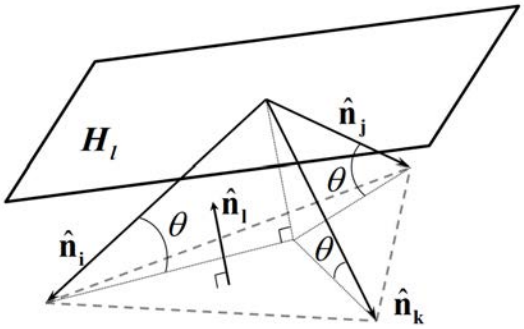

Figure 4: The three normals not positively spanning the entire space.

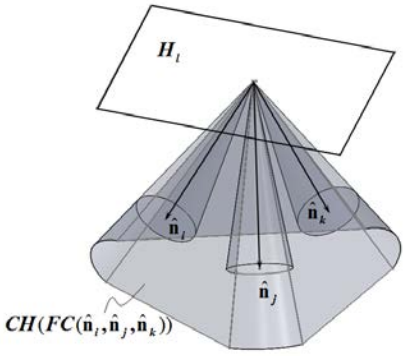

Figure 5: The convex-hull of the three normals not positively spanning the space.

of the friction cones partly cross to the other half space. It should be mentioned, if there exists a plane where all normals lie at one side, we can choose the one formed (according to Lemma 2) by the three normals constructing the cone. If such a triplet satisfying Lemma 2 does not exist, the set of normals does not positively span the force space $\mathbb{R}^3$. □

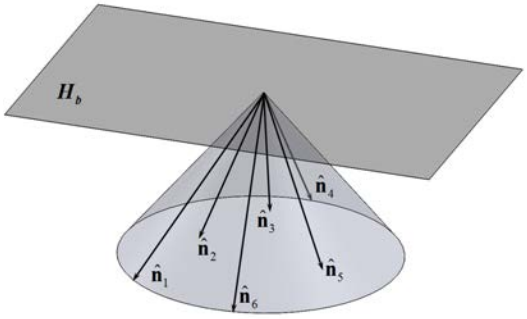

Figure 6: The minimal cone containing all of the normals.

We now use the previous three lemmas to form the following theorem, concluding a sufficient condition for a set of *n* normals **not** to positively span the force space and by that not to be a force closure grasp.

Theorem 3. *A sufficient condition for a set of n-frictional contacts grasp **not** to be force-closure is for the n normals at the contact points $\hat{\mathbf{n}}_1, ..., \hat{\mathbf{n}}_n$ is for the following two conditions to be satisfied:*

*(a) There does **not** exist a triplet of normals $\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j, \hat{\mathbf{n}}_k$ $(i, j, k = 1, 2, ..., n$ and $i \neq j \neq k)$ satisfying Lemma 2; and*

*(b) There does **not** exist a quartet combination of normals $\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j, \hat{\mathbf{n}}_k, \hat{\mathbf{n}}_h$ $(i, j, k, h = 1, 2, ..., n$ and $i \neq j \neq k \neq h)$ satisfying Lemma 3.*

Proof. According to Lemma 1, normals that positively span the wrench space $\mathbb{R}^6$ must span the subspace (force space) $\mathbb{R}^3$ as well. Since Lemma 3 is a necessary condition, we could claim its opposite, meaning; if its condition is not met the grasp is non-force-closure. Then, if it is non-force-closure, the normals do not span the entire force space. Since they do not span the force space there exists a plane where all of the normals lie at one side of the plane. If one vector would have passed to the other side, the whole space would have been spanned. Therefore, for the most, they span a half space and such a plane exists. Hence, if condition (b) of Theorem 3 is satisfied, there exists a plane where all normals lie at one side of the plane. Therefore, the normals (without the friction cones) do not positively span $\mathbb{R}^3$. And according to Lemma 4, if such a plane exists and condition (a) is satisfied, the friction cones of the normals do not positively span $\mathbb{R}^3$. Meaning, all of the normals and their friction cones lie at one half space and, therefore, the grasp is definitely not force-closure. Hence, the presented theorem is a sufficient condition for a grasp to be non-force-closure. □

The following lemma and proposition are used to implement condition (b) of Theorem 3 and are based on the ones proposed in [39].

Lemma 5. *Assume n normals $\hat{\mathbf{n}}_1, ..., \hat{\mathbf{n}}_n$ which positively span $\mathbb{R}^3$. For any rotational map R, the vectors $R\hat{\mathbf{n}}_1, ..., R\hat{\mathbf{n}}_n$ also positively span $\mathbb{R}^3$.*

Proof. Rotational mapping does not change the angles between the normals since $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j = R\hat{\mathbf{n}}_i \cdot R\hat{\mathbf{n}}_j$ and there-

fore if the normals satisfy Theorem 3, the rotated normals will also satisfy the theorem. □

Proposition 1. *[39] Given four normals* $\hat{\mathbf{n}}_\mathbf{i}, \hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}, \hat{\mathbf{n}}_\mathbf{h}$ *and a rotational map* R_c *(for* $c = i, j, k, h$*), which is set to map normal* $\hat{\mathbf{n}}_\mathbf{c} = (x_c\ y_c\ z_c)^T$ $(\|\hat{\mathbf{n}}_\mathbf{c}\| = 1)$ *to align with the z-axis, i.e.,* $R_c\hat{\mathbf{n}}_\mathbf{c} = (0\ 0\ 1)^T$*. A necessary condition for the set of the normals to span* $\mathbb{R}^3$ *is that there exists one normal* $\hat{\mathbf{n}}_\mathbf{i}$ *of the four that satisfy*

$$R_i\hat{\mathbf{n}}_\mathbf{s} = (x\ y\ z)^T \quad s.t.\ z < 0 \quad for \quad s = j, k, h \quad . \tag{10}$$

and the projection of the three normals $\hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}, \hat{\mathbf{n}}_\mathbf{h}$ *on the* $x - y$ *plane of* R_i *coordinate frame (where* $R_i\hat{\mathbf{n}}_\mathbf{i} = (0\ 0\ 1)^T$*) positively span the plane.*

Proof. Assume the three normals $\hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}, \hat{\mathbf{n}}_\mathbf{h}$ lie at one side of the half space of $\mathbb{R}^3$. Following from Lemma 5, rotation of the 4 normals by the same map is allowed. Therefore, rotational map R_i is defined to map the fourth normal $\hat{\mathbf{n}}_\mathbf{i}$ to align with the z-axis. If the z components of the vectors $R_i\hat{\mathbf{n}}_\mathbf{j}, R_i\hat{\mathbf{n}}_\mathbf{k}, R_i\hat{\mathbf{n}}_\mathbf{h}$ are negative, we can say that there exists a plane that separates $\hat{\mathbf{n}}_\mathbf{i}$ and $\hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}, \hat{\mathbf{n}}_\mathbf{h}$ to two half spaces of $\mathbb{R}^3$. Moreover, if the projection of the three normals $\hat{\mathbf{n}}_\mathbf{j}, \hat{\mathbf{n}}_\mathbf{k}, \hat{\mathbf{n}}_\mathbf{h}$ on the $x-y$ plane of R_i coordinate frame positively span $\mathbb{R}^2$, it means that the negative of $\hat{\mathbf{n}}_\mathbf{i}$ is inside the cone formed by the three normals. The analysis for positive spanning of $\mathbb{R}^2$ is done according to the condition presented in [9]. This follows from Lemma 3 defining the four normals to positively span $\mathbb{R}^3$. □

The presented sufficient condition for non-force-closure is a strong tool to filter-out candidate grasps that satisfy the condition and are definitely not force closure. The ones not satisfying the condition are to be further checked using the convex-hull method. Therefore, it is used to filter out candidate grasps and reduce runtime. The efficiency of this condition will be further investigated in the simulations section.

4. FCGS Generation Algorithm

In this section we parameterize the set of all feasible grasps into a feature space. We generate the set of all possible n-finger grasps for each object. We define a novel parametric representation of the grasp, invariant of any coordinate frame. The grasps that are force closure are to be represented as feature vectors in the *Force Closure Grasp Set* (FCGS). This section presents the proposed structure of the grasp feature vector and the method for constructing the FCGS.

4.1. Grasp feature vector

An n-finger grasp of an object B can be defined by a set of n contact points,

$$\mathcal{P} = \left\{\mathbf{p}_\mathbf{i} : \mathbf{p}_\mathbf{i} \in \mathbb{R}^3 \ for\ i = 1, ..., n\right\} \tag{11}$$

on the objects surface, and the normal to the objects surface at each point

$$\mathcal{N} = \left\{\hat{\mathbf{n}}_\mathbf{i} : \hat{\mathbf{n}}_\mathbf{i} \in \mathbb{R}^3 \ for\ i = 1, ..., n\right\} \quad . \tag{12}$$

We need to define a map T, mapping grasp j into a d-dimensional feature vector, injectively representing the grasp,

$$T : \{\mathcal{P}_j, \mathcal{N}_j\} \rightarrow \mathbf{e}_\mathbf{j} \in \mathbb{R}^d, \tag{13}$$

where $\mathbf{e}_\mathbf{j}$ is a d-sized feature vector $\mathbf{e}_\mathbf{j} = (u_1 ... u_d)^T$. This will enable the algorithm to map all possible grasps as feature vectors in the FCGS and intersect them later on. There is no analytical representation for map T and therefore we utilize a parameterization algorithm generating a feature vector from the representation in (11)-(12).

A triangulation of a set $\mathcal{P}$ is a partition of it into simplices, where the vertices are points of $\mathcal{P}$, such that the union of them equals $\mathcal{P}$. Given a set of contact points $\mathbf{p_1}, ...\mathbf{p_n}$, a triangulation of the point set can be made in order to form a polytope whose contact points are its vertices (see Figure 7a). The triangulation of the point set in $\mathbb{R}^3$ can be done with **triangulation algorithms** such as Quickhull [40], space sweep technique [41], or the algorithm proposed in [42]. A triangulation algorithm will output a triangles table K containing triplets of vertices forming the triangles of the polytope (Figure 7b).

We divide the proposed feature vector into two parts; first, is the constraints that define the polytope shape, and second is the constraints that define the contact point normals relative to the polytope geometry.

Proposition 2. *For an n-vertex polytope, there are* $3n - 6$ *degrees of freedom, which determine the polytope's shape.*

Proof. For an n-vertex polytope, each of the n vertices can be varied with three degrees of freedom (DOF), with a total of $3n$ DOF. However, of the total of $3n$ DOF, three correspond to translation and three to rotation of the polytope; so there are $3n - 6$ degrees of freedom for the polytope's shape. □

There are many possible representations for the polytope, some of them depending on the order in which we

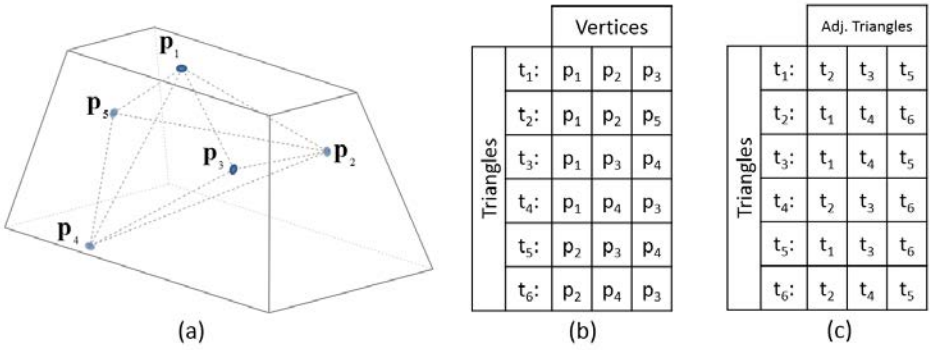

Figure 7: Example of (a) a polytope formed by the contact points, (b) its triangles table, and (c) the triangles adjacency table of the polytope.

parameterize its triangles. Therefore, we describe transformation map T as an algorithm which will uniformly parameterize the grasps polytope. We build the feature vector according to algorithm 1, which is based on parameterization of the triangles by a specific order, ensuring that the feature vector be injective to the polygon. The algorithm first parameterizes the triangle with the largest area and moves on to the one adjacent to it with the edge defined by γ_1. As mentioned in proposition 2, three constraints are needed to parameterize a triangle with total of $3n-6$ constraints for the entire polytope; therefore we need to constrain $\frac{3n-6}{3} = n-2$ triangles of the polytope to fully parameterize it. The parameterization moves on a chain of $n-2$ adjacent triangles, each with 3 parameters. Moreover, the algorithm describes the normals at the contact points relative to the polytope. Each normal needs 2 constraints in order to describe it; therefore we need $2n$ more parameters. Thus, the dimensionality of the feature vector is

$$d = 3n - 6 + 2n = 5n - 6 \tag{14}$$

The algorithm for constructing a $d = 5n-6$ dimensional feature vector for an n-finger grasp and its operation is as follows. The algorithm starts with constructing (step 1) a triangles table (like the one in Figure 7b), and define triangle t_1 as the one with the largest area (steps 2-5), where t_1 is denoted as the base of the polytope. It should be mentioned, that if two or more triangles have the same area within the floating point error, the polytope will be parameterized twice or more. In step 6, an adjacency table Y is formed (example in Figure 7c), denoting for each triangle its adjacency triangles according to common vertices in the triangles table K.

The generation algorithm outputs a sorted list of adjacent triangles for each triangle. Such a table is used to identify the next adjacent triangle to parameterize. Afterwards, we store the indices of triangle t_1 where v_1 and v_2 form the longest edge of the base triangle, v_3 is the third vertex of the triangle. The first parameter to be computed is length d_1 of $\overline{v_1v_2}$ (step 9) where $\mathbf{p_{K(t_i,v_j)}}$ is the location vector of vertex v_j of triangle t_i which are stored in table K. The for-loop (step 10 to 28) computes the shape parameters of the polytope, where for the first triangle (the base), we calculate 2 angles γ_1^1, γ_1^2 and the edge between them d_1. In the other $n-1$ triangles, for triangle t_i we calculate the two angles γ_i^1, γ_i^2 of the triangle that bounds the edge common to triangles t_i and t_{i-1}. Moreover, we calculate the angle ϑ_i between the two triangles t_i and t_{i-1} (step 15). Within each loop iteration, the next triangle t_{i+1} is chosen so that it will be adjacent to t_i but not adjacent to t_{i-1} (according to adjacency table Y). If two of those exist, the one that is adjacent to t_i with the longest edge will be chosen. This condition is made to ensure a single unique parameterization for all polytopes. After t_{i+1} is chosen, we update v_1, v_2, v_3 to store triangle t_{i+1}'s vertices. In the last for-loop (steps 29-32), each normal is parameterized to two angles ϕ_i^1, ϕ_i^2, which are the angles of the normal with triangles t_1 and t_2, respectively. These two triangles were chosen arbitrarily to ensure standard representation of the feature vectors.

Example 1. *A 4-finger grasp can be described by a tetrahedron (4 facet polytope - see Figure 8). According to Proposition 2, six constraints are needed to describe the tetrahedrons shape and 8 parameters to describe the normals directions relative to the tetrahedron, resulting in a 14-dimensional feature vector. Consider the algorithm to calculate face 1 formed by $\Delta p_1p_2p_3$ as the base triangle with the largest area. The parameters representing the shape of the tetrahedron, according to the algorithm, are given as follows. The length of the*

Algorithm 1 Feature vector generation.

Input: Grasp point set $\mathcal{P}_j$,normals at each point $\mathcal{N}_j$.

Output: Feature vector $\mathbf{e_j}$.

Generate triangles table K of $\mathbf{p_1}, ..., \mathbf{p_n}$ with the **triangulation algorithm**.
for $i = 1 \rightarrow 2n - 4$ **do**
  $A(i) = Area(K(i))$ //computes area of triangle i.
end for
$t_1 = \arg\max_i(A_i)$ //index of the largest area triangle denoted as the 'base' of the polytope.
Generate adjacency table Y.
$[v_1, v_2]$ = vertices indices of the largest edge of triangle t_1.
v_3 = index of the third vertex of triangle t_1.
$d_1 = \|\mathbf{p_{K(t_1,v_1)}} - \mathbf{p_{K(t_1,v_2)}}\|$ //length of longest edge of t_1.
for $i = 1 \rightarrow (n - 2)$ **do**
  $\gamma_i^1 = cos^{-1}\frac{(\mathbf{p_{K(t_i,v_2)}} - \mathbf{p_{K(t_i,v_1)}}) \cdot (\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_1)}})}{\|\mathbf{p_{K(t_i,v_2)}} - \mathbf{p_{K(t_i,v_1)}}\| \|\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_1)}}\|}$
  $\gamma_i^2 = cos^{-1}\frac{(\mathbf{p_{K(t_i,v_1)}} - \mathbf{p_{K(t_i,v_2)}}) \cdot (\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_2)}})}{\|\mathbf{p_{K(t_i,v_1)}} - \mathbf{p_{K(t_i,v_2)}}\| \|\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_2)}}\|}$
  $\mathbf{n_{t_i}^f} = \frac{(\mathbf{p_{K(t_i,v_2)}} - \mathbf{p_{K(t_i,v_1)}}) \times (\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_1)}})}{\|\mathbf{p_{K(t_i,v_2)}} - \mathbf{p_{K(t_i,v_1)}}\| \|\mathbf{p_{K(t_i,v_3)}} - \mathbf{p_{K(t_i,v_1)}}\|}$ //$\mathbf{n_{t_i}^f}$ is the normal to the surface of triangle t_i.
  if $i \neq 1$ **then**
    $\vartheta_i = cos^{-1}(\mathbf{n_{t_i}^f} \cdot \mathbf{n_{t_{i-1}}^f})$ //$\mathbf{n_{t_i}^f}$ is the normal to the surface of triangle t_i.
    t_{i+1} = triangle adjacent to $K(t_i)$ but not adjacent to $K(t_{i-1})$ according to Y.
    if t_{i+1} adjacent to t_i with edge v_1, v_3 **then**
      $v_2 = v_3$
    else
      $v_1 = v_3$
    end if
    v_3 =index of the third vertex of triangle t_{i+1}.
  else
    $v_2 = v_3$
    t_2 = triangle adjacent to t_1 on edge $\overline{v_1 v_2}$.
    v_3 = index of the third vertex of triangle t_2.
  end if
end for
for $i = 1 \rightarrow n$ **do**
  $\phi_i^1 = cos^{-1}(\hat{\mathbf{n}}_\mathbf{i} \cdot \mathbf{n_1^f})$ //angle of normal $\hat{\mathbf{n}}_\mathbf{i}$ with triangle t_1.
  $\phi_i^2 = cos^{-1}(\hat{\mathbf{n}}_\mathbf{i} \cdot \mathbf{n_2^f})$ //angle of normal $\hat{\mathbf{n}}_\mathbf{i}$ with triangle t_2.
end for
$\mathbf{e_j} = (\gamma_1^1 \ \gamma_1^2 \ d_1 \ \gamma_2^1 \ \gamma_2^2 \ \vartheta_2 \ ... \ \gamma_l^1 \ \gamma_l^2 \ \vartheta_l \ ... \phi_1^1 \ \phi_1^2 \ ... \ \phi_n^1 \ \phi_n^2)^T$

longest edge $\overline{p_1p_2}$ of the base triangle $t_1 = \Delta p_1p_2p_3$ (the largest area triangle)

$$d_1 = \|\mathbf{p_1} - \mathbf{p_2}\| \tag{15}$$

and the two angles adjacent to the edge

$$\gamma_1^1 = cos^{-1}\left(\frac{(\mathbf{p_3} - \mathbf{p_1}) \cdot (\mathbf{p_2} - \mathbf{p_1})}{\|\mathbf{p_3} - \mathbf{p_1}\|\|\mathbf{p_2} - \mathbf{p_1}\|}\right) \tag{16}$$

$$\gamma_1^2 = cos^{-1}\left(\frac{(\mathbf{p_1} - \mathbf{p_2}) \cdot (\mathbf{p_3} - \mathbf{p_2})}{\|\mathbf{p_1} - \mathbf{p_2}\|\|\mathbf{p_3} - \mathbf{p_2}\|}\right). \tag{17}$$

The next two angles are the angles of the adjacent triangle $t_2 = \Delta p_2p_3p_4$, which is adjacent to the shared edge $\overline{p_2p_3}$

$$\gamma_2^1 = cos^{-1}\left(\frac{(\mathbf{p_4} - \mathbf{p_2}) \cdot (\mathbf{p_3} - \mathbf{p_2})}{\|\mathbf{p_4} - \mathbf{p_2}\|\|\mathbf{p_3} - \mathbf{p_2}\|}\right) \tag{18}$$

$$\gamma_2^2 = cos^{-1}\left(\frac{(\mathbf{p_4} - \mathbf{p_3}) \cdot (\mathbf{p_2} - \mathbf{p_3})}{\|\mathbf{p_4} - \mathbf{p_3}\|\|\mathbf{p_2} - \mathbf{p_3}\|}\right) \tag{19}$$

and the angle between the two triangles

$$\vartheta_2 = cos^{-1}(\mathbf{n_2^f} \cdot \mathbf{n_1^f}) \ . \tag{20}$$

As we have $n = 4$ contact points, we only need $n - 2 = 2$ triangles to parameterize, t_1 and t_2.

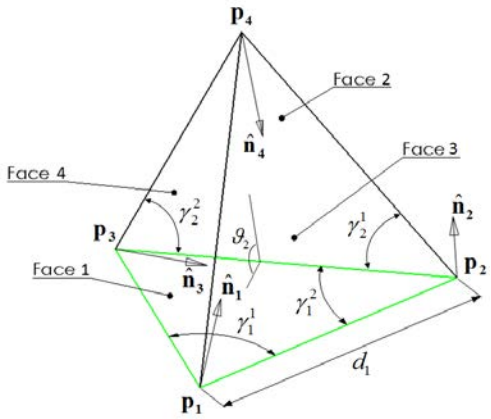

Figure 8: Tetrahedron representing a 4 finger grasp to be parameterized using 14-dimensional feature vector.

The normals direction representation is given by

$$\phi_i^1 = cos^{-1}(\hat{\mathbf{n}}_\mathbf{i} \cdot \mathbf{n_2^f}),\ i = 1, 2, 3, 4\ , \tag{21}$$

$$\phi_i^2 = cos^{-1}(\hat{\mathbf{n}}_\mathbf{i} \cdot \mathbf{n_1^f}),\ i = 1, 2, 3, 4 \tag{22}$$

and therefore, the feature vector will be

$$\mathbf{e} = (\gamma_1^1\ \gamma_1^2\ d_1\ \gamma_2^1\ \gamma_2^2\ \vartheta_2\ \phi_1^1\ \phi_1^2\ \phi_2^1\ \phi_2^2\ \phi_3^1\ \phi_3^2\ \phi_4^1\ \phi_4^2)^T. \tag{23}$$

Example 2. *A 3-finger grasp of object B is described by a single triangle with 3 vertices. The configuration of the grasp is illustrated in Figure 9. The position of the 3 fingers relative to each other can be injectively represented as a triangle by two angles γ_1, γ_2 and the edge length between them d_1, given by equations (15)-(17). However, for three contact points, the polytope degenerates into a triangle and therefore this case is not fully covered by algorithm 1. Therefore, we describe the normals at the contact points by two angles. Angle ϕ_i is the angle between the normal $\hat{\mathbf{n}}_\mathbf{i}$ and the normal to the triangle surface $\mathbf{n^f}$ given by*

$$\phi_i = cos^{-1}(\hat{\mathbf{n}}_\mathbf{i} \cdot \mathbf{n^f}),\ i = 1, 2, 3 \tag{24}$$

and angle ϑ_i is the angle of the normal's projection on the triangle surface with the adjacent triangle's edge, given by Equation (25)

$$\vartheta_i = \begin{cases} \frac{\pi}{2} - sgn(((\hat{\mathbf{n}}_\mathbf{3} \times \mathbf{n^f}) \times \hat{\mathbf{a}}_{\mathbf{3,1}}) \cdot \mathbf{n^f}) \\ \qquad cos^{-1}((\hat{\mathbf{n}}_\mathbf{3} \times \mathbf{n^f}) \cdot \hat{\mathbf{a}}_{\mathbf{3,1}}), \quad i = 3 \\ \frac{\pi}{2} - sgn(((\hat{\mathbf{n}}_\mathbf{i} \times \mathbf{n^f}) \times \hat{\mathbf{a}}_{\mathbf{i,i+1}}) \cdot \mathbf{n_f}) \\ \qquad cos^{-1}((\hat{\mathbf{n}}_\mathbf{i} \times \mathbf{n^f}) \cdot \hat{\mathbf{a}}_{\mathbf{i,i+1}}), \quad i = 1, 2 \end{cases} \tag{25}$$

where $\hat{\mathbf{a}}_{\mathbf{i,j}} = \frac{\mathbf{p_j} - \mathbf{p_i}}{\|\mathbf{p_j} - \mathbf{p_i}\|}$. The feature vector for the 3-finger grasp will be a 9-dimensional vector:

$$\mathbf{e} = (\gamma_1\ \gamma_2\ d_1\ \phi_1\ \phi_2\ \phi_3\ \vartheta_1\ \vartheta_2\ \vartheta_3)^T. \tag{26}$$

4.2. Grasp feasibility criteria

Since we would like to identify all feasible grasps of an object, Algorithm 2 is presented. This algorithm receives the query grasp and first checks if it satisfies the necessary condition of Theorem 2. This condition filters out non-force-closure grasps with low computation cost. If the condition is met, it then computes the convex hull of the primitive wrenches and checks if the force-closure condition (6) is met, using Theorem 2. This condition identifies whether the grasp is force closure. Moreover, to be on the safe side, we check if the quality of the grasp is above a predefined value, Q_d. For now, the value for Q_d is chosen manually according to the estimated number of force-closure grasps to be outputted; however, in practice the minimal quality value

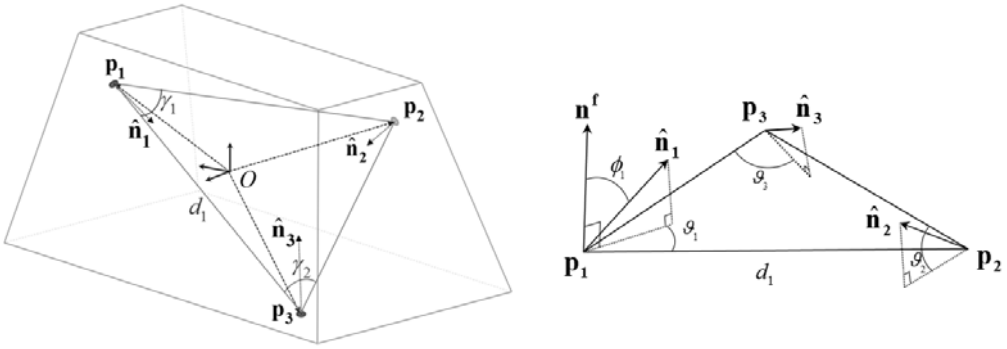

Figure 9: Three contact point grasp.

should be set according to the task to be done. The algorithm returns true if all of these conditions are true and therefore the grasp is feasible. Algorithm 2 defines

Algorithm 2 Grasp feasibility.

Input: An n-finger grasp j of object B, $\{\mathcal{P}_j, \mathcal{N}_j\}$.
Output: True/False - grasp j force-closure/non-force-closure.

if condition (8) is true **then**
  Map $\{\mathcal{P}_j, \mathcal{N}_j\}$ to a system of $n \cdot s$ wrenches $\mathcal{W}_j = (\mathbf{w_{11}}, \mathbf{w_{12}}, ...\mathbf{w_{1s}}, ..., \mathbf{w_{n1}}, \mathbf{w_{n2}}, ..., \mathbf{w_{ns}})$.
  Compute $CH(\mathcal{W}_j)$.
  if $O \in$ *interior*$(CH(\mathcal{W}_j))$ **then**
    Compute the quality measure Q.
    if $Q \geq Q_d$ **then**
      return True.
    end if
  end if
end if
return False.

whether a grasp is feasible, and we can now sample all (up to mesh size) feasible grasps and parameterize them to feature vectors. These feature vectors construct the FCGS which is presented next.

4.3. FCGS generation

We generate the FCGS by exhaustive search of all possible grasps of object B. Algorithm 3 receives a 3D mesh of the original CAD object, consisting a set of points $\tilde{\mathcal{P}} \in \mathbb{R}^3$ and a set of normals at each point $\tilde{\mathcal{N}} \in \mathbb{R}^3$. The size of the mesh is defined according to the grasp tolerances allowed by the manufacturing demands, as will be discussed later on. For a given number of contacts, the algorithm goes over all possible n-finger grasps $\mathcal{P}_j$, and selects only the feasible ones. This is achieved by using Algorithm 2, filtering out all grasps that are not force-closure, and bounding the quality measure of the grasps to be above a predefined value Q_d. The ones selected are transformed into feature vector $\mathbf{e_j}$ and added to the grasp set $\mathcal{E} \subset \mathbb{R}^d$, which is the output of the algorithm. Algorithm 3 is inspired by the one proposed by [43].

Algorithm 3 FCGS generation.

Input: Mesh of object B to be grasped, $\{\tilde{\mathcal{P}}, \tilde{\mathcal{N}}\}$.
Output: FCGS $\mathcal{E} = (\mathbf{e_1}, ..., \mathbf{e_v})$ of object B.

Generate grasp j defined by $\mathcal{P}_j = (\mathbf{p_1}, ..., \mathbf{p_n}) \in \tilde{\mathcal{P}}$. //by sequential scan over the mesh points.
if $\mathcal{P}_j$ is feasible (according to algorithm 2) **then**
  Map grasp j to feature vector $\mathbf{e_j} = (u_1 \ ... \ u_d)^T$ //according to Algorithm 1.
  Label $\mathbf{e_j}$ as force-closure and add to set $\mathcal{E}$.
  Store a pointer from $\mathbf{e_j}$ to $\mathcal{P}_j$.
end if
if $\mathcal{E} = (\mathbf{e_1}, ..., \mathbf{e_v})$ is **not** fully labeled **then**
  go to 1
else
  return grasp set $\mathcal{E}$.
end if

For the q query objects $B_1, ..., B_q$, we apply Algorithm 3 and generate the FCGS $\mathcal{E}_1, ..., \mathcal{E}_q$ of all the objects. In the next section we present the proposed method for nearest-neighbor search and classification in order to find common vectors between the sets, i.e., to find common grasps.

5. Similarity Search Algorithm

Given q sets of vectors of the FCGS $(\mathcal{E}_1, ..., \mathcal{E}_q) \subset \mathbb{R}^{d \times q}$, one for each object, a similarity algorithm pro-

posed in this section will find common vectors of the sets where the distance between them is smaller than a predefined tolerance. This will be done using Nearest-Neighbor search and classification.

5.1. Nearest Neighbor Search

The database structure used in this work is the well-known *kd*-tree. Each FCGS set is represented as a *d*-dimensional binary tree constructing an organized data structure that enables efficient nearest-neighbor search. The *kd*-tree data representation is based on dimension partitioning. For further information on the *kd*-tree refer to [44]. Several algorithms for nearest neighbor search that operates on *kd*-tree have been widely surveyed in literature [45]. They take advantage of the *kd*-tree structure for the search of a point in one set that is the closest to a query point in a different set. It should be noted that the nearest neighbor search algorithm uses the Euclidean distance criterion. However, due to the structure of the feature vector, our *d*-dimensional space is not homogenous and the Euclidean distance does not reflect the grasping demands. Therefore we need another metric, which will be used based on our grasping demands. Moreover, we are not bounded to the closest point to another; if our custom distance criteria fail, we can find the second closest point and so on. Therefore we use the *k*-nearest-neighbor search, which can find up to *k* closest points to a query point.

5.2. Similarity Join

We use a similarity join algorithm to pair points in each two *d*-dimensional sets $\mathcal{E}_i, \mathcal{E}_j$. Function *JoinFCGS* (Algorithm 4) receives the grasp sets $\mathcal{E}_1, ..., \mathcal{E}_q \in \mathbb{R}^d$ and for each two sets, using nearest neighbor (NN) search (implemented using the MATLAB[2] *knnsearch* function), find for each vector in one set the closest one in the other. Function *NearestNeighbor*, which is called within *JoinFCGS*, receives two FCGS sets $\mathcal{E}_i, \mathcal{E}_j$ and outputs two parallel sets $\mathcal{A}, \mathcal{B}$, which are stored in such a way that each vector $\mathbf{a_k} \in \mathcal{A}$ is the closest one to $\mathbf{b_k} \in \mathcal{B}$. As the *d*-dimensional space is not homogenous, each two vectors found cannot be considered as the same vector with only the Euclidean distance. Therefore, we add a custom distance metric that can be changed according to the properties of the grasp, e.g., coefficient of friction, object manufacturing tolerances, etc. Therefore we define the custom metric between two vectors $\mathbf{x} \in \mathcal{E}_i$ and $\mathbf{y} \in \mathcal{E}_i$

$$|\mathbf{x_i} - \mathbf{y_i}| \leq \varepsilon_i, \ \ for\ i = 1, ..., 3n-6 \qquad (27)$$

[2]Matlab® is a registered trademark of The Mathworks, Inc.

where $\varepsilon_1, ..., \varepsilon_{3n-6}$ are predefined tolerances. This means that the two feature vectors are to be inside a hyper-rectangle with edge lengths of $\varepsilon_1, ..., \varepsilon_{3n-6}$. Moreover, we demand that each two respective normals ($\hat{\mathbf{n}}_\mathbf{i}^\mathbf{x}$ from feature vector $\mathbf{x}$ and $\hat{\mathbf{n}}_\mathbf{i}^\mathbf{y}$ from feature vector $\mathbf{y}$) at the contact points (represented in feature vector $\mathbf{e_j}$ as ϕ_i^1, ϕ_i^2 - for example, equation (23)) to both be inside a friction cone, satisfying

$$\cos^{-1}(\hat{\mathbf{n}}_\mathbf{i}^\mathbf{x} \cdot \hat{\mathbf{n}}_\mathbf{i}^\mathbf{y}) \leq 2\alpha \cdot \tan^{-1}\mu, \ \ 0 < \alpha \leq 1 \qquad (28)$$

where α is a safety factor. This condition insures that the angle between each two respective normals is smaller than the friction cone angle multiplied by a predefined safety factor. Conditions (27) and (28) are implemented by function *IsSimilar* (Algorithm 5), which receives two feature vectors and returns *true* or *false* for satisfying the conditions. First, in the first loop, the algorithm checks whether condition (27) is satisfied for the first part of the feature vector (the shape part). Second, the algorithm restores the normal vectors according to the angles given in each feature vector and then compares them according to condition (28).

Algorithm 4 Function $JoinFCGS(\mathcal{E}_1, ..., \mathcal{E}_q)$

Input: FCGS of each object $\mathcal{E}_1, ..., \mathcal{E}_q$.
Output: Registry set $\mathcal{Z}$.

```
for i = 0 → q − 1 do
    for j = i + 1 → q do
        [A, B] = NearestNeighbor(E_i, E_j).
        for k = 1 → size(A) do
            if IsSimilar(a_k ∈ A, b_k ∈ B) = true then
                Z = InsertToZ(a_k, b_k, i, j)
            end if
        end for
    end for
end for
Z = RearrangeReg(Z)
return Z.
```

When two vectors from two different sets are close enough to be considered the same, they are to be inserted to a registry set $\mathcal{Z} \in \mathbb{R}^d$ and the pointer stored to their original grasp sets. The set $\mathcal{Z}$ is a *d*-dimensional database of vectors taken from $\mathcal{E}_1, ..., \mathcal{E}_q$. The vectors inserted to $\mathcal{Z}$ are the ones common to two or more sets of $\mathcal{E}_1, ..., \mathcal{E}_q$. Let $\mathcal{U}_q$ be a *q*-dimensional vector space of binary values, i.e., containing *q*-sized vectors of 0's and 1's. Each common vector $\mathbf{v_i} \in \mathbb{R}^d$ added to $\mathcal{Z}$ is coupled to a compatible vector $\tilde{\mathbf{v}}_\mathbf{i} \in \mathcal{U}_q$. Component *r* of vector $\tilde{\mathbf{v}}_\mathbf{i}$ denotes whether the vector is in the set $\mathcal{E}_r$ if labeled "1" and "0" if not. Therefore, two vectors,

Algorithm 5 Function $IsSimilar(x, y)$

Input: Feature vectors $\mathbf{x}$ and $\mathbf{y}$ representing n-finger grasps.

Output: Are $\mathbf{x}$ and $\mathbf{y}$ within the predefined tolerances (true/false).

for $i = 1 \rightarrow (3n - 6)$ **do**
  if $|x(i) - y(i)| > \varepsilon_i$ **then**
    return False
  end if
end for
for $i = (3n - 5) \rightarrow d$ **do**
  Calculate $\hat{\mathbf{n}}_{\mathbf{i}}^{\mathbf{x}}(x(i), x(i+n))$
  Calculate $\hat{\mathbf{n}}_{\mathbf{i}}^{\mathbf{y}}(y(i), y(i+n))$
  if $\cos^{-1}(\hat{\mathbf{n}}_{\mathbf{i}}^{\mathbf{x}} \cdot \hat{\mathbf{n}}_{\mathbf{i}}^{\mathbf{y}}) > 2\alpha \cdot \tan^{-1}\mu$ **then**
    return False
  end if
end for
return True

Algorithm 6 Function $InsertToZ(\mathbf{e_i}, \mathbf{e_j}, i, j)$

Input: Vector $\mathbf{e_i}$ in FCGS i and vector $\mathbf{e_j}$ in FCGS j.

Output: Updates registry set $\mathcal{Z}$ to contain $\mathbf{a}, \mathbf{b}$.

$\mathbf{v_{i,j}} = \left(\frac{\mathbf{e_i}(1)+\mathbf{e_j}(1)}{2} \ldots \frac{\mathbf{e_i}(d)+\mathbf{e_j}(d)}{2}\right)^T$
$\mathbf{u} = NearestNeighbor(\mathcal{Z}, \mathbf{v_{i,j}})//Find\ NN\ of\ \mathbf{v}.$
if $IsSimilar(\mathbf{v_{i,j}}, \mathbf{u}) = true$ **then**
  $\tilde{\mathbf{u}}(\mathbf{i}) = 1$
  $\tilde{\mathbf{u}}(\mathbf{j}) = 1$
else
  Add point $\mathbf{v_{i,j}}$ to $\mathcal{Z}$.
  $\tilde{\mathbf{v}}(\mathbf{i}) = 1$
  $\tilde{\mathbf{v}}(\mathbf{j}) = 1$
end if

$\mathbf{e_i} \in \mathcal{E}_i$ and $\mathbf{e_j} \in \mathcal{E}_j$, which are considered to be the same, are the input of procedure *InsertToZ* (Algorithm 6). It takes the mean vector $\mathbf{v_{i,j}}$ of the pair and checks whether it already exists in registry set $\mathcal{Z}$ (using $IsSimilar$). If it finds vector $\mathbf{u}$ the same as $\mathbf{v_{i,j}}$, it sets $\tilde{\mathbf{u}}_\mathbf{i}$ and $\tilde{\mathbf{u}}_\mathbf{j}$ to 1, meaning the point exists in $\mathcal{E}_i$ and $\mathcal{E}_j$. If it does not find the point in $\mathcal{Z}$, it adds it and than marks it in the compatible grasp sets.

After the construction of registry set $\mathcal{Z}$, function *RearrangeReg* at the end of algorithm 4 extracts the original vectors that constructed the vectors in $\mathcal{Z}$ from their affiliated FCGS. They are extracted according to the coupled binary vectors that denote in which FCGS they exist. This function recalculates each vector $\mathbf{v_i}$ in $\mathcal{Z}$ to be the mean vector of the original ones extracted from the FCGS. That way, the vectors in $\mathcal{Z}$ denote more accurately the original grasps in which they are originated from.

After a set of vectors common to two or more of the sets $\mathcal{E}_1, ..., \mathcal{E}_q$ are acquired, classification is needed to find the minimal number of grasp configurations that can grasp subsets of the objects. This is done using the classification algorithm presented next.

5.3. Classification

As we acquired a registry set $\mathcal{Z} \in \mathbb{R}^d$, we would now want to classify the set of objects to a minimal number of subsets, where each subset can be grasped by its respective grasp configuration.

Definition 3. *A subset of vectors $\mathcal{H} \subseteq \mathcal{Z}$ is said to cover*[3] *all the primitive sets $\mathcal{E}_1, ..., \mathcal{E}_q$ if there exists $\mathbf{u_1}, .., \mathbf{u}_\sigma \in \mathcal{H}$ such that for each $\mathcal{E}_i$, where $i = 1, ..., q$, there exist at least one $\mathbf{u_j} \in \mathcal{E}_i$ for some $j \in [1, \sigma]$.*

The d-dimensional registry set $\mathcal{Z}$ acquired in the previous algorithms contains the vectors $\mathbf{u_1}, ..., \mathbf{u_m} \in \mathcal{Z}$, which are common to two or more sets of $\mathcal{E}_1, ..., \mathcal{E}_q$. A compatible set $\tilde{\mathbf{u}}_\mathbf{1}, ..., \tilde{\mathbf{u}}_\mathbf{m} \in \mathcal{U}_q$ affiliates the vectors in $\mathcal{Z}$ to the FCGS in which they exist. $\tilde{\mathbf{u}}_\mathbf{i}$ is a q sized binary vector consisting of one in component j if $\mathbf{u_i}$ exists in $\mathcal{E}_j$ and zero if not. The next step is to classify all vectors in set $\mathcal{Z}$ to a minimum subset of vectors $\mathcal{H} \subseteq \mathcal{Z}$ that covers all the primitive sets $\mathcal{E}_1, ..., \mathcal{E}_q$. That is, classify the objects to a minimal number of subsets with its compatible common grasp for each subset.

Assume a vector $\mathbf{u_i} \in \mathcal{Z}$ and its compatible binary vector $\tilde{\mathbf{u}}_\mathbf{i} \in \mathcal{U}_q$ consisting one's or zero's. It can be said that a subset $\mathcal{H} \subseteq \mathcal{Z}$ where $\mathbf{u_1}, ..., \mathbf{u}_\sigma \in \mathcal{H}$, covers $\mathcal{E}_1, ..., \mathcal{E}_q$ if

$$\bigvee_{j=1}^{\sigma} \tilde{\mathbf{u}}_\mathbf{j} = (\vec{1})_{q\times 1} \quad . \tag{29}$$

This means we want to find a minimal number of binary vectors where their unification equals a vector of ones. That is, a minimal number of grasps that can grasp all objects. We seek to find a solution where σ is minimal and with priority equal to 1. If point $\mathbf{u_1}$ is found with its corresponding vector $\tilde{\mathbf{u}}_\mathbf{1}$ satisfying condition (29), it means that $\tilde{\mathbf{u}}_\mathbf{1} = (1, 1, ..., 1)^T$, meaning there is one grasp (defined by feature vector $\mathbf{u}_1$) that is common to all sets. If a number of vectors is needed

[3] The notion of cover is different here from the set cover problem discussed by [46], in which the problem is defined as the selection of as few as possible subsets from a collection of subsets such that every point in a universe set is contained in at least one of the selected subsets.

($\sigma > 1$) to cover the FCGS, it means that the set of objects is divided into σ subsets and each vector is in fact a grasp suitable for the specific subset of objects (see Example 5.3). We will finally obtain σ grasps for q objects divided into σ subsets according to the compatible grasp. Algorithm 7 presents function *Classification*, which receives the set $\mathcal{Z}$ and searches for the minimal number of vectors that satisfy condition (29). Sometimes it is possible to find more than one set of vectors that cover the FCGS; therefore, the function returns the set of vectors that have the maximum grasp quality.

Algorithm 7 Function *Classification*($\mathcal{Z}$)

Input: Registry set $\mathcal{Z}$.
Output: One or more common grasps.

$\sigma = 1$
while $\sigma <$ *size*($\mathcal{Z}$) **do**
  Find all possible combinations $\mathcal{H}_j = (\mathbf{u_1}, ..., \mathbf{u}_\sigma)$ in $\mathcal{Z}$ which satisfy $\bigvee_{i=1}^{\sigma} \tilde{\mathbf{u}}_\mathbf{i} = (\vec{1})_{q\times 1}$
  if success **then**
    Calculate $Q_j = \min_{Q}(\mathbf{u_1}, ..., \mathbf{u}_\sigma)$ of each $\mathcal{H}_j$.
    return $\mathcal{H}_j$ satisfying $\arg\max_{\mathcal{H}_j} Q_j(\mathcal{H}_j)$ // return the grasp with the highest quality measure.
  else
    $\sigma = \sigma + 1$
  end if
end while
return NULL // there is no common grasp.

Let registry set $\mathcal{Z}$ contain five feature vectors $\mathbf{u_1}, ..., \mathbf{u_5} \in \mathcal{U}_q$ that are common to two or more sets of $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4$ ($q = 4$ objects). After the application of function *JoinFCGS* on $\mathcal{E}_1, ..., \mathcal{E}_4$, each vector $\mathbf{u_j}$ is affiliated with a binary vector $\tilde{\mathbf{u}}_\mathbf{j}$. For example, the five outputted binary vectors are shown in (30). Binary vector $\tilde{\mathbf{u}}_\mathbf{1}$ has 1's in the first and fourth positions. Therefore, the respective feature vector $\mathbf{u_1}$ exists in FCGS $\mathcal{E}_1$ and $\mathcal{E}_4$, meaning the corresponding grasp configuration can grasp objects 1 and 4. Binary vector $\tilde{\mathbf{u}}_\mathbf{5}$ equals $(\vec{1})$ and therefore, it corresponds to a grasp configuration that can grasp all the objects. If such a single binary vector $\tilde{\mathbf{u}}_\mathbf{5}$ that satisfies condition (29) does not exist, we would search for the minimal set of binary vectors satisfying the condition. In this example, we can take vectors $\tilde{\mathbf{u}}_\mathbf{2}$ and $\tilde{\mathbf{u}}_\mathbf{4}$, where their union results in $(\vec{1})$. Therefore, we classify the set of objects to subsets, grasp 2 corresponds to grasp configuration $\mathbf{u_2}$ and will grasp objects 2 and 4. Grasp 4 corresponds to grasp configuration $\mathbf{u_4}$ and will grasp objects 1 and 3. Moreover, overlapping may occur, such as the union of $\tilde{\mathbf{u}}_\mathbf{1}$ and $\tilde{\mathbf{u}}_\mathbf{3}$, which results in the possibility of both grasp configurations grasping object 4. In such a case, object 4 will be grasped by the highest quality grasp of the two. In the classification process we divide the objects into classes, where each class has its own grasp configuration.

5.4. Main Algorithm

The main algorithm for finding the common grasp of a set of objects is given in Algorithm 8. It begins with the mesh of the each object and the marking of forbidden zones by the user. After the mesh is acquired, the generation of the FCGS can be done according to Algorithm 3. Next, the similarity join is executed according to Algorithm 4 in order to build the registry set $\mathcal{Z}$. Finally, classification is done according to Algorithm 7 to output the best grasp or grasps that are common to all of the objects.

Algorithm 8 Common grasp search

Input: 3D CAD's of objects $B_1, ..., B_q$.
Output: A common grasp for all objects or common grasps for subsets of the objects.

for $i = 1 \rightarrow q$ **do**
  Mesh object B_i.
  Manually label forbidden grasp regions on mesh of object B_i.
  Generate set $\mathcal{E}_i$ using Algorithm 3.
end for
$\mathcal{Z} = JoinFCGS(\mathcal{E}_1, ..., \mathcal{E}_q)$ using Algorithm 4.
$\mathcal{H} = Classification(\mathcal{Z})$ using Algorithm 7.
return $\mathcal{H} = (\mathbf{u_1}, ..., \mathbf{u}_\sigma)$

It should be noted that because we seek for a simple and minimal gripper, a minimal number of contact points is desired. Therefore, Algorithm 8 is run starting from $n = 3$ and if a solution is not found, n is increased until a solution is found.

The following Theorem presents the final claim for the algorithm to find a solution if one exists.

Theorem 4. *The algorithm is complete: If a solution of a single common grasp or a set of common grasps at the vertices of the objects meshes exists, the algorithm would find it. And if no solution exists, the algorithm reports no solutions and exits.*

Proof. The algorithm maps all possible grasps (up to mesh size) to find the feasible ones. The Feasible ones are those that are force-closure and have a quality measure greater than a defined lower bound. The feasible

$$\tilde{\mathbf{u}}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \quad \tilde{\mathbf{u}}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \quad \tilde{\mathbf{u}}_3 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \tilde{\mathbf{u}}_4 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad \tilde{\mathbf{u}}_5 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \tag{30}$$

grasps are parameterized to a feature vector in the feature space, injectively representing the grasp configuration as a polytope with no reference to any coordinate frame. The feature vectors are added to a set denoted as FCGS of the compatible object. Once all feasible grasps of all objects are mapped to the FCGS sets, Nearest neighbor search is done to find pairs of common vectors among the sets. The pairs found are checked to satisfy tolerance demands and further added to a registry set of common vectors. Classification is then done to find the minimum set of vectors from the registry set that covers all of the FCGS sets. The set consists of feature vectors found to be common grasps of subsets of the objects. The algorithm aims to find a single vector that exists in all of the FCGS or a minimal set of vectors for subsets of objects. The vectors found are in fact the grasp configurations that are able to grasp the compatible subset of objects. As we go through all possible n-finger grasp combinations (up to mesh size), therefore, the algorithm will certainly find a common grasp or a set of common grasps if such exist. Thus, if a single grasp for all objects or a set of grasps for subsets of the objects exist, the algorithm will find them. □

6. Complexity analysis

Assume q objects to be grasped, discretized to a mesh with maximal size k. We generate the FCGS of all possible n-finger grasps for each object. Let N be the maximum number of possible grasps of an object, and it will be the number of possibilities to choose n contact points from k possibilities without repetitions, when the selection order is not important. Therefore, the value of N is $\begin{pmatrix} k \\ n \end{pmatrix} = \frac{k!}{n!(k-n)!}$. The complexity of the Quickhull algorithm [40] for generation of the convex-hull is in the order of $O(n_w \log(n_w))$, where n_w is the number of wrenches forming the convex-hull. In the worst case, we generate a convex-hull for all (up to mesh size) n-finger grasps with discretization of the friction cone to an s-sided convex cone. The complexity for generation of q FCGS sets with N possible grasps is $O(q \cdot N \cdot n \cdot s \cdot \log(n \cdot s))$. We use the Quickhull algorithm as a triangulation algorithm as well (Section 4.1). We perform up to $q \cdot N$ triangulation (for q objects with up to N possible grasps each) of the n-contact points to an n-vertices polytope and therefore it is done with $O(qN \cdot n \log n)$ complexity. Nearest neighbor search, using the kd-tree, between two sets of N vectors each is done by comparing for each vector in one set to all vectors in the second set, and this is done with $O(N \log N)$ complexity. We search for nearest neighbors between each two possible combinations of sets, and there are $\begin{pmatrix} q \\ 2 \end{pmatrix} = \frac{q!}{2!(q-2)!}$ possibilities. Therefore, the search for pairs between all combinations of two sets of size N takes a number of inspections in the order of $O(\frac{q!}{2!(q-2)!} N \log N)$.

Let $m \leq N$ be the number of points in $\mathcal{Z}$ that are common to two or more sets. For each pair found, we search $\mathcal{Z}$ for its existence in it using nearest-neighbor search. In total, m inspections are done with $O(m \log m)$ complexity. In the worst case, $m = N$ and assuming $n \ll k$, summing up all the algorithms parts complexities yields an overall time complexity of $O(k^n)$.

The complexity is exponential to the number of contact points. However, it should be noted that the complexity presented is the worst case possible; the high-complexity comes from the analysis of the convex-hull and by adding Theorem 3 we filter out a large quantity of non-force-closure grasps without generating the convex-hull. Moreover, the FCGS for each object can be computed in parallel, which significantly decreases runtime. Parallel computation is implemented in the following simulations.

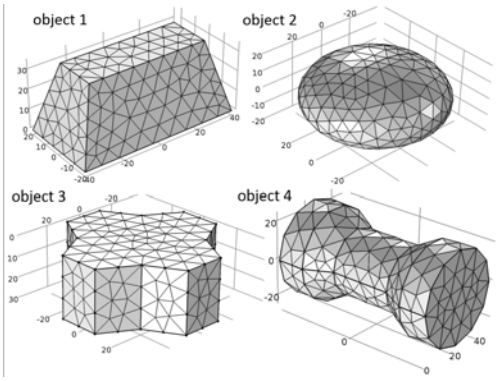

Figure 10: Four meshed objects.

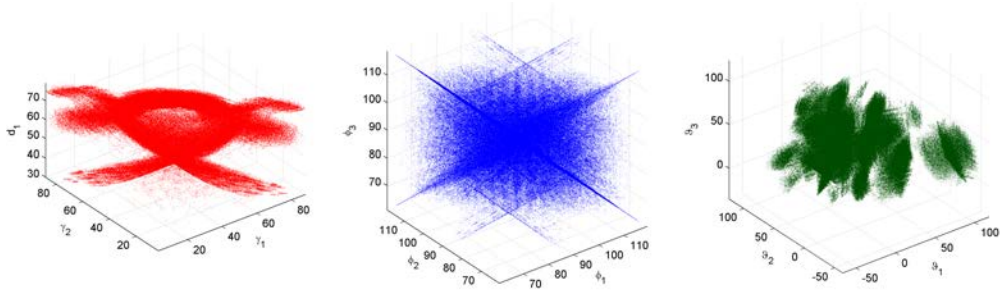

Figure 11: FCGS of object 3.

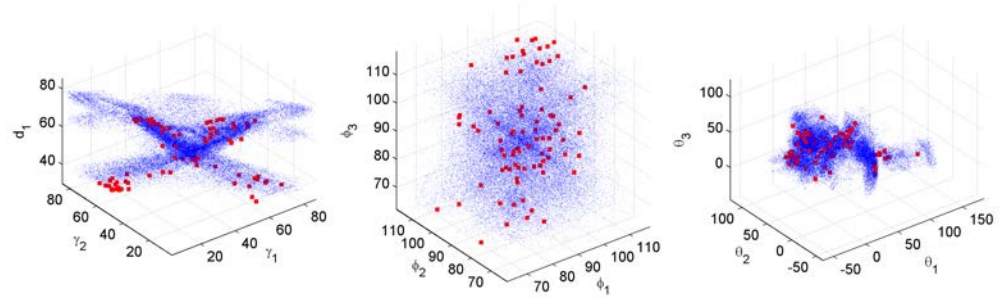

Figure 12: Registry set $\mathcal{Z}$: vectors that are common to two or more objects (blue points) and vectors that are common to all objects (red squares).

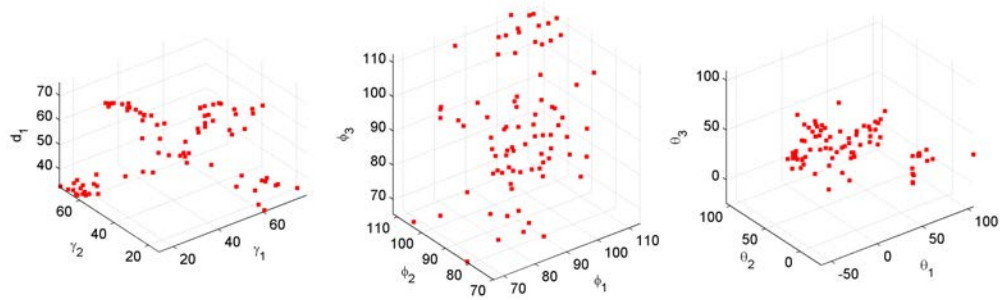

Figure 13: The vectors in registry set $\mathcal{Z}$ that are common to all objects.

7. Test-run of the Algorithm

The following test-runs of the proposed algorithm were implemented in Matlab on an Intel-Core i7-2620M 2.7GHz laptop computer with 8GB of RAM. The operation of the algorithm was done using the MATLAB parallel computing toolbox in order to reduce runtime. The following tests present an example of the algorithm for 3-finger frictional grasps of four objects.

7.1. Implementation and Results

Four objects ($q = 4$) were tested for the implementation of the algorithm and are shown in Figure 10. The objects were meshed using COMSOL Multiphysics[4] to a $k = 450$ triangular mesh. We implement the algorithm with a 3-finger frictional grasp, where the friction cones at each contact point are modeled as 5-sided convex cones ($s = 5$). Each object has an average of 915,630 feasible grasps. For the generation of the FCGS, the construction of the grasp feature vector is done according to Algorithm 1 and as described in example 2 (Section 4.1). We filter-out force-closure grasps that have a quality measure smaller than $Q_d = 0.1$ (refer to Algorithm 2). Figure 11 shows one generated FCGS for object 3. Because it is a 9-dimensional space and only

[4]COMSOL Multiphysics is a registered trademark of COMSOL AB.

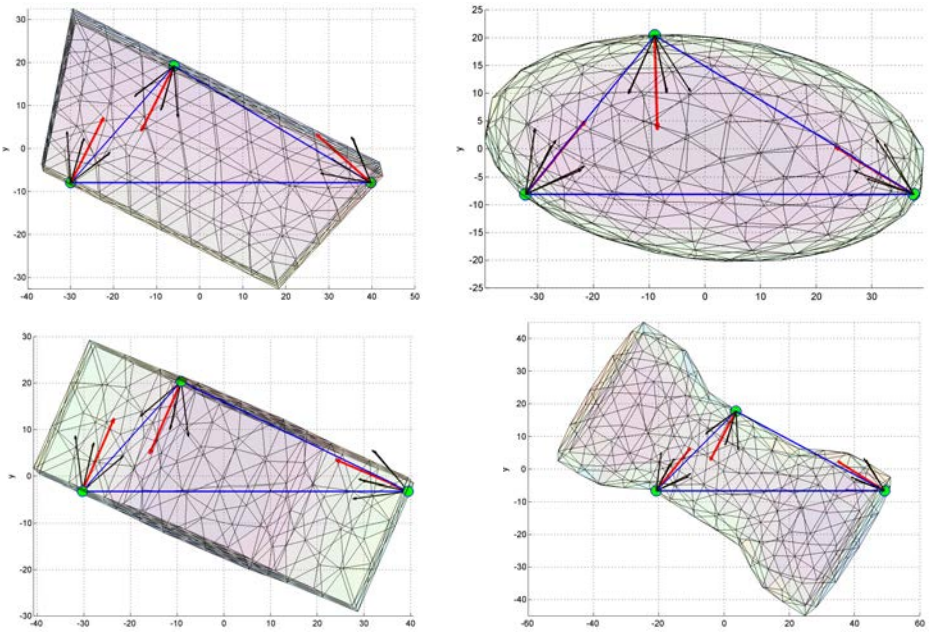

Figure 14: A 3-fingers common grasp illustrated on the four objects.

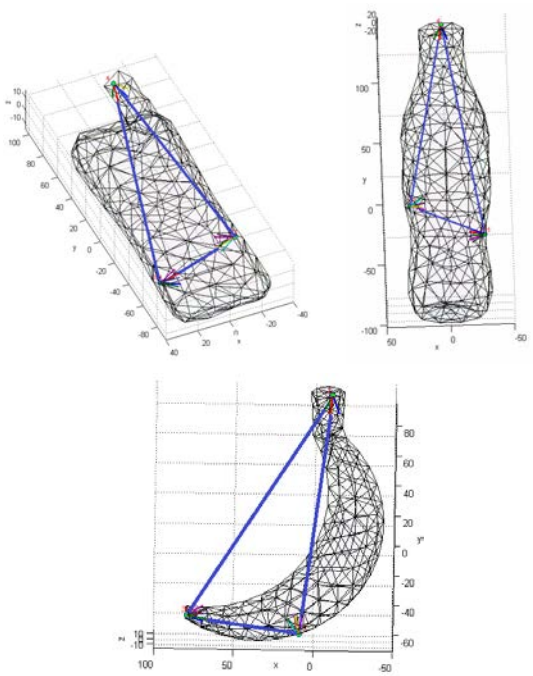

Figure 15: A 3-fingers common grasp of a phone, a soda bottle and a banana.

for illustration, the set is shown in three 3-dimensional projections.

For the similarity search algorithm, the tolerances of the triangles shape $\varepsilon_1[mm], \varepsilon_2[mm], \varepsilon_3[^\circ]$ were chosen such that the edges will not extend by more than 4% of their original length. The parameters to compare (with condition (28)) normals at the contact points are $\alpha = 0.5$ (50% of the friction angle) and $\mu = 1.15$. The friction coefficient value was extracted from an experiment on the materials to be used in the experiments presented in the next section. In the experiment, a surface made of the objects material was tilted until a small object made of the fingertip material that was placed on it started to slip. The angle of the surface in the slippage point was 49^o and its tangent is 1.15, which is the friction coefficient of the two materials [47]. Figure 12 presents the output of the similarity search. Registry set $\mathcal{Z}$ is illustrated and contains 53,796 grasps that are common to two or more objects. Moreover, classification of set $\mathcal{Z}$ provides 82 grasps that are common to all objects and can be seen as red squares in Figure 13. The output of the algorithm would be one grasp out of the 82 with the highest quality measure. The computation run time for this test case took approximately 30 hours. Figure 14 presents the best grasp with quality measure $Q = 0.41$. This grasp is the best common grasp for all four tested objects. Minor differences can be seen between the grasps as the tolerances allowed. The differences can

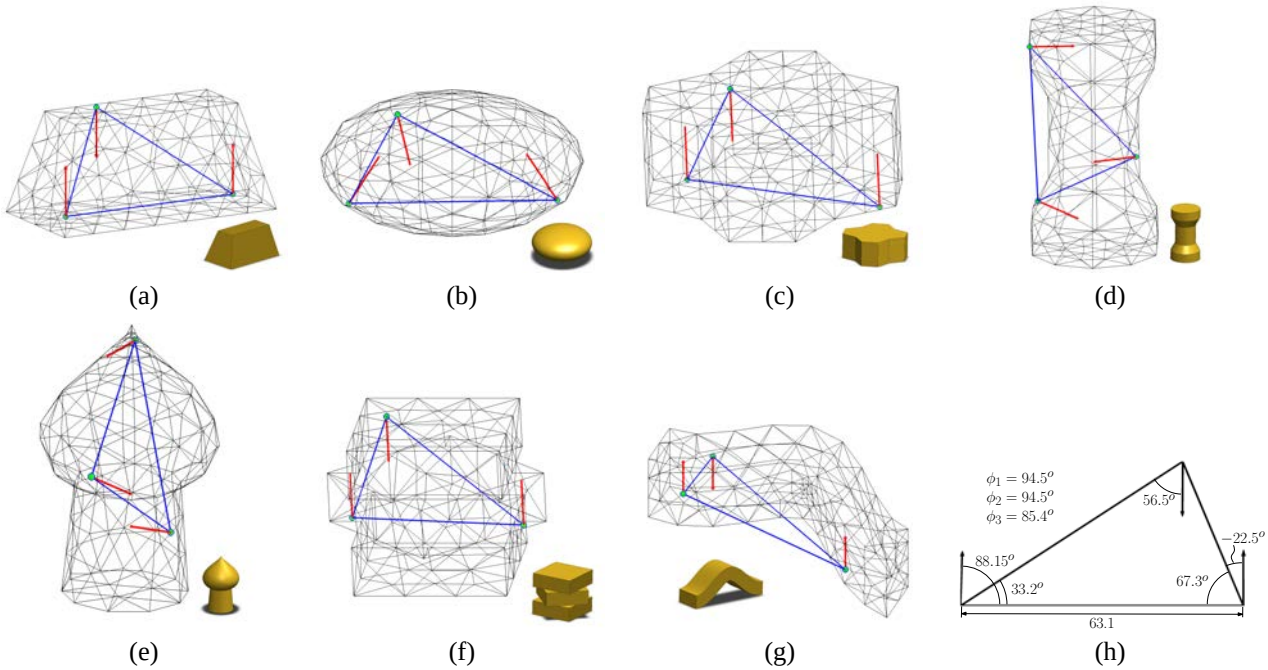

Figure 16: (a)-(g) A 3-fingers common grasp for seven objects and (h) the gripper configuration.

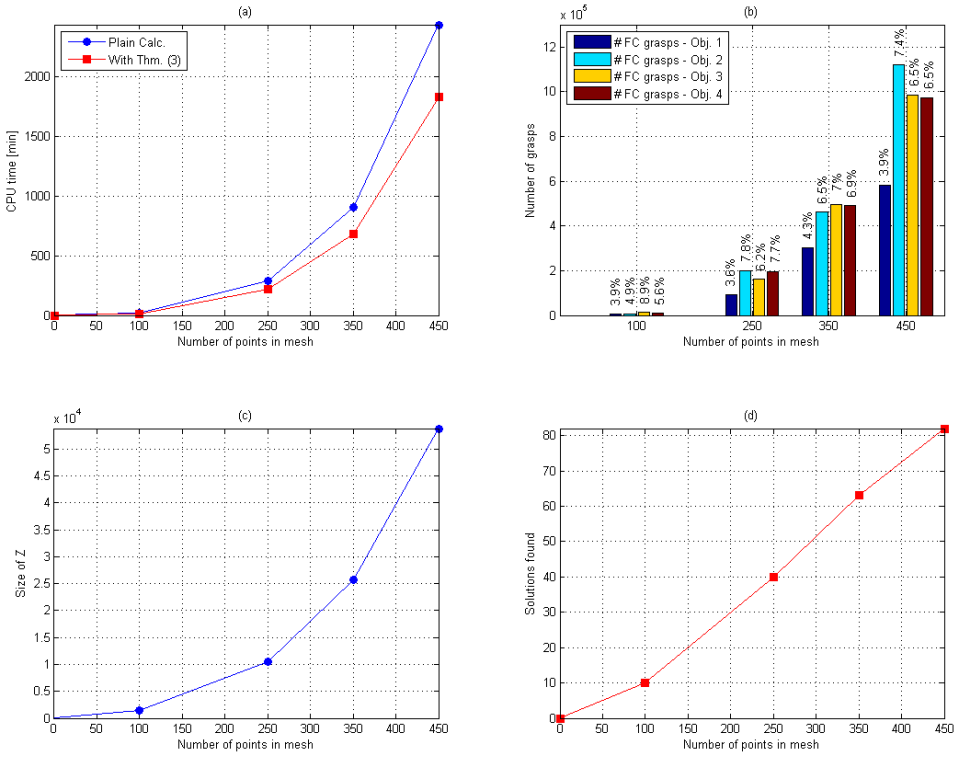

Figure 17: Performance parameters of the algorithm relative to the number of points in the mesh, (a) solution time, (b) total number of grasps and its percent of the number of possible grasps for the object, (c) number of grasps common to 2 or more shapes (size of Z), (d) number of solutions found.

mostly be seen in the normals directions; however, due to the friction cone, such deviation between the normals is allowed. This will be verified in the experiments. The mean feature vector as defined in (26) according to Algorithm 1 is

$$\mathbf{e} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ d_1 \\ \phi_1 \\ \phi_2 \\ \phi_3 \\ \vartheta_1 \\ \vartheta_2 \\ \vartheta_3 \end{pmatrix} = \begin{pmatrix} 46^o \\ 29.6^o \\ 69.7mm \\ 96.5^o \\ 92.1^o \\ 90.5^o \\ 36.6^o \\ -7.5^o \\ 16.6^o \end{pmatrix}. \tag{31}$$

This feature vector is the common gripper design parameters that will be used in the experiments presented in the next section. We present test-runs for 3- and 4-finger grasps. The choice for the proper number of fingers is defined by manufacturing demands, usually to reduce stresses on the grasped object due to weight or operational acceleration. However, with no such demands, the algorithm would choose the minimal number of fingers possible for a feasible grasp.

Figure 15 illustrates a common grasp example of three irregular objects with mesh of $k = 400$. Furthermore, we demonstrate the run of the algorithm with a larger selection of objects. Figures 16a-16g show seven objects and the common grasp configuration. The run was implemented with the same conditions as previously described, and with mesh size $k = 250$. After runtime of 16 hours, 111 common grasps were found and the one with the highest quality measure $Q = 0.32$ is presented in Figure 16h.

Figure 19 demonstrates the implementation of the algorithm for a 4-fingers grasp where $k = 100$. However, it is hard to illustrate such a grasp as a figure, and deviations can be seen between objects as result of different orientations and allowed tolerances. The run time for the 4-fingers computation was approximately 51 hours.

The outputted grasp configuration can now be used to manufacture a simple gripper. The mechanical design is straightforward. The fingers are located at the triangles (in the 3 finger case) vertices. Moreover, each finger has a linear DOF to apply contact force on the object in the computed normals direction.

The proposed algorithm could also be used for regrasping the same object in different areas on its surface with a simple grasp configuration. For example, a mug can be grasped on its handle or on its top. Thus, we can define these different regions as different meshes and search for a their common grasp. Figure 18 shows results of a test-run on such mug. Figure 18a shows the chosen regions to search for a grasp. The friction coefficient was chosen to be $\mu = 0.6$. The most qualitative common grasp is shown in Figure 18b. Another possible grasp is a pinching-like configuration shown in Figure 18c.

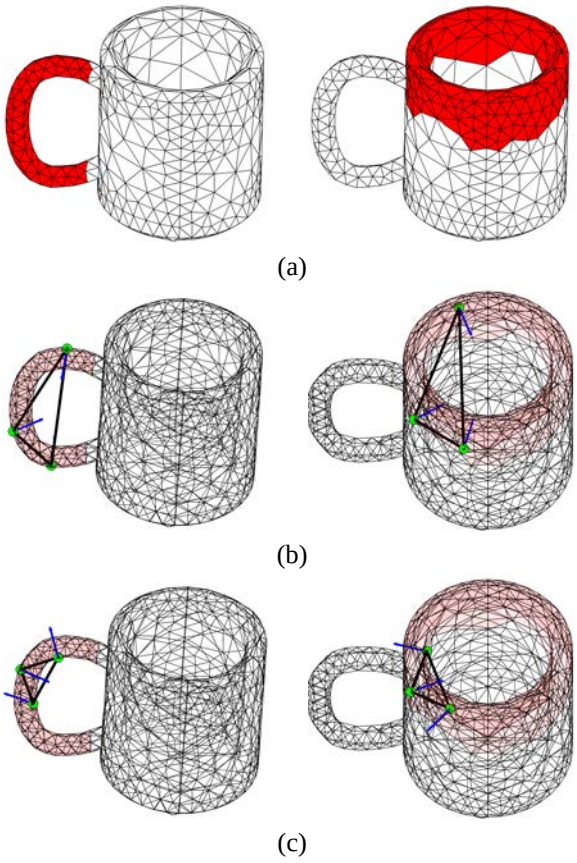

Figure 18: Implementation of the algorithm for regrasping a mug with a single grasp configuration: (a) The red meshes are the allowed region for grasping (color figure). (b)-(c) possible common grasps of the handle and the mugs top region.

7.2. Performance

In Figure 17, some performance parameters of the 3D-OCOG algorithm are shown. Figure 17a presents the CPU runtime of the algorithm as a function of the mesh number. Exponential behavior of the run-time can be seen as a function of the mesh size; this is expected from the $O(k^n)$ complexity calculated previously. Significant improvement can be seen using Theorem 3, which decreased the runtime by approximately 25%. It should be mentioned that the implementation of the algorithm was used with Matlab Parallel Computing Tool-

box. In tests we have made, this parallel computation decreased the runtime by 73%. It should be noted that although the runtime is excessive, it performs offline. Moreover, compared to manual gripper design time of several weeks or months, this is a large improvement.

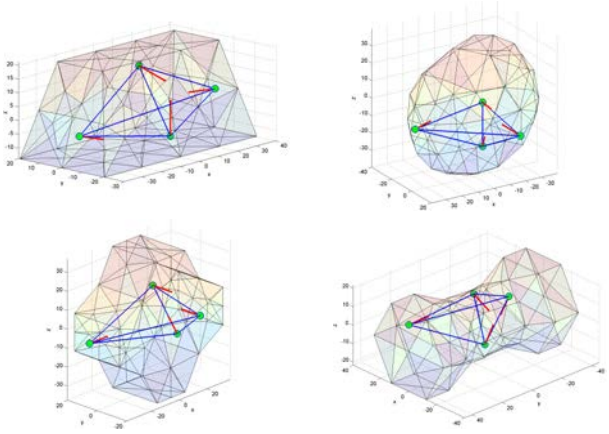

Figure 19: A 4-fingers common grasp illustrated on the four objects.

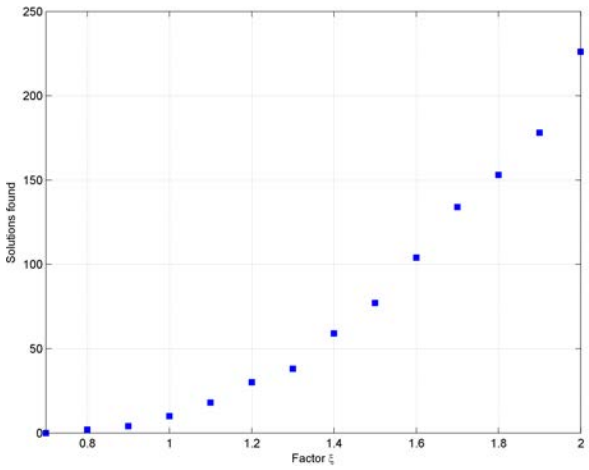

Figure 20: Sensitivity analysis with change of tolerance values.

Figure 17b shows the number of force-closure grasps for each object and its percent of the number of possible grasps (including those that are not force closure) of the object. There is small percentage of feasible grasps over all possible grasps and Theorem 3 contributed to filtering some of them out before the convex-hull criterion. Figure 17c presents the number of grasps that are common to two or more objects. Both Figures 17b and 17c show exponential behavior of the number of grasps relative to mesh size. Figure 17d presents the number of solutions found as a function of the mesh number. It

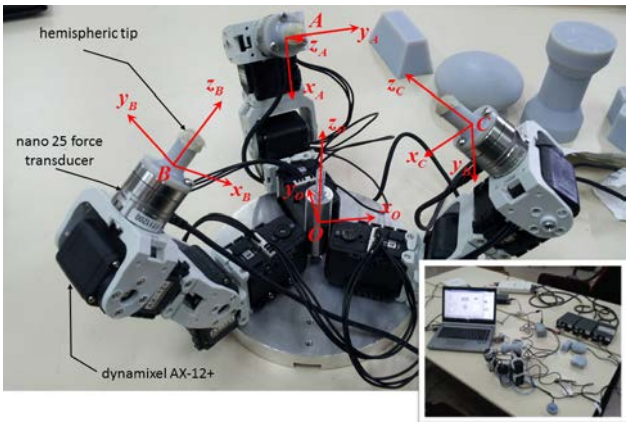

Figure 21: The experimental setup.

can be seen that as we increase the number of triangles in the mesh, we can acquire more solutions. Therefore, as we increase our mesh number, we can calculate more accurate common grasps by the stringent similarity demands defined in Algorithm 5.

Sensitivity analysis was done to examine the number of solutions output with the change in the tolerances. The data were calculated on a 3-fingers grasp with mesh size of k=100. The tolerances presented in the test-run were multiplied by a factor ξ so that the new tolerances are $(\varepsilon_1^{new} \ldots \varepsilon_3^{new})^T = \xi \cdot (\varepsilon_1 \ldots \varepsilon_3)^T$, and the friction angle is $\xi\alpha \tan\mu$. Figure 20 presents the change in the number of solutions as a function of factor ξ. Great sensitivity can be seen with change of the tolerances, which concludes that we can acquire many more solutions if we are willing to accept reduced accuracy.

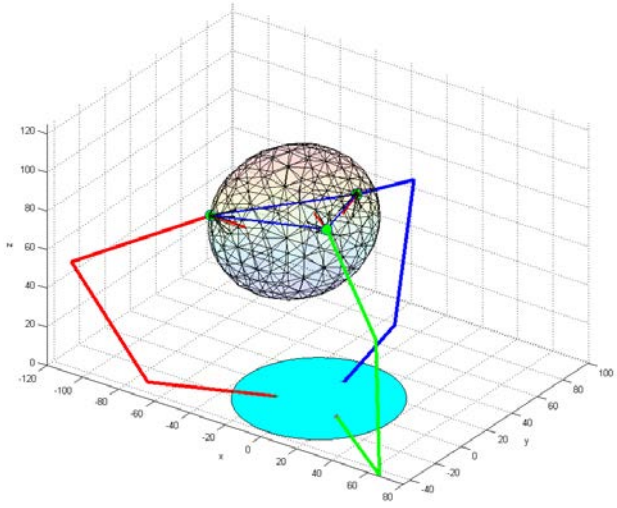

Figure 22: Simulation of the grasp planner.

8. Experiments

A designated experimental system was built to test the 3-finger grasp results outputted from the test-runs. The system is shown in Figure 21 and is composed of 3 robotic fingers with 3-DOF each. The experimental system and some experiments can also be seen in the enclosed video clip. Each finger is built from 3 Dynamixel AX-12 actuators (Bioloid robotic kit), which also serve as the joints. According to the grasp solution, the program automatically computes the angles of the joints such that the grasp triangle is formed between the fingertips, where its plane is horizontal (Figure 22). Moreover, application of forces at the contact points are done by small movements of the fingertips in the direction of the normals according to the outputted grasp configuration. Movement of the fingertips is calculated using a kinematical model of the gripper built for the experiment.

Each finger is equipped with an ATI Nano25 force/torque transducer. The transducer is used for acquiring contact force data at each finger. An external load applied to the objects simulates a disturbance force. All fingers have 5[mm] radius hemispheric caps on their tips to have a point contact. The force/torque transducers were connected to an NI Data Acquisition (DAQ) board. Force data from the transducers was acquired and processed using the Matlab DAQ Toolbox. The force data of the transducers are acquired in the sensors coordinate frames and therefore they are all rotated to a central coordinate frame O.

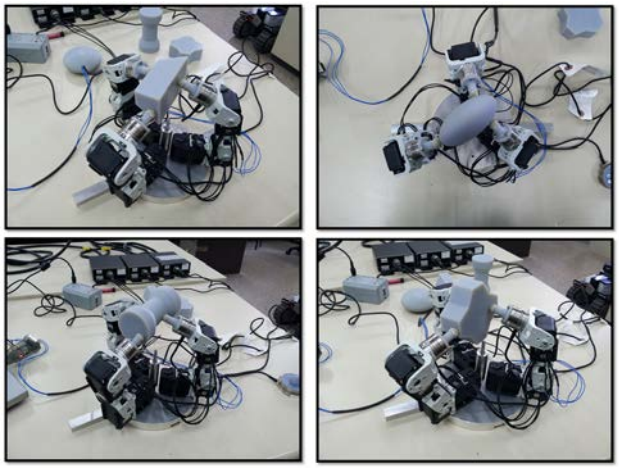

Figure 23: The common grasp of the four objects.

Figure 23 shows the calculated common grasp tested on the 4 objects. The objects were produced using rapid prototyping with Connex[5] 500 3D printer. For the four grasps, the common grasp is defined by the feature vector in (31) and is shown in Figure 24. Note that the normal directions are almost parallel to the triangle surface. We present the results of the experiment on this specific grasp where an external force was applied to the objects in direction $-z_0$ (down). The estimated magnitude of the external force applied to each object is presented in Figure 25. The external force is estimated using equilibrium equations. The results for the reaction forces on the fingertips are shown in Figure 26. The forces are represented in reference frame O. Figure 27 presents the calculation results of the angle between the force vector and the normal at each contact point. It can be seen that when the angles at each grasp are smaller than the fric-

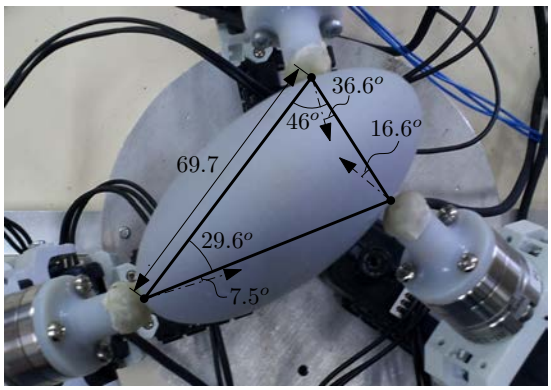

Figure 24: The grasp triangle defined by the mean feature vector.

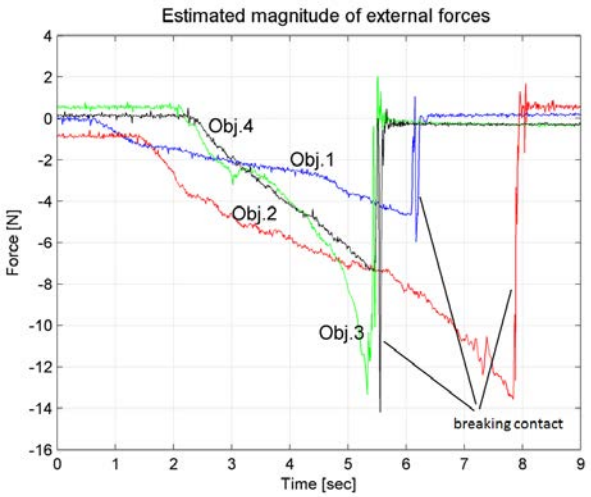

Figure 25: Estimated magnitude of external forces applied to the objects in z direction.

[5]Connex 500 is a registered trademark of Objet Ltd.

tion angle (marked by a dashed line) the grasp is stable. However, when one of the angles exceeds the friction angle, the fingers lose their grip of the object. During the experiments, slippage was observed when exceeding the friction angle, thus validating our results. There are minor deviations in the results due to minor measurement errors of the transducers and the joint angles, which lead to transformation errors. Deviations might also occur due to non-uniform surface roughness, which affects the friction angle. However, despite the errors and deviations, stability of the grasp is maintained under the friction conditions and therefore we can say that this specific common grasp configuration is feasible and stable for the four tested objects.

The same experiment was done under several other external load directions resulting stable grasps until the friction angle is exceeded. Moreover, other grasp configurations outputted from the algorithm were examined with similar results. These experiments validate the grasps and show that they are feasible common grasps for the set of objects.

9. Conclusions

The 3D-OCOG algorithm presented in this paper is based on the search of all possible force closure grasps for each object and their mutual intersection to find the common ones. Each object's force closure grasps are represented as a set in a high-dimensional space, where each point in the set is a feature vector parameterization of the grasp. The feature vector is the key element of the algorithm as it represents the grasp configuration injectively and invariant of any coordinate frame. Such representation of the grasp enables comparison between grasps and efficient similarity search based on nearest-neighbor search. Classification is done between grasps common to two or more sets to find the minimum set of grasps that are able to grasp the set of objects. Simulation results from a Matlab implementation were used to generate the best common grasp for four objects and were followed by experimental verification. The test-runs and experiments verified the feasibility of the proposed algorithm.

The proposed algorithm discretized the objects to find common grasps. The probability to find a common grasp increases as the mesh size increases, however we pay with increased runtime. Moreover, two grasps are considered the same if they are within the boundaries of defined tolerances. Thus, the common grasp will not fit exactly to the objects. In order to avoid increasing the mesh size, future work will deal with post-processing refinement of the common grasp over the object to find an optimized configuration and locations on the objects to achieve accurate fit. This will involve defining an optimization problem constraining the contact points to be on the surface of the objects within bounded regions. These regions should be defined to be inside *Independent Contact Regions* found using the algorithm proposed in [48]. In that case, force closure grasp is maintained within these regions. Such refinement feature will enable the run of the algorithm with relatively coarse mesh and larger tolerances.

The proposed algorithm could also be used for grasping with only two fingers. However, to do so we must leave out the force-closure constraint. In that way, the grasp would not be able to resist wrenches in some directions. Future work could consider modification of the proposed algorithm to two finger grasps. In such case, the algorithm would generate a feature vector parameterizing a single line between the contact points. A difficulty would be on dealing with excessive feasible solutions in such parameterization. Nevertheless, this would make redundant the need for force-closure checks. Thus, it could be a classical implementation of a task based quality measure to select only grasps that can resist the wrenches exerted in the intended task.

It should be mentioned that the size of all analyzed objects should be in the same scale in order to acquire solutions. As the difference in scale between objects increases, the probability to find a common grasp decreases. Such a problem can be dealt in a future work by adding degrees of freedom to the gripper to overcome scale differences that are expressed in the feature vectors. Performance analysis shows that larger mesh results in longer solution time but produces more common grasps. However, the nature of the algorithm enables parallel processing of q objects, each of them on a separate thread, which enables reduction of overall runtime by almost $1/q$.

Future work will involve reducing solution time by using more or other sophisticated filtering conditions. As we require more accurate solutions the convex-hull computation gets more "expensive" and such conditions could replace it or filter-out non-feasible grasps prior to the convex-hull computation. Moreover, by changing the similarity search algorithm to find points in the FCGS that are similar in only $d-1$ dimensions, we can add degrees of freedom to a finger to minimize the distance of the remaining dimension and by this adding more common grasp solutions. Such addition can be beneficial where no solution is found due to the large number of objects, objects scale differences, or when a more accurate solution is needed.

The presented algorithm provides a minimum num-

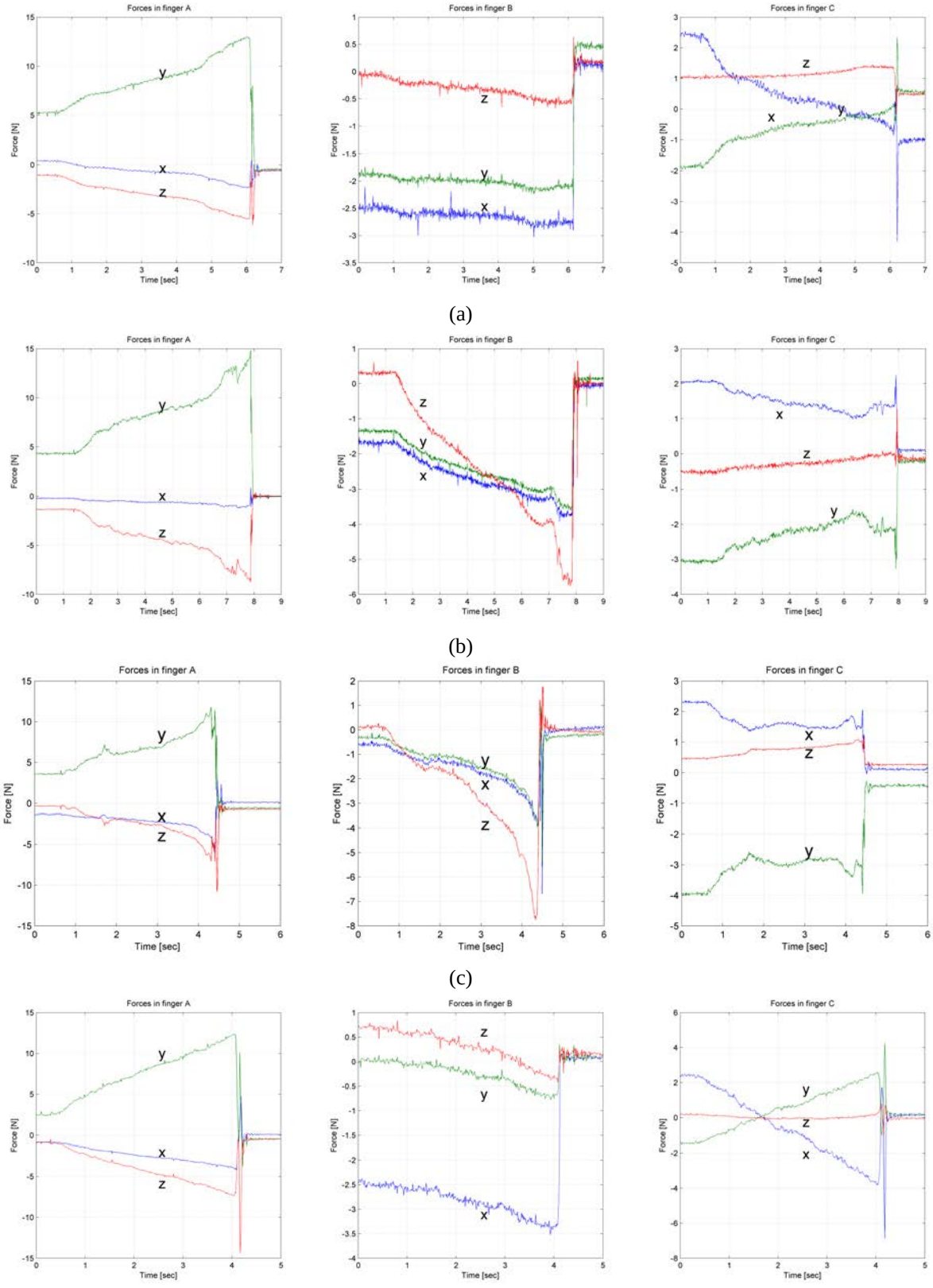

Figure 26: Forces at the fingertips (relative to the *O* reference frame) grasping (a) object 1, (b) object 2, (c) object 3, and (d) object 4.

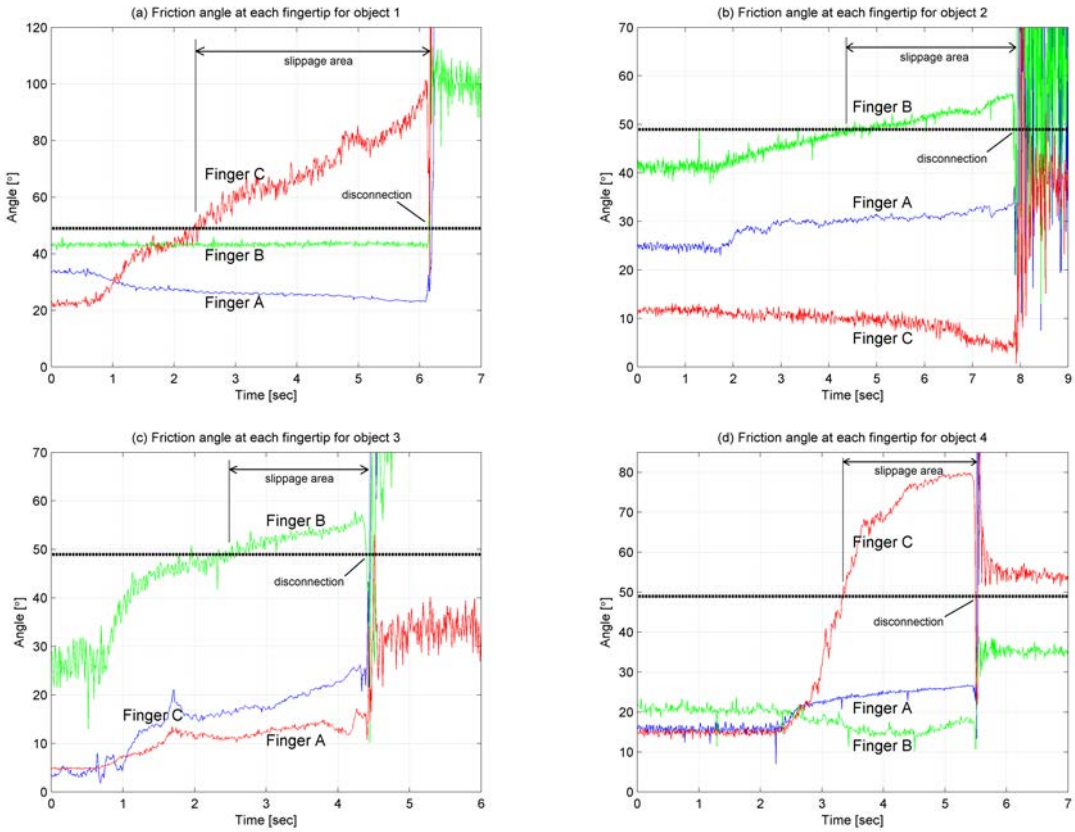

Figure 27: The friction angles at the fingertips change in time. The dashed line is the material friction angle.

ber of grippers needed to grasp a set of parts in a manufacturing line. That is, a single gripper could grasp several parts for several tasks and by that reducing the number of robotic arms in the plant, gripper design and manufacturing time and at the end reduce the final product cost.

Acknowledgements

This work was supported by General Motors Ltd. under PO No. IMEUS-NA-UNIV-11-025, and was partially supported by the Helmsley Charitable Trust through the Agricultural, Biological and Cognitive Robotics Center of Ben-Gurion University of the Negev, and the Pearlstone Center for Aeronautical Engineering Studies.

References

[1] A. Burlacu, C. Copot, C. Lazar, Predictive control architecture for real-time image moments based servoing of robot manipulators, Journal of Intelligent Manufacturing 25 (5) (2014) 1125–1134.

[2] H. Chaudhary, V. Panwar, R. Prasad, N. Sukavanam, Adaptive neuro fuzzy based hybrid force/position control for an industrial robot manipulator, Journal of Intelligent Manufacturing (2014) 1–10.

[3] S. Daoud, H. Chehade, F. Yalaoui, L. Amodeo, Efficient metaheuristics for pick and place robotic systems optimization, Journal of Intelligent Manufacturing 25 (1) (2014) 27–41.

[4] Z. Jakovljevic, P. Petrovic, V. Mikovic, M. Pajic, Fuzzy inference mechanism for recognition of contact states in intelligent robotic assembly, Journal of Intelligent Manufacturing 25 (3) (2014) 571–587.

[5] M. Liu, J. Ma, L. Lin, M. Ge, Q. Wang, C. Liu, Intelligent assembly system for mechanical products and key technology based on internet of things, Journal of Intelligent Manufacturing (2014) 1–29.

[6] D. Scrimieri, R. Oates, S. Ratchev, Learning and reuse of engineering ramp-up strategies for modular assembly systems, Journal of Intelligent Manufacturing (2013) 1–14.

[7] M. Mehrabi, A. Ulsoy, Y. Koren, Reconfigurable manufacturing systems: Key to future manufacturing, Journal of Intelligent Manufacturing 11 (4) (2000) 403–419.

[8] R. Muller, M. Esser, M. Vette, Reconfigurable handling systems as an enabler for large components in mass customized production, Journal of Intelligent Manufacturing 24 (5) (2013) 977–990.

[9] A. Sintov, R. J. Menassa, A. Shapiro, OCOG: A common grasp computation algorithm for a set of planar objects, Robotics and

Computer-Integrated Manufacturing 30 (2) (2014) 124 – 141.
[10] A. Sintov, S. Raghothama, R. Menassa, A. Shapiro, A common 3-finger grasp search algorithm for a set of planar objects, in: Proceeding of the IEEE International Conference on Automation Science and Engineering (CASE), 2012, pp. 1091–1096.
[11] A. Sintov, R. Menassa, A. Shapiro, On the computation of a common n-finger robotic grasp for a set of objects, International Journal of Mechanical Science and Engineering 7 (11) (2013) 34 – 41.
[12] R. M. Murray, Z. Li, S. S. Sastry, A Mathematical Introduction to Robotic Manipulation, 1st Edition, CRC Press, 1994.
[13] B. Mishra, J. T. Schwartz, M. Sharir, On the existence and synthesis of multifinger positive grips, Algorithmica 2 (1987) 541–558.
[14] N. Niparnan, A. Sudsang, Computing all force-closure grasps of 2d objects from contact point set., in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 1599–1604.
[15] C. Ferrari, J. Canny, Planning optimal grasps, in: Proceedings of the IEEE International Conference on Robotics and Automation, 1992, pp. 2290–2295.
[16] J. Bohg, A. Morales, T. Asfour, D. Kragic, Data-driven grasp synthesis - a survey, IEEE Transactions on Robotics 30 (2) (2014) 289–309.
[17] M. Roa, R. Suarez, Geometrical approach for grasp synthesis on discretized 3d objects, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2007, pp. 3283–3288.
[18] M. Wang, An optimum design for 3-d fixture synthesis in a point set domain, IEEE Transactions on Robotics and Automation 16 (6) (2000) 839 –846.
[19] J. Ponce, B. Faverjon, On computing three-finger force-closure grasps of polygonal objects, IEEE Transactions on Robotics and Automation 11 (6) (1995) 868 –881.
[20] Y. H. Liu, Computing n-Finger Form-Closure grasps on polygonal objects, The International Journal of Robotics Research 19 (2) (2000) 149–158.
[21] A. T. Miller, P. K. Allen, Graspit! a versatile simulator for robotic grasping, Robotics & Automation Magazine, IEEE 11 (4) (2004) 110–122.
[22] C. Borst, M. Fischer, G. Hirzinger, Grasping the dice by dicing the grasp, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 4, 2003, pp. 3692–3697 vol.3.
[23] A. Sintov, A. Shapiro, An analysis of grasp quality measures for the application of sheet metal parts grasping, Autonomous Robots (2015) Available online.
[24] Z. Li, S. Sastry, Task oriented optimal grasping by multifingered robot hands, in: Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 4, 1987, pp. 389–394.
[25] E. Chinellato, R. Fisher, A. Morales, A. del Pobil, Ranking planar grasp configurations for a three-finger hand, in: Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 1, 2003, pp. 1133 – 1138 vol.1.
[26] B. H. Kim, B. J. Yi, S. R. Oh, I. H. Suh, Non-dimensionalized performance indices based optimal grasping for multi-fingered hands, Mechatronics 14 (3) (2004) 255 – 280.
[27] N. Niparnan, T. Phoka, A. Sudsang, Heuristic approach for multiple queries of 3d n-finger frictional force closure grasp., in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 1817–1822.
[28] R. Prado, R. Suarez, Heuristic grasp planning with three frictional contacts on two or three faces of a polyhedron, in: Proceedings of the 6th IEEE International Symposium on Assembly and Task Planning: From Nano to Macro Assembly and Manufacturing, 2005, pp. 112 –118.
[29] L. Balan, G. M. Bone, Automated gripper jaw design and grasp planning for sets of 3d objects, J. Field Robotics 20 (3) (2003) 147–162.
[30] A. Rodriguez, M. T. Mason, Grasp invariance, Int. Journal of Robotic Research 31 (2) (2012) 236–248.
[31] R. Detry, C. Ek, M. Madry, J. Piater, D. Kragic, Generalizing grasps across partly similar objects, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2012, pp. 3791–3797.
[32] L. Museros, Z. Falomir, F. Velasco, L. Gonzalez-Abril, I. Mart, 2d qualitative shape matching applied to ceramic mosaic assembly, Journal of Intelligent Manufacturing 23 (5) (2012) 1973–1983.
[33] M. Novotni, R. Klein, A geometric approach to 3d object comparison, International Conference on Shape Modeling and Applications (2001) 154–166.
[34] R. Osada, T. Funkhouser, B. Chazelle, D. Dobkin, Shape distributions, ACM Transactions on Graphics 21 (4) (2002) 807–832.
[35] R. Ohbuchi, T. Otagiri, M. Ibato, T. Takei, Shape-similarity search of three-dimensional models using parameterized statistics, in: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications, 2002, pp. 265–274.
[36] Y. Li, N. S. Pollard, A shape matching algorithm for synthesizing humanlike enveloping grasps, in: Proceedings of the 5th IEEE-RAS International Conference on Humanoid Robots, 2005, pp. 442–449.
[37] M. Roa, R. Suarez, Regrasp planning in the grasp space using independent regions, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 1823 –1829.
[38] C. Borst, M. Fischer, G. Hirzinger, A fast and robust grasp planner for arbitrary 3d objects, in: Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 3, 1999, pp. 1890 –1896 vol.3.
[39] N. Niparnan, A. Sudsang, A heuristic approach for computing frictionless force-closure grasps of 2d objects from contact point set, in: Proceedings of the IEEE Conference on Robotics, Automation and Mechatronics, 2006, pp. 1 –6.
[40] C. Bradford-Barber, D. P. Dobkin, H. Huhdanpaa, The quickhull algorithm for convex hulls, ACM Trans. Math. Softw. 22 (4) (1996) 469–483.
[41] F. P. Preparata, M. I. Shamos, Computational Geometry, Springer-Verlag, New York, 1985.
[42] D. Avis, H. El-Gindy, Triangulating point sets in space, Discrete & Computational Geometry 2 (1987) 99–111.
[43] M. A. Roa, R. Suárez, J. Rosell, Grasp space generation using sampling and computation of independent regions, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 2258–2263.
[44] A. W. Moore, Efficient memory-based learning for robot control, Ph.D. thesis, Cambridge, UK (1990).
[45] J. H. Friedman, J. L. Bentley, R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software 3 (3) (1977) 209–226.
[46] U. Feige, A threshold of ln n for approximating set cover, J. ACM 45 (4) (1998) 634–652.
[47] K. Grote, E. Antonsson, Springer Handbook of Mechanical Engineering, no. 10, Springer, 2009.
[48] M. Roa, R. Suarez, Independent contact regions for frictional grasps on 3d objects, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2008, pp. 1622 –1627.