Dual-Arm Grasp Detection of Chairs

Philipp Badenhoop

1 Introduction

In the field of robotics, we ubiquitously encounter problems that are easy for humans but turn out to be hard to implement on machines. Teaching robots grasping skills is a complex challenge as it typically involves to develop robust perception, planning and control algorithms. During the last years, the task of autonomous bin-picking has received a lot of attention [9] [8]. The general goal is to train a robot to move small objects from one container to another using a single gripper. While these robots already achieve promising success rates, they are only limited to single-arm manipulation.

In this paper, we address the problem of grasping large objects with multiple arms. In particular, we concentrate on the goal of detecting grasp poses for novel objects, but of familiar category, namely chairs. Chairs have a number of interesting properties to consider for research. First, they cannot be efficiently manipulated with only one arm in most cases. Secondly, for each arm, we have to find 6D gripper poses. This differs in complexity to typical bin-picking tasks where it is often enough to work in a planar space. Thirdly, chairs can be grasped in a number of different ways. Our goal is on the one hand to yield grasps with physically plausible stability and on the other hand to find those grasps that can be reached most efficiently by the robot arms. Finally, we aim for a method that avoids the need for tedious labeling as it is often required for deep learning applications. Our proposed multi-arm grasp detection algorithm addresses all of these aspects.

In this work, we evaluate our method purely in simulations using Gazebo [7] and ROS [17]. Our task setup is depicted in Figure 1. There are two Universal Robot UR10 industrial robot manipulators [2] with attached Robotiq 2f 140 dual-jaw grippers [1] that are placed diagonally with respect to the world frame. Two Kinect depth cameras are located along the other diagonal and the coordinate transformations between the camera and the robots are known. The goal is to detect the 6D gripper poses for each arm such that the robots can lift the chair off the ground. While it is also important to develop suitable planning algorithms to move the object to a desired position on a collision-free path, this work only focuses on the perception problem.

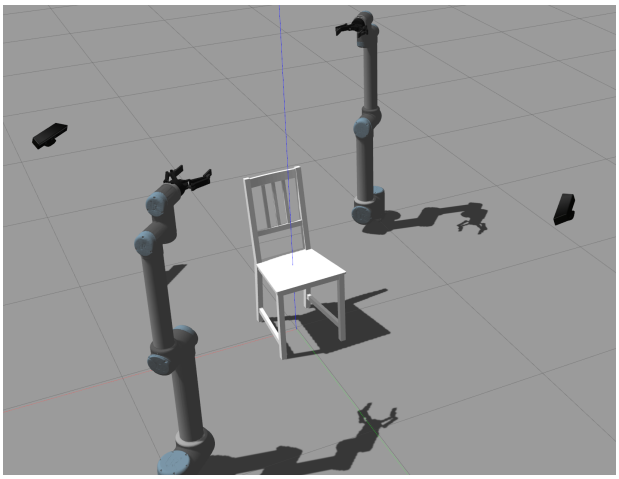


Fig. 1: The task of grasping a chair using two industrial robot manipulators with dual-jaw grippers, two Kinect depth cameras and known coordinate transformations.

This paper is organized as follows: Section 2 presents related work. Section 3 describes our approach while Section 4, the experiments and results. Finally, discussion and conclusions are given in Section 5.

Author
University
E-mail: email

2 Related Work

There has been extensive research in the field of robot grasping over the last decades. A large number of contributions have been devoted to the task of bin-picking. Its simple setup of using a camera to capture small objects on a flat surface makes this problem especially applicable for Deep Learning and Reinforcement Learning approaches. Instead of manually annotating grasp poses on images, more recent works perform data collection by letting either simulated or real robots execute a large number of grasp trials [8] [9]. Although these approaches yield promising results, the training procedure requires a huge amount of time and resources. Furthermore, it is not clear how to translate these methods to more general, full 6D grasp pose detection systems yet.

In the domain of dual-arm grasping, research has mainly focused on the bimanual planning problem [18] [15] [16]. However, quite recently, there has been more effort towards the perception and detection part. Exciting work in the field of humanoid dual-arm grasping of familiar objects has been done by Pavlichenko et al. [12]. The authors employ latent non-rigid shape registration to obtain functional grasp descriptors for the first hand and sample supporting grasp poses for the second hand using Dexterity Network [12]. While the proposed method is able to learn shape related features to infer the functional grasp descriptor, it only allows for a single functional grasp per object category. This makes it only suboptimal for manipulating larger objects that can be grasped in a variety of different ways. Furthermore, in order to train the linear regressor that maps the latent representation of the deformation field to a grasp descriptor, it needs multiple 3D models, each annotated with 6D finger poses for a given object category and its functional grasp. It is not made clear how to obtain these labels.

3 Methodology

Our proposed method for detecting grasp poses for dual-arm manipulation is depicted in Figure 2. Based on the analysis of Bohg et al. [3] on existing grasp detection systems, one can describe the algorithm as a data-driven approach which is split into an offline and an online phase. In the offline phase, the goal is to create a database of grasp experience for a set of 3D example models given as meshes and point clouds. For each such model, we sample tool center point (TCP) poses from the model's point cloud that have a high likelihood of being a valid single arm grasp. We refer to this process as *grasp hypothesis sampling*. Next, we filter out those grasp hypotheses that directly result in a collision state

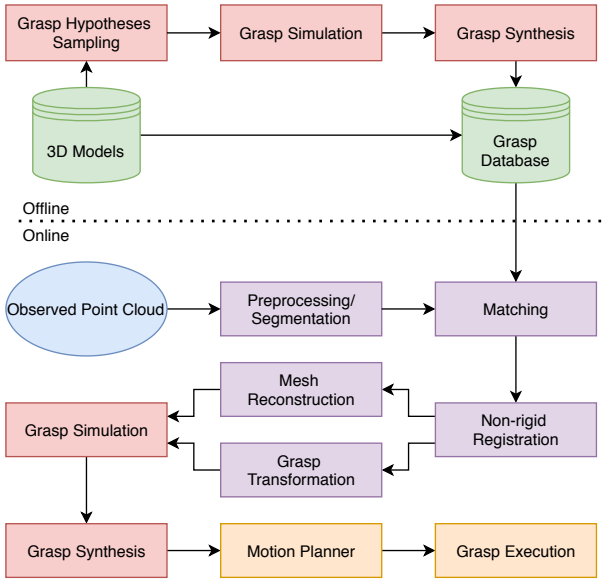


Fig. 2: The proposed pipeline divided into an offline and an online phase. In the offline phase, pairs of sampled and simulated single-arm grasp hypotheses are ranked based on a grasp quality metric before they are stored inside a grasp database. During the online phase, the object of interest is segmented from the scene point cloud and matched with the models of the database. The matched database model is warped to obtain a deformation field that is used to transform the stored grasps into the observed space where they are simulated on the reconstructed mesh. Finally, the grasps are ranked according to grasp quality and reachability and the best candidate is chosen for execution.

between the 3D gripper model of the manipulator in the opened state and the model's mesh. Then, at each filtered pose, a grasp is simulated by closing each of the gripper's fingers individually until contact with the object has been made. This way we obtain the contact normals which are used to compute grasp quality metrics during *grasp synthesis*. Grasp synthesis is the process of ranking all pairs of grasp hypotheses in order to select the best n dual-arm grasps which are finally stored inside the grasp database.

During the online phase, we first combine, preprocess and segment the observed point clouds from the depth cameras to retrieve the points of the object of interest. The obtained cloud is matched against the clouds of the database models to find the model that is most similar to the observed one. Now, we warp the model point cloud to align with the segmented cloud by performing non-rigid registration. This allows us to reconstruct the mesh from the registered cloud and transform the grasps stored inside the database into

the observed space. Similarly to the offline phase, the transformed grasps are run through the grasp simulator and ranked by grasp quality and reachability. The best grasp candidate is sent to the motion planner for execution on the robots.

We like to note that we mentioned the matching phase only for the sake of completeness as we did not actually implemented it in this work. In the following, we will explain all other individual steps in the proposed pipeline in more detail.

3.1 Grasp Hypothesis Sampling

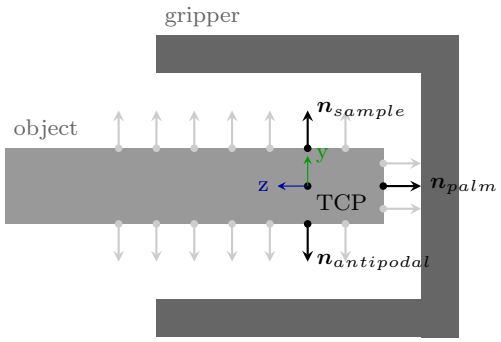


Fig. 3: Sketch of the idea behind grasp hypothesis sampling. For a given sampled point there must exists an antipodal and a palm point in a local neighborhood which have to satisfy certain positional and angular constraints.

We strive to find a mechanism that allows us to efficiently sample TCP poses that result in valid grasps on a model in our database. For a given uniformly sampled point $\boldsymbol{x}_{sample}$ of the model's point cloud, we use very basic geometric assumptions of the dual-jaw gripper's shape such that we can either reject the sampled point or quickly evaluate a suitable TCP position and orientation from the local neighborhood of points. This idea is made clear in Figure 3. If we consider $\boldsymbol{x}_{sample}$ as the contact point where one of the gripper pads touches the object, then there must exist an *antipodal point* $\boldsymbol{x}_{antipodal}$ whose distance is less than the distance between the gripper pads d. Furthermore, the angle between the normals $\boldsymbol{n}_{sample}$ and $\boldsymbol{n}_{antipodal}$ should ideally be 180°. The midpoint between $\boldsymbol{x}_{sample}$ and $\boldsymbol{x}_{antipodal}$ defines the position of the TCP. Next, we find a third point $\boldsymbol{x}_{palm}$ that is located near the gripper's palm. The distance between the TCP position and $\boldsymbol{x}_{palm}$ must be less than the gripper's pad length l and the angle between $\boldsymbol{n}_{sample}$ and $\boldsymbol{n}_{palm}$ should ideally be 90°. Finally,

Algorithm 1: Grasp Hypothesis Sampling

Input : Point cloud with normals: $\boldsymbol{\mathcal{X}}$
Input : Position and normal of the sampled point: $\boldsymbol{x}_{sample}$, $\boldsymbol{n}_{sample}$
Input : Gripper pad distance: d
Input : Gripper pad length: l
Input : Angle thresholds: ϕ, ψ, γ
Output: TCP position $\boldsymbol{p} \in \mathbb{R}^3$ and orientation given as rotation matrix $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ or REJECT

$\boldsymbol{x}_{antipodal} \leftarrow \arg\min_{\boldsymbol{x}}\{$
  $\pi - \angle(\boldsymbol{n}_{sample}, \boldsymbol{n}) + \angle(\boldsymbol{n}_{sample}, \boldsymbol{x}_{sample} - \boldsymbol{x})$
  $\mid (\boldsymbol{x}, \boldsymbol{n}) \in \boldsymbol{\mathcal{X}}$
  $\wedge \|\boldsymbol{x}_{sample} - \boldsymbol{x}\| < d$
  $\wedge \angle(\boldsymbol{n}_{sample}, \boldsymbol{n}) \geq \phi$
  $\wedge \angle(\boldsymbol{n}_{sample}, \boldsymbol{x}_{sample} - \boldsymbol{x}) \leq \psi\}$
if $\boldsymbol{x}_{antipodal}$ *does not exist* **then**
  return REJECT
end
$\boldsymbol{x}_{palm} \leftarrow \arg\min_{\boldsymbol{x}}\{$
  $\frac{1}{\pi/2}|\pi/2 - \angle(\boldsymbol{n}_{sample}, \boldsymbol{n})| + \frac{1}{l}\|\boldsymbol{x}_{sample} - \boldsymbol{x}\|$
  $\mid (\boldsymbol{x}, \boldsymbol{n}) \in \boldsymbol{\mathcal{X}}$
  $\wedge \|\boldsymbol{x}_{sample} - \boldsymbol{x}\| < l$
  $\wedge |\pi/2 - \angle(\boldsymbol{n}_{sample}, \boldsymbol{n})| \leq \gamma\}$
if $\boldsymbol{x}_{palm}$ *does not exist* **then**
  return REJECT
end
$\boldsymbol{p} \leftarrow \frac{1}{2}(\boldsymbol{x}_{sample} + \boldsymbol{x}_{antipodal})$
$\boldsymbol{r}_y \leftarrow \boldsymbol{n}_{sample}$
$\boldsymbol{r}_z \leftarrow \frac{-\boldsymbol{n}_{palm} - \pi_{\boldsymbol{n}_{sample}}(-\boldsymbol{n}_{palm})}{\|-\boldsymbol{n}_{palm} - \pi_{\boldsymbol{n}_{sample}}(-\boldsymbol{n}_{palm})\|}$
$\boldsymbol{r}_x \leftarrow \boldsymbol{r}_y \times \boldsymbol{r}_z$
$\boldsymbol{R} \leftarrow (\boldsymbol{r}_x\ \boldsymbol{r}_y\ \boldsymbol{r}_z)$
return $\boldsymbol{p}$, $\boldsymbol{R}$

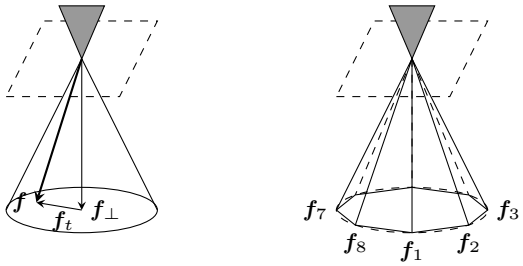
Fig. 4: The grasp hypothesis sampling algorithm. Given a sampled point from a point cloud the algorithm either rejects the sample or outputs a TCP pose based on the found antipodal and palm points.

the y-axis of the TCP frame is given by $\boldsymbol{n}_{sampled}$ and the z-axis is defined as

$$\boldsymbol{r}_z = \frac{-\boldsymbol{n}_{palm} - \pi_{\boldsymbol{n}_{sample}}(-\boldsymbol{n}_{palm})}{\| - \boldsymbol{n}_{palm} - \pi_{\boldsymbol{n}_{sample}}(-\boldsymbol{n}_{palm})\|}$$

where $\pi_u(v)$ is the projection of v onto u. The complete algorithm to determine the TCP pose based on the sampled, antipodal and palm points is given in Figure 4.

We like to point out that the given constraints certainly do not guarantee a collision free grasp and therefore it is essential to still perform real collision checking on successful completion of the algorithm. Furthermore, it should be mentioned that this method may not work well on arbitrary objects but turns out to be very practical when applied to chair models.



(a) (b)

Fig. 5: For any given contact, in order to prevent slippage, the magnitude of force that is acting along the tangential plane $\|\boldsymbol{f}_t\|$ must be less or equal the magnitude of the contact normal force times the friction coefficient $\mu\|\boldsymbol{f}_\perp\|$. Hence, the combined contact force $\boldsymbol{f}$ has to lie inside a friction cone as depicted in (a). This cone is approximated with an m sided pyramid (b).

3.2 Grasp Synthesis

In order to obtain dual-arm grasps, we simply combine pairs of grasp hypotheses which we also refer to as *grasp candidates*. The grasp synthesizer first creates all possible pairs of hypotheses that survived the collision checking and grasp simulation steps. Consequently, the run-time complexity of this process is $\mathcal{O}(N^2)$ where N is the number of remaining grasp hypotheses. Then, each candidate is ranked based on a weighted sum of quality and reachability metrics though reachability is only considered during the online phase.

In the offline phase, we output the best M candidates and store them inside the grasp database. This, however, poses a problem as candidates with almost equal score are likely to be very similar or even the same grasps. In contrast to that, one of our main goals was to be able to grasp the same object in multiple ways. Therefore, we perform non-maximum suppression on the set of ranked candidates to filter out groups of grasps that are similar in terms of position and orientation and only keep each group's best candidate.

3.3 Grasp Analysis

In our analysis, a grasp is solely defined by the set of contacts between the surface of the object and the grippers' fingers. The applied forces and torques at each contact are determined using the hard finger contacts model with friction [4] where we consider exactly one contact per finger. We reject grasps that do not satisfy the force closure property and evaluate the v_1 quality measure proposed by Ferrari and Canny [6]. In the following, we briefly explain the employed concepts.

3.3.1 Contact Model

At each contact, a finger exerts a force $\boldsymbol{f}_\perp$ to the object that is perpendicular to the contact surface. According to Coulomb's law, slippage of the contact is prevented if the tangentially acting force component $\boldsymbol{f}_t$ satisfies

$$\|\boldsymbol{f}_t\| \leq \mu\|\boldsymbol{f}_\perp\| \tag{1}$$

where μ is the *friction coefficient*. Given three-dimensional forces, the constraint is represented as a friction cone as illustrated in Figure 5(a). For computational reasons, the cone is often approximated with an m-sided pyramid [10], seen in Figure 5(b) (in our case, we use $m = 8$). By doing so, the combined force $\boldsymbol{f}$ that is acting at the contact has to be represented as a convex combination of the m boundary vectors of the pyramid to prevent slippage.

3.3.2 Grasp Wrench Space

To examine grasp stability, we are interested in answering the question of which forces and torques a particular grasp can apply to compensate any potential disturbances made to the object. A common way of deriving grasp metrics is by analysing the *grasp wrench space* (GWS) [10].

A *wrench* combines both 3-D force and 3-D torque into a single 6-D vector. Now, for each force vector $\boldsymbol{f}_{i,j}$ of the the pyramid that approximates the friction cone of contact i, we can define a corresponding wrench

$$\boldsymbol{w}_{i,j} = \begin{bmatrix} \boldsymbol{f}_{i,j} \\ \lambda(\boldsymbol{d}_i \times \boldsymbol{f}_{i,j}) \end{bmatrix}$$

where $\boldsymbol{d}_i$ is the vector going from the object's center of gravity to the contact point. Furthermore, we set $\lambda = \frac{1}{r}$ with r being the maximum radius of the object from its center of gravity. The rational behind λ is that it relates units of torques to units of forces. In addition, we assume unit normal forces at each contact to compute $\boldsymbol{f}_{i,j}$.

Using these wrenches, we can build the space of wrenches W_{L_1} that can be applied to the object given that the contact normal force magnitudes maximally sum up to one:

$$W_{L_1} = \left\{ \boldsymbol{w} \;\middle|\; \boldsymbol{w} = \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_{i,j}\boldsymbol{w}_{i,j} \right. \\ \left. \wedge \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_{i,j} \leq 1 \;\wedge\; \alpha_{i,j} \geq 0 \right\}. \tag{2}$$

This is one of the definitions of the grasp wrench space that Ferrari and Canny [6] proposed. The second definition is the W_{L_∞} space which assumes that the magnitude of each individual contact normal force is less or equal to one. Despite being more intuitive, we opt for the W_{L_1} version because it can be built efficiently by computing the convex hull of the wrenches $\boldsymbol{w}_{i,j}$

$$W_{L_1} \equiv ConvexHull\left(\bigcup_{i=1}^{n}(\boldsymbol{w}_{i,1},\ldots,\boldsymbol{w}_{i,m})\right). \tag{3}$$

3.3.3 Force Closure

Until now, we established a way to determine which wrenches a given grasp can generate to balance potential external wrenches such as gravity while imposing limits on the contact normal forces. A natural question arises: Can a grasp support wrenches in all directions of the $\mathbb{R}^6$? If this can be answered positively, the grasp is said to be in *force closure* [10]. This property can be easily verified as it is equivalent to checking whether the origin is strictly contained inside the convex hull of the GWS. Hence, a given grasp is in force closure if every hyperperplane that encloses the convex hull has a negative offset.

3.3.4 Grasp Quality Measures

The epsilon measure ϵ_1 is a widely adopted metric for ranking force closure grasps based on their ability to resist arbitrary external wrenches [3]. In particular, it represents the radius of the $\mathbb{R}^6$ ball that just fits inside the GWS. Thus, it is the magnitude of the wrench that resists the worst possible disturbance wrench given that the sum of magnitude of all normal forces is exactly one. One can easily compute ϵ_1 as it is the absolute offset of the hyperplane that is closest to the wrench origin.

Unfortunately, we found that this quality is rather agnostic to where the grippers are positioned relative to each other. This means that it is possible to have a high epsilon score even if one hand is quite near to the other one or both are grasping at the same side of the object. Therefore, we decided to use the v_1 measure which is the volume of the convex hull [10]. Based on our experiments, this metric gives much more plausible grasps. While it still happens that the v_1 measure scores high on grasps that are located on the same side of the chair, it leads to a better spread of the grippers.

Nevertheless, we decided to augment the grasping score with the distance between the positions of the grasps normalized by the length of the diagonal d of the bounding box of the model. The combined grasp quality Q is given as

$$Q_{grasp} = w_{v_1} v_1 + w_{dist}\frac{\|\boldsymbol{p}_1 - \boldsymbol{p}_2\|}{d}. \tag{4}$$

where $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are the grasp positions and w_{v_1} and w_{dist} are weights to control the influence of the corresponding metric.

3.4 Reachability Analysis

During the online phase, we need to consider the reachability of each grasp with respect to the position of the robot manipulators. We simply assign a grasp to the arm that is closest to it. This of course means that we reject grasp candidates that assign a certain pose multiple times to the same arm. In addition to that, we reject grasps for which the kinematics solver cannot find a solution.

Furthermore, we employ two heuristics that favor grasps that are easiest to reach. The first one is to prefer TCP poses whose positions are closer to the assigned manipulator's base. Our second heuristic looks at the angle in the x-y-plane between the vector going from the position of the corresponding arm's base to the position of the grasp and the z-axis of the TCP pose. The resulting final metric for evaluating the reachability can be stated as follows:

$$Q_{reach} = \frac{1}{2}\sum_{i=1}^{2}\left[w_{pos}\left(1 - \frac{\|\boldsymbol{p}_i - \boldsymbol{b}_i\|}{r}\right) + w_{ang}\left(1 - \frac{\angle_{xy}\left(\boldsymbol{p}_i - \boldsymbol{b}_i, \boldsymbol{R}_{i,z}\right)}{\phi}\right)\right] \tag{5}$$

where $\boldsymbol{b}_i$ is the base position of the arm that is assigned to grasp i, $\boldsymbol{R}_{i,z}$ is the direction of the z-axis of grasp i, r is the maximum allowed distance between the base and the TCP pose and ϕ is the maximum allowed angle. The weights w_{pos} and w_{ang} determine the contribution of the two measures.

3.5 Preprocessing and Segmentation

We now go into the details of the first part of the steps that happen during the online phase. First, we receive point clouds from both cameras and merge them into a single point cloud. Then, a voxel grid filter is applied to reduce the number of points. Next we apply a statistical outlier removal filter [14] to the downsampled output. We now filter out the ground plane using sample consensus segmentation. Finally, we use euclidean clustering to extract the cluster that contains the most

points which we hypothesize to contain the object of interest.

3.6 Non-Rigid Registration

The following shows how we transfer grasping knowledge to new instances of familiar objects. To do so, the goal is to transform a grasp given in model space into the observed space while considering the local geometric shape of the grasped object. Therefore, we warp the point cloud of the matched model to align with the segmented cloud. This is achieved using the non-rigid *coherent point drift* (CPD) registration algorithm proposed by Myronenko and Song [11]. We briefly give an overview of CPD and explain how we use the resulting deformation field to transform the grasps appropriately.

3.6.1 Coherent Point Drift

CPD considers two point sets $\boldsymbol{X}_{N\times D} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)^T$ and $\boldsymbol{Y}_{M\times D} = (\boldsymbol{y}_1, \dots, \boldsymbol{y}_M)^T$ where $\boldsymbol{Y}$ represents the template points that should be aligned with the observed data points $\boldsymbol{X}$. The alignment problem is viewed from a probabilistic perspective that is to represent $\boldsymbol{y}_1, \dots, \boldsymbol{y}_M$ as the centroids of a Gaussian Mixture Model (GMM) and $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ as samples drawn from it. The objective is to estimate the GMM with maximum likelihood for the given data points while at the same time making sure that the topological structure of the template points is preserved. Especially the last part is of importance as we want points that are close to each other to move coherently. Minimizing the following cost function is equivalent to maximizing the likelihood of the GMM [13]:

$$E(\theta, \sigma^2) = -\sum_{n=1}^{N} \log \sum_{m=1}^{M} e^{-\frac{1}{2\sigma^2}\|\boldsymbol{x}_n - \mathcal{T}(\boldsymbol{y}_m, \theta)\|^2} + \frac{\lambda}{2}\phi(\boldsymbol{Y}) \tag{6}$$

where $\mathcal{T}(\boldsymbol{y}_m, \theta)$ is a function parameterized by θ that transforms template points to data points. While the first term of Equation 6 serves as a penalty on the distance between the points, the second term uses ϕ as a regularizer to enforce motion coherence.

CPD defines the non-rigid transformation $\mathcal{T}$ as

$$\mathcal{T}(\boldsymbol{Y}, \boldsymbol{W}) = \boldsymbol{Y} + \boldsymbol{G}\boldsymbol{W} \tag{7}$$

which is the initial template point set $\boldsymbol{Y}$ plus a displacement given by the matrix multiplication of a fixed gaussian kernel matrix $\boldsymbol{G}_{M\times M}$ and the estimated kernel weights $\boldsymbol{W}_{M\times D}$. We also refer to $\boldsymbol{W}$ as the *deformation field*. The elements in $\boldsymbol{G}$ correspond to the distance measured between the points $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ using the kernel $\mathcal{G}(\cdot, \cdot)$:

$$g_{ij} = \mathcal{G}(\boldsymbol{y}_i, \boldsymbol{y}_j) = \exp\left(-\frac{1}{2\beta^2}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|\right). \tag{8}$$

Note, that the parameter β controls how much influence points have on each other. The smaller this value, the more interaction we have between the points.

To minimize the cost function given in Equation 6, CPD applies an Expectation Maximization (EM) algorithm. For the sake of brevity, we do not show the full algorithm here but rather refer to the work by Myronenko and Song [11] to get further insight.

3.6.2 Grasp Transformation

CPD only outputs is the deformation field $\boldsymbol{W}$ that is used to displace each individual template point. So how can we actually use this matrix to transform points that do not belong to $\boldsymbol{Y}$? For a new point $\boldsymbol{z}$, we assume motion coherence with the template points that are close to it. Thus, we use a weighted mean displacement of the k-nearest neighbors that are contained in $\boldsymbol{Y}$ to transform $\boldsymbol{z}$.

Let $\mathcal{I}_k(\boldsymbol{z})$ be the set of indices corresponding to the k nearest neighbors of $\boldsymbol{z}$ in $\boldsymbol{Y}$. We define the transformation $\mathcal{F}$ that maps a point from the template space to the observed space as:

$$\mathcal{F}(\boldsymbol{z}) = \boldsymbol{z} + \frac{\sum_{i\in\mathcal{I}_k(\boldsymbol{z})} \mathcal{G}(\boldsymbol{z}, \boldsymbol{y}_i)\boldsymbol{W}^T\boldsymbol{g}_i}{\sum_{i\in\mathcal{I}_k(\boldsymbol{z})} \mathcal{G}(\boldsymbol{z}, \boldsymbol{y}_i)} \tag{9}$$

where $\boldsymbol{g}_i$ is the i-th row vector of the kernel matrix $\boldsymbol{G}$.

In order to transform grasp poses, we still need to handle orientations. Given rotation matrix $\boldsymbol{R}$, the idea is to warp each basis vector $\boldsymbol{r}_i$ into the data space using $\mathcal{F}$ and orthonormalize the resulting vectors. The first part is done by applying the following transformation $\mathcal{L}$ to the direction $\boldsymbol{r}$:

$$\mathcal{L}(\boldsymbol{r}) = \mathcal{F}(\alpha\boldsymbol{r} + \boldsymbol{p}) - \mathcal{F}(\boldsymbol{p}) \tag{10}$$

where $\boldsymbol{p}$ is the position of the grasp we want to transform. Here, α determines, how far is looked into the direction of $\boldsymbol{r}$. As $\boldsymbol{r}$ has a length equal to one, this parameter has to be reasonably adapted according to the units of measure in the observed space. Finally, we orthonormalize in the following manner:

$$\begin{aligned}
\boldsymbol{u}_z &= \mathcal{L}(\boldsymbol{r}_z) \\
\boldsymbol{u}_x &= \mathcal{L}(\boldsymbol{r}_x) - \pi_{u_z}(\mathcal{L}(\boldsymbol{r}_x)) \\
\boldsymbol{u}_y &= \mathcal{L}(\boldsymbol{r}_y) - \pi_{u_z}(\mathcal{L}(\boldsymbol{r}_y)) - \pi_{u_x}(\mathcal{L}(\boldsymbol{r}_y)) \\
\boldsymbol{R}_{new} &= \left(\frac{\boldsymbol{u}_x}{\|\boldsymbol{u}_x\|}\ \frac{\boldsymbol{u}_y}{\|\boldsymbol{u}_y\|}\ \frac{\boldsymbol{u}_z}{\|\boldsymbol{u}_z\|}\right).
\end{aligned}$$

Note, that we give highest priority to the z-direction as this is the main axis of the gripper.

3.6.3 Mesh Reconstruction

We finally discuss how we reconstruct a mesh from the warped point cloud. One big problem with CPD is that the gaussian kernel has M^2 entries. This limits the number of points we can register as the algorithm quickly becomes extremely slow. To resolve this issue, we first use a voxel grid filter to reduce the number of points of the template cloud to about 300-500. The downsampled cloud is then used for registration. Then, we apply $\mathcal{F}$ to transform every point of the original template cloud individually. This approach is orders of magnitudes faster than registering the cloud directly while still providing enough deformation quality. For the mesh reconstruction itself, we use a greedy triangulation algorithm [5].

4 Experiments

In the following, we first evaluate the grasps that are generated on 3D models in the offline phase and then show experimental results on how grasping knowledge is transferred to new instances.

4.1 Grasp Database Generation

We generated grasp databases on four different models of different geometric complexity as seen in Figure 6. In all experiments, we used $w_{v_1} = 1.0$ and $w_{dist} = 0.5$. The displayed poses are taken from the best 50 grasp candidates returned by the grasp synthesizer. Note, that we additionally filtered out poses whose z-coordinate is less than 0.2 to avoid storing grasps that likely result in collision between the manipulator and the ground.

4.1.1 Hypothesis Sampling

For sampling grasp hypotheses, we ran our proposed algorithm in Figure 4 on 10'000 uniformly sampled points from the model's point cloud. The point clouds are created with uniform mesh sampling and have a density of one point per cubic millimeter. It is important to have high point densities as we need to be able to extract the normals that are associated with the palm directions on thin surfaces such as leg or plate edges.

In order to evaluate our hypotheses sampling algorithm, we can compare the number of samples that were accepted, rejected due to invalid constraints or rejected due to checked collisions. We naturally expect the vast majority of samples to be rejected. However, the question is how many times we can avoid expensive collision checking by observing that the antipodal and palm point constraints are not fulfilled.

For each model in Figure 6, Figure 7 shows the number of accepted and rejected samples. Two main conclusions can be drawn from this plot. First, the number of rejections due to constraint violations is on average 7.7 times higher than the number of rejections due to collision checking. For the most basic model, the method checks for collisions in only 5 % of the samples. This number rises to 18 % in the case of the desk chair but we think that this is still a good result. Secondly, the number of accepted grasps significantly drops with the model's complexity. It is a logical consequence that with higher non-graspable surface area, sampling based approaches become more inefficient. In this regard, the desk chair really pushes the boundaries of this method while keeping only 1.4 % of the total number of generated samples. On the other hand, more than every fifth pose was accepted on the first chair which is quite many considering the fact that points are sampled uniformly across the surface of the model.

4.1.2 Grasp Synthesis

Using our grasp quality metric, we noticed that the two gripper poses tend to be spread out as far as possible. Figure 6(a) demonstrates this effect nicely as the grasps on the back plate of the chair are located directly at the corners. The grasps are mostly diagonally placed meaning that for example, the first arm goes to the chair's top-left corner while the second one picks a position at the front-right leg. Furthermore, the direction of the gripper poses tend to be orthogonal instead of parallel which is expected when using the v_1 quality.

We noticed that problems may arise when grasps are located near the intersection of two chair components such as a leg and the seat plate. In this case, it is more likely that the grasp result in a collision after being warped to a newly seen object. The chair in Figure 6(k) shows some instances where this issue might happen. Using our approach it is hard to explicitly discount these kinds of grasps.

4.2 Detection

We test our proposed grasping pipeline in a simulated environment using Gazebo. The constructed scenes always contain the chair to be grasped at the origin of the world frame. We set the location of the manipulator bases to (1,1,0) and (-1,-1,0) respectively. The cameras are also placed diagonally at (1,-1,1) and (-1,1,1). This setup allows the robots to reach pretty much any position on the grasped object. Instances where the robots successfully lifted the target chair are shown in Figure 8.

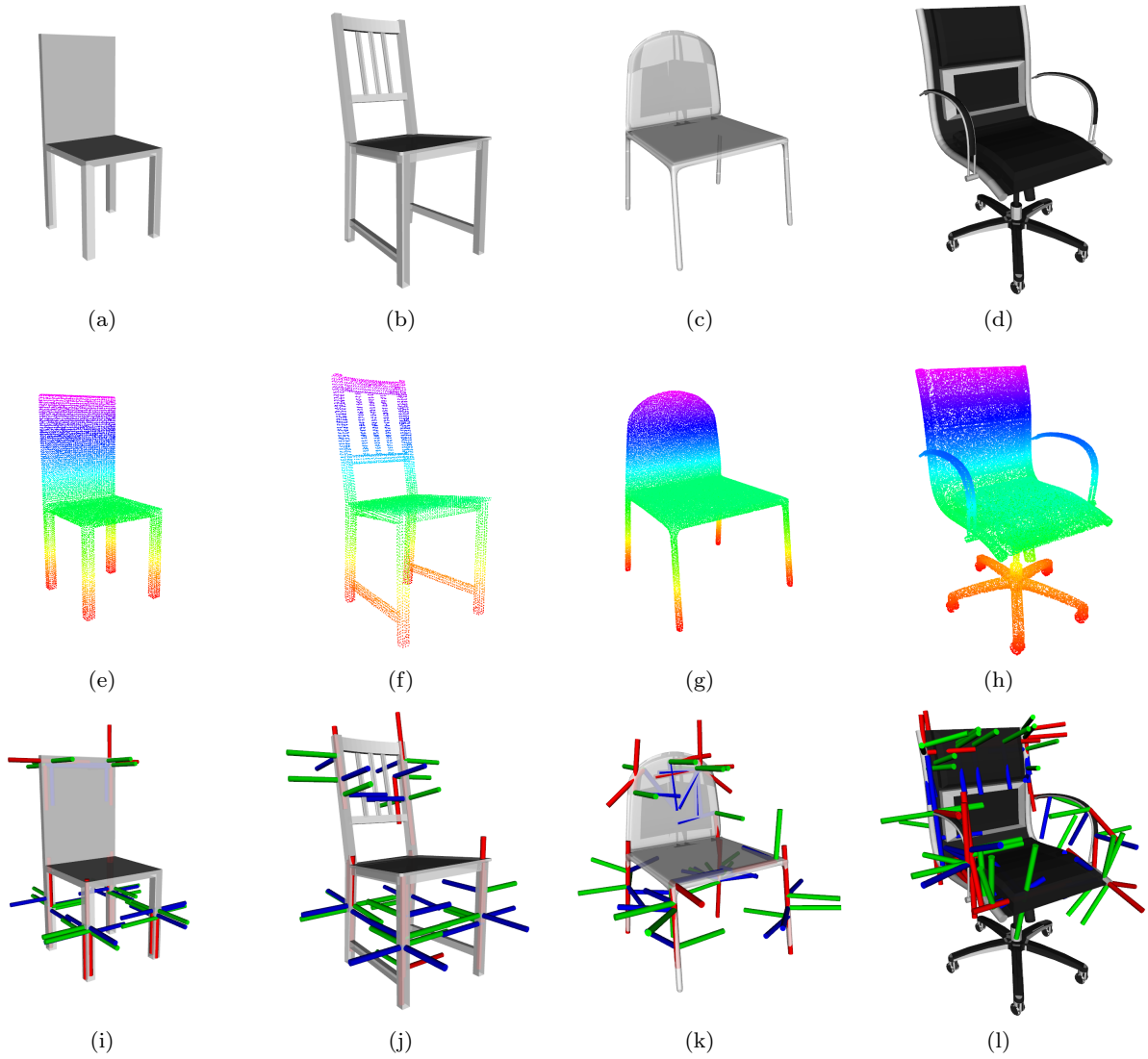


Fig. 6: Grasp generation on different models. Each model is given as mesh (a-d) and as point cloud (e-h). The generated grasp poses (i-l) are collected from the 50 best dual-arm grasps. The blue cylinders represent the z-direction of the grasps (see Figure 3 for reference).

In all experiments, we generated the grasp database using only the model shown in 6(a). Consequently, this model is the one that is registered to the observed point clouds. To parameterize CPD, we used the values $\lambda = 3$ and $\beta = 3$ and set the maximum number of iterations to 30. Figure 9 depicts meshes that were reconstructed from the registered point clouds of two example chairs during the detection phase of the pipeline.

We can see that the non-rigid registration is able to deform the template chair to match the size of the target object. More importantly, it is able to bend the back plate and the legs to align with local curvatures. Figure 10 shows that the orientation of the grippers almost perfectly matches the orientation of the grasped parts. This indicates that the grasp transformation works as expected.

4.3 Limitations

It is generally hard to directly compare the results of different grasping algorithms. The performance of grasping systems depends on a variety of different factors including the particular choice of the manipulator and gripper, the lighting of the environment and obviously, the objects that should be grasped themselves. We understand that we are only partially able to reason about the practicality of our proposed method in the real

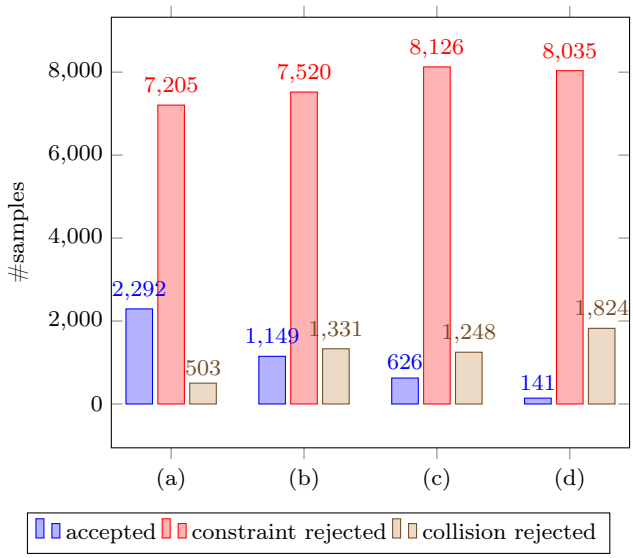


Fig. 7: For each of the models (a-d) in Figure 6, we ran our hypothesis sampling algorithm on 10'000 uniformly sampled points and counted how often the samples were accepted, rejected due to violated constraints or rejected after collision checking.

world given that we only tested it in simulations. However, we would like to point out general limitations of our proposed method that we think are important to consider when deploying our dual-arm grasping system to a real application.

One issue that we encountered with the setup is the fact that the cameras do not capture the chair's seat plate from the bottom. Therefore, the observed point cloud has only one layer of points compared to the fully captured template model cloud that has two layers associated with the seat plate. The problem is that CPD just tries to minimize the total alignment error and thus, the aligned plate is centered around the single layer of observed points thereby creating an undesired offset. This offset can be clearly seen in Figure 11(a). A simple workaround is to manually remove the bottom points of the seat plate from the template cloud, however, this fix should be considered quite unmethodical.

While trying out different kinds of chairs we noticed that it is generally hard to figure out whether a particular grasp is rather inappropriate for our specific type of gripper. For example, Figure 11(b) depicts the manipulator trying to grasp at a place that would be a good fit for the database model but not for the displayed chair. We found that CPD is unable to deform the template cloud with high level of details as would be required in this case. Lowering the parameters λ and β is unfortunately not an option as this worsens the deformation quality on a global scale.

Finally, we want to mention that CPD only provides satisfying results when the shapes of the registered objects are similar enough. Figure 11(c) and 11(d) show the attempt of aligning our database model to a chair whose backrest is formed like an arc. The registration can definitely be considered failed. Therefore, if the grasping system should be able to perform well on a large variety of different objects of familiar type, we recommend to generate grasps from more than a single model and implement a point cloud matching algorithm to obtain the model that is most similar to the observed one.

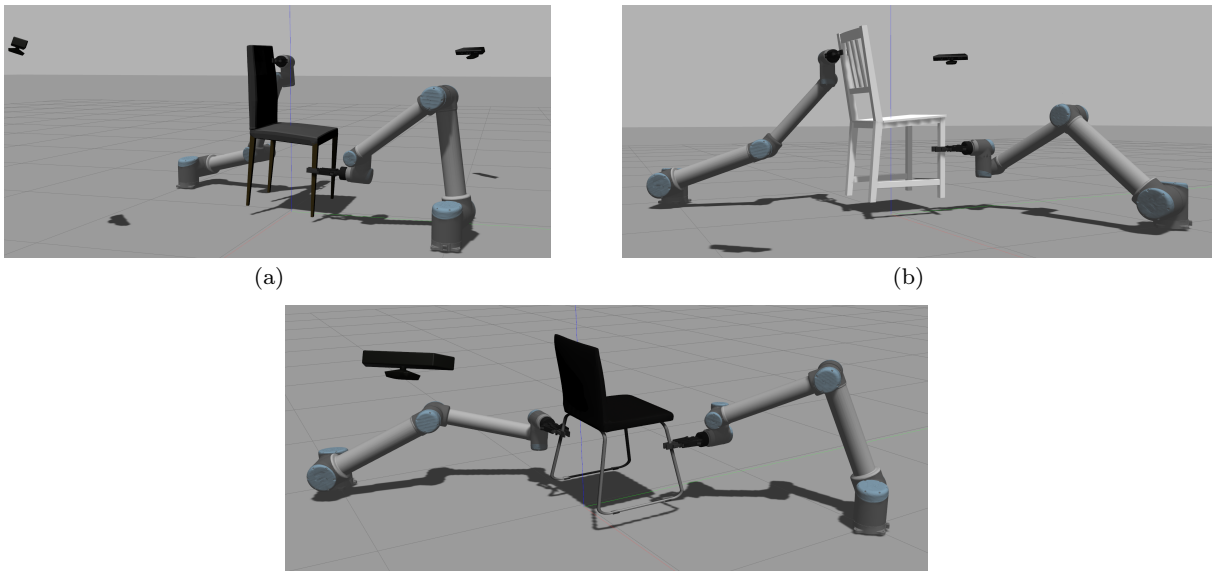
5 Conclusions

In this work we implemented a pipeline for dual-arm grasping of large objects of familiar kind with particular focus on chairs. We proposed a method, to efficiently sample grasp hypotheses and synthesize dual-arm grasps based on common measures that are derived from the grasp wrench space analysis. A key component of the ability to transform grasping knowledge to newly seen instances arises through the usage of non-rigid registration. In simulations, we showed that our method is able to detect grasps to successfully lift chairs that are similar in shape compared to the model used to create the grasp database. An advantage of our approach is that it only requires a model of an example chair as input and relatively short offline processing time in order to start detecting grasps.

For future work, we would like to evaluate how our grasping pipeline performs in the real world. Besides that, work has to be done to improve the registration process as we noticed that the coherent point drift algorithm performs poorly in certain scenarios. Finally, it would be interesting to examine the usage of different types of robot hands for dual-arm grasping.

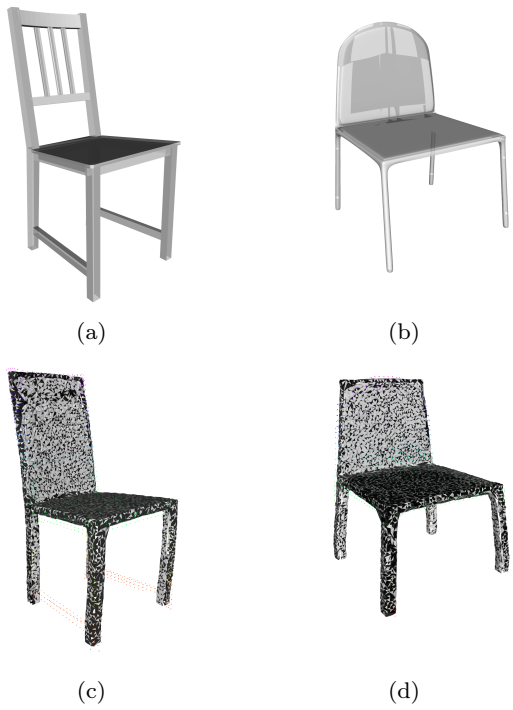
References

1. Robotiq 2f 140 product page. URL https://robotiq.com/products/2f85-140-adaptive-robot-gripper.
2. Universal robot ur10 product page. URL https://www.universal-robots.com/products/ur10-robot/.
3. J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis - A survey. *CoRR*, abs/1309.2660, 2013. URL http://arxiv.org/abs/1309.2660.
4. C. Borst, M. Fischer, and G. Hirzinger. Grasping the dice by dicing the grasp. In *Proceedings*



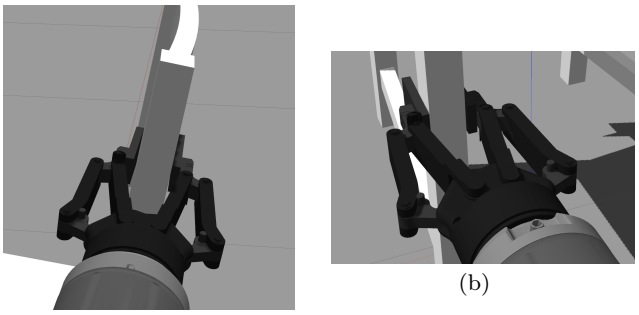
(a) (b) (c)

Fig. 8: Simulations run in Gazebo, where the manipulators successfully grasped and lifted a chair.



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

Fig. 9: Reconstructed meshes (c-d) obtained from registered point clouds of the observed chairs (a-b). The template cloud used in the non-rigid registration process is shown in Figure 6(e). Note, that the dark wholes are only caused by a displaying issue and not by missing triangles in the mesh.



(a) (b)

Fig. 10: Detailed view of executed grasps on the model shown in Figure 6(b). The orientations of the grippers align well with the grasped parts of the chair.

2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), volume 4, pages 3692–3697 vol.3, 2003.

5. M. Dickerson, R. Drysdale, S. McElfresh, and E. Welzl. Fast greedy triangulation algorithms. *Computational Geometry*, 8, 07 1997. doi: 10.1016/S0925-7721(97)89149-3.
6. C. Ferrari and J. Canny. Planning optimal grasps. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, pages 2290–2295 vol.3, 1992.
7. N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Con-*

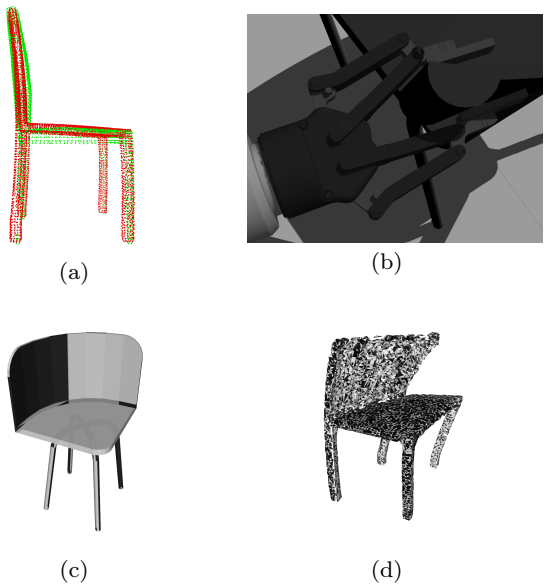


Fig. 11: Summarizing the limitations of the registration-based method. CPD has problems aligning the seat plate (a). Here the green cloud corresponds to the template cloud and the red one is the observed cloud. In (b), a grasp is executed at a place which is rather inappropriate for the gripper. Figures (c-d) show, that the non-rigid registration performs poorly when the target object has a significantly different shape than the template model.

ference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), volume 3, pages 2149–2154 vol.3, 2004.

8. S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *CoRR*, abs/1603.02199, 2016. URL http://arxiv.org/abs/1603.02199.
9. J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, and K. Goldberg. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26):eaau4984, 2019.
10. A. T. Miller and P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics Automation Magazine*, 11(4):110–122, 2004.
11. A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12): 2262–2275, 2010.
12. D. Pavlichenko, D. Rodriguez, C. Lenz, M. Schwarz, and S. Behnke. Autonomous bimanual functional regrasping of novel object class instances. *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pages 351–358, 2019.
13. D. Rodriguez, C. Cogswell, S. Koo, and S. Behnke. Transferring grasping skills to novel instances by latent space non-rigid registration. *CoRR*, abs/1809.05353, 2018. URL http://arxiv.org/abs/1809.05353.
14. R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3d point cloud based object maps for household environments. *Robot. Auton. Syst.*, 56(11):927–941, Nov. 2008. ISSN 0921-8890. doi: 10.1016/j.robot.2008.08.005. URL https://doi.org/10.1016/j.robot.2008.08.005.
15. S. Y. Shin and C. Kim. Human-like motion generation and control for humanoid's dual arm object manipulation. *IEEE Transactions on Industrial Electronics*, 62(4):2265–2276, 2015.
16. C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic. Dual arm manipulation - a survey. *Robotics Auton. Syst.*, 60:1340–1353, 2012.
17. Stanford Artificial Intelligence Laboratory et al. Robotic operating system. URL https://www.ros.org.
18. N. Vahrenkamp, E. Kuhn, T. Asfour, and R. Dillmann. Planning multi-robot grasping motions. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 593–600, 2010.