I Get the Hang of It! A Learning-Free Method to Predict Hanging Poses for Previously Unseen Objects

Wanze Li, Lexin Pan, Boren Jiang, Yuwei Wu, Weixiao Liu, *Member, IEEE*, and Gregory S. Chirikjian

***Abstract*—The action of hanging previously unseen objects remains a challenge for robots due to the multitude of object shapes and the limited number of stable hanging arrangements. This paper proposes a learning-free framework that enables robots to infer stable relative poses between the object being hung (object) and the supporting item (supporter). Our method identifies potential hanging positions and orientations on previously unseen supporters and objects by analyzing the hanging mechanics and geometric properties. An evaluation policy is designed to match potential hanging positions and directions and to optimize the relative hanging poses. Experiments were conducted in both simulation and real-world scenarios. The success rates of our strategy outperform the state-of-the-art baseline method. The proposed method was also tested on unhangable pairs of objects and supporters and results show that our algorithm can reject false positive hanging properly. Finally, we ran experiments under different scanning conditions. Experimental results indicate that although the success rate decreases as the quality of the scan decreases, it remains at a high level. More details and Supplementary Material can be found at our project webpage.**

***Index Terms*—Calibration and identification, computational geometry, domestic robotics.**

I. Introduction

OVER the past few decades, the field of robotics has made tremendous strides, leading to a significant impact on our daily lives. Household robotics, a critical sector within the robotics industry, has gained increasing attention. It is widely believed that in the near future, robots will be capable of assisting humans in performing routine tasks, such as housework and elderly care [1]. However, certain everyday tasks, such as hanging objects, remain challenging for robots.

Manuscript received 18 October 2023; accepted 5 February 2024. Date of publication 21 February 2024; date of current version 11 March 2024. This letter was recommended for publication by Associate Editor K. Hang and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported in part by National Research Foundation, Singapore, under its Medium Sized Centre Programme - Centre for Advanced Robotics Technology Innovation (CARTIN), under Grant A-0009428-08-00, and in part by AME Programmatic Fund under Project MARIO A-0008449-01-00. *(Corresponding author: Gregory S. Chirikjian.)*

Wanze Li, Lexin Pan, Boren Jiang, and Yuwei Wu were with the Department of Mechanical Engineering, National University of Singapore, Singapore 117575 (e-mail: li_wanze@u.nus.edu).

Weixiao Liu was with the Department of Mechanical Engineering, National University of Singapore, Singapore 117575, and also with the Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218 USA.

Gregory S. Chirikjian was with the Department of Mechanical Engineering, National University of Singapore, Singapore 117575. He is now with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716 USA (e-mail: gchirik@udel.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3368297, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3368297

Hanging objects is a routine activity in our daily lives, involving tasks such as placing mugs on mug trees, clothes on racks, and pans on hooks. Therefore, enabling robots to hang everyday objects on various supporting items could increase convenience and improve domestic robotics. However, the diverse geometry of everyday objects and the limited number of stable hanging postures bring significant challenges for robotic hanging, especially in unseen environments.

Compared to popular topics such as grasping and planning, the problem of hanging objects with robots has received relatively little attention. Object hanging is typically used to validate methods for other robotics problems [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], like objects placement and affordance representation. For instance, Jiang et al. [2] proposed a Support Vector Machines (SVM) based supervised learning algorithm to identify object placement. The algorithm fits a function of features to estimate the stability and orientation of a placement. They tested their method via hanging objects on hooks, but the rate of success was only 40%.

Manuelli et al. [3] developed a framework to represent the objects with keypoints for category-level manipulation, including hanging a mug on a mug tree. Their method relies on manually labeled data for training, which limits their efficiency and capacity to manipulate novel object categories. Similarly, Simeonov et al. [7] used Neural Descriptor Fields (NDFs) and annotated keypoints to infer relationships between pairs of unknown objects for specific tasks, including hanging mugs. Ruiz et al. [4], [5] demonstrated a method for predicting the geometric affordance of a scene with Interaction Tensor and used hanging coat hangers as a test. Their approach requires examples of the interaction between similar pairs of objects as prior knowledge, limiting performance in novel environments. Furthermore, these works generally focus on a specific pair of object and supporter, such as hanging a mug on a mug tree, lacking universality in hanging arbitrary objects.

In recent years, some groups have developed methods that specifically address the problem of hanging objects [12], [13], [14]. You et al. [12] proposed a reinforcement learning-based approach that can hang a wide range of objects on various supporting items. Takeuchi et al. [13], [14] used a Generative Adversarial Network (GAN) to generate 3D models with different shapes. Then they trained a deep neural network with these 3D models to estimate hanging points of an unknown object. Both of these methods require training on large amounts of data.

Existing research on objects hanging mainly uses learning-based methods. These methods require extensive training data and often have limited adaptability to novel situations that differ significantly from the training data. In contrast, when humans

Fig. 1. Examples of daily objects hanging with single contact point.

hang objects, we can recognize and extract suitable geometric shapes for hanging such as hooks and holes on both the object and the supporter [15]. Additionally, we can intuitively predict hanging stability based on analyzing the force between two objects. For example, while hanging a stir-fry spoon on a hook, humans usually hang it in a pose where the spoon body is lower than the handle (as shown in Fig. 1) to prevent the spoon from rotating after release. To adapt the philosophy of human perception and action in hanging objects, in this paper we propose a three-stage learning-free algorithm capable of predicting a stable hanging pose for previously unseen objects and supporters. A suitable hanging pose is defined as a collision-free pose in which the object does not fall down after release. In this work, we mainly focus on the hanging case with only one contact point between the object and the supporter. As shown in Fig. 1, such an assumption includes adequate hanging scenarios in daily life and more complex situations will be explored in the future. The workflow of the proposed hanging pose prediction method is shown in Fig. 2. The algorithm takes the mesh model of the object and the point cloud of the supporter as inputs, both of which can be obtained by 3D scanning. Then inputs are processed separately to detect the suitable hanging positions and directions on both the object and supporter. Each position is represented as a 3D keypoint and each direction is represented as a key-vector. Subsequently, all keypoints and key-vectors on the object and supporter are aligned with each other to identify all possible matches. These matches are evaluated and ranked to determine proper hanging poses. For hanging execution, the hanging poses prediction method is integrated with an RRT-connect planner [16].

Compared to existing methods for robotic hanging objects, the proposed work eliminates the need for training data. Moreover, our method is based on the analysis of the mechanics and geometric properties, resulting in greater accuracy in unseen objects and supporters. The main contributions of this paper include: (1) a learning-free algorithm to predict hanging poses of previously unseen object-supporter pairs, (2) a simple, straightforward, yet effective evaluation method to evaluate the quality of each hanging candidate, and (3) validation of the proposed approach by hanging arbitrary objects on arbitrary supporters in simulation and real-world under different scanning conditions. To the best of our knowledge, we are the first to finish the real-world robotic hanging experiments with arbitrary objects and supporters. The results demonstrate the superior performance of our method over the state-of-the-art.

II. Method

This section introduces the details of the proposed hanging poses detection method. We first present the theoretical foundations behind the intuition of our method in Section II-A. Then we explain how to detect hanging positions and hanging directions on objects (Section II-B) and supporters (Section II-C). Finally, the procedure of matching and evaluating potential hangings is introduced in Section II-D. Some results of different steps are shown in Fig. 2.

A. Support Force Analysis

When an object is stably hung on a supporter, the support force must balance the gravity of the object. Since there is only one contact point, the support force should be colinear with the center of mass (CoM) of the object. Normal vectors can point either inside ($\mathbf{n}_2$ in Fig. 3(a)) or outside ($\mathbf{n}_1$ in Fig. 3(a)) the object. In this work, the direction pointing outside the object is selected as the normal vector direction. Therefore, as illustrated in Fig. 3(b), the direction of the support force on the object should be opposite to the normal vector at the contact point [17]. A possible contact point $\mathbf{p}_h$ on object satisfies:

$$\frac{(\mathbf{h}_c - \mathbf{p}_h) \cdot \mathbf{n}_h}{\|\mathbf{h}_c - \mathbf{p}_h\|} = 1 \tag{1}$$

where $\mathbf{n}_h$ is the normal vector at $\mathbf{p}_h$ with a unit length, $\mathbf{h}_c$ is the 3D position of the object CoM. In addition, the direction of the support force offered by the supporter is opposite to the gravity and along the normal vector at the contact point on the supporter. Therefore, a potential contact point $\mathbf{p}_s$ on supporter satisfies:

$$-\mathbf{g} \cdot \mathbf{n}_s = 1 \tag{2}$$

Here, $\mathbf{g}$ is the direction of gravity and $\mathbf{n}_s$ is the unit normal vector at $\mathbf{p}_s$. While these conditions are necessary but not sufficient, they are enough to provide a proper initial guess for the following procedures. In this paper, the z direction is antiparallel with the gravitational direction.

B. Hangability Detection

Hangability is defined as the ability of an object to be hung steadily. The goal of hangability detection is to find all possible hanging positions within the object and corresponding hanging directions. The workflow is shown in Fig. 4.

Sampling, Potential Contact Points Selection and Clustering: To leverage the rules described in (1), a point cloud (Fig. 4(b)) $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^N \in R^{3\times N}$ is sampled from the mesh via Poisson disk sampling [18]. $N = 10000$ is the predefined number of sampling points. Then points satisfying the following relationship are considered as potential contact points (red points in Fig. 4(c)) and form a subset $\mathbf{P}_h$ of $\mathbf{H}$:

$$\mathbf{P}_h \triangleq \left\{\mathbf{h} \in \mathbf{H} | \frac{(\mathbf{h}_c - \mathbf{h}) \cdot \mathbf{n}_h}{\|\mathbf{h}_c - \mathbf{h}\|} > \alpha_h, \mathbf{h}_c = \frac{1}{N}\sum_{i=1}^{N}\mathbf{h}_i\right\} \tag{3}$$

Here, $\mathbf{n}_h$ is the normal vector at $\mathbf{h}$, $\mathbf{h}_c$ is the mean of points in point cloud as the estimation of object CoM and $\alpha_h \in (0, 1]$ is a threshold. To reduce the computation and accelerate following steps, $\mathbf{P}_h$ is then clustered using the Mean Shift Algorithm [19]. Only clusters with enough number of points are kept for the following steps, i.e.:

$$\mathbf{P}_h^i \subseteq \mathbf{P}_h, i = 1, \ldots, M, s.t. |\mathbf{P}_h^i| > l_c \tag{4}$$

M is the number of remained clusters, which is determined automatically after clusters selection and l_c is a predefined threshold. In practice, we find that sometimes the center of a cluster may be located inside the model, resulting the inaccuracy of the following steps. To address this issue, the algorithm selects

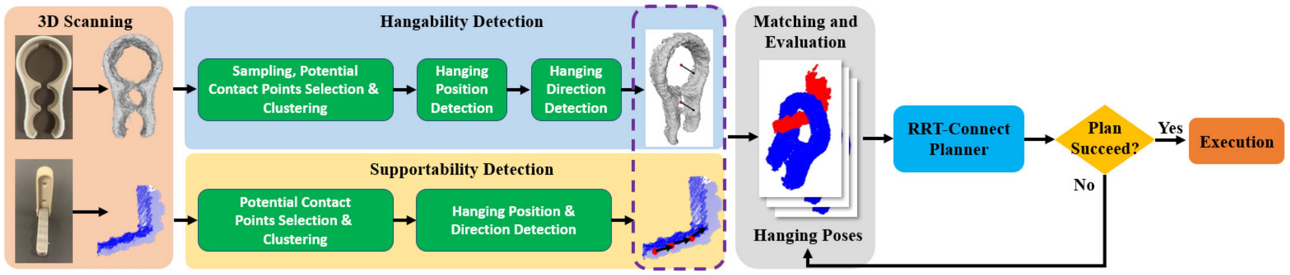
3D Scanning
Hangability Detection
Sampling, Potential Contact Points Selection & Clustering
Hanging Position Detection
Hanging Direction Detection
Supportability Detection
Potential Contact Points Selection & Clustering
Hanging Position & Direction Detection
Matching and Evaluation
Hanging Poses
RRT-Connect Planner
Plan Succeed?
Yes
No
Execution

Fig. 2. Workflow of the proposed method. The entire procedure includes 3D scanning, hangability detection, supportability detection, matching and evaluation, and RRT-Connect planning. In this figure, red points represent hanging positions and black arrows represent hanging directions.

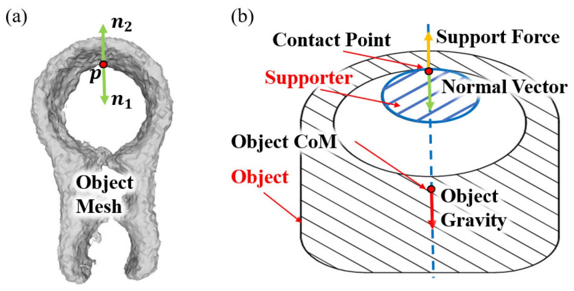

Fig. 3. (a) Direction of the normal vector. $\mathbf{n}_1$ and $\mathbf{n}_2$ are two possible directions of the normal vector at point $\mathbf{p}$ on the mesh. $\mathbf{n}_1$ points outside the mesh and $\mathbf{n}_2$ points inside the mesh. (b) The cross-section of the object and the supporter in a stable hanging pose. The CoM, contact point, normal vector at the contact point, support force and gravity of the object are labeled. Object is marked by black shaded lines and supporter is marked by blue shaded lines.

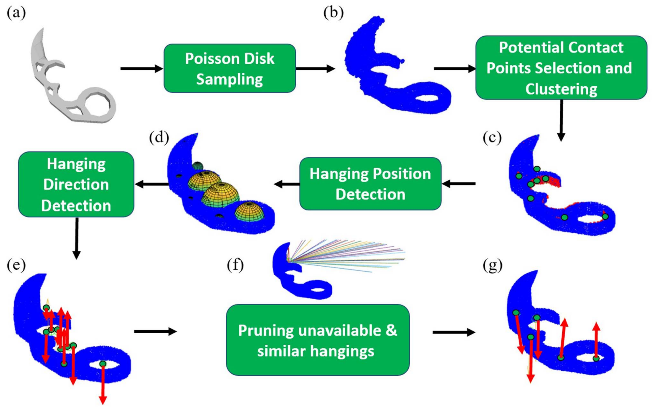
(a)
Poisson Disk Sampling
(b)
Potential Contact Points Selection and Clustering
(c)
(d)
Hanging Position Detection
Hanging Direction Detection
(e)
(f)
Pruning unavailable & similar hangings
(g)

Fig. 4. Procedure for hangability detection. (a) The mesh model of the object obtained in 3D scanning. (b) The point cloud of the object after Poisson disk sampling. (c) The point cloud after contact points selection and clustering. Potential contact points are marked as red. Cluster centroids are marked as green. (d) Point cloud after hanging position detection. Spheres represent free space near hanging positions. (e) Hanging positions and directions after hanging direction detection. Hanging positions are labeled by green dots and hanging directions are represented as red arrows. (f) An example of an unavailable hanging position, because too many rays go to infinity. (g) Hanging positions and directions after pruning unavailable or similar hangings.

the closest point to the centroid of each cluster as its substitute. These points (green dots in Fig. 4(c)) are denoted as $\{\mathbf{p}_h^k\}_{k=1}^M$.

Hanging Position Detection: As shown in Fig. 3(b), in the stable state, the supporter is below the object at the contact point. Consequently, there must be sufficient free space near the contact point to accommodate the supporter. The next step is to detect the hanging position, which is defined as the center of the free space. The procedure of hanging position detection is shown in Fig. 5. For every point $\mathbf{p}_h^i \in \{\mathbf{p}_h^k\}_{k=1}^M$, a ray is cast toward object CoM $\mathbf{h}_c$. The intersection point between the ray and mesh of object is denoted as $\mathbf{q}_h^i$ (see Fig. 5(a)). If there is no interaction, $\mathbf{q}_h^i$ is set to $\mathbf{h}_c$. Next, N_h points (black dots in Fig. 5(b)) are evenly sampled along the line segment $\overline{\mathbf{p}_h^i\mathbf{q}_h^i}$. N_h is a predefined value and selected as 10. The farthest sampled point from the object point cloud $\mathbf{H}$ is selected as the hanging position, denoted as $\mathbf{c}_h^i$ (see Fig. 5(c)). The distance from $\mathbf{c}_h^i$ to $\mathbf{H}$ is denoted as free space radius (FSR), represented as r_h^i. As shown in Figs. 4(d) and 5(c), the free space between $\mathbf{p}_h^i$ and $\mathbf{q}_h^i$ can be represented as a sphere with $\mathbf{c}_h^i$ as the center and r_h^i as the radius. Finally, hanging positions and corresponding free spaces are filtered based on two conditions: (1) $r_h^i > r_\alpha$ where r_α is a threshold for FSR, to ensure a sufficiently large free space for supporter placement and (2) $S[\mathbf{c}_h^i] > 0$ where $S[\mathbf{c}_h^i]$ is the signed distance function (SDF) value at $\mathbf{c}_h^i$, ensuring that $\mathbf{c}_h^i$ is outside the object to avoid collisions. Only the $(\mathbf{c}_h^i, r_h^i)$ pairs that satisfy these conditions are kept for the following procedure.

Hanging Direction Detection: The next step involves determining the optimal hanging directions for hanging positions ($\mathbf{c}_h^i$) identified in the previous step. Section II-A shows that the vector $\mathbf{n}_i = \frac{(\mathbf{q}_h^i - \mathbf{p}_h^i)}{||\mathbf{q}_h^i - \mathbf{p}_h^i||}$ in Fig. 6 should coincide with the gravity direction at a stable hanging pose. Thus, the only degree of freedom for the object orientation is the rotation about $\mathbf{n}_i$. Therefore, possible hanging directions at $\mathbf{c}_h^i$ can be enumerated by $\mathbf{v}_j^i = R_n(\beta_j)\mathbf{u}_i, j = 1, \ldots, N_p$, where $\mathbf{u}_i$ is a random vector that is perpendicular to $\mathbf{n}_i$ (see Fig. 6(a) and (b) for more details). $R_n(\beta_j)$ rotates $\mathbf{u}_i$ about $\mathbf{n}_i$ for β_j, $\beta_j = \frac{j\pi}{N_p}$ and N_p is a predefined number of sampled directions.

As shown in Fig. 6(b), the optimal hanging direction is determined by casting rays from $\mathbf{c}_h^i$ within the plane that is vertical to $\mathbf{v}_j^i$. Then the $\mathbf{v}_j^i$ with the most number of rays that intersect with the object is selected as the hanging direction. The objective for hanging direction detection is to find the most proper direction for the supporter to 'insert' into the object at each hanging position. Fig. 3(b) displays that if the object is sliced by a plane that is perpendicular to the hanging direction at the hanging position $\mathbf{c}_h^i$, most range around $\mathbf{c}_h^i$ should be surrounded by the object surface to provide enough contact with the supporter. Therefore, the hanging direction can be determined by searching

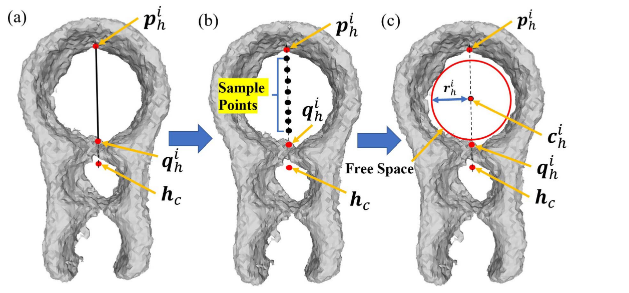

Fig. 5. Procedure of hanging position detection on the object. (a) Position of potential contact point $\mathbf{p}_h^i$, object CoM $\mathbf{h}_c$ and intersection point $\mathbf{q}_h^i$. (b) Sampling points on $\overline{\mathbf{p}_h^i\mathbf{q}_h^i}$. Sampled points are marked with black dots. (c) Hanging position selection. The cross-section of the free space is represented as the red circle. The hanging position ($\mathbf{c}_h^i$) and free space radius ($\mathbf{r}_h^i$) are also labelled.

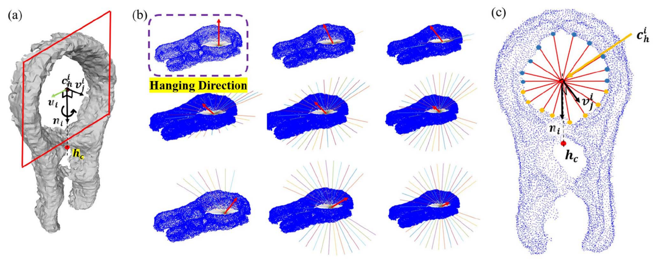

Fig. 6. (a) Schematic of hanging direction detection. $\mathbf{n}_i$ points from hanging position ($\mathbf{c}_h^i$) to object CoM ($\mathbf{h}_c$). $\mathbf{u}_i$ is a random vector that is perpendicular to $\mathbf{n}_i$. The sampled hanging direction $\mathbf{v}_j^i$ is obtained by rotating $\mathbf{u}_i$ around $\mathbf{n}_i$. The plane containing $\mathbf{c}_h^i$ and perpendicular to $\mathbf{v}_j^i$ is marked as red lines. (b) Visualizes the hanging direction detection. Red arrows represent sampled directions $\mathbf{v}_j^i$. Rays are cast from the $\mathbf{c}_h^i$ within the plane that is perpendicular to $\mathbf{v}_j^i$. The direction labeled by the dashed box is selected as the final hanging direction because most number of rays intersect with the object. (c) Shows more details about how ray casting works. Blue dots represent contact points higher than the hanging position and orange dots represent contact points lower than the hanging position.

the sampled vector $\mathbf{v}_j^i$ at $\mathbf{c}_h^i$ with most object surfaces around. Concretely, for each potential hanging direction $\mathbf{v}_j^i$ at $\mathbf{c}_h^i$, N_r rays (thin lines in Fig. 6(b)) are cast uniformly from $\mathbf{c}_h^i$ within the plane (marked red in Fig. 6(a)) perpendicular to $\mathbf{v}_j^i$. N_r is a predefined sampling number. More rays intersecting with the model means that the hanging position is more completely encircled by the object in that ray surface, making the corresponding hanging direction $\mathbf{v}_j^i$ more appropriate for hanging. For instance, the hanging of the object along the direction $\mathbf{v}_j^i$ labeled by a dashed box in Fig. 6(b) is better than other sampled directions. Then for each sampled hanging direction $\mathbf{v}_j^i$, the following values can be calculated:

$$m_1^j = \frac{\#(\text{contact points above } \mathbf{c}_h^i)}{N_r}$$
$$m_2^j = \frac{\#(\text{contact points below } \mathbf{c}_h^i)}{N_r}$$
$$m^j = \frac{\#(\text{contact points})}{N_r} \tag{5}$$

Here, m^j represents the proportion of rays that contact with object mesh. m_1^j and m_2^j are the proportions of rays that intersect with the object model above (blue points in Fig. 6(c)) and below (orange points in Fig. 6(c)) $\mathbf{c}_h^i$ at the stable hanging pose. In other words, m_1^j and m_2^j represent how well the hanging position is encircled by the object above and below the supporter at the hanging pose, respectively. The direction $\mathbf{v}_j^i$ with largest m^j is selected as the hanging direction for $\mathbf{c}_h^i$, denoted as $\mathbf{v}_h^i$. Corresponding m^j, m_1^j and m_2^j are denoted as m_i, m_i^1 and m_i^2, respectively. To further remove unfeasible hangings, only $(\mathbf{c}_h^i, \mathbf{v}_h^i)$ pairs that satisfy

$$m_i^1 < \alpha_1 \text{ and } m_i^2 < \alpha_2 \tag{6}$$

are retained for following steps. α_1 and α_2 are two thresholds. Here, α_1 is larger than α_2 because the portion of the object above the hanging position actually makes contact with the supporter during hanging.

Pruning Similar Hangings: Finally, potential hangings are filtered by merging the hanging positions and directions with similar free space size or hanging direction. Two pairs of potential hanging positions and directions $(\mathbf{c}_h^i, \mathbf{v}_h^i)$ and $(\mathbf{c}_h^j, \mathbf{v}_h^j)$ with FSRs r_h^i and r_h^j are considered similar if they satisfy:

$$\frac{\left|r_h^i - r_h^j\right|}{\max(r_h^i, r_h^j)} < r_\beta \text{ and } \left|\mathbf{v}_h^i \cdot \mathbf{v}_h^j\right| > \beta \tag{7}$$

r_β and β are two user-defined thresholds. If the conditions are met, the hanging position-direction pair with the higher m_i value is retained.

C. Supportability Detection

Similar to hangability, supportability is defined as the ability to provide support to an object hanging on it. As shown in Fig. 2, the supportability detection is to identify proper hanging positions and directions on the supporter. The detailed procedure is shown in Fig. 7(a). Intuitively, a smaller cross-sectional area (blue shaded in Fig. 3(b)) of the supporter component under the object makes it easier to place the supporter along the normal vector of that cross-section. For instance, hanging objects along the orange arrow in Fig. 7(b) is easier than along the orange arrow in Fig. 7(c). Points on the supporter point cloud are firstly selected according to (2) and clustered (Fig. 7(a.2)). Then for each cluster, the algorithm slices the supporter point cloud along different directions (Fig. 7(a.3)) at the center of every cluster. The direction with the minimum cross-sectional area is selected as the hanging direction and the centroid of cross section is selected as the hanging position.

Specifically, given a point cloud of supporter $\mathbf{S}$, the normal vectors of potential contact points must point in the opposite direction of gravity according to (2). Hence, points with normal vectors that satisfy $-\mathbf{g} \cdot \mathbf{n}_s^i > \alpha_s$ are selected as potential contact points, where $\mathbf{n}_s^i$ is the normal vector at the point for checking. $\alpha_s \in (0, 1)$ is a hyperparameter. Similar to the process of object, potential contact points are clustered into the M_c clusters with the Mean Shift method, expressed as $\{\mathbf{P}_s^i\}_{i=1}^{M_c}$ with cluster centroids $\{\mathbf{p}_s^i\}_{i=1}^{M_c}$. M_c is defined automatically in clustering. Since centroids of the cluster may not be present in the $\mathbf{S}$, the algorithm adjusts each $\mathbf{p}_s^i$ as the closest point to the cluster centroid (orange dots in Fig. 7(a.2)) in $\mathbf{S}$. Results after

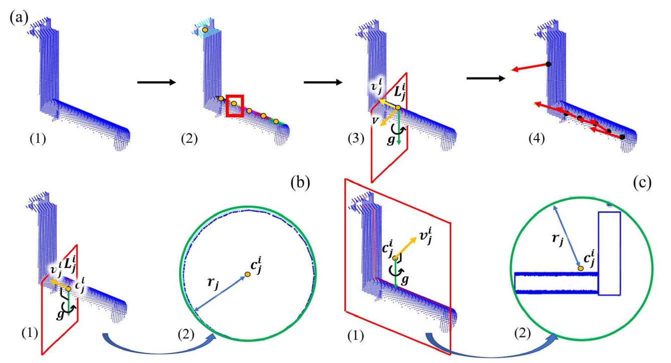

Fig. 7. (a) Procedure of supportability detection on the supporter. a.1 Shows the point cloud of the supporter. a.2 Shows the point cloud after initial selection and clustering. Potential contact points of different clusters are labelled with different colors and cluster centroids are represented as orange points. a.3 Visualizes the point cloud slicing. The slicing plane $\mathbf{L}_j^i$ is represented as red lines and points belonging to the sliced point cloud are represented as red dots. $\mathbf{L}_j^i$ contains the cluster centroid (orange point) and is parallel with gravity $\mathbf{g}$ (green arrow). $\mathbf{v}$ is a random vector that is perpendicular to $\mathbf{g}$. The normal vector of the slicing plane ($\mathbf{v}_j^i$) is generated by rotating $\mathbf{v}$ about $\mathbf{g}$ and represented as a yellow arrow. a.4 Visualizes the final results. Hanging directions are represented by red arrows and hanging positions are represented by black dots. (b) And (c) shows examples of sliced supporter point clouds with different slicing directions. Slicing planes are represented as red lines. Points belonging to the sliced point cloud are marked as red. Hanging positions and hanging directions are marked as orange points and orange arrows respectively. (b) Shows the slicing with a small boundary sphere and (c) The sliced point cloud with a large boundary sphere. (b.2) And (c.2) show the sliced point clouds and boundary spheres after being projected to slicing planes. Hanging positions $\mathbf{c}_j^i$ and FSRs r_j are also labeled.

potential contact points selection and clustering are shown in Fig. 7(a.2).

To determine the hanging positions and hanging directions of supporter, the algorithm employs M_s planes $\{\mathbf{L}_j^i\}_{j=1}^{M_s}$ to slice the point cloud $\mathbf{S}$ at each cluster center $\mathbf{p}_s^i$ (see Fig. 7(a.3)). M_s is a predefined hyperparameter. $\mathbf{L}_j^i$ contains $\mathbf{p}_s^i$ and has the normal vector $\mathbf{v}_j^i = R_g(\beta_j)\mathbf{v}, j = 1, \ldots, M_s$. $\mathbf{v}$ is a vector that is perpendicular to gravitational direction and $R_g(\beta_j)$ rotates $\mathbf{v}$ about gravitational direction for $\beta_j = \frac{j\pi}{M_s}$. The algorithm slices $\mathbf{S}$ with $\mathbf{L}_j^i$ by selecting all points in $\mathbf{S}$ that have a distance to $\mathbf{L}_j^i$ smaller than a threshold d_l (see Fig. 7(b.1) and (c.1)). Then these points are clustered using the Agglomerative Clustering Algorithm [20] and the cluster contains the $\mathbf{p}_s^i$ is chosen for subsequent steps. The geometric center of the selected cluster is denoted as $\mathbf{c}_j^i$ (orange dots in Fig. 7(b) and (c)). Next, a boundary sphere is generated for each cluster, centered at $\mathbf{c}_j^i$. The radius r_j of the sphere is defined as the distance from $\mathbf{c}_j^i$ to the farthest point from $\mathbf{c}_j^i$ in the cluster. Fig. 7(b.2) and (c.2) show the cross sections formed by intercepting the boundary sphere with $\mathbf{L}_j^i$ (green circles), corresponding center $\mathbf{c}_j^i$ and radius r_j. For each $\mathbf{p}_s^i$, the normal vector of the slicing plane with the smallest boundary sphere is selected as the hanging direction (yellow arrows in Fig. 7(a.4)) of the supporter, denoted as $\mathbf{v}_s^i$. The center of the selected boundary sphere is regarded as the corresponding hanging position (red dots in Fig. 7(a.4)), denoted as $\mathbf{c}_s^i$. The radius of the smallest boundary sphere is denoted as r_s^i. Fig. 7(b) and (c) demonstrate the situations with small and large boundary spheres, respectively. Finally, hanging positions $\{\mathbf{c}_s^i\}_{i=1}^{M_c}$, hanging directions $\{\mathbf{v}_s^i\}_{i=1}^{M_c}$ and boundary circle radii $\{r_s^i\}_{i=1}^{M_c}$ are stored for following steps.

D. Matching and Evaluation

The final step of the proposed method involves the object/supporter matching and selecting the most appropriate hanging pose. For each hanging position-direction pair of the object ($\mathbf{c}_h^i$, $\mathbf{v}_h^i$) and the supporter ($\mathbf{c}_s^j$, $\mathbf{v}_s^j$), the algorithm first aligns the hanging directions $\mathbf{v}_h^i$ and $\mathbf{v}_s^j$. Next, the hanging position $\mathbf{c}_h^i$ on the object is moved to coincide with the supporter hanging position $\mathbf{c}_s^j$. Finally, the object is rotated about the hanging direction $\mathbf{v}_s^j$ until the vector points from $\mathbf{c}_h^i$ to the object CoM is parallel to the gravitational direction. These steps result in a transformation $T_{ij} \in SE(3)$. Matches with a free space radius $r_h^i < \alpha_m r_s^j$ are ignored, where r_h^i is the FSR of the object and r_s^j is the boundary sphere radius of the supporter. $\alpha_m \in (0, 1]$ is a hyperparameter. This ensures that the object can be placed without interference in the free space around the supporter.

In practice, there may be multiple potential hanging positions on both the object and the supporter, and not all matches are feasible. Hence, it is necessary to evaluate these matches and select viable ones. The matching evaluation is implemented by calculating and ranking the value of a cost function which considers three criteria: collision, distance to the supporter boundary and contact area.

Collision: To ensure successful hanging, it is essential that the predicted hanging pose is collision-free. If the object collides with the supporter at the predicted hanging pose T_{ij}, some points of the supporter point cloud would be inside the object mesh transformed by T_{ij}. These points are denoted as $\mathbf{S}_{collision} \subseteq \mathbf{S}$. The cost of collision between the object and the supporter is defined as:

$$S_c = |\mathbf{S}_{collision}| \tag{8}$$

Distance to Supporter Boundary: Another criterion for evaluating the hanging pose is the distance between the hanging position and the boundary of the supporter. If the hanging position $\mathbf{c}_s^j$ of the supporter is very close to the edge of the supporter, the object is more likely to fall off with disturbance. Hence, it is safer for the hanging position to be far away from the boundary of the supporter. Considering $\mathbf{c}_s^j$ and corresponding hanging direction $\mathbf{v}_s^j$, the cost of distance to the boundary can be represented as:

$$S_b = \frac{\max(d_1, d_2)}{d_1 + d_2} \tag{9}$$

where d_1 and d_2 are distances from $\mathbf{c}_s^j$ to the convex hull of supporter point cloud along $\mathbf{v}_s^j$. A smaller S_b indicates that the hanging position is farther away from the boundary of the supporter, making it less prone to falling off.

Contact Area: As shown in (4), for each hanging position $\mathbf{c}_h^i$ of the object, there is a corresponding cluster of potential contact points $\mathbf{P}_h^i$. The number of points in $\mathbf{P}_h^i$ represents the size of the contact area that can support stable hanging. Therefore, selecting the hanging position $\mathbf{c}_h^i$ with a larger $|\mathbf{P}_h^i|$ is more likely to result in a stable hanging. The cost of the contact area is defined as:

$$S_a = 1 - \frac{|\mathbf{P}_h^i|}{\max(\{|\mathbf{P}_h^k|\}_{k=1}^{N_k})} \tag{10}$$

where N_k is the number of hanging positions on the object after the hangability detection. A smaller S_a indicates a larger contact

TABLE I
PARAMETERS USED IN EXPERIMENTS

Name	Value	Name	Value	Name	Value	Name	Value
α_h, α_s	0.8	N_h	10	l_c	20	α_m, γ_1	1.0
N_p, N_r	50	α_1	0.7	α_2	0.5	γ_2, γ_3	1.0
M_s	20	d_l, r_α	0.002	r_β	0.1	β	0.9

area between the object and the supporter, which makes it more likely to result in a stable hanging.

The total cost is computed as:

$$S_{total} = \gamma_1 S_c + \gamma_2 S_b + \gamma_3 S_a \tag{11}$$

γ_1, γ_2 and γ_3 are three hyperparameters within (0,1]. The success of the hanging is more likely when the S_{total} value is smaller. Hence, all predicted hanging poses are ranked based on their S_{total} value in ascending order, the smaller S_{total} value the better. Finally, a motion planner checks whether a path exists from the pose with the lowest cost one by one until find a practical hanging pose. More details about the motion planning can be found in the Supplementary Material.

III. Experiments

Our experiments aim at evaluating the following questions: (1) How stable are hanging poses predicted by our method? (2) How well does the robot execute these hanging poses? (3) How does scanning quality affect the performance of the algorithm? The parameters used in the experiments are manually set as values in Table I based on the author's daily experience. In the experiments, the mesh models are reconstructed using TSDF Fusion [21], and point clouds and meshes are processed by Open3D [22]. Please note that although we generate meshes with TSDF Fusion, our method doesn't have any limitations about how to reconstruct models. Any kind of mesh reconstruction method, like [23], could be used. Algorithms are implemented in Python on a computer equipped with AMD Ryzen^{TM} 9 5950X (3.4 GHz).

A. Experiments in Simulation

To test the stability of the predicted poses, we conduct experiments in an open-source simulation environment PyBullet [24]. Object meshes and supporter point clouds are generated from both multiple and single depth images to test the algorithm sensitivity to the scanning quality. Then our algorithm predicts the hanging poses. Finally, the object is loaded at the pose with the smallest cost and released to evaluate its stability. The hanging pose is considered stable if it is collision-free and the object doesn't fall on the ground after release. We also rerun the code provided in [12] as the baseline. In [12], the authors claim the 68.3% success rate with 3104 pairs of objects and supporters for testing, but the paper doesn't provide enough details about the 3104 pairs they used to reproduce their result. Therefore, we run simulated experiments with different 1600 pairs from the dataset introduced in [12] on both baseline and our method. Details about the testing set we used will be provided online so that intereseted readers can reproduce. Results of the simulated experiments are shown in Table II.

B. Experiments in the Real World

To further test the executability of the proposed method, we also conduct experiments in real world with a robot. The dataset used for real-world experiments contains 14 daily objects for hanging and 10 supporting items (displayed in Fig. 8(b)). 4 out of the 14 objects for hanging are unhangable, which enables us to test whether the method proposes false positive predictions. We recruit 5 volunteers to annotate the objects. An object is considered hangable if more than 3 out of 5 volunteers think there exist suitable hanging poses for all supporters with one contact point. Similarly, if more than 3 volunteers believe the object cannot be hung on any supporter, it is classified as unhangable. Six supporting items are normal IKEA hooks for hanging objects. Two are special supporters: a hand model and a supporter made by gluing a marker pen into a piece of acrylic board. The other two supporters are 3D-printed supporters from the dataset used in the simulation experiments. All of these objects and supporters are previously unseen by the robot. As shown in Fig. 8(a), we build a robot system with a Franka Emika robot arm mounted with a gripper and a Primesense RGB-D camera. The robot system is controlled to scan the object and supporter, manipulate, and hang the object onto the supporter. In each trial, our algorithm takes the mesh and point cloud as input and predicts potential hanging poses. Ten predicted poses with the lowest S_{total} value are selected and input to the motion planner until finding an executable pose. Then the robot only executes that hanging prediction.

Three distinct experiments are launched to validate the accuracy and stability of our method. (1) Accuracy validation: We first run an experiment with 14 objects for hanging and 10 supporting items which gives a total of 100 trials for feasible hangings and 40 trials for unfeasible hangings. For each trial, the object is scanned from 10 different views and the supporter is scanned from 3 views. The outcomes are shown in Table III. (2) Sensitivity to object scanning: In the second experiment, we assess our method's resilience to variations in object mesh quality by scanning the object from different numbers of viewpoints. Specifically, we conduct three groups of experiments with 10 hangable objects and 1 supporter. In each group, objects are scanned from varying numbers of viewpoints (10/3/1 views respectively), while the supporter is consistently scanned from 3 viewpoints. The results are shown in Table IV. (3) Sensitivity to supporter scanning: Similarly, in the third experiment, we examine the robustness of our method with respect to the scanning of the supporter. Here, we conduct experiments involving 2 hangable objects and all 10 supporters. The objects are consistently scanned from 10 viewpoints, while the supporters are scanned from 3 or 1 viewpoint(s). The results are shown in Table V.

IV. Discussion

Hanging Accuracy: Table II illustrates that the hanging accuracy of our method significantly outperforms the baseline method [12]. Particularly, with the same input (**single view** for both object and supporter), the hanging success rate of our method (71.2%) is still higher than the success rate of baseline (64.4%). In addition, the baseline model is trained on 16195 pairs of objects and supporters. On the contrary, the proposed geometric-based method doesn't require any data for training and has better stability while facing previously unseen objects and supporters.

TABLE II
HANGING POSE PREDICTION ACCURACY (%) WITH DIFFERENT KINDS OF OBJECTS IN SIMULATED EXPERIMENTS

Methods	Mean	Bag	Cap	Clothes Hanger	Utensil	Headphone	Knife	Mug	Racquet	Scissor	Wrench	Others
Ours (24V + 3V)	**83.3**	91.0	64.0	95.7	41.3	91.0	66.7	83.9	40.4	93.1	74.7	89.1
Ours (24V + 1V)	**81.0**	86.5	68.6	91.4	42.5	86.5	68.4	79.2	40.4	90.8	75.9	88.3
Ours (3V + 1V)	**79.6**	90.5	58.1	87.7	28.8	85.5	64.9	78.6	34.6	90.1	78.2	89.4
Ours (1V + 1V)	**71.2**	77.5	26.7	85.9	33.8	77.5	78.9	47.6	32.7	87.0	88.5	84.9
Omnihang [12]	**64.4**	69.8	60.5	61.3	46.3	76.8	50.9	57.1	73.1	67.2	58.6	61.5

a]The scanning conditions are displayed in the table, the condition for objects followed by the condition for supporters. For example, '3V + 1V' means three viewpoints for object scanning and 1 viewpoint for supporter scanning.
The bold values are the mean accuracy over all objects with different methods.

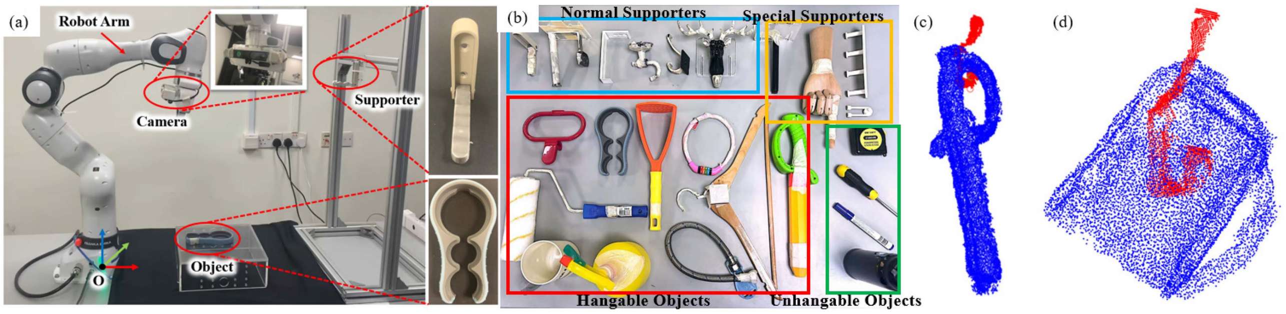

Fig. 8. (a) Setup of real-world experiments. (b) Dataset used for experiments. 10 hangable objects are enclosed in a red box. 4 unhangable objects are enclosed in a green box. 6 normal IKEA supporters are enclosed in a blue box and 2 special supporters are enclosed in an orange box. (c) A failure example of unstable hanging. (d) A failure example of a hanging prediction with collision.

TABLE III
HANGING ACCURACY OF REAL-WORLD EXPERIMENTS

Object Types	Total	Hanging Detected	Within Workspace	Execution Success	Rate of Success
Hangable	100	97	96	77	77.8%
Not Hangable	40	1	0	0	97.5%

TABLE VI
FAILURE REASONS DISTRIBUTION

Type	Unstable Hanging Prediction	Prediction with Collision	No Hanging Detected	Problems in Execution	Total
Number	7	5	3	7	22

TABLE IV
HANGING ACCURACY OF OBJECTS SCANNED IN DIFFERENT NUMBER OF VIEWS

View Number	Total Trials	Hanging Detected	Execution Success	Rate of Success
10	10	9	9	90%
3	10	10	8	80%
1	10	10	7	70%

TABLE V
HANGING ACCURACY OF SUPPORTERS SCANNED IN DIFFERENT NUMBER OF VIEWS

View Number	Total Trials	Hanging Detected	Execution Success	Rate of Success
3	20	18	18	90%
1	20	18	13	65%

Results of real-world experiments further display the reliability and the high accuracy of our approach. As shown in Table III, in real-world experiments our method identifies hanging poses for 97 pairs out of 100 pairs of hangable objects and supporters and the robot successfully executes 77 of them. In one trail all 10 predicted poses are out of the robot workspace. Since such a situation depends on the robot's position and is not the focus of our algorithm, this case was excluded. Only one hanging pose is detected for the 40 unhangable pairs, which displays the stability of our method for false positive cases. A comprehensive breakdown of the failure reasons is displayed in Table VI. 7 failure cases are caused by unstable predicted hanging poses. Although some predicted poses can theoretically afford stable hanging (like the one shown in Fig. 8(c)), the objects at these poses are likely to fall off under disturbance. 5 failures are due to predicted hanging poses in the presence of collision (Fig. 8(d)). The algorithm doesn't find any hanging positions/directions on the object for 3 pairs, which results in another 3 failures. These three kinds of failure can be improved by methods that can yield more potential hanging poses like reducing the cluster size after contact point selection. However, such modification can also increase the computational demands and reduce the processing speed. Moreover, increasing the mesh and point cloud quality can also increase the hanging accuracy. The other 7 failures can be attributed to problems in execution like the object dragged by the gripper after hanging, which can be further improved by optimizing the pick-and-hang trajectory.

Sensitivity to Scanning: Tables II, IV and V demonstrate that the hanging accuracy reduces as scan quality decreases but still remains high. Notably, even with meshes and point clouds reconstructed from single-view scanning, our method still achieves a high success rate and outperforms the baseline method. One reason for the reduction in accuracy is that when scanning quality is low, some structural elements that can afford hanging, like the cup handle, are incomplete and corresponding hangings cannot be detected. Moreover, if the reconstructed mesh and point cloud are incomplete, the algorithm may treat a hanging pose with collision as a collision-free one. For example, as displayed in Fig. 8(d), although the supporter penetrates the wall of the cup, the algorithm considers the pose collision-free. Because the cup wall is incomplete and contains a 'fake' hole. Finally, potential hanging positions are filtered by checking the SDF value to ensure they are outside the object. Low-quality scanning can introduce inaccuracies in the SDF values, rendering this step less effective. However, as long as the essential structure required for hanging remains relatively intact in meshes and point clouds, our algorithm can still detect the corresponding hanging and furnish accurate predictions. Consequently, the hanging accuracy under poor scanning conditions remains high.

Limitations: Although the experimental results are encouraging, our method comes with certain limitations. The most notable limitation is that the proposed method exclusively addresses the hanging situation with single-point contact. While this encompasses a reasonably broad range of scenarios, it does exclude some situations, such as hanging a goblet on a wine glass rack or hanging garments, which involve multiple contact points and more complex interactions. Indeed, the hanging accuracy of some categories of objects (like utensil and racquet) in Table II are significantly lower than other categories because these objects are difficult to be hung with single contact. Moreover, the current method mainly focuses on the hanging poses prediction and execution can be further improved by optimizing the trajectory generation. The research about solving these issues is planned for future.

V. Conclusion and Future Work

This paper proposes a learning-free method for hanging detection of previously unseen objects and supporters. The proposed method first finds potential hanging positions and directions on both the object and supporter. Then these hanging positions and directions are matched and evaluated to obtain feasible hanging poses. We also run experiments in simulation and real-world environments to validate the proposed method. Results show that our approach achieves a high success rate, outperforming the baseline results. Moreover, the accuracy remains high under poor scanning conditions. Future work will focus on exploring more complex hanging scenarios, such as hanging with multiple contact points and the optimization of the entire pick-and-hang procedure.

Acknowledgment

The authors thank Xin Meng at NUS for programming assistance in the experiments.

References

[1] R. Bogue, "Domestic robots: Has their time finally come?," *Ind. Robot: An Int. J.*, vol. 44, no. 2, pp. 129–136, 2017.

[2] Y. Jiang, M. Lim, C. Zheng, and A. Saxena, "Learning to place new objects in a scene," *Int. J. Robot. Res.*, vol. 31, no. 9, pp. 1021–1043, 2012, doi: 10.1177/0278364912438781.

[3] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "KPAM: Keypoint affordances for category-level robotic manipulation," in *Proc. Int. Symp. Robot. Res.*, 2022, pp. 132–157.

[4] E. Ruiz and W. Mayol-Cuevas, "Geometric affordance perception: Leveraging deep 3D saliency with the interaction tensor," *Front. Neurorobot.*, vol. 14, 2020, Art. no. 45. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnbot.2020.00045

[5] E. Ruiz and W. Mayol-Cuevas, "Where can I do this? Geometric affordances from a single example with the interaction tensor," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2192–2199.

[6] D. Estevez, J. Victores, R. F. Fernandez, and C. Balaguer, "Towards clothes hanging via cloth simulation and deep convolutional networks," *Simul. Notes Eur.*, vol. 31, pp. 169–176, 2019.

[7] A. Simeonov et al., "SE(3)-equivariant relational rearrangement with neural descriptor fields," in *Proc. Conf. Robot Learn.*, 2023, pp. 835–846.

[8] P. Xu, H. Cheng, J. Wang, and M. Q.-H. Meng, "Learning to reorient objects with stable placements afforded by extrinsic supports," *IEEE Trans. Automat. Sci. Eng.*, to be published, doi: 10.1109/TASE.2023.3314461.

[9] J.-S. Ha, D. Driess, and M. Toussaint, "Deep visual constraints: Neural implicit models for manipulation planning from visual input," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 10857–10864, Oct. 2022.

[10] D. Driess, J.-S. Ha, M. Toussaint, and R. Tedrake, "Learning models as functionals of signed-distance fields for manipulation planning," in *Proc. Conf. Robot Learn.*, 2022, pp. 245–255.

[11] J. A. Haustein, K. Hang, J. Stork, and D. Kragic, "Object placement planning and optimization for robot manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 7417–7424.

[12] Y. You, L. Shao, T. Migimatsu, and J. Bohg, "OmniHang: Learning to hang arbitrary objects using contact point correspondences and neural collision estimation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5921–5927.

[13] K. Takeuchi, I. Yanokura, Y. Kakiuchi, K. Okada, and M. Inaba, "Automatic hanging point learning from random shape generation and physical function validation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 4237–4243.

[14] K. Takeuchi, I. Yanokura, Y. Kakiuchi, K. Okada, and M. Inaba, "Automatic learning system for object function points from random shape generation and physical validation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 2428–2435.

[15] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychol. Rev.*, vol. 94, no. 2, pp. 115–147, 1987.

[16] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. Int. Conf. Robot. Automat. Millennium Conf.. IEEE Int. Conf. Robot. Automat. Symposia Proc.*, 2000, pp. 995–1001.

[17] H. Goldstein, C. Poole, J. Safko, and S. R. Addison, "Classical mechanics, 3rd ed.," *Amer. J. Phys.*, vol. 70, no. 7, pp. 782–783, Jul. 2002, doi: 10.1119/1.1484149.

[18] C. Yuksel, "Sample elimination for generating poisson disk sample sets," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 25–32, 2015.

[19] D. Comaniciu and M. Peter, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.

[20] M. L. Zepeda-Mendoza and O. Resendis-Antonio, *Hierarchical Agglomerative Clustering*. New York, NY, USA: Springer, 2013, pp. 886–887, doi: 10.1007/978-1-4419-9863-7-1371.

[21] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proc. 23rd Annu. Conf. Comput. Graph. Interactive Techn.*, 1996, pp. 303–312.

[22] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," 2018, *arXiv:1801.09847*.

[23] N. Rüegg, S. Tripathi, K. Schindler, M. J. Black, and S. Zuffi, "BITE: Beyond priors for improved three-D dog pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 8867–8876.

[24] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016. [Online]. Available: http://pybullet.org