Learning to Place Objects onto Flat Surfaces in Human-Preferred Orientations

Rhys Newbury*, Kerry He*, Akansel Cosgun and Tom Drummond
Monash University, Australia

Abstract—We study the problem of placing a grasped object on an empty flat surface in a human-preferred orientation, such as placing a cup on its bottom rather than on its side. We aim to find the required object rotation such that when the gripper is opened after the object makes a contact with the surface, the object would be stably placed in the desired orientation. We use two neural networks in an iterative fashion. At every iteration, Placement Rotation CNN (PR-CNN) estimates the required object rotation which is executed by the robot, and then Placement Stability CNN (PS-CNN) estimates if the object would be stable if it is placed in its current orientation. In simulation experiments, our approach places objects in human-preferred orientations with a success rate of 86.1% using a dataset of 18 everyday objects. A real world implementation is presented, which serves as a proof-of-concept for direct sim-to-real transfer. We observe that sometimes it is impossible to place a grasped object in a desired orientation without re-grasping, which motivates future research for grasping with intention to place objects.

I. INTRODUCTION

Everyday objects are usually placed in certain orientations that are convenient to humans. For example, a cup is designed to be placed on its bottom rather than on its side. Placing objects down properly in orientations preferable to humans is a fundamental skill for service robots. For example, a robot that is unpacking the dishwasher should place plates, glasses and bowls on shelves in certain orientations. Research in robotic manipulation over the past decades has mostly focused on how to pick up objects [1], with recent works utilizing the advances in deep learning [2]–[5]. However, what to do with the object after it has been grasped has largely been overlooked in the field. In fact, the most common practice in pick-and-place robotic manipulation scenarios is to drop the object at a height without any consideration to its resulting pose [3], [5], [6]. Only a handful of researchers have studied how to place the grasped object down, as reviewed in Section II. Furthermore, no work to our knowledge has leveraged deep learning for the object placement problem.

In this paper, we study the problem of placing a grasped object down on an empty flat surface in the human-preferred orientation. We consider the solution to the problem as finding the required object rotation such that when the robot end-effector is lowered until a contact is made and the gripper is opened, the object would be stably placed in the human-preferred orientation. We present two neural networks, Placement Rotation Convolutional Neural Network (PR-CNN) and Placement Stability Convolutional Neural Network (PS-CNN) that are used in an iterative algorithm. Both networks take input depth images of the grasped object (with the end-effector) from three viewpoints. At every iteration, the PR-CNN recommends an object rotation to achieve the human-preferred orientation, which the robot executes, and then PS-CNN estimates the stability of the proposed object orientation. The algorithm is stopped in one of two conditions: 1) PR-CNN’s output is the rotational identity which suggests that the object orientation has converged or 2) The maximum number of iterations is reached, at which point the rotation that yields the the maximum predicted stability is chosen among all iterations. The iterative approach helps in getting more observations from the object and correcting the errors of PR-CNN. Stability estimation provided by PS-CNN is useful when the output of the PR-CNN results in oscillations in the object orientations, which is a common failure mode when distinguishing features that could typically be used the determine the upright orientation is occluded in the depth images. We train both networks in simulation using 18 objects we picked from the KIT object database [7]. We report on the experimental results starting from random object orientations in the robotic gripper.

The contributions of this paper are four-fold:

• PR-CNN: Learning the required rotation to reach desired object orientations, from depth images of the gripper holding an object only, without explicit object
pose detection.
- PS-CNN: Learning to estimate whether an object would end up in a stable orientation, again from depth images without object pose detection.
- Iterative Placement with Stability: An algorithm that iteratively uses PR-CNN to move the arm and PS-CNN to assess the hypothetical stability, which helps gather more observations and correct errors.
- An implementation on a robotic system, demonstrating the feasibility of direct sim-to-real transfer.

The organization of the paper is as follows. We first review the relevant literature in Section II. We then describe the problem in Section III. Our approach, called Iterative Placement with Stability, is detailed in Section IV. Experimental setup is presented in Section V before presenting the results and discussing failure modes in Section VI. Robot implementation details are described in Section VII before concluding in Section VIII.

II. RELATED WORK

A. Robotic Placing

In one of the earliest implementations of robotic object placement, Edsinger and Kemp [8] use a compliant robotic arm to place an object onto a shelf by moving the arm to a fixed configuration and then lowering the end-effector using force control, hence utilizing contact with the environment. However, this approach assumes that the pose of the object in the gripper is known, which is unavailable for unknown objects. Since then, many researchers approached the placement problem analytically, attempting to find flat features on the object and the surface on which the object can be placed [9]–[13]. Baumgart [9], [10] find stable poses of the object analytically by finding a point of first object contact, followed by rotating the object such that additional contact points are found. Their approach runs in real time, however does not take into account preferred placement orientations. Harada [11], [12] matches planar surface patches on the object with planar surface patches in the environment, which allows finding placements on large, flat surfaces, but also less obvious placements such as a mug hanging on a flat bar. Their approach, however, requires the 3D model of the object and its pose. Haustein [13] presents a similar approach and uses Monte Carlo Tree Search to optimize motion planning to reach the stable pose.

Majority of recent approaches in robotic manipulation is driven by machine learning approaches, and object placement is no exception. The approach presented by Jiang [14], [15] uses learning on hand-designed features and successfully places known objects stably 98% of the time and new objects 82% of the time. Paolini [16] estimates the probability of a successful placement, then attempts to solve for the most likely placement location, given a grasped object. Fu [17] use hand chosen features to try and find the upright orientation of man-made objects.

B. Planning for Placement

Object placement is also studied in the task and motion planning context. Alami [18] first formalized robotic manipulation as atomic actions to achieve a higher-level task in configuration space. Grasping is not the only way to place objects to their correct places as objects can also be moved using non-prehensile manipulation. Scholz [19] optimizes for the configuration of tabletop objects and plans pushing actions to achieve the desired table configuration. Cosgun [20] plans for a sequence of pushing actions on a cluttered table in order to make space to place a new object.

C. Representing Rotations in Neural Networks

Rotations can be represented in many ways, such as rotation matrices, Euler angles, quaternions and axis-angle representations. The choice of rotation representation has an important effect on the performance of machine learning models when inputs and/or outputs include rotations [21]. A common issue of angular representations is their discontinuity stemming from their periodic nature. Theoretical analyses suggest that smooth functions [22] or functions which have stronger continuity properties have lower approximation errors [23], [24]. Zhou [21] posits that SO(3) rotations are discontinuous under any representation using four or fewer dimensions, and propose a six dimensional representation for SO(3) rotations which they show to outperform all other typical representations, in the context of machine learning. We adopt this rotational representation in this paper.

III. PROBLEM DESCRIPTION

Fig. 2 illustrates the axes and angles we define for an object. The coordinate frame defined by the axes \((x, y, z)\) represents an arbitrary and fixed world frame, in which gravity is acting in \(-z\) direction. We define the stable axis \(\hat{x}_S\) as the orthogonal unit vector to the surface on which objects are placed. The placement surface is set as the infinite \(xy\)-plane, hence we set \(\hat{x}_S\) to be in the direction of the \(z\)-axis. We assign an upright vector \(\hat{v}_S\) attached to an object, such that \(\hat{v}_S = \hat{x}_S\) when the object is in its human-preferred orientation. An important property of object placement on an uncluttered infinite plane is that the placement is independent of any rotation about the stable axis \(\hat{x}_S\). To illustrate this, any rotation of the mug shown in Fig. 2 about \(\hat{x}_S\) can be reframed as a rotation of the global reference frame about the same axis. Because of this, there exists a

![Image](image_url)
continuous set of rotations $R \in SO(3)$ that can orient an object to a human-preferred orientation. We uniquely define the ground truth rotation $R_T$ as the shortest possible rotation required to move the object from its current orientation to a human-preferred orientation. The ground truth rotation $R_T$ can therefore be found as the transformation required to rotate the unit vector $\hat{v}_S$ to $\hat{x}_S$. This rotation is calculated most easily using axis-angle representation, which is defined by an angle $\theta$ rotated about an axis $n$. The ground truth rotation angle $\theta_T \in [-\pi, \pi]$ is the angle between unit vectors $\hat{v}_S$ and $\hat{x}_S$ and can be found by:

$$\theta_T = \arccos (\hat{x}_S \cdot \hat{v}_S)$$ (1)

The ground truth axis $n_T$ is defined as the axis perpendicular to both $\hat{v}_S$ and $\hat{x}_S$ and can be found by:

$$n_T = \hat{x}_S \times \hat{v}_S$$ (2)

We consider the robotic placement problem as follows. The robot starts with an object already in hand and the task is to successfully place the object down. A solution to the problem is a proposed object rotation that would result in a particular object orientation (the human-preferred orientation in this case). We assume that the robot has no a priori knowledge about the object class, 3D model or the human-preferred orientation, however it has access to depth cameras and joint force sensors. We use a simple placing behavior: the end-effector is lowered along the $-z$ direction with a fixed orientation until a contact with the table is felt. The gripper fingers are then opened and the end-effector is retracted along the reverse direction of the end-effector. Successful placement onto the tabletop requires that the object is stable and in the human-preferred orientation under gravitational and contact forces after release.

IV. ITERATIVE PLACEMENT WITH STABILITY

We propose an iterative, learning-based approach to robotic object placement. Three depth cameras are used in order to observe the object from different viewpoints. We use two neural networks:

- A network takes the depth images as input and outputs the rotation that should be applied to the object so that it results in a human-preferred orientation. We call this network PR-CNN: Placement Rotation Convolutional Neural Network.
- A network takes the depth images as input and estimates the confidence level that the object would be stable if it is placed in its current orientation. We call this network PS-CNN: Placement Stability Convolutional Neural Network.

Note that the two networks have different criteria. PR-CNN estimates the required rotation towards the ground truth orientation, whereas PS-CNN considers the physics when the object is released. Moreover, a stable placement does not necessarily mean the object would end up in a human-preferred orientation. For example, if a cup is placed upside down, the placement would be stable but not in the human-preferred orientation.

At each iteration, we first get a proposed rotation from PR-CNN and apply the rotation on the object. Assuming the gripper and grasped object act as a single rigid body (i.e. no slipping), we can execute object rotations by applying the same rotation to the robotic gripper. We then estimate the stability of the resulting object orientation using PS-CNN. This is repeated until PR-CNN’s output converges within a threshold of $\epsilon$ to the rotational identity since theoretically an object which is already at a stable orientation will yield a rotational identity when evaluated via PR-CNN. If, however, the maximum number of iterations is reached without convergence, we pick the rotation that yielded the highest stability among all the iterations, as estimated by PS-CNN. The pseudocode can be seen in Algorithm 1.

Algorithm 1 Proposed placement algorithm

1. function PLACEOBJECT
2.     for $i = 1 \rightarrow max_{iter}$ do
3.         rotation[$i$] ← PR-CNN(img)
4.         robot_rotate_by(rotation[$i$])
5.         orientation[$i$] ← get ee rotation()
6.         stability[$i$] ← PS-CNN(img)
7.         if $|rotation[i]| < \epsilon$ then
8.             return
9.         end if
10.     end for
11.     max_index ← max(stability)
12.     robot_rotate_to(orientation[max_index])
13. end function

The iterative approach helps with fine-tuning the object orientation especially if the object is slipping in the gripper, as well as getting new observations from the object. The use of the PS-CNN in evaluating stability helps in resolving the situations when proposed rotations show oscillatory behavior, which can happen with near symmetric objects such as cups. A more detailed discussion on common failure modes can be found in Section VI-C.

A. Placement Rotation CNN (PR-CNN)

We learn the required rotation that is applied to the object that would result in a human-preferred orientation. The ground truth rotation $R_T$ is obtained analytically using the methodology outlined in Section III and a single network is trained on all the objects in a given dataset.

$$Q_{\theta,1} = R$$ (3)

$$\theta^* = \arg \min_{\theta \in \Theta} E_{R,u}[L_{geodesic}(R, Q_{\theta,1}(u))]$$ (4)

Where $L_{geodesic}$ is the geodesic loss function (Equation 5) and $\Theta$ defines the parameters of PR-CNN.

PR-CNN inherits its architecture from DenseNet-121 [25] and is pre-trained on ImageNet [26]. PR-CNN defines the set of parameters $\Theta$ used to represent the required rotation function $Q_{\theta,1}$. PR-CNN takes as input three $64 \times 64$ depth images of the object. The depth values are saturated at 0.5m and then normalised between $[-0.5, 0.5]$. Modifications were made to the final layers to output a six dimensional repre-
representation of $SO(3)$ rotations as proposed by Zhou [21]. This represents the first two columns of a rotation matrix, which is then converted to a full rotation matrix using the Gram-Schmidt-like process described by Zhou [21]. Furthermore, we use a similar loss function to Zhou [21], which is the geodesic distance between the output and the ground truth:

$$L_{\text{geodesic}} = \arccos \left( \frac{\text{tr}(R_z R_z^{-1}) - 1}{2} \right)$$

(5)

where $\text{tr}(R)$ is the matrix trace operator. For a rotation matrix, this is defined as:

$$\text{tr}(R) = R_{00} + R_{11} + R_{22}$$

(6)

B. Placement Stability CNN (PS-CNN)

We learn the estimated stability of an object being placed in its current orientation. The ground truth binary label is obtained by physics simulation.

$$Q_{\theta, 2} = E[S|u]$$

(7)

Our goal is to attempt to learn this function over a series of objects and orientations that classifies stability according to a binary success metric.

$$\theta^* = \arg\min_{\theta \in \Theta} E_{s,u}[L(S, Q_{\theta, 2}(u))]$$

(8)

Where $L$ is the binary cross-entropy loss function and $\Theta$ defines the parameters of PS-CNN.

PS-CNN is based on VGG16 [27] and is pretrained on ImageNet [26]. PS-CNN defines the set of parameters $\Theta$ used to represent the stability function $Q_{\theta, 2}$. The first four convolutional blocks are frozen and only the final layers are fine tuned. The input to the network is a depth image stacked and scaled to $224 \times 224$. Similar prepossessing steps as PR-CNN were applied to the depth images. Modifications were made to the final layer to output one number, with sigmoid activation. This number represents the confidence of stability if the object was placed in the current orientation.

V. EXPERIMENTS

A. Object Models

We picked 18 everyday objects from the KIT object models database [7], which provides 3D meshes of the objects. The simulated renderings of all 18 objects can be seen in Fig. 3. The objects were picked such that each one had a well defined, single upright vector (we avoided objects with multiple stable axes, such as a coke can or cereal box). The upright vector for each object model is annotated with the human-preferred orientation. We use a scene where three depth cameras were positioned orthogonally from each other, to the front, left and right of the gripper, at a 25cm radius around the point $p_c$ at which objects and the gripper interact (as seen in Fig. 1).

B. Data Collection

We used the PyRep toolkit [28] for the simulation environment. To generate a data point, we randomly pick an object from the 18 objects, and randomly sample orientations with slight positional variations around $p_c$. We collect the three depth camera images along with the ground truth rotations that would lead the object to the human-preferred orientations and binary labels indicating whether the object is in a stable placement orientation or not.

In order to evaluate how our approach performs, we collected two datasets: data with and without the robot arm in simulation. Both datasets contain 85,000 data points each.

Without Robot: In order to evaluate the feasibility of our approach, we first experiment on a setup where there is no robot involved and the object is moved in simulation directly. This mode is simpler, because there are no contact forces between the gripper and the object that can affect the object orientation, and we can move the objects without kinematic constraints. Placement stability is evaluated by setting the position of the object to the lowest possible point without colliding with the placement surface, then enabling gravitational physics to simulate placement.

With Robot: Random object grasps are sampled with a Panda Franka robotic arm. Placement stability is evaluated by using inverse-kinematics to rotate and place the object onto a surface. Placement is achieved by lowering the end effector until a contact is detected via force sensing of the robot joints. In order to increase the manipulability of the robot arm, objects are placed on a surface elevated from the ground.

An important distinction between the datasets is that in the “With Robot” dataset, the robot arm appears in the depth images along with the grasped object, whereas in the “Without Robot” dataset the depth images contain only the object of interest.

C. Methods

We compared the performance of several placement methods to benchmark our proposed algorithm against.

- **Random placement (RND):** Object rotation is randomly sampled.
- **Single pass (SP):** A single pass of PR-CNN is used to determine the object rotation.
- **Iterative (ITR):** PR-CNN is run iteratively until the identity rotation is achieved or a maximum number iterations (5 in this case) has been reached.
• **Iterative with Stability (ITR-S):** Our full approach combining PR-CNN and PS-CNN, as detailed in Section IV.

D. Metrics

• **Success Rate:** The percentage of placements where the steady-state object orientation is within $\pm 15^\circ$ of the human-preferred ground truth orientation.

• **Stability Rate:** The percentage of placements where the object stays stationary for a minute and the final object orientation is within $\pm 15^\circ$ of the initial placement orientation. This metric was adopted from [15].

• **Angular Error:** The average angle difference between the upright vector $\hat{v}^S$ and the stable axis $\hat{x}^S$.

For the “With Robot” experiments, placements that lead to kinematically infeasible orientations were ignored and did not contribute towards the results.

VI. RESULTS

A. Without Robot

The aggregate results on all the objects is shown in Table I. RND expectedly performed very poorly with 3.6% success rate. It had, however, achieve a moderate 45.8% stability rate. The discrepancy is because objects can be stably placed in orientations other than the human-preferred orientation.

| Method | Success Rate | Stability Rate | Angular Error |
|--------|--------------|----------------|--------------|
| RND    | 3.6%         | 45.8%          | 87.0°        |
| SP     | 97.9%        | 98.6%          | 4.2°         |
| ITR    | 98.4%        | 98.4%          | 3.1°         |
| ITR-S  | 97.0%        | 97.1%          | 3.5°         |

TABLE I: Placement results without the robot, tested on all objects

**SP, ITR and ITR-S** all performed similarly, achieving 97.9%, 98.4% and 97.0% success rates respectively. ITR performed the best in terms of success rate and angular error, and outperformed the full approach ITR-S. Fig. 6 shows the placement success rate for each object and method. It is interesting to note that all methods except RND achieved a 100% success rate for 7 out of 18 objects.

These results suggest that the advantage of the stability metric is not apparent when there is no robot arm involved and even the SP method is sufficient for a reasonable performance.

B. With Robot

The aggregate results on all the objects is shown in Table II. Compared to the previous experiment without the robot arm, all methods performed worse in all metrics, due to the inherent difficulty of the scenario of the robot hand being in the images and the contact physics between the end-effector and the object. The success rates were 72.3%, 84.6% and 86.1% for SP, ITR and ITR-S approaches respectively, and the full approach ITR-S performed the best in all metrics.

By introducing the robotic arm, network errors are more pronounced, which allows iterative approaches to compensate for the errors of SP. This is most noticeable for objects which SP struggles with such as the Toothpaste and various cup models as seen in Figure 7. This is also apparent during training when the PR-CNN training error is still high, as seen in Figure 4. The only exception to this is before the network is trained, at which point the network is outputting random rotations. This validates our intuition that an iterative approach is able to compensate for errors in the network as long as the network is trained to an extent such that it is able to move in the general direction towards the ground truth optimum.

C. Failure Modes

A common failure mode of our approach is sometimes placing the objects upside-down, which occurs with objects that look symmetrical, such as the cups, ShowerGel and ToothPaste object models. This occurs most often when the object is in an orientation such that distinguishing features that could typically be used to determine an upright orientation, such as the handle or mouth of a cup, are not visible to the cameras. These objects also result in oscillatory behaviour when using the ITR, as the model oscillates between two perceived stable orientations. When this occurs, the angular error of object orientation can be very high depending on which iteration the algorithm is stopped. This

![Fig. 5: Angular error between object orientation and the ground truth human-preferred orientation using ITR over 15 iterations. Most objects yield convergent behaviour such as the HamburgerSauce object (solid line), however, some objects like Glassbowl yield oscillatory behaviour (dashed line), causing the object to rotate between two perceived stable orientations.](image)

![Fig. 4: Success rate for SP and ITR methods while PR-CNN was training on data without a robotic arm. ITR consistently outperformed SP during training, and converged to a steady state much faster.](image)

![Table II: Placement results with the robot in simulation tested on all objects.](image)
Success rate
RND SP ITR ITR-S

Fig. 6: Placement success rates categorised by object models without the robot. Each object was evaluated 50 times. Success rates are very high across the board, with little difference between SP, ITR and ITR-S approaches, which is expected as there is little error that the iterative approaches need to compensate for.

Fig. 7: Placement success rates categorised by object models evaluated with the robot. Each object was evaluated 50 times. Both ITR and ITR-S consistently outperform SP. The largest performance increases between ITR and ITR-S occur for objects which SP and ITR tend to have difficulty with, such as the toothpaste and red cup object models.

is most noticeable for the Glassbowl object, which exhibited the most oscillatory behavior out of our object set. In Fig.5 the angular error is shown for each iteration for two objects. While HamburgerSauce object shows the expected behavior, the behavior is oscillatory for Glassbowl. An advantage of ITR-S is to solve the oscillation problem by testing for stability at each iteration and outputting a highly stable orientation among all iterations.

Another common failure mode occurs with intrinsically unstable objects, such as objects with a small base area relative to their total volume. Objects such as the Toothpaste and Pitcher are examples of objects exhibiting this trait. Although the angular error of the network to the ground truth pose is similar to other more intrinsically stable objects, the success rate of placements is noticeably lower. Due to the different intrinsic stability properties of different objects, it is difficult to benchmark our results against other works.

VII. ROBOT IMPLEMENTATION

To demonstrate the feasibility of our proposed approach and the potential for sim-to-real transfer, we implemented our approach on a Franka Panda robotic arm. The system consists of three computers connected via TCP/IP where each computer is connected to a Realsense D435 RGB-D camera. The reason for using multiple computers was that the high USB bandwidth required for realsense cameras. Each computer uses the Melodic version of the Robot Operating System (ROS) [29] running on Ubuntu 18.04 LTS operating system. We use MoveIt [30] motion planning framework for planning and control.

The physical setup is similar to simulation, with the three RGB-D cameras placed orthogonally from each other. We moved the location of the shelf to the right of the arm due to physical constraints. The depth images are inpainted using OpenCV [31] to assist in noise removal from the depth
images. The output of the network is then processed by PR-CNN, which is trained only in simulation, to calculate the new rotation. The object is then lowered onto the shelf using force-feedback to detect when the object is in collision with the shelf. We 3D-printed 6 of the 18 objects in our dataset and the robot was able to successfully place the objects in certain initial object orientations. At the time of writing, the system was not robust enough for a full evaluation. This is due to the limited considerations during motion planning and noisy depth images, which were not considered during the training of PR-CNN.

VIII. Conclusion and Future Work

In this work, we proposed an approach to rotate grasped objects into orientations such that they can be placed in stable, human-preferred orientations. We show the feasibility of learning to place objects from depth images without object detection or explicit pose estimation. Our experimental results suggest that iterative approaches such as ITR and ITR-8 show improvements over the single pass approach. Our work also shows potential for sim-to-real transfer learning, and justifies the need for more research to make the system more robust and generalize the approach to different classes and shapes of objects.

There are many interesting future research directions for this work. First, our current iterative approaches only reevaluates the object’s orientation after it has completed the rotation, making it slow to react to disturbances. A closed-loop reactive approach, analogous to grasping in [5], that is able to reevaluate the rotation at every time step will be much more efficient at dealing with these problems, but still potentially yield the same benefit as our iterative approaches. Second, our approach is designed for objects with a single defined preferred orientation, however some objects such as cans and boxes have multiple preferred orientations. It is possible to extend our representation to include such objects, which may also help to reduce the oscillatory behaviour of iterative approaches. Third, we have observed that in many situations, it is impossible to place to object in the preferred orientation, either the arm would make contact with the surface or that the desired pose is kinematically infeasible. It would be interesting to couple our work with grasping - such that robot picks up the object in an orientation that enables placement in a desired orientation.

References

[1] J. Bohg, A. Morales, T. Asfour, and D. Kragic, “Data-driven grasp synthesis survey,” IEEE Transactions on Robotics, 2014.
[2] I. Lenz, H. Lee, and A. Saxena, “Deep learning for detecting robotic grasps,” The International Journal of Robotics Research, 2015.
[3] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics,” 2017.
[4] L. Pinto and A. Gupta, “Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours,” in IEEE International Conference on Robotics and Automation (ICRA), 2016.
[5] D. Morrison, P. Corke, and J. Leitner, “Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach,” arXiv preprint arXiv:1804.05172, 2018.
[6] C. R. Garrett, T. Lozano-Prez, and L. P. Kaelbling, “Frob: Leveraging symbolic planning for efficient task and motion planning,” The International Journal of Robotics Research, 2018.
[7] A. Kasper, Z. Xue, and R. Dillmann, “The kit object models database: An object model database for object recognition, localization and manipulation in service robotics,” The International Journal of Robotics Research, 2012.
[8] A. Edsinger and C. C. Kemp, “Manipulation in human environments,” in IEEE-RAS International Conference on humanoid Robots, 2006.
[9] J. Baumgartl, P. Kaminsky, and D. Henrich, “A geometrical placement planner for unknown sensor-modelled objects and placement areas,” in IEEE International Conference on Robotics and Biomimetics (ROBIO), 2013.
[10] J. Baumgartl, T. Werner, P. Kaminsky, and D. Henrich, “A fast, gpu-based geometrical placement planner for unknown sensor-modelled objects and placement areas,” in 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014.
[11] K. Harada, T. Tsuji, K. Nagata, N. Yamanobe, H. Onda, T. Yoshimi, and Y. Kawai, “Object placement planner for robotic pick and place tasks,” in IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012.
[12] K. Harada, T. Tsuji, K. Nagata, N. Yamanobe, and H. Onda, “Validating an object placement planner for robotic pick-and-place tasks,” Robotics and Autonomous Systems, 2014.
[13] J. A. Haustein, K. Hang, J. Stork, and D. Kragic, “Object placement planning and optimization for robot manipulators,” arXiv preprint arXiv:1907.02555, 2019.
[14] Y. Jiang, C. Zheng, M. Lim, and A. Saxena, “Learning to place new objects,” Proceedings - IEEE International Conference on Robotics and Automation, 2011.
[15] Y. Jiang, M. Lim, C. Zheng, and A. Saxena, “Learning to place new objects in a scene,” The International Journal of Robotics Research, 2012.
[16] R. Paolini, A. Rodriguez, S. S. Srinivasa, and M. T. Mason, “A data-driven statistical framework for post-grasp manipulation,” The International Journal of Robotics Research, 2014.
[17] H. Fu, D. Cohen-Or, G. Dror, and A. Sheffer, “Upright orientation of man-made objects,” in ACM SIGGRAPH, 2008.
[18] R. Alami, T. Simeon, and J.-P. Laumond, “A geometrical approach to planning manipulation tasks. the case of discrete placements and grasps,” 1990.
[19] J. Scholz and M. Stilman, “Combining motion planning and optimization for flexible robot manipulation,” in IEEE-RAS International Conference on Humanoid Robots, 2010.
[20] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman, “Push planning for object placement on cluttered table surfaces,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011.
[21] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, “On the continuity of rotation representations in neural networks,” in CVPR, 2019.
[22] Z.-B. Xu and F.-L. Cao, “Simultaneous lp-approximation order for neural networks,” Neural Networks, 2005.
[23] Z. Xu and F. Cao, “The essential order of approximation for neural networks,” Science in China Series F: Information Sciences, 2004.
[24] Z. Chen and F. Cao, “The construction and approximation of neural networks operators with gaussian activation function,” in Mathematical Communications, 2013.
[25] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[26] J. Deng, W. Dong, R. Socher, L. jia Li, K. Li, and L. Fei-fei, “Imagenet: A large-scale hierarchical image database,” in In CVPR, 2009.
[27] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014.
[28] S. James, M. Freese, and A. J. Davison, “Pyrep: Bringing v-rep to deep robot learning,” arXiv preprint arXiv:1906.11776, 2019.
[29] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in ICRA workshop on open source software, 2009.
[30] I. Sucan and S. Chitta, “Moveit motion planning framework,” 2019. [Online]. Available: [http://moveit.ros.org]
[31] G. Bradski, “The OpenCV Library,” Dr. Dobb’s Journal of Software Tools, 2000.