IFOR: Iterative Flow Minimization for Robotic Object Rearrangement

Ankit Goyal[1,2]*, Arsalan Mousavian[1], Chris Paxton[1], Yu-Wei Chao[1], Brian Okorn[1,3]*
Jia Deng[2], Dieter Fox[1]
[1]NVIDIA, [2]Princeton University, [3]Carnegie Mellon University

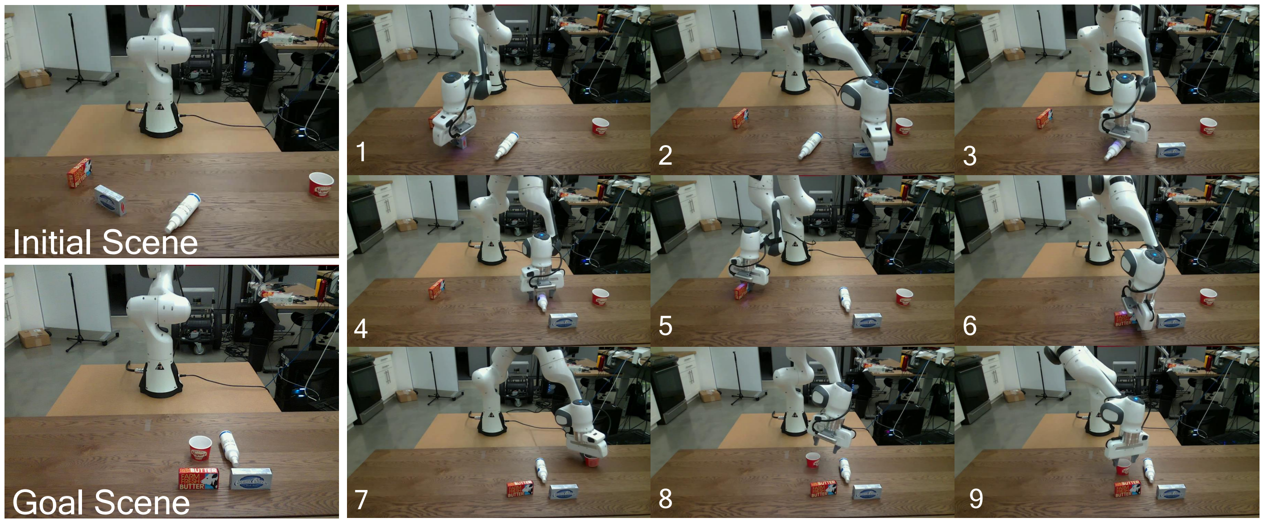

Figure 1. An example of *IFOR* being applied to real data. The initial and goal scenes are shown on the left. Our approach allows the robot to repeatedly identify transformations that will minimize the flow for various objects between the current and goal scenes. It can then repeatedly grasp, move, and place objects, rotating as necessary, in order to achieve the configuration in the goal scene. The system is trained completely on synthetic data and transfers to the real world in zero-shot manner.

Abstract

Accurate object rearrangement from vision is a crucial problem for a wide variety of real-world robotics applications in unstructured environments. We propose IFOR, *Iterative Flow Minimization for Robotic Object Rearrangement, an end-to-end method for the challenging problem of object rearrangement for unknown objects given an RGBD image of the original and final scenes. First, we learn an optical flow model based on RAFT to estimate the relative transformation of the objects purely from synthetic data. This flow is then used in an iterative minimization algorithm to achieve accurate positioning of previously unseen objects. Crucially, we show that our method applies to cluttered scenes, and in the real world, while training only on synthetic data. Videos are available at* https://imankgoyal.github.io/ifor.html.

1. Introduction

Object rearrangement is the capability of an embodied agent to physically re-configure the objects in a scene into a desired goal configuration [2]. It is an essential skill in day-to-day activities like setting a dining table, putting away groceries, and organizing a desk. Endowing robots with this capability is crucial for deploying them to assist people with everyday tasks [2].

*Work done while authors were interns at NVIDIA

With varying task setups, the desired goal state can be provided in different forms, for instance, a compact state representation [32, 67] or natural language descriptions [36, 55]. In this work, we address the rearrangement task where the goal state is specified by an RGB-D image [34, 51], as shown in Fig. 1. This setup lends itself well to many scenarios where the goal state can be snapped once, either in the first place or from a one-time demonstration. For instance, a user can set the dining table once to their preference and take a photo, and a robot assistant can restore the table back to the desired state from any configuration.

Traditionally, object rearrangement problems have been studied in the robotics community, often in the context of Task and Motion Planning (TAMP) [15]. Despite much recent progress [10, 16, 26], most TAMP approaches still rely on a strict set of assumptions on the perception front. First, the objects and scenes are often assumed to be known a priori, provided with high fidelity 3D models. This makes the approaches difficult to be deployed in unseen environments or environments without models. Second, given the models, the planning front often assumes accurate pose information at the input. This makes explicit object pose estimation [33, 35, 50, 61, 66] a necessary part of the pipeline, and the full system susceptible to pose estimation error from real vision systems.

Recent efforts in robotics have attempted to relax these constraints by leveraging the power of deep learning. A recent approach called NeRP, proposed by Qureshi et al. [51], has allowed for rearranging objects unseen at the training time, by representing the observed objects with learned embeddings. It also removes the need of explicit object pose estimation for planning by leveraging recent progress on learning-based grasp planners [62] and collision detectors [7]. However, NeRP only allows moving objects with 2D in-plane translations on the table surface and allows no change in their orientation. This prevents its applications in realistic scenarios that require moving objects with more complex transformations such as those shown in Fig. 1.

We propose a new approach to image-guided robotic object rearrangement with RGB-D input. It achieves, for the first time to the best our of knowledge, the ability to handle unknown objects with translation as well as planar rotations.

The key to our method is re-formulating object rearrangement as an iterative minimization of optical flow between the current observed image and the goal image. By using optical flow as an intermediate representation, we can capitalize on the cutting edge development in flow estimation models [63]. Although these flow models were originally developed for consecutive video frames with small pixel displacements, we show that with proper training, these models can excel at estimating flow with large displacements from arbitrary transformation of objects. Using this estimated flow, together with the depth input and generic object segmentation models [69], we can obtain dense 3D correspondences for each object. This provides a general representation that allows us to solve for the desired transformation of objects with simple optimization. Furthermore, with such a general representation, our method trained entirely on synthetic data transfers well to the real world in a zero shot manner.

To summarize, we introduce *IFOR*, **I**terative **F**low Minimization for Unseen **O**bject **R**earrangement. *IFOR* is to our knowledge the first system capable of rearranging unseen objects, given an RGB-D image goal, that handles both translation and rotation. Our approach is trained solely on simulation data and transfers to the real world in a zero-shot manner. Finally, we perform a set of experiments showing our method allows rearrangement of novel objects in cluttered scenes with a real robot.

2. Related Work

Robotic Object Manipulation. Our work falls in the broad area of robotic manipulation. Conventional manipulation systems often take a modular approach, decomposing the full system into perception, planning, and actuation components. The perception module is charged with estimating the state of the environment, e.g., detecting and segmenting objects [4, 6, 22, 38, 70] and estimating their 6D poses [33, 35, 50, 61, 66]. With the perception output, the planning module then searches for a sequence of actions to accomplish the manipulation task. In robotics, this is conventionally formalized and studied in the problem of Task and Motion Planning (TAMP) [10, 15, 16, 26]. However, setting up a real-world TAMP system often requires substantial task-specific knowledge and accurate 3D models of the environment, significantly limiting the environments to which the system can generalize. To address this challenge, recent work has adopted deep learning-based approaches for robotic manipulation, for instance, on grasp planning [44, 47, 48, 62, 65], motion planning [7, 57], and reasoning about spatial relations [20, 36, 49].

Our work is concerned with rearranging objects, an area that has a long history in robotics [31, 32, 34, 51, 54] but has recently gained traction in the vision and learning communities [2, 19, 39, 67] thanks to the advances in simulation platforms. The works most relevant to ours are that of Labbé et al. [34] and NeRP [51], which also address the rearrangement task with the goal state specified by an image. A key step to address this problem is to establish the correspondence of objects between the current and goal image and solve for their desired transformations. However, both [34] and [51] only consider 2D planar translation with no orientation change. This is arguably because their feature descriptors for objects are generic and not rotation sensitive. In contrast, we propose to use optical flow as the low-level feature descriptors, which can be naturally used to infer the full 6D transformations. In parallel to our work, recent efforts have also addressed rearrangement particularly learned from human demonstrations [14, 72] and also with different goal specifications such as language [36, 55].

Optical Flow and Feature Correspondence. Optical flow is a long-standing vision problem that addresses the estimation of pixel motions between two video frames. Alike other sub-fields in computer vision, the take-off of deep learning has replaced the traditional pipelines for optical flow with learning-based end-to-end architectures [1, 23, 27, 30, 41, 58–60, 63, 73]. In tackling object rearrangement, our work proposes to use the predicted optical flow between the current and goal image to establish the desired object transformations. However, rather than estimating flow from consecutive video frames, we demonstrate that state-of-the-art models can excel at predicting flow with large displace-

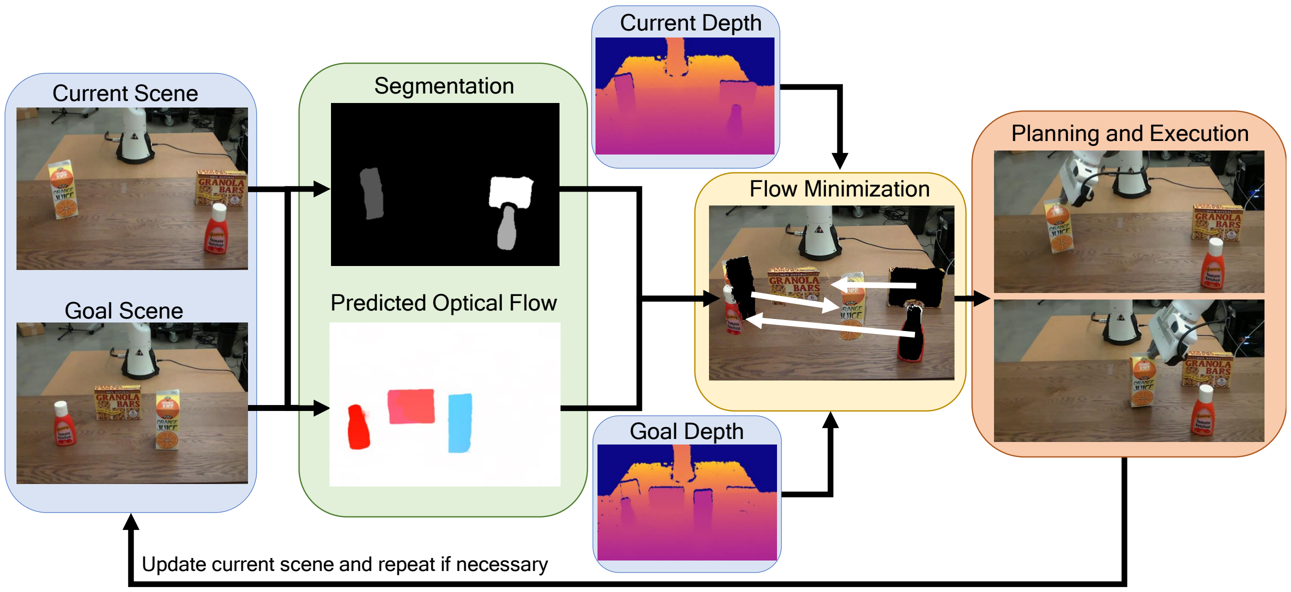

Figure 2. Overview of the *IFOR* algorithm. *IFOR* takes as input RGB+D images of a current and a goal scene, and uses these to make predicts as to which objects should move and by which transformations, using RAFT to estimate optical flow. This is then sent to a robot planning and execution pipeline which is capable of grasping of unknown objects and motion planning in scenes with unknown geometry.

ments from arbitrary goal images in object rearrangement.

In addition to optical flow, our work can potentially also leverage the body of work on scene flow, which directly predicts the motion in 3D rather than in the 2D image space. Recent work has studied various setups for scene flow including predicting from monocular frames [24], stereo images [3, 29, 43], RGB-D pairs [25, 42, 64], to 3D point clouds [17, 21, 37, 45]. Finally, our flow prediction task is closely related to the problem of establishing feature correspondences [11, 52, 71, 74] in 3D reconstruction (e.g., SfM) and visual localization (e.g., SLAM).

Embodied AI. Our work fits well with recent trends in embodied AI. Initially, this work largely centered around the family of tasks pertained to navigation [8, 9, 53, 56, 68], but has gradually grown to encompass tasks pertained to physical manipulation [12, 28, 69]. Moreover, a recent notable article by Batra et al. [2] has recognized the rearrangement problem as a "canonical task" for evaluating embodied AI. Our work drives progress in this exact frontier. And unlike most prior embodied AI works, which run evaluation only in simulation, we also evaluate the performance of our method on a real-world robotic platform.

3. Method

IFOR takes as input the RGB-D images of the current and the goal scene, and iteratively generates a pick-and-place action for one object at a time. At each iteration, the RGB-D image of current and goal scene is passed through two components: (1) perception and (2) planning (Fig. 2).

The perception component is responsible for estimating the relative transformation of all objects between the current and goal scene. Given the estimated transforms, the planning component selects an object to be moved along with the required transformation, by taking into account collision and kinematic feasibility. Finally, after executing the planned pick-and-place action, the system will take a new observation of the scene and repeat the process.

3.1. Perception

The task of the perception component is to segment out objects in the current and goal images, establish their correspondences, and predict a 6-DoF transformation for each object from its current pose to its goal pose. We achieve this with a pipeline that combines optical flow estimation, unseen object segmentation, and a RANSAC-based transformation optimization.

Optical Flow For Rearrangement. The first step is to estimate the optical flow between the current and the goal image. This provides us with a pixel-level correspondence between them. In conventional settings, optical flow is generally estimated between temporally close images in a video. The displacement of the flow is often small as temporally close images do not differ much from one another. In fact, this small displacement prior is crucial for classical methods like Lucas-Kanade method [40]. This assumption however does not hold in rearrangement, as the objects could be moved a large distance and rotated by large angles from the initial scenes. This makes classical approaches of optical

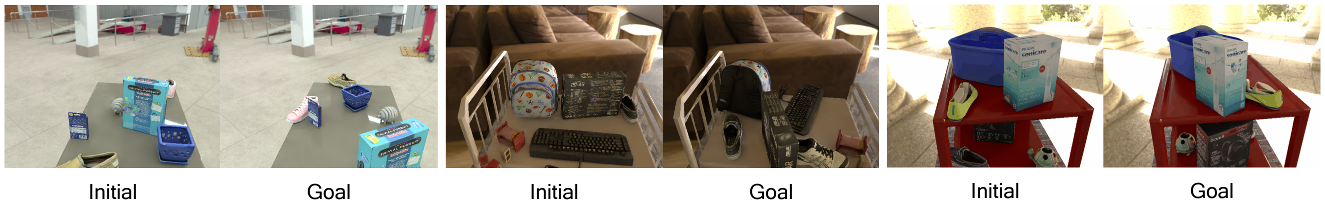

Figure 3. Examples from the synthetic dataset used for training RAFT. Images come in pairs of initial and final images, each containing assorted objects in clutter. Between the two images, there is a large, randomly-sampled transformations. Retraining RAFT on these large discontinuities is essential to our approach.

flow estimation ill-suited for rearrangement.

Contrary to classical methods, deep learning based optical flow methods like Recurrent All Pairs Field Transforms (RAFT) [63] learn to make predictions from data. But, they too were developed for and trained on estimating flow in videos and hence pre-trained RAFT model does not work well on rearrangement scenes. However, RAFT's underlying architecture does not make any assumption about the flow displacement being small, as it compares every pixel in one image to every other pixel in the other image. Consequently, given suitable data, RAFT can be trained for object rearrangement with large translations and rotations.

RAFT. Recurrent All Pairs Field Transforms (RAFT) estimates optical flow by constructing a 4D correlation volume of which each pixel in one image is compared to every pixel in the other image [63]. It then updates the flow estimates using a recurrent unit, starting with zero optical flow at all locations. In each iteration, the recurrent unit does a lookup around the current estimate of flow to decide how to update the flow estimates.

During training, RAFT applies supervision over all these intermediate flow estimates made by the recurrent unit. Suppose, $\{\mathbf{f}_1, ..., \mathbf{f}_N\}$ are the N intermediate flow estimates and $\mathbf{f}_{gt}$ is the ground-truth flow, then the loss ($\mathcal{L}$) is defined as the weighted sum of l_1 distance between estimated and ground-truth flow. Specifically,

$$\mathcal{L} = \sum_{i=1}^{N} \gamma^{N-i} ||\mathbf{f}_{gt} - \mathbf{f}_i||_1, \tag{1}$$

where γ is a discount factor of value 0.8. For more details, please refer to the work by Teed an Deng [63].

Synthetic Data. To train RAFT for object rearrangement, we create a visually realistic dataset of synthetic scenes. In these scenes, objects are placed on supports like tables or beds. We sample the supports from ShapeNet [5] and the objects from the Google Scanned Dataset [18]. We use the NViSII renderer [46] to render scenes with realistic lighting via ray tracing, and diverse view points by randomizing camera poses. We also randomize the lighting of the scenes, the texture of the supports and the background image. In the end, we created a dataset of $\sim$54K training samples and 1000 test samples. Some samples of our synthetic data are shown in Fig. 3.

Unseen Object Segmentation. Optical flow alone is not sufficient to estimate the transformation of the object as they lack the grouping or "objectness" information. Moreover, unlike most prior segmentation methods that only segment object classes they were trained on, we need a zero-shot segmentation method to deal with unseen objects at test time. We use pre-trained UCN [70], which is trained to segment unknown objects from RGB-D image of the scene. It learns per-pixel embedding such that pixels belonging to the same object instance have similar embeddings but different from other object instances in the scene. At the test time, objects are segmented using mean-shift clustering.

Relative Object Transform from Flow. The next step is to estimate the relative transformation of the objects between the two frames. The relative transformation of each object is computed by first unprojecting each pixel to 3D from the depth map of the scene and camera intrinsics. The 3D correspondence between current image and goal image is computed from predicted flow and unprojected points.

We solve for a rigid body transformation for each object by minimizing the error in position after applying the transformation. Let $P_c \in \mathbb{R}^{3\times n}$ be the 3D points of an object in the current scene and $P_g \in \mathbb{R}^{3\times n}$ be the corresponding points in the goal scene. We then estimate a rigid-body rotation R and translation T by solving the following optimization problem using SVD decomposition:

$$\arg\min_{R,T} ||(R \cdot P_c + T) - P_g||_2. \tag{2}$$

In practice, we observed that the matched correspondences from flow can contain many outliers. This is especially prominent when the object undergoes extreme transformations (e.g., large rotations), resulting in only a small portion of the object surface that is visible in both images. To handle the outliers reliably, we adopt RANSAC [13] in solving the relative pose. In Table 1, we show that RANSAC is effective in removing outliers and estimating accurate transformations. In Figure 4, we show qualitative

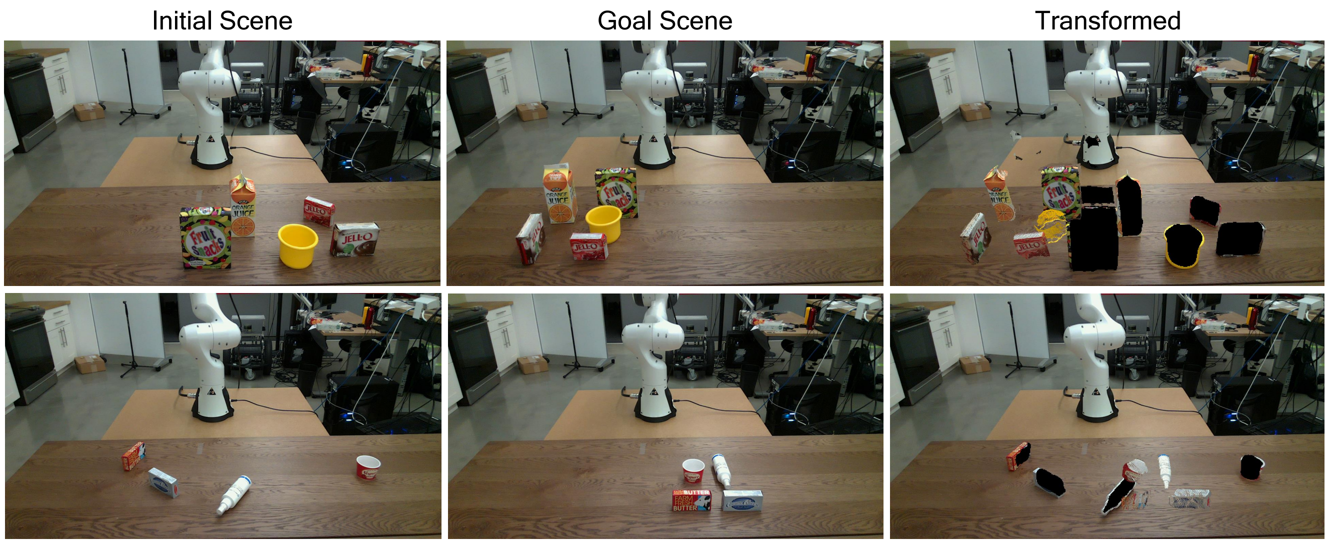

Figure 4. Examples of different real-world scenes warped after object points have been transformed according to poses estimated by *IFOR*. Poses are estimated according to predicted optical flow features.

Algorithm 1 Pseudo Code for Planning Module in IFOR

```
1:  Input: Objects $[O_1, \dots, O_n]$, Estimated transformation $[T_1, \dots, T_n]$
2:  for $i = 1$ to $n$ do
3:      Calculate score $S_i$ for object $O_i$
4:  Sort objects $[O_1, \dots, O_n]$ and transformations $[T_1, \dots, T_n]$ based on score $[S_1, \dots, S_n]$
5:  Let sorted object be $[\bar{O}_1, \dots, \bar{O}_n]$ and sorted transformations be $[\bar{T}_1, \dots, \bar{T}_n]$
6:  ObjectMoved = False
7:  for $i = 1$ to $n$ do
8:      Collision = if $\bar{O}_i$ collides when applied transformation $\bar{T}_i$
9:      if Collision is False then
10:         Move object $\bar{O}_i$ with relative transform of $\bar{T}_i$
11:         ObjectMoved = True
12:         break loop
13: if ObjectMoved is False then
14:     for $i = 1$ to $n$ do
15:         FreeSpaceFound = Find free space $\bar{F}_i$ to place $\bar{O}_i$
16:         if FreeSpaceFound is True then
17:             Move $\bar{O}_i$ to $\bar{F}_i$
18:             break loop
```

examples of the transformations estimated by RANSAC using our trained RAFT model.

3.2. Planning and Execution

Given the list of desired object transformations, the planner produces a pick-and-place action to execute, while taking into account various kinematic and geometric constraints. Our planning algorithm iterates over the list of desired object transformation from perception module, and finds out which objects can be moved directly using the predicted transform. The planner classifies each relative transform as feasible if the object at the predicted transform is not colliding with any other objects in the scene. We used pre-trained SceneCollisionNet [7] to check the collision of the object at the predicted transformation. The feasible objects are then ranked based on score $S = |r| + \lambda|t|$, where r is the relative rotation transformation in radians, t is the relative translation in cm and $\lambda = 0.2$ in our experiments. The planning algorithm greedily picks objects with larger relative transformations.

If the policy is not able to find a feasible movement for any object, it will try to move one of the objects to a random collision-free location. Provided that the estimated transformation of the objects are correct, the proposed planning and execution policy is guaranteed to converge and successfully rearrange the objects, i.e., in the worse case, it will move all but one object to collision-free locations, followed by moving each of them to the goal location one by one.

The system terminates when the estimated change in rotation and translation for all objects is smaller than than a fixed threshold. Specifically, we use a threshold of $10°$ for rotation and $5\ cm$ for translation. Our experiments show that this heuristic is quite effective in handling challenging object rearrangement scenarios in the real world (discussed in Sec. 4.1). The pseudo-code for the planning algorithm is outlined in Algo. 1.

4. Experiments

We present results in two settings. First, we perform an integrated system comparison in the real world, with a phys-

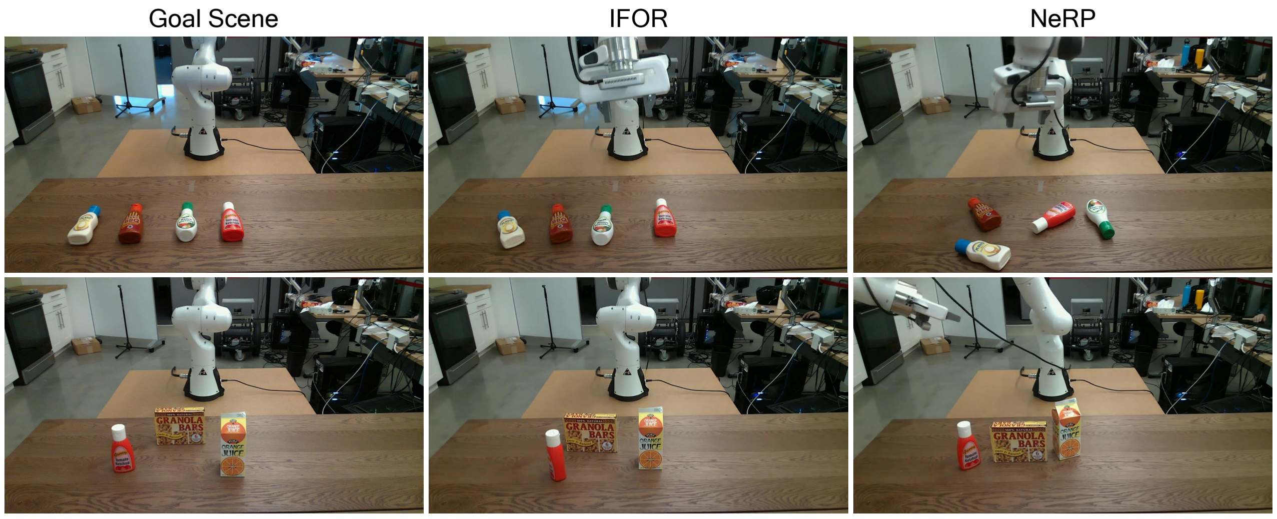

Figure 5. Two examples comparing the qualitative performance of NeRP [51] to *IFOR*. In both of these examples, *IFOR* both more closely matches the goal image, and matches the orientation much more precisely than NeRP does.

Method	Rot. $\in [-60°, 60°]$ Median Rot. Err. (in °)	Rot. $\in [-60°, 60°]$ Median Pos. Err. (in cm)	Rot. $\in [-180°, 180°]$ Median Rot. Err. (in °)	Rot. $\in [-180°, 180°]$ Median Pos. Err. (in cm)
Learning Baseline	22.8	8.1	33.8	8.2
Pretrained RAFT + RANSAC	20.6	46.5	67.8	43.1
Rearrangement RAFT + RANSAC	**3.6**	**1.2**	**13.7**	**2.7**

Table 1. Performance of various approaches in estimating pose transformation from flow in one-step on simulated data. We find that RANSAC outperforms the learning based method for finding the relative transform. In addition, pre-trained RAFT model performs poorly on the rearrangement scenes where the object displacements are larger than video sequences.

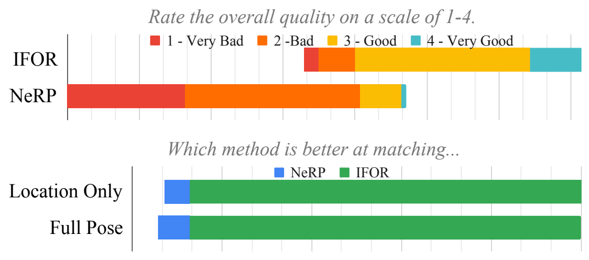

Figure 6. User scores for *IFOR* vs. NeRP [51]. When asked to rate performance of the two methods on a scale of 1-4, users preferred *IFOR* by a wide margin. Users chose *IFOR* over NeRP in almost all situations, when looking at either position only (94%) or full pose (position and orientation, 92%).

ical robot picking and placing objects using either *IFOR* or the previous state of the art [51]. Second, we perform large scale ablation studies in simulation to evaluate the effect of each design choice in *IFOR*.

4.1. Real World Experiments

We used a Franka Panda robot to conduct the real world experiments. The world is perceived from an external RealSense L515 camera, and a wrist-mounted RealSense D415 camera. The external camera is used for planning with *IFOR*, collision avoidance, and controlling the robot. The wrist mounted camera is only used for grasping to improve the robustness of the system.

IFOR generates a list of objects and transforms representing the placement location. These poses are passed to the pick-and-place system which takes the ordered list of actions from the planner. It selects the first object that can be grasped and places it at the desired location. Grasps for each object are computed by Contact-GraspNet [62] and the robot motions are generated using a model predictive control pipeline and SceneCollisionNet. Refer to [7] for details of the pick and place system. None of the components are trained on any real objects nor the objects we tested on.

We evaluated on 6 scenes, where each scene has between 2 to 5 objects in the initial configuration and a distinct goal configuration. Although it is not possible to replicate the exact initial conditions in the real world, we tried to duplicate the initial configuration visually, and used the same RGB-D goal image for testing both methods. In order to quantitatively evaluate the performance of the methods in

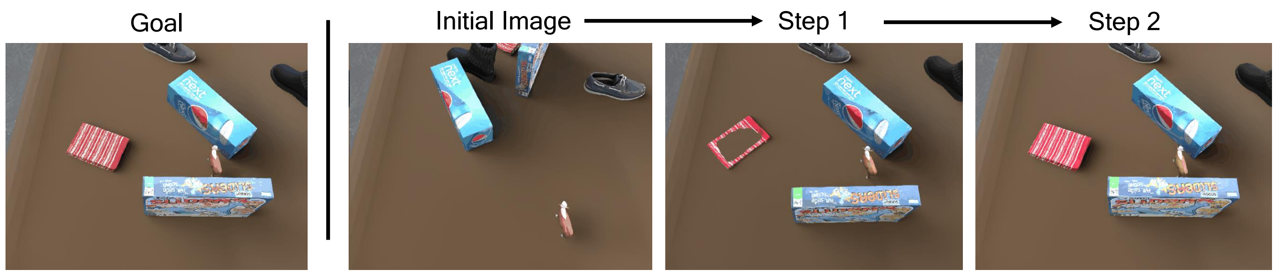

Figure 7. An example showing the sequence of transformations executed by our model on the synthetic data. Objects are teleported into the next position as predicted by IFOR without collision checking. This figure only shows the first two steps, but minor refinements are still possible.

the real world, we conducted a user study with 10 users, where we asked users to select the method that performed better: *IFOR* or NeRP [51]. We also asked users to rate *IFOR* and NeRP on a scale of 1-4, where 1 is "very bad" and 4 is "very good".

All the components of the pick-and-place system were the same for both *IFOR* and NeRP, except the estimation of the objects' final pose. Fig. 5 shows a qualitative comparison of *IFOR* and NeRP given similar target images. The video of the experiments are included in the Supplementary Materials. Since NeRP does not handle changing the orientation, we asked users to rank the two methods only based on translation as well as considering both rotation and translation. Fig. 6 shows that users significantly prefer *IFOR* over NeRP both on the translation only setting and full pose setting. NeRP failures were often due to an incorrect correspondence between the current image and target image. In addition, the learned placement generator does not generalize well to arbitrary rearrangements. *IFOR*, on the other hand, can find corresponding objects and their relative pose transform reliably from the predicted optical flow.

4.2. Ablation Studies

We conducted our ablation studies on a synthetic dataset which consisted of 200 tabletop scenes with randomized lighting, texture and background images. Each scene contained between 1 and 9 objects. Given a random target scene, the current scene was generated by applying a random planar rotation and translation to each object. We first define the evaluation metrics below and then provide the analysis on different ablation studies.

Metrics. We report the median translation and rotation error averaged over all the objects. We also report the percentage of objects that are within a threshold of rotation and position error for different thresholds. In addition, for Table 2, we divide the scenes by three task difficulty levels: "easy" for position error less than 2 cm and rotation error less than 5°; "medium" for position error less than 5 cm and rotation error less than 10°; and "hard" for position error less than 10 cm and rotation error less than 15°.

Learning-based versus RANSAC for relative transform prediction. We explored learning based solution to predict transformation from flow instead of optimizing it with RANSAC. For estimating location, we predict the heat-map of the centroid of the object in the goal scene given the flow and depth image of the scene. For estimating rotation, we use a cropped image of the flow around the object and regress the change in rotation. Table 1 compares the accuracy of the learning-based baseline with our RANSAC-based optimization. Our RANSAC-based optimization with re-trained RAFT (rearrangement RAFT) achieves significantly lower errors.

Pre-trained RAFT versus Rearrangement RAFT. We compared the accuracy of single step transformation prediction between pre-trained RAFT which is trained on video sequences vs rearrangement RAFT where it is trained on our synthetic data with large translation and orientation changes. Table 1 shows that training RAFT on rearrangment scenes is cruicial for achieving high accuracy for predicting relative transforms of the objects.

Teleporting objects versus planning one action at the time. In order to provide an approximate upper-bound performance of *IFOR*, we implemented a baseline where the policy computes transformations for all the objects and move all of them at once and observe the scene again. Fig. 7 shows the execution of teleport policy in simulation. This policy is clearly not practical in any real robotic setting but removes planning constraints and achieves significantly better results as shown in first row of Table 2. In addition, we can see the the heuristic planning does not lose the performance compared to the teleport policy which shows its effectiveness.

Ground-truth Collision Checker versus Learned Collision Checker [7]. An important component of *IFOR* is the collision checker which indicates whether a predicted rotation and translation should be accepted by the policy or not. Table 2 shows that the placement accuracy drops due to imperfections of the learned collision checker. *IFOR* ben-

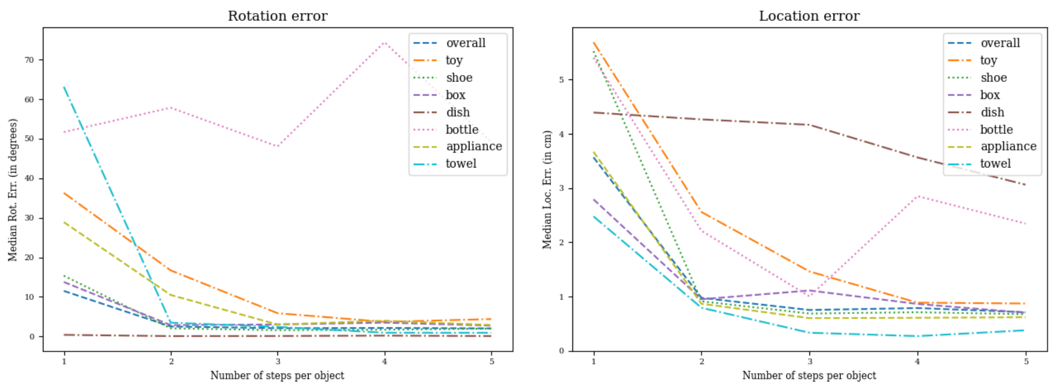

Figure 8. Per-class results over time, showing both position and rotation error. Objects from the Google Scanned Objects dataset [18] were divided into different groupings containing at least 10 distinct models. This class breakdown shows that the system has the most trouble with object classes that have a lot of physical symmetry; the "bottle" category includes objects where the only orientation distinction is often a label.

GT Coll. Check	Learned Coll. Check	Planar Rot.	Planning	Median $\|\Delta\theta\|$ (in °)	Median $\|\Delta t\|$ (in cm)	$\|\Delta t\| < 2cm$ $\|\Delta\theta\| < 5°$	$\|\Delta t\| < 5cm$ $\|\Delta\theta\| < 10°$	$\|\Delta t\| < 10cm$ $\|\Delta\theta\| < 15°$
		✓		1.3	0.58	72.8%	78.4%	79.5%
✓		✓	✓	1.4	0.64	70.0%	81.8%	84.0%
✓			✓	3.5	1.05	59.7%	74.2%	76.9%
	✓	✓	✓	1.6	1.09	49.2%	64.7%	68.1%
	✓		✓	4.5	2.04	37.0%	53.5%	57.9%

Table 2. Ablation results for IFOR in simulation. First row corresponds to the teleportation baseline discussed in Sec.4.2, and serves as an no-planning upper-bound. Our planning policy achieves close to this no-planning upper-bound. Further, we show that IFOR benefits from better collision detection and assumptions about planar rotations.

efits from improvements in collision checking for arbitrary scenes.

Planar Rotation versus SO(3) Rotation. Given the correspondences in 3D, we optimize for relative $SE(3)$ transform using RANSAC. However, we can use the knowledge that objects rotations are planar and only keep the rotation component around z-axis. Table 2 shows that adding planar rotation assumption improves the accuracy significantly.

Object-wise Performance Analysis. In order to get more insight into the performance of the system, we divided the objects of [18] into 7 classes that are shown in Fig. 8. The general trend among the categories is that the accuracy improves with the increase in the number of steps which stems from having more overlapping views of the object. The category that *IFOR* struggles with most is *bottle*. Bottles are challenging because the method needs to match not only the geometry of object but also needs to match the texture details on the surface of the bottles, which can be quite challenging for smaller objects where there is not enough resolution on those objects. We can see a similar issue with the *dish* category, which includes pots, pans, and mugs. Unlike with bottles, these objects tend to have little to no texture and are mostly symmetric.

5. Discussion

We proposed *IFOR*, an iterative optical flow based system for object rearrangement. To the best of our knowledge, *IFOR* is the first method that can solve the image-based object rearrangement task, with both translation and rotation, for unseen objects. We conducted experiments with a robot in the real world to show its effectiveness and generalization to real perception data. Our results show a significant performance boost over previous work. In addition, we conducted a comprehensive analysis on simulation data to evaluate the effect of each component of our approach.

There are multiple promising future directions for this work. The first is to extend *IFOR* to $SE(3)$ poses. Even though there is no inherent assumption about planar rotation in our method, we only evaluated the object rearrangement problem with planar rotations because the models are only trained on data with planar rotations.

Another interesting direction would be to monitor the scene during robot execution to account for any in-gripper object motions and update the placement pose accordingly.

References

[1] Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. Learning optical flow from still images. In *CVPR*, 2021. 2

[2] Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A challenge for embodied AI. *arXiv preprint arXiv:2011.01975*, 2020. 1, 2, 3

[3] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios? In *ICCV*, 2017. 3

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[5] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Su, Li Su, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. 4

[6] Michael Danielczuk, Matthew Matl, Saurabh Gupta, Andrew Li, Andrew Lee, Jeffrey Mahler, and Ken Goldberg. Segmenting unknown 3D objects from real depth images using Mask R-CNN trained on synthetic data. 2019. 2

[7] Michael Danielczuk, Arsalan Mousavian, Clemens Eppner, and Dieter Fox. Object rearrangement using learned implicit collision functions. In *ICRA*, 2020. 2, 5, 6, 7

[8] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *CVPR*, 2018. 3

[9] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, Luca Weihs, Mark Yatskar, and Ali Farhadi. RoboTHOR: An open simulation-to-real embodied AI platform. In *CVPR*, 2020. 3

[10] Danny Driess, Jung-Su Ha, and Marc Toussaint. Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image. In *RSS*, 2020. 1, 2

[11] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable CNN for joint description and detection of local features. In *CVPR*, 2019. 3

[12] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ManipulaTHOR: A framework for visual object manipulation. In *CVPR*, 2021. 3

[13] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, Jun 1981. 4

[14] Peter R. Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. In *CoRL*, 2018. 2

[15] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(1):265–293, 2021. 1, 2

[16] Caelan Reed Garrett, Chris Paxton, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Dieter Fox. Online replanning in belief space for partially observable task and motion problems. In *ICRA*, 2020. 1, 2

[17] Zan Gojcic, Or Litany, Andreas Wieser, Leonidas J. Guibas, and Tolga Birdal. Weakly supervised learning of rigid 3D scene flow. In *CVPR*, 2021. 3

[18] Google Research. Google scanned objects. https://app.ignitionrobotics.org/GoogleResearch/fuel/collections/Google%20Scanned%20Objects, 2021. 4, 8

[19] Ankit Goyal and Jia Deng. PackIt: A virtual environment for geometric planning. In *ICML*, 2020. 2

[20] Ankit Goyal, Kaiyu Yang, Dawei Yang, and Jia Deng. Rel3D: A minimally contrastive benchmark for grounding spatial relations in 3D. In *NeurIPS*. 2020. 2

[21] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. HPLFlowNet: Hierarchical permutohedral lattice FlowNet for scene flow estimation on large-scale point clouds. In *CVPR*, 2019. 3

[22] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 2

[23] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, 2019. 2

[24] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *CVPR*, 2020. 3

[25] Junhwa Hur and Stefan Roth. Self-supervised multi-frame monocular scene flow. In *CVPR*, 2021. 3

[26] Brian Ichter, Pierre Sermanet, and Corey Lynch. Broadly-exploring, local-policy trees for long-horizon task planning. *arXiv preprint arXiv:2010.06491*, 2020. 1, 2

[27] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 2

[28] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The robot learning benchmark & learning environment. *RA-L*, 5(2):3019–3026, 2020. 3

[29] Huaizu Jiang, Deqing Sun, Varun Jampani, Zhaoyang Lv, Erik Learned-Miller, and Jan Kautz. SENSE: A shared encoder network for scene-flow estimation. In *ICCV*, 2019. 3

[30] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *ICCV*, 2021. 2

[31] Ivan Kapelyukh and Edward Johns. My house, my rules: Learning tidying preferences with graph neural networks. In *CoRL*, 2021. 2

[32] Jennifer E. King, Marco Cognetti, and Siddhartha S. Srinivasa. Rearrangement planning using object-centric and robot-centric action spaces. In *ICRA*, 2016. 1, 2

[33] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *ECCV*, 2020. 2

[34] Yann Labbé, Sergey Zagoruyko, Igor Kalevatykh, Ivan Laptev, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Monte-Carlo Tree Search for efficient visually guided rearrangement planning. *RA-L*, 5(2):3715–3722, 2020. 1, 2

[35] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6D pose estimation. In *ECCV*, 2018. 2

[36] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. StructFormer: Learning spatial structure for language-guided semantic rearrangement of novel objects. *arXiv preprint arXiv:2110.10189*, 2021. 1, 2

[37] Xingyu Liu, Charles R. Qi, and Leonidas J. Guibas. FlowNet3D: Learning scene flow in 3D point clouds. In *CVPR*, 2019. 3

[38] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2

[39] Ziyuan Liu, Wei Liu, Yuzhe Qin, Fanbo Xiang, Songyan Xin, Maximo A. Roa, Berk Calli, Hao Su, Yu Sun, and Ping Tan. OCRTOC: A cloud-based competition and benchmark for robotic grasping and manipulation. *arXiv preprint arXiv:2104.11446*, 2021. 2

[40] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. Vancouver, British Columbia, 1981. 3

[41] Kunming Luo, Chuan Wang, Shuaicheng Liu, Haoqiang Fan, Jue Wang, and Jian Sun. UPFlow: Upsampling pyramid for unsupervised optical flow learning. In *CVPR*, 2021. 2

[42] Zhaoyang Lv, Kihwan Kim, Alejandro Troccoli, Deqing Sun, James M. Rehg, and Jan Kautz. Learning rigidity in dynamic scenes with a moving camera for 3D motion field estimation. In *ECCV*, 2018. 3

[43] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *CVPR*, 2019. 3

[44] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *RSS*, 2017. 2

[45] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *CVPR*, 2020. 3

[46] Nathan Morrical, Jonathan Tremblay, Yunzhi Lin, Stephen Tyree, Stan Birchfield, Valerio Pascucci, and Ingo Wald. NViSII: A scriptable tool for photorealistic image generation. *arXiv preprint arXiv:2105.13962*, 2021. 4

[47] Douglas Morrison, Juxi Leitner, and Peter Corke. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. In *RSS*, 2018. 2

[48] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-DOF GraspNet: Variational grasp generation for object manipulation. In *ICCV*, 2019. 2

[49] Chris Paxton, Chris Xie, Tucker Hermans, and Dieter Fox. Predicting stable configurations for semantic placement of novel objects. In *CoRL*, 2021. 2

[50] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *CVPR*, 2019. 2

[51] Ahmed H. Qureshi, Arsalan Mousavian, Chris Paxton, Michael C. Yip, and Dieter Fox. NeRP: Neural rearrangement planning for unknown objects. In *RSS*, 2021. 1, 2, 6, 7

[52] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 3

[53] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied AI research. In *ICCV*, 2019. 3

[54] Daniel Seita, Pete Florence, Jonathan Tompson, Erwin Coumans, Vikas Sindhwani, Ken Goldberg, and Andy Zeng. Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In *ICRA*, 2021. 2

[55] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and where pathways for robotic manipulation. In *CoRL*, 2021. 1, 2

[56] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 3

[57] Anthony Simeonov, Yilun Du, Beomjoon Kim, Francois R Hogan, Joshua Tenenbaum, Pulkit Agrawal, and Alberto Rodriguez. A long horizon planning framework for manipulating rigid pointcloud objects. In *CoRL*, 2020. 2

[58] Austin Stone, Daniel Maurer, Alper Ayvaci, Anelia Angelova, and Rico Jonschkowski. SMURF: Self-teaching multi-frame unsupervised RAFT with full-image warping. In *CVPR*, 2021. 2

[59] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T. Freeman, and Ce Liu. AutoFlow: Learning a better training set for optical flow. In *CVPR*, 2021. 2

[60] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 2

[61] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3D orientation learning for 6D object detection from RGB images. In *ECCV*, 2018. 2

[62] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-GraspNet: Efficient 6-DoF grasp generation in cluttered scenes. In *ICRA*, 2021. 2, 6

[63] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 2, 4

[64] Zachary Teed and Jia Deng. RAFT-3D: Scene flow using rigid-motion embeddings. In *CVPR*, 2021. 3

[65] Chenxi Wang, Hao-Shu Fang, Minghao Gou, Hongjie Fang, Jin Gao, and Cewu Lu. Graspness discovery in clutters for fast and accurate grasp detection. In *ICCV*, 2021. 2

[66] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martin-Martin, Cewu Lu, Li Fei-Fei, and Silvio Savarese. DenseFusion: 6D object pose estimation by iterative dense fusion. In *CVPR*, 2019. 2

[67] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *CVPR*, 2021. 1, 2

[68] Fei Xia, Amir R. Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: Real-world perception for embodied agents. In *CVPR*, 2018. 3

[69] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A SimulAted Part-based Interactive ENvironment. In *CVPR*, 2020. 2, 3

[70] Yu Xiang, Christopher Xie, Arsalan Mousavian, and Dieter Fox. Learning RGB-D feature embeddings for unseen object instance segmentation. In *CoRL*, 2020. 2, 4

[71] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, 2018. 3

[72] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In *CoRL*, 2020. 2

[73] Feihu Zhang, Oliver J. Woodford, Victor Adrian Prisacariu, and Philip H.S. Torr. Separable flow: Learning motion cost volumes for optical flow estimation. In *ICCV*, 2021. 2

[74] Qunjie Zhou, Torsten Sattler, and Laura Leal-Taixe. Patch2Pix: Epipolar-guided pixel-level correspondences. In *CVPR*, 2021. 3