💃 Baggie Boogie 💃: A Robotic Journey with Transporter Networks

Zilin Si, Jennifer Yang, Bardienus P. Duisterhof, Hung-Jui Huang, Andrew Wong

***Abstract*—Pick and place is a critical prerequisite for many applications in industrial, pharmaceutical, and household tasks. Robots working in the kitchen will need to pick and place food items, but also manipulation, pick and place tools. Pick and place typically requires precise and robust manipulation for task completion, and can be subdivided in pick (including grasping), and placing. The robot needs to first identify the world pose of the object, either implicitly or explicitly, to then grasp the object at a robust grasp. One line of work for such tasks is known as imitation learning: to learn from expert demonstrations.**

Here we propose to use imitation learning, more specifically behavior cloning, to achieve robust (not so accurate) pick and place of deformable objects. The proposed approach leverages a modified version of transporter networks, learning transport in robot coordinates instead of image coordinates - ommiting the need for camera-to-robot calibration. We also contribute an automated data generation pipeline, learning from a state machine instead of expert demonstrations. The results suggest a test error of less than 8 cm, which proves to be precise enough for robust grasping. Possible sources of error are the imprecise data collection process (inconsistent grasping), insufficient data, insufficient data diversity, and network architecture.

I. Introduction

The act of picking and placing is fundamental to numerous applications in industrial, pharmaceutical, and household settings. For instance, in kitchen operations, robots are not only required to handle food items but also to manipulate tools for picking and placing. This process involves precise and resilient manipulation for successful task execution and can be broken down into two main stages: picking (which includes grasping) and placing. To accomplish these tasks, the robot must initially determine the object's world pose, either implicitly or explicitly, and then execute a secure grasp. Robustness is important, as robots often perform sequential tasks such as stacking, where errors may compound. Other requirements include: (1) work with relatively little training data, (2) precision for tasks like cracking eggs, and (3) hardware-agnostic (after training).

An approach commonly employed for such tasks is a modular approach which leverages a depth camera, RGB camera, combined with grasp detection and object detection. A key limitation of this approach is that these approaches might fail at the long tail of the distribution. For example, the grasp detector might struggle in presence of an unknown fruit, and the object detector might struggle with a new color kitchen counter top, or different lighting.

Imitation learning has been proposed as a paradigm to learn from expert demonstrations, typically in a narrow domain. This allows a user to use robot demonstrations (e.g., teleoperation, sometimes videos), and have the robot reproduce the task with some generalization. The advantage of this approach over more generic pre-trained approaches is that the performance in a narrow domain is likely to far exceed pre-trained approaches. Challenges include, but are not limited to, multi-modal behavior, noisy labels, small datasets, and distribution shift.

Here we propose to modify Transporter Networks for the task of pick and place. First, we deploy an automated data collection method. The method uses RGB images and camera calibration with color masking to arrive at rough grasps for pick and place. Using a dataset collected with this method, we then train a modified version of transporter networks and show robust performance at test time. The modified version, 'Baggie Boogie', infers robot grasp locations directly instead of grasp locations in image space. This has makes calibration unnecessary, though it does make it more challenging to transfer the policy to new setups. 'Baggie Boogie' makes use of ResNet18 as a visual encoder, adding both complexity and hopefully better generalization.

The results show errors of roughly 8 cm for picking and 6 cm for placing. We find the accuracy is sufficient to grasp deformable objects thanks to their compliance. The performance could be improved significantly by, for example: (1) deploying actual human policies, which might be more consistent, (2) avoiding learning transport in world coordinates, use calibration instead, (3) larger dataset with more diversity to avoid errors due to domain shifts, and (4) the model architecture may be improved by using more suitable visual encoders.

In summary, we contribute:

- A modified implementation of transporter networks which also implicitly learns camera calibration as part of the learning process.
- A method for automated collection of expert data.
- Experiments that show the implementation, which we

$\langle bduister, hungjuih, zsi, jennifey, twong2\rangle$@andew.cmu.edu

term 'Baggie Boogie', is robust in real-world pick and place.

II. Related Work

A. Behavior Cloning

Performing complex manipulation remains a considerable challenge for robots. A promising approach is behavior cloning, in which a human expert demonstrates executions of a manipulation task and the robot learns to clone the demonstrated policy. Behavior cloning techniques can be categorized into two main classes based on policy representation. Explicit policy cloning involves learning a function that directly maps observations to actions [10] [1]. For example, Pomerleau [8] trained a neural network to take in camera images and laser scan data and output steering wheel angles. Similarly, [2] used neural networks to map raw camera images directly to steering commands. Zhang et al. [11] had users teleoperate robots and learn neural network policies mapping images to actions. Implicit policy methods learn energy-based models (EBM) that define action probabilities conditional on observations [3] [4]. Learning implicit policies can better handle multi-modal demonstrations [3]. Since our demonstration data is uni-modal, we apply explicit policy approaches for simplicity.

B. Pick and Place

Pick-and-place manipulation is a foundational capability for robots. A common approach involves two sequential phases: object detection followed by grasping. Zeng et al. [9] proposed scanning objects and matching them to pre-scanned models to estimate pose, then planning grasps based on the detected pose. Miller et al. [6] modeled objects as geometric primitives and planned grasps accordingly. Affordance-based methods comprise another approach, directly predicting grasp poses from camera images without explicit object identity prediction [7] [5]. By avoiding the need to recognize specific objects, affordance methods can exhibit better generalizability to novel objects. In this work, we adopt an affordance-based approach to perform pick-and-place.

III. Method

A. Calibration

Geometric camera calibration is a critical prerequisite for many vision-based robotics tasks. It typically involves recovering the intrinsic and extrinsics. The intrinsics includes an unprojection and projection function. The unprojection function infers a bearing vector from each pixel location, whereas the projection function infers image space coordinates for any point in the world. The extrinsics describe the 6 DoF pose of an image sensor w.r.t. other frames such as the robot frame or a common world frame. We now describe the separate calibration steps performed.

Img2Robot: The automated data collection pipeline needs a calibration to infer robot coordinates for points in the image. With this calibration, we can mask out the bag, and find an appropriate robot pick location. To achieve this, we move the end effector to several positions in the workspace, recording end effector position as well as image coordinates. We then fit a homography, assuming low distortion, and use that homography as the calibration.

Cam2World: The extrinsic calibration between camera and the world was more straightforward, by moving an aruco marker inside the FOV of the camera with known intrinsics, we retrieve the pose of the aruco marker - and with that the camera pose w.r.t. the aruco marker.

Robot2World: To calibrate the rigid body transform between the robot and a known world frame, we move the end-effector to three positions: (1) the world origin $\mathbf{v}_o$, (2) a point along the x-axis $\mathbf{v}_x$, and (3) a point along the y-axis $\mathbf{v}_y$. We then define the 3×4 transformation matrix T as:

$$T = \begin{bmatrix} \frac{\mathbf{v}_x - \mathbf{v}_o}{|\mathbf{v}_x - \mathbf{v}_o|} & \frac{\mathbf{v}_y - \mathbf{v}_o}{|\mathbf{v}_y - \mathbf{v}_o|} & \frac{\mathbf{v}_x - \mathbf{v}_o}{|\mathbf{v}_x - \mathbf{v}_o|} \times \frac{\mathbf{v}_y - \mathbf{v}_o}{|\mathbf{v}_y - \mathbf{v}_o|} & \mathbf{v}_o \end{bmatrix} \quad (1)$$

B. Data Collection

To collect data, we created a simple, autonomous pick and place system. This system collected images of workspace and the bag locations to be used in the model training process. We also treated this system as the baseline.

First, we use the Kinect camera to gather and store an RGB image of the workspace. Then, the image is analyzed with OpenCV to detect and record the bag's center location and rotational angle. This information is transmitted to the Franka Arm, which prompts the arm to move to the bag and rotate the gripper to the appropriate corresponding angle. Next, the bag is lifted and moved to a random location. The gripper is rotated back to its original orientation, aligning the bag square to the workspace. Subsequently, the bag is placed down onto the workspace. Finally, the Kinect camera collects another RGB image of the workspace to get the final bag location. This pipeline ran autonomously for 300 iterations, saving data every 25 iterations.

Every random placement of the bag is at least 20 cm from its previous location. This is to ensure that there is ample variance in the data being collected, and the full workspace can be explored. This enables the model to better generalize to different locations.

To ensure the quality of the data being collected, correctional measures are put in place. We included a slight offset that is added to the detected center location to account for camera distortion. We also correct for the bag rotation every iteration to ensure that rotation errors are corrected immediately and do not propagate. One common OpenCV detection error would be that the bag would be detected at approximately a 45 degree angle, contrary to its actual orientation. This discrepancy resulted in the arm trying to pick up the bag at an angle, significantly

Fig. 1: Goal-Annotated Network Input

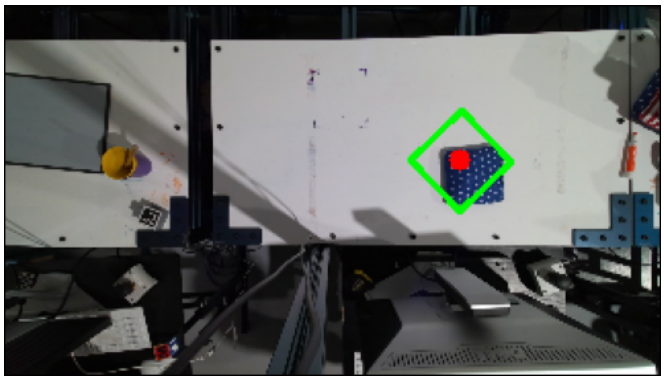

Fig. 2: OpenCV Bag Detection Errors

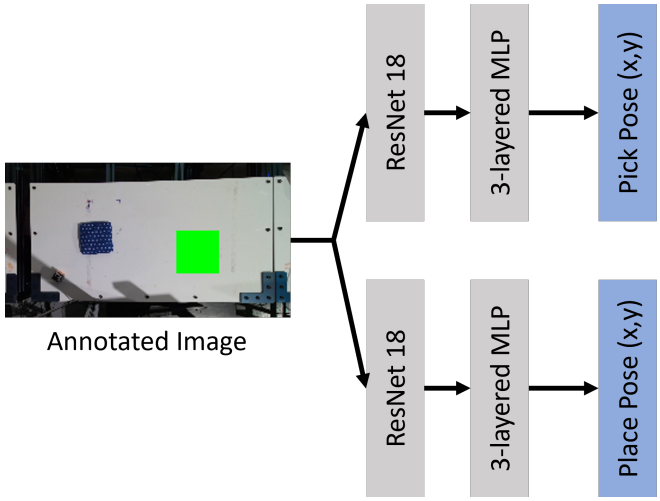

Fig. 3: The picking and placing network.

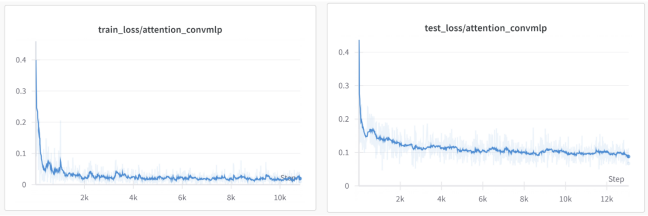

Fig. 4: Training and validation losses.

deviating from the true center. This error is depicted in Figure 2. To account for this error, we would not angle the gripper and increase the offset for the bag pick location in these situations.

The collected images are post processed to create goal-annotated images, as seen in Figure 1.

C. Picking

The objective of the picking network is to determine the optimal grasping location on a bag within a robotic coordinate system, given an input image of the bag with the annotated goal. To accomplish this, we implement a CNN model that takes the bag image as input and directly outputs a two-dimensional pose prediction (Figure 3). The network is trained on a dataset of 200 bag images, with supervision provided in the form of ground truth grasping locations. We optimize the parameters of the network by minimizing the L1 loss between the predicted and ground truth poses.

D. Placing

Similarly, a CNN with an architecture analogous to the picking network is trained to predict the two-dimensional pose for placing the bag within the scene (Figure 3). This CNN was developed using the same input image and outputs the target pose. As with the picking network, an L1 loss function was utilized during training procedures. We attempted to implement the transporter network architecture as proposed in [10]. However, the absence of precise calibration hindered the accurate execution of the cropping and rotation procedures in the transporter network methodology.

E. Implementation Details

For both picking and placing, we use the same network architectures which take one single image as input and predict a 2D location for pick or place as output. Our models include CNN-based image feature embedding layers and MLP-based location prediction layers. For the CNN module, we use a pre-trained resnet18, and for the MLP module, we use three linear layers (1000-128-32-2) with RELU activation function. During the training, we used Adam optimizer with 1 as batch size and $1e-4$ as learning rate. The training is supervised with L1 losses between the predicted and ground truth 2D locations.

In total, we use 200 data points as the training set and 50 data points as the validation set and directly test on the real-world setup. The training and validation losses are plotted in Fig. 4 where both converged after $10,000$ training iterations. The averaged picking and placing errors on the validation set are 7.8 cm and 5.8 cm.

IV. Results

A. Simulation

The codes of Transporter Net were originally open-sourced in Tensorflow, and we used the adapted PyTorch

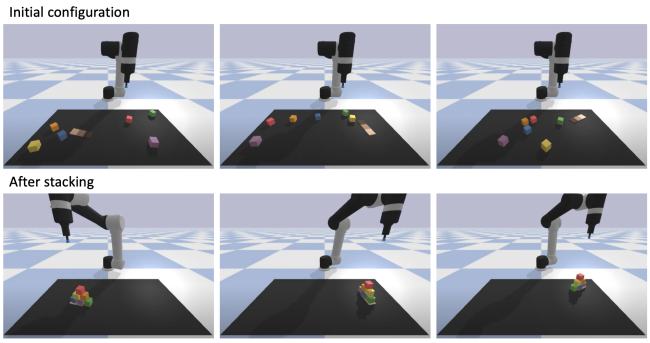

Fig. 5: Pyramid stacking with Transporter Net in simulation.

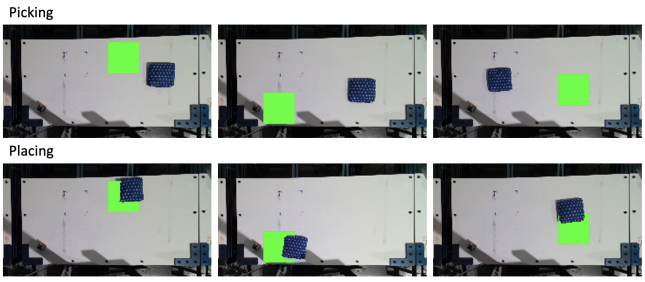

Fig. 6: Real-world results with virtually labeled target boxes.

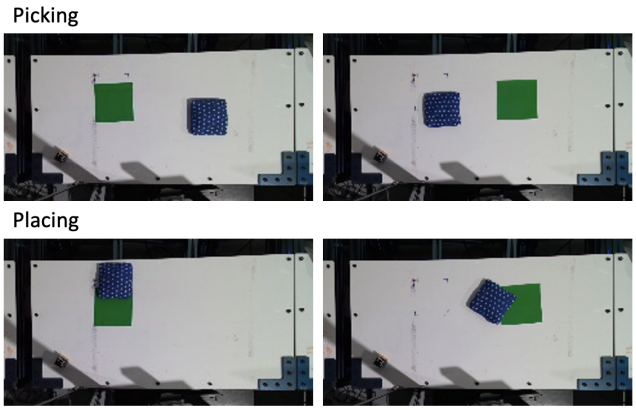

Fig. 7: Real-world results with real target boxes.

version https://github.com/thomaschabal/transporter-nets-torch for the pyramid stacking task in simulation. For each trajectory, six blocks are randomly initialized on the table, and a robot arm with a suction cup gripper will execute six times of pick-and-place to stack the blocks in a pyramid shape within the marked target area. To be fair to compare with the real-world pick-and-place task, we collect the same amount of 200 training data points and 50 validation data points in simulation for training. We also trained the Transport Net for $10,000$ iterations. And we test on 50 more sets in the simulation where the accuracy is 100%. Some qualitative results are shown in Fig. 5 and we can see all the blocks are successfully stacked in the pyramid shape and in the target area.

B. Real - Data augmented

We first evaluate our proposed approach on augmented real-world data. Similarly to the procedure of data collection and annotation as described in Section III-B, we first randomly place a bag on the table within the camera view. Then we randomly define a placing location and augment the image by adding the target location in the image as a green block. Then we use this augmented image as the input for both the picking and placing models and infer the predicted picking and placing locations. The robot will first go to the predicted picking location and grasp the bag, and then send the bag to the predicted placing location and drop the bag. The grasping and dropping heights are pre-defined given the bag's size. Here we show some results in Fig. 6. From the results, we can see that the bag can be transported to the target location but with certain errors. We observed that the errors are accumulated from both picking and placing, such as the gripper didn't grasp at the center of the bag, and then the predicted placing location also has an offset shift on top of the picking error.

C. Real - Green paper in real

We further evaluate our models on the real-world setup by using a printed green card as the target placing location instead of virtual augmentation. As shown in Fig. 7, we demonstrate that without fine-tuning, we can directly deploy the trained models on the real-world setup and conduct the pick-and-place task.

To minimize the difference between the augmented data and the real-world data, we applied two additional data processing: First, we detected the RGB values of the green card from the images, and then re-annotated the training data by using the green box with the same RGB values; Secondly, instead of using a bounding box as the target location as shown in Fig. 2, we use a solid green area as shown in Fig. 1. We observed an improvement of placing prediction accuracy by using the solid target area which might come from better detection.

V. Conclusion and Limitations

Limitations One of the major limitations of this work is the model's dependence on high data quality. Although the data collection pipeline is designed to detect the middle of the bag and pick it up at that location, this was often not the case. Even with the offset to account for camera distortion or detection errors, the true pickup location would be varied so we could not guarantee that the bag was being picked up at the exact center. Instead, the bag would often to picked up at the side or the corner. Because of these errors, our model accuracy is compromised. Although the model is decent, it is not precise. As a result, we are unable to accurately place a bag and thus unable to stack the bags.

Conclusion In this project, we have demonstrated

a simple method for imitation learning. We modified transporter networks to come up with 'Baggie Boogie', an algorithm that infers robot grasp coordinates directly instead of the flow in image space. We observe results with an average test accuracy of 8cm in picking and 5 cm in placing. We find deformable grasping from demonstrations is actually particularly challenging because of the compliance: if grasps are not consistent, the model might not learn a consistent pick and place location. With rigid objects with less compliance, more precise pick and place locations would be necessary.

VI. Future Work

For future work, we would like to improve the picking model to pick up the bags more accurately. We would also like to improve the placing model so that we can stack the bags. This would require a large amount of higher quality data collection. Once the picking and placing models have been improved, we would like to explore different stacking patterns. For example, we would like to explore the possibility of stacking the beanbags all vertically or in a pyramid-like shape. We think it would also be interesting to explore generalizing to different bag colors, shapes, and sizes. Finally, we think it would be interesting to explore generalizing to other deformable objects.

References

[1] Yahav Avigal, Lars Berscheid, Tamim Asfour, Torsten Kröger, and Ken Goldberg. Speedfolding: Learning efficient bimanual folding of garments, 2022.

[2] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016. URL http://arxiv.org/abs/1604.07316.

[3] Pete Florence, Corey Lynch, Andy Zeng, Oscar Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. *CoRR*, abs/2109.00137, 2021. URL https://arxiv.org/abs/2109.00137.

[4] Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. Strictly batch imitation learning by energy-based distribution matching, 2021.

[5] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018. doi: 10.1177/0278364917710318.

[6] A.T. Miller, S. Knoop, H.I. Christensen, and P.K. Allen. Automatic grasp planning using shape primitives. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pages 1824–1829 vol.2, 2003. doi: 10.1109/ROBOT.2003.1241860.

[7] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *CoRR*, abs/1509.06825, 2015. URL http://arxiv.org/abs/1509.06825.

[8] Dean Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D.S. Touretzky, editor, *Proceedings of (NeurIPS) Neural Information Processing Systems*, pages 305 – 313. Morgan Kaufmann, December 1989.

[9] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker Jr., Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. *CoRR*, abs/1609.09475, 2016. URL http://arxiv.org/abs/1609.09475.

[10] 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. *CoRR*, abs/2010.14406, 2020. URL https://arxiv.org/abs/2010.14406.

[11] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635, 2018. doi: 10.1109/ICRA.2018.8461249.