Supplementary Material:

Mobile Robot Manipulation using Pure Object Detection

Per-Object TFOD Benchmark Results. We provide *per-object* **T**ask-Focused **F**ew-Shot **O**bject **D**etection (TFOD) benchmark results in Figure 7, which correspond to the ClickBot $k = 1, 2, 4$ few-shot example configurations in Table 5. As in Table 5, we find opportunities for innovation across all settings, especially one- or two-shot detection. The Wood, Chips Can, and Box of Sugar are particularly inaccurate for $k < 4$. Unsurprisingly, the $k = 4$ configuration has the best performance for all objects with the exception of Gelatin.

We expect performance improvements across all objects and few-shot configurations with future few-shot object detection research. In practice, such methodological advances will also improve robot task performance and reduce overall annotation requirements.

Camera Movement and Learned Visual Servo Control. We plot the camera movements for learning visual servo control in Figure 8 with the corresponding learned parameters originally shown in Figure 4.

For camera motion (Δx), ClickBot repeats eight movement *commands* comprising the permutations of {-5, 0, 5} cm across the x and y axes (e.g., x = -5, y = 5 for the second Broyden update). However, ClickBot's base movements are imprecise for small motions, so the *actual* measured movement distance we use for the update is slightly less (e.g., Base Forward = -2.7 cm and Base Lateral = 2.5 cm of actual motion for the second update). Nonetheless, the actual motion profile is sufficient to learn ClickBot's visual control, which we use for all experiments in Section 5.

Depth Estimate Convergence. In Section 4.3, we introduce ClickBot's active depth estimation, which continually processes incoming data while approaching objects for grasping. We provide an example depth convergence plot in Figure 9, which corresponds to the Chips Can result in Figure 1. ClickBot advances in 0.05 m increments, so the depth estimate generally completes with the object between 0.15 m to 0.2 m away. In this example, after the grasp camera moves 0.15 m, the Chips Can's final estimated depth is 0.18 m, which leads to a successful grasp of the Chips Can.

As discussed in Section 4.3, ClickBot estimates object depth from detection by comparing changes in bounding box size (i.e., optical expansion) with the corresponding camera movement, which we obtain using robot kinematics. We use the Box_{LS} equation [17, (9)] within our active depth estimation approach to process all available observations in a least-squares formulation, thus, our depth estimate generally improves as more data are collected. Finally, the depth estimate's accuracy significantly improves as the object gets closer and exhibits more rapid optical expansion.

Individual Trial Results for Task-Focused Annotation. We provide the task-focused few-shot annotation results for individual trials in Table 6. All Mean results are the same as those originally shown in Table 3. Remarkably, no experiment configuration uses more than a minute of human annotation time per object, which is approximately the same amount of time required to annotate a single segmentation mask and much less than the time required to generate a 3D object model.

We discuss a few notable individual trial results. For the Visual Servo and Depth Benchmark on the Food: Chips Can, Potted Meat, Plastic Banana trial, ClickBot learns the Find, Move, and Depth tasks for all objects without prior annotation using 3 task-focused examples. For Pick-and-Place *in Clutter* with Prior Annotation on the Food: Box of Sugar, Tuna, Gelatin trial, ClickBot requires only 1 task-focused Move example to transfer learning from the prior pick-and-place task to learn pick-and-place in clutter. Finally, for Pick-and-Place *in Clutter* on the Food: Chips Can, Potted Meat, Plastic Banana trial, ClickBot learns all tasks for all objects in a cluttered setting without prior annotation using 7 task-focused examples.

ClickBot-Generated Map for Dynamic Pick-and-Place. We provide an example ClickBot-generated map in Figure 10, which corresponds to the dynamic pick-and-place result originally shown in Figure 6.

ClickBot uses the same few-shot detection model with it's head-mounted RGBD camera, which enables ClickBot to map any RGB-based bounding box to a median 3D point using the corresponding depth image. Using this map for the Find task, ClickBot quickly identifies the closest grasp object and subsequent placement location even after a grasped object is blocking ClickBot's grasp camera.

Cleaning Scattered Cups with Dynamic Pick-and-Place. We show ClickBot cleaning scattered cups with changing placement locations in Figure 11, which corresponds to the final dynamic pick-and-place experiment in Section 5.5.

Supplementary Videos are provided at https://youtu.be/Bby4Unw7HrI. Videos include a detection-based manipulation overview, the learning visual servo control experiment from Section 5.2, and two example dynamic pick-and-place experiments from Section 5.5, which includes ClickBot cleaning scattered objects with moving placement locations at over 120 picks-per-hour.

Application Novelty. We provide an extended discussion on application novelty to supplement the main paper. We also provide an application comparison of related work in Table 7. All citations are with the main paper's References.

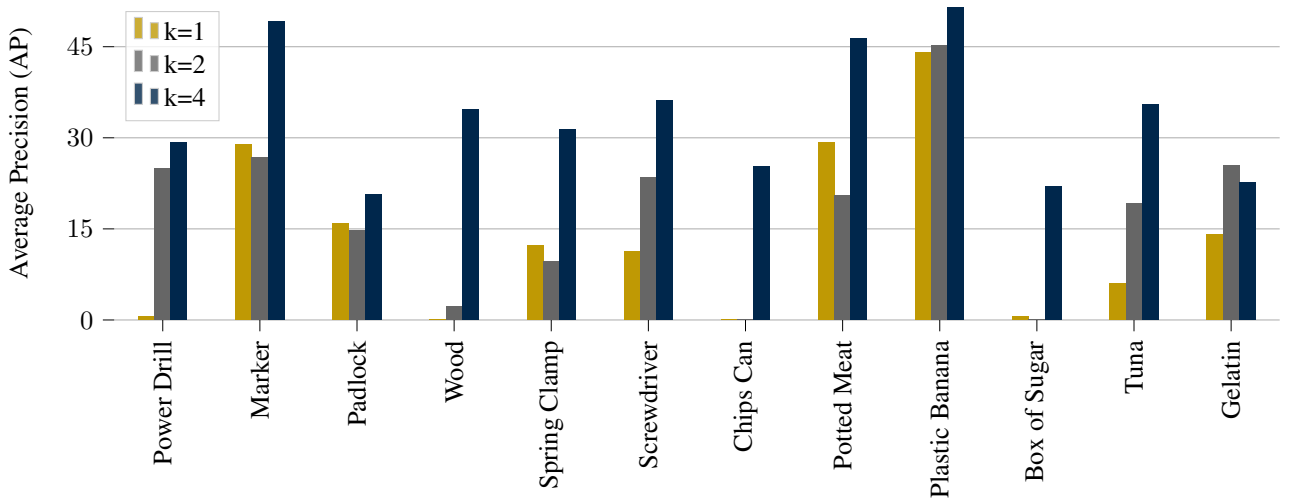

Figure 7. **Per-Object Task-Focused Few-Shot Object Detection (TFOD) Benchmark Results**. All TFOD test results correspond to the baseline ClickBot method in Table 5. There are many opportunities for future improvements, especially for $k = 1, 2$ few-shot examples.

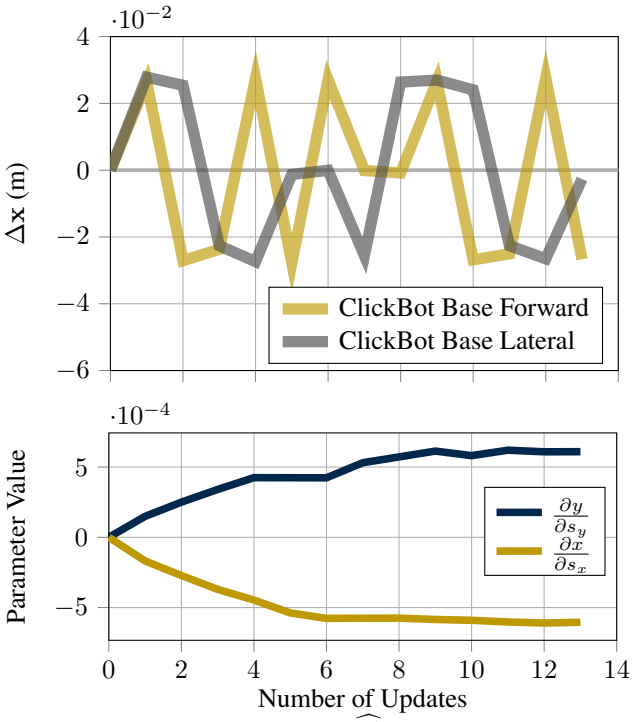

Figure 8. **Learned Visual Control $\widehat{\mathbf{L}_\mathbf{s}^+}$ Parameter Convergence with Camera Movement**. ClickBot learns detection-based visual servo control in 13.3 seconds after 13 camera movements (top) and corresponding Broyden updates (4) (bottom). Subsequently, ClickBot uses this learned visual control in all other experiments.

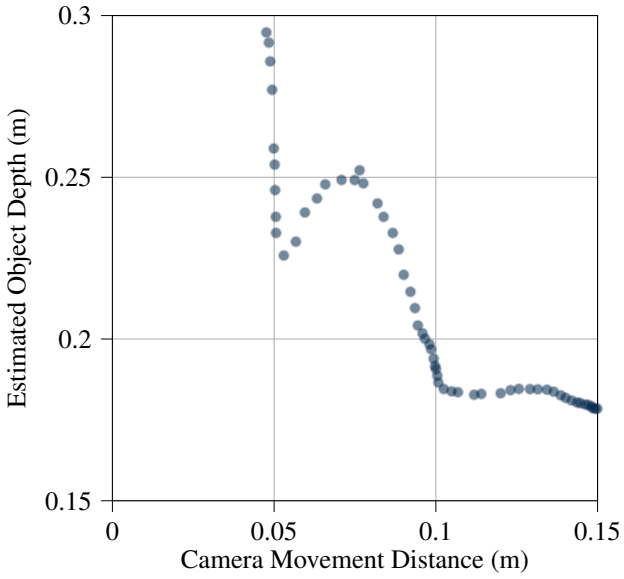

Figure 9. **Depth Estimate Convergence**. We plot the depth estimate corresponding to the Chips Can result in Figure 1. ClickBot actively estimates an object's depth as it approaches for grasping, and this depth estimate convergences as the camera moves closer and collects more data. Notably, the object's depth relative to the camera decreases with camera movement.

A fundamental asset of robot perception is the opportunity to learn beyond static datasets from a robot's own surroundings. Consequently, the robotics community has developed innovative solutions wherein robots actively learn in a fixed workspace. To estimate an object's pose [62], robots interact with 3D-modeled objects [6] to create new training data [11, 39]. To grasp objects [12], robots perform large-scale data collection individually [48] or with other robots [32]. To place objects, robots learn residual physics to toss objects into out-of-reach boxes [66]. However, all of these robots learn in a fixed workspace, and scaling these solutions to mobile operation remains an open problem. On

Table 6. **Task-Focused Few-Shot Annotation Results (Individual Trials)**. All results are from a single consecutive set of trials. Clicks are the number of annotated bounding boxes, which each require 7 seconds (see user study [26]). Note that Clicks per Few-Shot Example varies with the number of in-view objects. CPU refers to training time. Mean results are the same as those originally shown in Table 3.

	Number of Task-Focused Few-Shot Examples Generated (E)					Requirements Per Object Class			
						Annotation	Robot	CPU	
Task-Focused Learning Experiment Trial	Find	Move	Depth	Grasp	Total	Clicks	Time (seconds)		
Learning Visual Control (Section 5.2)	1	0	N/A	N/A	1	1.0	7.0	13.3	227
Visual Servo and Depth Benchmark (Section 5.3)									
Tool: Power Drill, Marker, Padlock	**1**	1	3	N/A	5	3.7	25.7	22.9	381
Tool: Wood, Spring Clamp, Screwdriver	**1**	1	**2**	N/A	4	3.7	25.7	10.9	309
Food: Chips Can, Potted Meat, Plastic Banana	**1**	**0**	**2**	N/A	**3**	**2.7**	**18.7**	**9.3**	**233**
Food: Box of Sugar, Tuna, Gelatin	**1**	1	4	N/A	6	3.7	25.7	30.0	460
Kitchen: Mug, Softscrub, Skillet with Lid	**1**	1	5	N/A	7	5.0	35.0	30.0	536
Kitchen: Plate, Spatula, Knife	**1**	**0**	3	N/A	4	**2.7**	**18.7**	18.0	304
Shape: Baseball, Plastic Chain, Washer	**1**	2	3	N/A	6	4.7	32.7	18.6	457
Shape: Stacking Cup, Dice, Foam Brick	**1**	1	3	N/A	5	3.7	25.7	21.5	387
Mean	1.0	0.9	3.1	N/A	5.0	3.7	26.0	20.2	383
Pick-and-Place with Prior Annotation (Section 5.4)									
Tool: Power Drill, Marker, Padlock	**0**	**0**	**0**	4	4	3.0	21.0	27.3	307
Tool: Wood, Spring Clamp, Screwdriver	**0**	**0**	1	**2**	**3**	**2.7**	**18.7**	**21.9**	**231**
Food: Chips Can, Potted Meat, Plastic Banana	1	**0**	1	3	5	3.7	25.7	32.3	378
Food: Box of Sugar, Tuna, Gelatin	**0**	1	3	**2**	6	4.3	30.3	35.0	457
Mean	0.3	0.3	1.3	2.8	4.5	3.4	23.9	29.1	343
Pick-and-Place *in Clutter* with Prior Annotation (Section 5.4)									
Tool: Power Drill, Marker, Padlock	1	**0**	**0**	3	4	2.7	18.7	32.6	374
Tool: Wood, Spring Clamp, Screwdriver	**0**	2	**0**	3	5	3.7	25.7	34.6	387
Food: Chips Can, Potted Meat, Plastic Banana	1	**0**	**0**	3	4	3.3	23.3	25.7	309
Food: Box of Sugar, Tuna, Gelatin	**0**	1	**0**	**0**	**1**	**1.0**	**7.0**	**0.2**	**76**
Mean	0.5	0.8	0.0	2.3	3.5	2.7	18.7	23.2	287
Pick-and-Place (Section 5.4)									
Tool: Power Drill, Marker, Padlock	**1**	1	**2**	5	9	6.7	46.7	61.0	689
Tool: Wood, Spring Clamp, Screwdriver	**1**	1	**2**	**3**	7	6.0	42.0	38.3	543
Food: Chips Can, Potted Meat, Plastic Banana	**1**	**0**	**2**	**3**	**6**	**5.3**	**37.3**	**33.4**	**457**
Food: Box of Sugar, Tuna, Gelatin	**1**	1	4	4	10	6.0	42.0	73.0	770
Mean	1.0	0.8	2.5	3.8	8.0	6.0	42.0	51.4	615
Pick-and-Place *in Clutter* (Section 5.4)									
Tool: Power Drill, Marker, Padlock	**1**	2	5	5	13	10.0	70.0	97.0	1,008
Tool: Wood, Spring Clamp, Screwdriver	**1**	**0**	4	3	8	**5.7**	**39.7**	60.2	614
Food: Chips Can, Potted Meat, Plastic Banana	**1**	2	**2**	**2**	**7**	6.0	42.0	**33.0**	**540**
Food: Box of Sugar, Tuna, Gelatin	**1**	4	6	3	14	8.3	58.3	79.2	1,082
Mean	1.0	2.0	4.3	3.3	10.5	7.5	52.5	67.3	811

the other hand, this paper uniquely addresses the problem of robot learning in a mobile application setting. Mobile application challenges include moving cameras, changing environments, and dynamic grasp positioning for a mobile robot and dexterous workspace that move in the world frame.

For robot perception without learning, robot mobility has been achieved using closed-form visual servo control (VS), i.e., using visual data as input to a servo feedback control loop [8, 9, 24]. VS achievements include positioning UAVs [18, 41] or wheeled robots [38, 40] and mobile manipulation [30, 57]. However, all of these mobile VS robots use structured visual features (e.g., fiducial markers or LED panels). On the other hand, this paper introduces a new detection-based approach to robot learning to learn VS, depth estimation, and grasping on a mobile robot in unstructured settings, thus extending VS to new applications where the environment and objects can change.

Achieving the mobile manipulation results in this paper required innovation across each of the detection-based tasks. For detection-based visual servo control (Section 4.2), we 1) develop a set of detection-based features that account for detection errors and multiple objects and 2) define a novel update formulation that learns visual servo control on average in less than 14 s and reduces learning variability by 65-85% relative to prior visual servo approaches (Table 1). For detection-based depth estimation (Section 4.3), we adopt our previous least squares formulation [17, (9)] into a new active detection framework that improves depth estimation during the grasp approach while mitigating proximity-based detection errors. To our knowledge, this paper is the first work to use detection-based depth estimation in a real-time application. For detection-based grasping (Section 4.4), we use a novel active multi-view grasp selection approach that requires only bounding

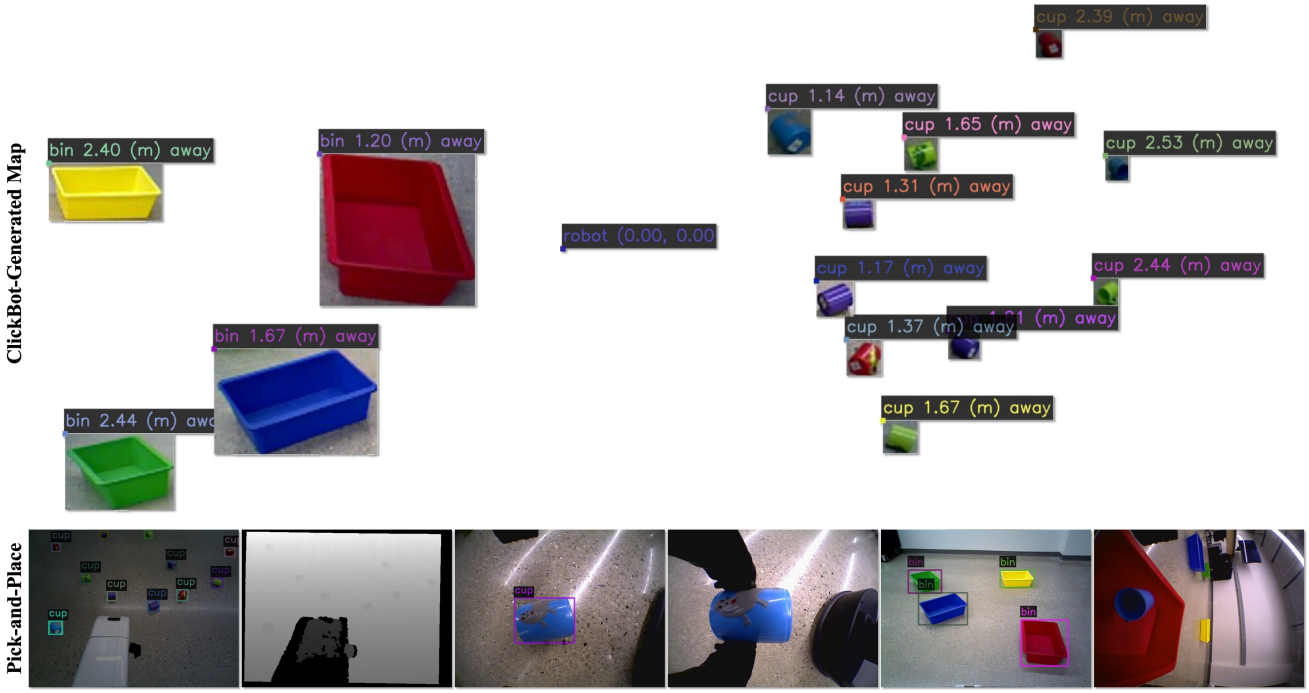

Figure 10. **ClickBot-Generated Map for Pick-and-Place with Dynamic Locations**. In dynamic pick-and-place (bottom), ClickBot uses detection with an RGBD camera to locate and grasp scattered objects (left) and similarly uses detection to find a suitable placement location (right). Here, we show the ClickBot-generated map (top) corresponding to the pick-and-place result originally shown in Figure 6.

Table 7. **Application Comparison of Related Work**. To our knowledge, this is the first work to use a real robot to learn few-shot mobile manipulation for novel objects.

	Vision		Robot		
	Few-Shot Object Learning	Non-Structured Visual Features	Collect Train Data	Manipulate Objects	Mobile Operation
Few-Shot Object Detection e.g. [10]	Yes	Yes	N/A	N/A	N/A
Train Detector with Robot Data e.g. [3]	No	Yes	Yes	No	Yes
Classic Visual Servo Control e.g. [57]	No	No	N/A	Yes	Yes
Learned Visual Manipulation e.g. [27]	No	Yes	Yes	Yes	No
Mobile Visual Manipulation e.g. [16]	No	Yes	No	Yes	Yes
ClickBot (Ours)	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

boxes. To our knowledge, this paper is the first work to grasp objects entirely from detection. Finally, we introduce TFOD (Section 3) to learn all of these tasks for new objects and settings on a real robot in a variety of experiments (Section 5), thereby validating our detection-based method.

Challenges, Solutions, and Future Work. We discuss a few implementation challenges we found in experiments and suggest some corresponding solutions and future work.

Moving HSR's grasp camera (Section 5.1) in close proximity to objects can cause the objects to blur, making detection more difficult. However, if blur causes an error, ClickBot uses the blurred image for few-shot annotation to update its detection model, which we found improves detection performance on blurred images. Another solution to decrease blur is to scale the control input $\mathbf{v}$ (3) to slow down the grasp camera when visual servoing to an object.

Our current approach to detection-based grasping (Section 4.4) uses an overhead antipodal grasp at the center of an object. However, HSR's gripper (Section 5.1) is too small to grasp some of the YCB Dataset objects using this approach (e.g., the Skillet with Lid and Plate shown in Figure 5). One solution is to train the detector to generate bounding boxes on *only* the graspable portion of each object (e.g., the Skillet with Lid's handle). Regardless, some objects are simply not graspable from overhead (e.g., the Plate). In future work, we will expand our approach to include alternative grasp strategies (e.g., lateral grasping at an object's side) when the detected object is too large for overhead grasping.

Our baseline detection model is based on a Faster R-CNN [51] configuration available on the popular and open source Detecton2 platform [61]. As discussed in Section 5.1, one motivation for using this baseline was ease of reproducibility for our experimental results. On the other hand, few-shot object detection (FSOD) is becoming a hotly studied area of object detection with increasingly rampant advances, even within just the past year

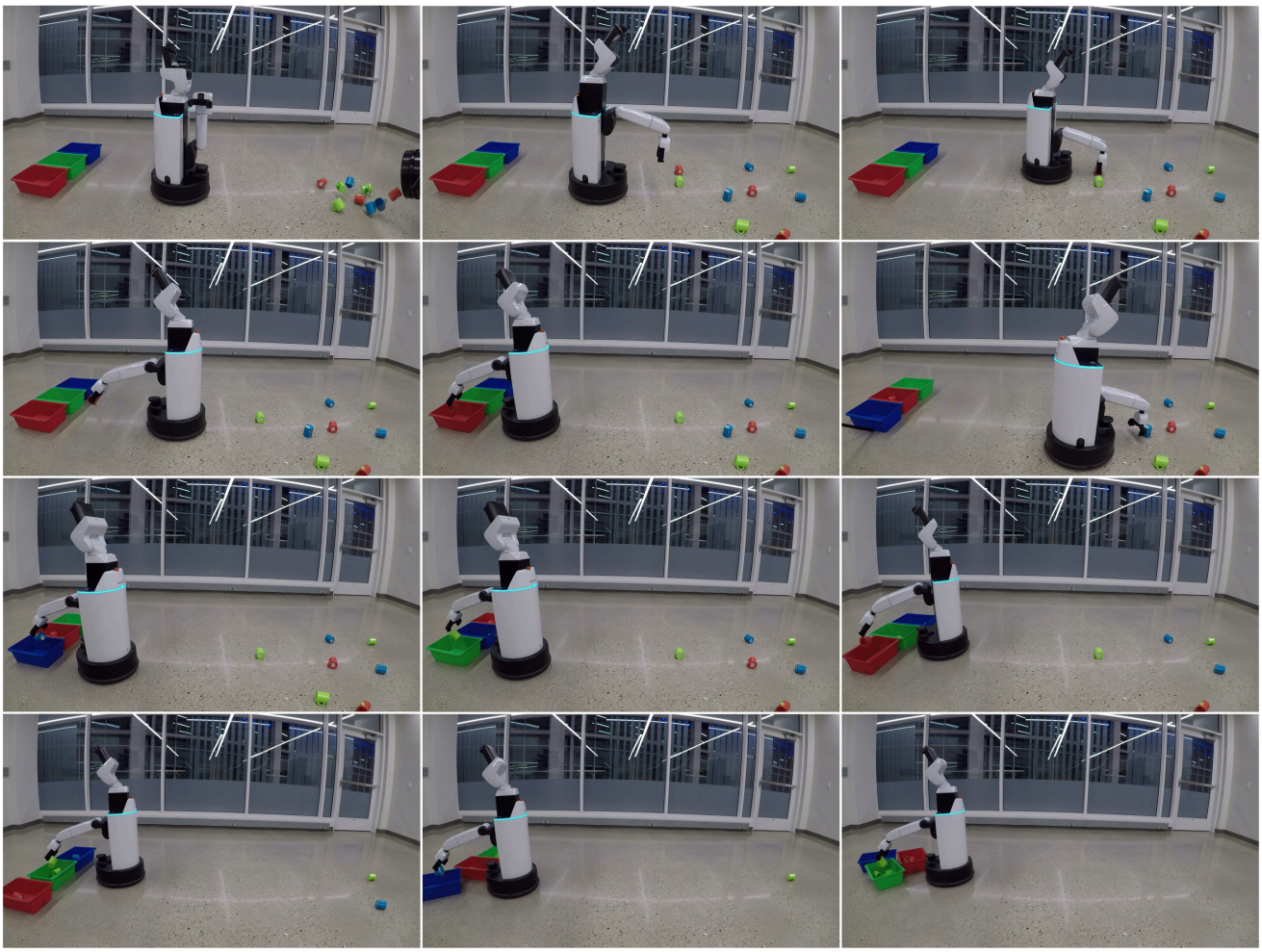

Figure 11. **Cleaning Scattered Cups with Dynamic Pick-and-Place**. After we scatter cups on the ground, ClickBot uses its head camera to locate the cups and then its grasp camera to grasp the closest cup (first row, left to right). Next, ClickBot uses its head camera to locate the bins and then places the grasped red cup in the red bin to match color. As ClickBot grasps the second cup, we also rearrange the bins (second row). ClickBot places each cup in the correctly colored bin, regardless of our rearranging the bins after each placement (third row). ClickBot cleans up all nine cups in 260 seconds, which equates to a rate of 124.6 mobile picks per hour (bottom row). To our knowledge, there is no precedent for this rate of vision-based mobile robot manipulation in the literature (see video at https://youtu.be/giiSYDwZM4c).

[14, 20, 23, 35, 33, 34, 49, 54, 60, 68, 67, 69]. We openly admit that our results will likely improve with FSOD algorithms that are more advanced than our initial baseline approach. While running more experiments with current and future FSOD algorithms to reduce annotation requirements and improve task performance is an area of future work, this paper currently provides a new TFOD Benchmark that makes robot-collected data and corresponding annotations publicly available for research.[1] Thus, with this paper, we are encouraging the object detection research community to join us in this effort to perform and evaluate methods in this new task-focused setting for robot manipulation, which will guide future innovation toward increasingly reliable few-shot detection for robotics applications.

[1]Dataset website: https://github.com/griffbr/TFOD

Acknowledgment. Toyota Research Institute provided funds to support this work.