Extended Abstract

Motivation The increasing adoption of robotics in service industries, particularly restaurants and hospitality, highlights a critical need for automated solutions that extend beyond traditional serving tasks. While robotic dish-serving systems are becoming more common, the challenges of automating post-meal cleanup, such as table clearing and dish sorting, remain largely unaddressed. These environments are inherently unstructured and dynamic, presenting significant hurdles for robotic automation due to the highly variable nature of items and their unpredictable arrangements. This project investigates robotic tray sorting as a foundational step toward addressing these complexities, aiming to develop a robust solution for automated object organization that can operate effectively in such challenging settings.

Method We formulate the robotic tray sorting task as a Markov Decision Process (MDP) and employ the Soft Actor-Critic (SAC) algorithm, chosen for its sample efficiency and handling of continuous action spaces crucial for precise manipulation. The robot's objective is to accurately pick and place objects into designated sorting zones within a simulated MuJoCo environment. To enhance generalization beyond static object sets, we introduce two key extensions: object feature conditioning, where observations are augmented with RGB-like feature vectors to encode appearance properties (e.g., color), allowing the policy to infer behavior for unseen instances; and task-conditioned policy learning, where a one-hot task identifier is appended to the observation space, enabling a single policy to learn various sorting behaviors (e.g., standard, partial, size-based, color-based sorting).

Implementation Our experimental setup consists of two main components: a procedural generation pipeline for diverse tray scenes and a custom Gym-compatible `TraySortingEnv` for SAC agent training and evaluation. One thousand distinct tray scenes are generated in MuJoCo, each containing five objects (plate, cup, tissue, bowl, spoon) with realistic `.stl` meshes and unique RGBA colors, randomly positioned within a fixed tray area while adhering to minimum separation constraints. Each scene includes corresponding color-coded sorting zones. The `TraySortingEnv` defines a continuous 4D action space for pick/place coordinates and gripper signals, and a high-dimensional observation space encoding object properties, held-object status, and target zone information. The dense reward structure includes bonuses for successful picks, accurate placements, fast execution, and episode completion, alongside penalties for failures and inefficiency.

Results Initial training with five *seen* objects yielded perfect performance: 100% grasp success, 100% correct zone rate, and 100% episode success with sub-centimeter placement accuracy ($\sim$0.017m). To test generalization, we conducted a zero-shot experiment by withholding the *tissue* object during training and introducing it only during evaluation. A baseline SAC agent, trained on four objects, achieved 100% grasp success but saw its correct zone rate drop to 89.8% due to misplacement of the unseen *tissue* object. However, by incorporating **object-feature conditioning** and a color-similarity reward, the agent's correct zone rate significantly improved to 95.8%, with 85% of *tissue* objects correctly placed, demonstrating strong generalization. Multi-task learning, though theoretically appealing, resulted in a degraded performance (94.1% grasp success, 86.6% correct zone rate) on the standard sorting task, suggesting task interference within this specific environment.

Conclusion This project successfully validated Soft Actor-Critic's capabilities for robotic tray sorting and, more importantly, demonstrated that integrating object-level features and a color-based reward mechanism provides significant generalization power, allowing the system to robustly handle unseen objects with a high success rate of 95.8%. While multi-task learning showed context-dependent limitations in this study, the overall framework provides a strong foundation for developing adaptable robotic systems. This research contributes to addressing the challenges of automation in unstructured service environments like post-meal cleanup. Future work will extend these methods to tackle more advanced challenges, including occlusions, ambiguous object identities, and more diverse manipulation behaviors, ultimately moving closer to fully autonomous and flexible robotic solutions.

Robotic Tray Sorting with Soft Actor-Critic and Task-Conditioned Learning

Keyuan Wu
SCPD
Stanford University
keyuanwu@stanford.edu

Abstract

This paper investigates *robotic tray sorting* as a foundational step towards automating post-meal cleanup in service industries. We employ the *Soft Actor-Critic (SAC)* algorithm in a simulated MuJoCo environment to address the challenges posed by unstructured "dirty table" scenarios. Initially, the robot achieves a 100% sorting success rate with five pre-defined objects. To improve generalization, we explore *object feature conditioning* and *multi-task learning*. By introducing an unseen object during testing, the system maintains a remarkable 95.8% success rate through object feature augmentation and a *color-based reward function*. While multi-task learning showed limited benefits in this specific context due to task overlap, the results underscore the promising potential of our approach for developing adaptable robotic systems. This work details the methodologies employed, analyzes their effectiveness, and discusses implications for future advancements in robotic manipulation within unstructured environments.

1 Introduction

The increasing adoption of robotics in service industries, particularly in restaurants and hospitality, highlights a growing need for automated solutions beyond traditional serving tasks. While robotic dish-serving systems are becoming more common, the challenges of automating post-meal cleanup, such as table clearing and dish sorting, remain largely unaddressed. These tasks are inherently complex due to the highly variable and unstructured nature of "dirty table" environments, demanding significant generalization capabilities from robotic systems. This project explores robotic tray sorting as a foundational step towards addressing this challenge, focusing on developing a robust and generalizable solution for automated object organization.

We initially approach the problem within a simplified environment: a tray containing five distinct objects, each with a predetermined sorting zone. The robot's task is to accurately pick and place each object into its designated location. For this pick-and-place task, Soft Actor-Critic (SAC) was chosen as the reinforcement learning algorithm due to its sample efficiency and ability to handle continuous action spaces, which are crucial for precise robotic manipulation. In a controlled setting where all five objects were present during the training phase, the robotic system achieved a 100% success rate in sorting.

However, real-world applications necessitate the ability to handle novel or previously unseen objects. To address this critical aspect of generalization, we investigated two primary approaches: incorporating object features and implementing multi-task learning. By introducing an unseen object during the testing phase, we observed that the success rate remained remarkably high, reaching 95.8%. This demonstrates the promising potential of our proposed methods in enabling robotic systems to adapt to variations in their environment and improve their generalization capabilities for complex sorting tasks.

This project aims to detail the methodologies employed, analyze their effectiveness, and discuss the implications for future advancements in robotic manipulation within unstructured environments.

2 Related Work

Our project builds upon foundational advancements in deep reinforcement learning and their applications in robotic manipulation. This section reviews key contributions that inform our methodology and contextualize our research.

We heavily rely on the Soft Actor-Critic (SAC) algorithm, introduced by Haarnoja et al. (2018). This model-free, off-policy deep reinforcement learning algorithm is chosen for its sample efficiency, ability to handle continuous action spaces, and entropy-regularized objective, all crucial for learning complex robotic grasping and manipulation policies (1).

Our research aligns with recent work in robotic dishware manipulation. Voysey et al. (2021) explored autonomous dishwasher loading, utilizing YOLO for object detection in a real-world setting (2). While they did not employ reinforcement learning, their work highlights the potential and challenges of vision-based robotic sorting. Our project offers a complementary perspective by employing a simulated environment and integrating SAC for policy learning. Furthermore, Nwakeze et al. (2025) and Chen et al. (2023) successfully integrated YOLO for object detection with SAC for learning grasp poses, validating our approach of combining vision and reinforcement learning (3) (4).

To achieve generalization to unseen objects, we draw inspiration from multi-task reinforcement learning with context-based representations by Sodhani, Zhang, and Pineau (2021). Their methods for learning generalizable policies across multiple related tasks through context-based representations are highly relevant to our goal of enabling our robot to handle novel object configurations by learning a more robust and adaptable policy (5).

3 Method

We formulate robotic tray sorting as a Markov Decision Process (MDP) defined by $(\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma)$, where $\mathcal{S}$ is the observation space, $\mathcal{A}$ is the continuous action space, r is the reward function, $\mathcal{P}$ defines the transition dynamics, and $\gamma \in [0, 1)$ is the discount factor. At each timestep t, the agent observes a state $s_t \in \mathcal{S}$, takes an action $a_t \in \mathcal{A}$, receives a scalar reward $r_t = r(s_t, a_t)$, and transitions to the next state s_{t+1}.

The action space $\mathcal{A} \subset \mathbb{R}^4$ consists of normalized coordinates (x, y, z) for pick/place locations, along with a gripper signal that indicates whether to grasp or release. The observation space encodes the scene layout, including object positions, object classes, sorting zone coordinates, and currently held object status.

We adopt the Soft Actor-Critic (SAC) algorithm to learn an optimal stochastic policy $\pi(a_t \mid s_t)$. SAC maximizes expected reward while encouraging high-entropy policies for better exploration. The objective is defined as:

$$J(\pi) = \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} \left[\sum_{t=0}^{T} \gamma^t \left(r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot \mid s_t)) \right) \right]$$

where $\mathcal{H}(\pi)$ is the entropy of the policy and α is a temperature coefficient that balances exploration and exploitation.

To improve generalization across object types and configurations, we introduce two key extensions:

- **Object Feature Conditioning.**
  Rather than relying solely on symbolic object class IDs, we augment object observations with RGB-like feature vectors that encode appearance properties such as color. This allows the policy to generalize to novel object instances (e.g., unseen object classes like "tissue") based on shared visual characteristics. These features are concatenated with spatial object data to form a richer observation embedding.

- **Task-Conditioned Policy Learning.**
  We extend the observation space to include a one-hot task identifier, allowing a single policy to learn multiple sorting behaviors simultaneously. These tasks include standard sorting, partial-object sorting, size-based sorting, and color-based matching. The resulting policy is conditioned on both the current state and the task context: $\pi(a_t \mid s_t, z)$, where z is the task encoding. This enables the agent to adapt its behavior based on the specified sorting objective.

The reward function is structured to promote both grasp reliability and placement accuracy. Rewards are given for successful grasps, accurate placement into the correct sorting zone (based on class or feature alignment), and episode completion. Distance-based penalties are used to encourage precise object placement, and bonuses are added for completing placements efficiently or matching zone attributes like color in color-based tasks.

Together, these methodological components support both task flexibility and generalization, enabling the robot to operate effectively in scenes that vary in object composition, spatial layout, and sorting criteria.

4 Experimental Setup

The experiment setup comprises two main components: (1) procedural generation of realistic tray scenes using MuJoCo and (2) the development of a Gym-compatible simulation environment tailored for training and evaluating Soft Actor-Critic (SAC) agents under a continuous action space and task-conditioned reward structure.

4.1 Tray Scene Generation in MuJoCo

To expose the agent to diverse learning scenarios, we procedurally generate 1000 tray scenes in the MuJoCo physics engine. Each scene contains five distinct object classes—*plate*, *cup*, *tissue*, *bowl*, and *spoon*—modeled as scaled `.stl` meshes with realistic dimensions and unique RGBA colors for visual distinction and ground-truth annotation.

Objects are randomly placed within a fixed $1.1\,\text{m} \times 0.85\,\text{m}$ tray area using rejection sampling to maintain a minimum 3 cm separation between bounding boxes. Metadata such as object name and (x, y) positions is saved in `.json` format for scene reconstruction.

Each scene includes five fixed-size, color-coded sorting zones ($0.25\,\text{m} \times 0.25\,\text{m}$) placed along the bottom edge of the tray, each mapped to one object class. A simple robot arm model with two revolute joints and a static gripper is included for reference, though not actuated.

A top-down camera captures 640×480 RGB images, accompanied by a secondary rendering with labeled bounding boxes for optional detection tasks. Each scene consists of:

- a MuJoCo XML model,
- a rendered RGB image,
- a labeled ground-truth image,
- a `.json` object placement file.

This pipeline supports randomized environment resets and diverse training scenarios during reinforcement learning. An example is shown in Figure 1.

4.2 Tray Sorting Environment for Reinforcement Learning

`TraySortingEnv` is implemented as a `gym.Env` simulating sequential pick-and-place operations, where the agent places each object into its matching color-coded zone. Each episode features up to five objects sampled from the generated scenes.

- **Action and Observation Spaces.**
  Actions are 4D continuous vectors:

$$(a_x, a_y, a_z, g), \quad a_i \in [-1, 1]$$

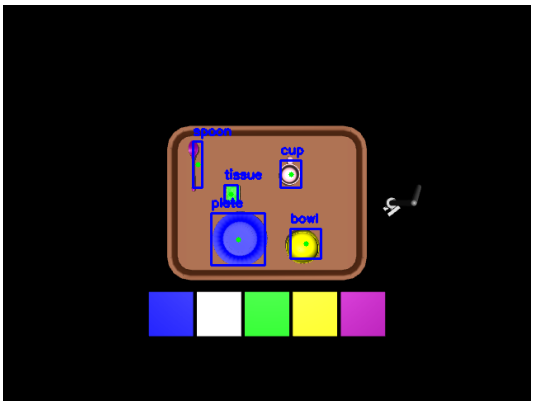

Figure 1: Annotated top-down view of a procedurally generated tray scene with ground-truth object bounding boxes.

where (a_x, a_y) are spatial coordinates, a_z is unused, and g is a grip signal: positive for pick, negative for place.

Observations include:

- Encoded positions, sizes, and one-hot class vectors for up to 5 objects,
- Held-object status,
- Sorting zone positions,
- Correct target zone for the held object.

This representation supports object-aware and task-conditioned learning.

- **Environment Dynamics.**
  At each step, the agent picks or places an object. A pick succeeds if the grip signal is positive and close to an unheld object. A place succeeds if a held object is released near its target zone.
  Success is judged using a 0.1 m distance threshold from the zone center.
- **Reward Structure.**
  A dense reward encourages accurate, efficient behavior:
  - $+0.5$ for successful pick,
  - $3.0 \cdot \exp(-1.5 \cdot d)$ for precise placement,
  - $+0.5$ bonus for correct zone placement,
  - $+0.2$ if placed within two steps of pick,
  - $+5.0$ for episode completion,
  - Penalties: -0.1 for failed pick, -0.005 per step while holding, -0.01 per step overall, and extra penalty beyond 10 steps.
- **Reset and Termination.**
  On `reset()`, a random `.json` file reconstructs object placements. Episodes end after all objects are placed or when the step limit is reached. Metrics such as grasp rate, placement accuracy, and final reward are recorded.

5 Results

5.1 Training and Evaluation Performance

Training Results

Figure 2 shows the SAC agent's training performance over 100,000 timesteps. The mean episode reward (right) steadily increases and plateaus at 25.1, near the theoretical maximum of 26.0:

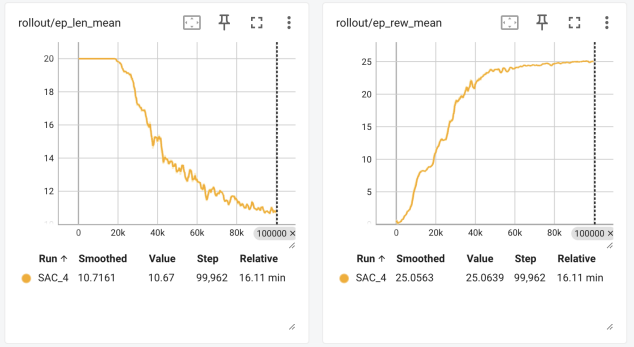

Figure 2: Training curves on 5-object scenes.

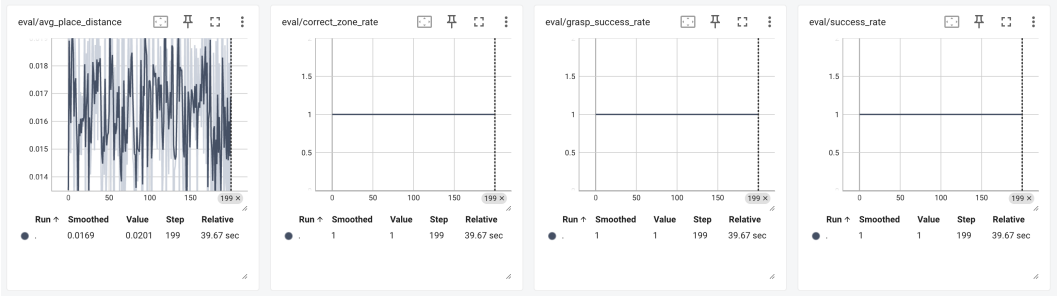

Figure 3: Evaluation curves on 5-object scenes.

Maximum Reward Breakdown:

- Per object: 0.5 (grasp) + 3.0 (placement) + 0.5 (zone bonus) + 0.2 (fast drop) = 4.2
- For 5 objects: $5 \times 4.2 = 21.0$
- Completion bonus: $+5.0$

This indicates the agent reliably completes all pick-and-place tasks with high precision. The episode length (left) drops from 20 to about 10.7 steps, close to the optimal 10 steps (2 per object), showing efficient behavior with minimal wasted actions.

Evaluation Results

The trained agent is evaluated on 200 held-out tray scenes. As shown in Figure 3, it achieves perfect performance across all metrics:

- **Grasp Success Rate:** 100%
- **Correct Zone Rate:** 100%
- **Episode Success Rate:** 100%
- **Avg. Placement Distance:** ~ 0.017 m

These results confirm the agent's ability to detect, pick, and accurately place objects into target zones with sub-centimeter precision. The lack of performance variance shows strong generalization across randomized placements of the five known object types.

This demonstrates SAC's effectiveness for structured pick-and-place tasks under consistent object and reward conditions. The following sections examine how performance holds under more challenging generalization settings.

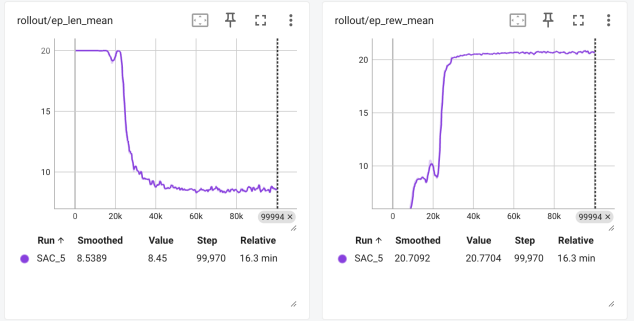

Figure 4: Training curves on 4-object scenes, baseline.

5.2 Generalization to Unseen Object

While the SAC agent trained on fixed tray configurations performs nearly perfectly, such setups are unrealistic. In practical robotic settings, agents must generalize to new object combinations—including those not seen during training. To test this, we designed a zero-shot generalization experiment by withholding the *tissue* object during training and introducing it only in evaluation.

Experimental Setup

Training Set: Tray scenes contain only four object classes—*plate*, *cup*, *bowl*, and *spoon*. The *tissue* object is completely excluded.

Testing Set: Scenes include all five objects, with *tissue* introduced as an unseen class.

The action space, reward function, and zone locations remain unchanged. The agent receives no prior knowledge about object identity and must rely solely on spatial and visual cues during evaluation.

This setup simulates a zero-shot setting where the agent must manipulate a novel object using a policy trained on only a subset of classes.

Training Results on Seen Objects

A new agent is trained exclusively on 4-object tray scenes. As shown in Figure 4:

- **Episode Reward:** The mean reward increases and plateaus around 20.8—slightly below the theoretical maximum of 21.8.
- **Episode Length:** The agent quickly converges to an average of $\sim$8.5 steps per episode. This aligns with the reduced task size (4 objects per episode).

These results confirm that the agent effectively learns to sort the seen objects and maintains strong performance even with reduced object diversity.

Baseline for Generalization

This serves as a baseline for evaluating zero-shot generalization under the following conditions:

- No retraining or fine-tuning is performed with *tissue*.
- No object feature augmentation or conditioning is used.
- Scene complexity increases, but the architecture and reward remain unchanged.

Evaluation Results

Figure 5 presents evaluation metrics:

- **Grasp Success Rate:** 100%.

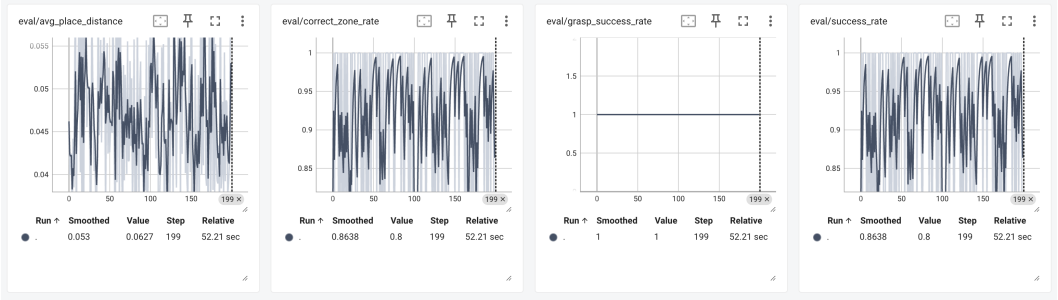

Figure 5: Evaluation curves on 4-object scenes, baseline.

- **Correct Zone Rate:** Drops to $\sim$89.8%. While the agent attempts to place all objects, it frequently misplaces the *tissue* due to the absence of a learned class-to-zone mapping.
- **Episode Success Rate:** Also $\sim$89.8%. One incorrect placement (typically *tissue*) causes the full episode to be marked incomplete.
- **Average Placement Distance:** Increases to $\sim$0.051 meters—more than 3$\times$ greater than the $\sim$0.017 meters observed in full-class training. This reflects reduced placement precision for the unseen object.
- **Tissue Object Summary:** Across 200 evaluation episodes, the agent successfully picks and places the *tissue* in every instance. However, only 57% of placements are in the correct zone. The average placement error is 0.103 meters—substantially higher than for seen classes—indicating difficulty in accurately matching the tissue to its target.

Analysis

These results show that the agent generalizes well for grasping, even with unseen objects, but fails to generalize zone assignment to the *tissue* class. This is due to the absence of semantic grounding or class-based policy logic.

This experiment highlights the limitations of vanilla RL policies trained on static object sets and motivates future enhancements such as:

- **Object-Feature Conditioning:** Integrating RGB embeddings or class attributes to inform policy decisions.
- **Task-Conditioned or Multi-Task Learning:** Training across diverse goal structures to develop more flexible and adaptive behavior.

5.3 Object-Feature Conditioning for Generalization

To address the performance degradation observed during generalization, we explore whether incorporating object-specific features can help the SAC agent infer correct placement behavior for previously unseen object classes. The central idea is to augment the observation space with semantic and visual cues, enabling the policy to generalize beyond class identity alone.

Method

We introduce two key modifications to the tray environment:

Object Feature Encoding: Each object is associated with a fixed-length feature vector, including:

- RGB color (serving as a semantic cue aligned with zone color),
- Normalized spatial features (position and size).

Reward Augmentation: The reward function includes a color similarity bonus based on the Euclidean distance between an object's RGB color and that of its target zone:

$$r_{\text{color}} = \max\left(0,\ 1 - \|c_{\text{obj}} - c_{\text{zone}}\|\right)$$

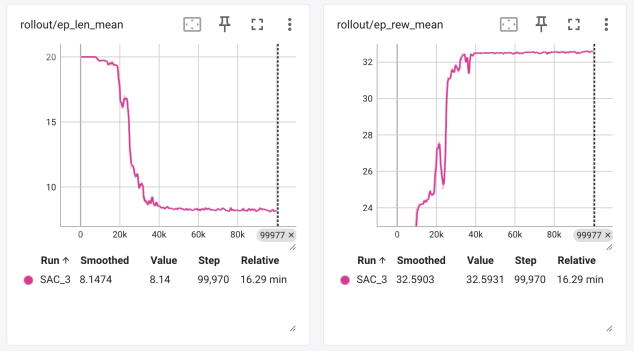

Figure 6: Training curves on 4-object scenes with object-feature conditioning.

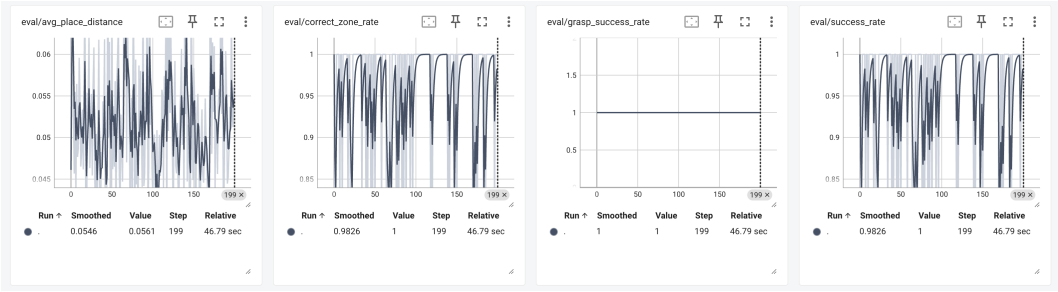

Figure 7: Evaluation curves on 4-object scenes with object-feature conditioning.

This encourages the agent to place objects near zones with visually matching attributes, even for novel object classes not encountered during training.

Training Results

As shown in Figure 6:

- **Mean Episode Reward:** Reaches approximately 32.6, approaching the theoretical maximum of 33.8. This reflects both the added reward signal and the agent's enhanced ability to leverage feature cues for correct placement.
- **Mean Episode Length:** Remains efficient, converging to approximately 8.14 steps—comparable to baseline models.

These results indicate that the agent successfully integrates visual features and color-based reward shaping to refine its policy and improve generalization.

Evaluation Results

Figure 7 presents evaluation metrics(trained with object features, zone features, and color similarity reward):

- **Grasp Success Rate:** Maintains 100%.
- **Correct Zone Rate:** Improves to 95.8% —a significant gain over baseline's 89.8%. This confirms that color-based reward and zone features enable the agent to generalize class-zone associations.
- **Episode Success Rate:** Also reaches 95.8%.
- **Average Placement Distance:** Slightly higher at $\sim$0.067 meters (vs. 0.051 in baseline), but still within the correct zone threshold.

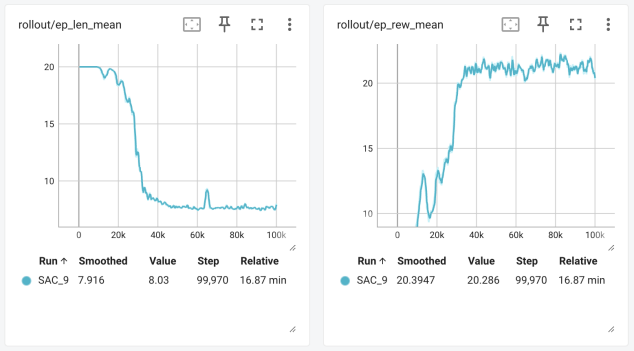

Figure 8: Training curves on 4-object scenes with multi-task policy learning.

- **Tissue Object Summary:** The tissue object is picked and placed in every episode. Placement in the correct zone improves to 85%, and the average distance to the correct zone is reduced to 0.067 meters.

Analysis

These results confirm that integrating object features, zone color input, and color-aware reward shaping provides the strong generalization ability—allowing the agent to infer correct behavior for previously unseen classes like tissue.

5.4 Task-Conditioned Learning

To further explore generalization in robotic sorting, we trained a multi-task SAC agent in a shared environment encompassing four distinct task variants:

- **Task 0 – Standard Sorting:** Place all objects into their class-specific zones.
- **Task 1 – Partial Sorting:** Randomly select 3 out of 5 objects for sorting.
- **Task 2 – Size-Based Sorting:** Assign zones based on ascending object size.
- **Task 3 – Color-Based Sorting:** Match objects to the zone with the most similar RGB color.

Each episode includes a one-hot task ID appended to the observation space, allowing the policy to condition on the active task. This setup enables investigation into whether a shared multi-task policy can:

- Learn transferable skills (e.g., grasping, transport),
- Differentiate tasks based on embeddings,
- Improve sample efficiency or policy robustness through cross-task transfer.

Training Results

As shown in Figure 8, the multi-task SAC agent exhibited stable convergence and strong performance across all task types:

- The overall reward curve converged around 20.3, and mean episode length dropped from 20 to $\sim$8 steps.
- Task-specific success rates for Tasks 0–3 approached or reached 1.0.
- Learning was balanced across tasks, indicating that the shared policy effectively generalized to all task configurations.

These outcomes suggest that the multi-task agent successfully learned both general grasp-and-place behaviors and task-specific placement logic.

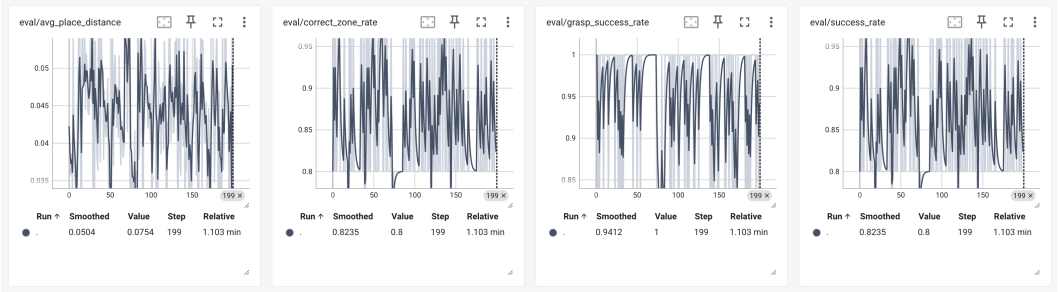

Figure 9: Evaluation curves on 4-object scenes with multi-task policy learning.

Evaluation Results on Task 0

Despite strong training performance, evaluation on **Task 0** (Standard Sorting) revealed that the multi-task model underperforms relative to its single-task counterpart:

Figure 9 presents evaluation metrics:

- **Grasp Success Rate:** Drops to 94.1%, below the baseline (100%). This suggests some degradation in the agent's ability to consistently pick up objects.
- **Correct Zone Rate:** Drops to 86.6% — a notable decrease from the baseline's 89.8%. This confirms that the multi-task agent struggled more with placing objects into their designated zones, likely due to shared policy interference across tasks.
- **Episode Success Rate:** Also drops to 86.6%..
- **Average Placement Distance:** $\sim$0.043 meters (vs. 0.052 meters in baseline). While numerically lower, this can be misleading — many failed placements for the tissue object were not counted toward correct placements, which artificially lowers this metric.
- **Tissue Object Summary:** The tissue object was picked and placed only 70% of the time. Correct zone placement was just 35%, a significant decline compared to the baseline (57%). The average distance to the correct zone increased to 0.108 meters, indicating difficulty in generalizing to this unseen object.

These results suggest that while the multi-task policy learned Task 0 reasonably well, performance was hindered—likely due to task interference or reduced specialization from shared capacity.

Analysis

Multi-Task RL Is Suboptimal in This Setting Although multi-task reinforcement learning is promising in theory, this environment presents challenges that limit its effectiveness:

- **Low Inter-Task Diversity:** All tasks share the same object set, tray geometry, and manipulation mechanics. Differences arise mainly in zone assignment logic or object count, reducing the need for distinct skill acquisition.
- **Minimal Task Conflict:** Unlike classical multi-objective RL, the tasks here do not pose conflicting objectives. Most share similar grasping and motion strategies, with modest variations in placement.
- **Shared Optimal Policy Structure:** The optimal behaviors—grasping, transporting, and placing—are largely consistent across tasks, limiting the benefit of distinct task-specific representations.
- **No Gains in Sample Efficiency:** Each task is simple enough to be solved efficiently by a single-task SAC agent. Multi-tasking yields little improvement in convergence speed or robustness.

Metric	Single-task (5 seen objects)	Single-task (4 seen objects) baseline	Single-task (4 seen objects) Object Features	Multi-task (4 seen objects)
Grasp Success Rate	1.000	1.000	1.000	0.941
Correct Zone Rate	1.000	0.898	0.958	0.866
Success Rate	1.000	0.898	0.958	0.866
Avg Placement Dist. (m)	0.017	0.051	0.067	0.043

Table 1: Evaluation metrics across different training configurations: single-task with all seen objects, single-task baseline with domain shift, single-task with object feature conditioning, and multi-task reinforcement learning.

Evaluation Metrics Summary

6 Discussion

This study explored robotic tray sorting using a Soft Actor-Critic (SAC) agent in a simulated MuJoCo environment, with an emphasis on generalization and task flexibility. Our initial single-task SAC agent achieved excellent performance when trained and tested on a fixed set of five object classes. However, this setup was highly idealized and lacked the variability typically encountered in real-world applications, such as novel objects or domain shifts.

To introduce a more realistic challenge, we simulated a domain shift by training on only four object classes and testing on all five, including a previously unseen *tissue* object. Under this condition, the baseline SAC model retained a high grasp success rate (100%) but showed decreased accuracy in correct zone placement and placement precision for the novel object (89.8% correct zone rate, 0.051 m average placement distance). This highlights the limitation of vanilla RL policies when faced with novel class-to-zone assignments without semantic grounding.

To address this generalization gap, we implemented object-feature conditioning. By augmenting the observation space with object-level RGB feature vectors, adding zone color encodings, and modifying the reward function to include a color similarity bonus, we substantially improved performance. The agent achieved near-perfect generalization (95.8% correct zone rate) across all metrics, demonstrating the effectiveness of rich feature inputs for adaptive policy learning.

We also explored a multi-task reinforcement learning setup by defining four distinct sorting tasks and training a shared SAC policy conditioned on task identifiers. While theoretically promising for skill transfer, this approach underperformed in our environment. The tasks, despite differing in sorting criteria, shared similar object sets, tray layouts, and manipulation mechanics. This low inter-task diversity limited the effectiveness of task-specific representations. As a result, the multi-task agent underperformed on the standard sorting task (Task 0) compared to the single-task baseline (86.6% vs. 89.8% correct zone rate), and showed weaker generalization to the tissue object (35% vs. 57% correct placements). These findings suggest that in low-diversity task settings, a specialized single-task policy may outperform shared multi-task policies due to reduced interference and greater task focus.

7 Conclusion

This project successfully demonstrated the robust application of Soft Actor-Critic (SAC) for robotic tray sorting in simulated environments, achieving 100% task success with known object classes. Importantly, we showed that incorporating object-level features and a color-based reward significantly improves generalization, enabling a 95.8% success rate even when handling previously unseen objects. This approach directly addresses the core challenge of variability in unstructured environments like post-meal cleanup.

While multi-task learning showed limited benefit in this specific domain due to task overlap, the underlying framework remains promising for more complex manipulation tasks. This research provides a solid foundation for developing adaptable robotic systems. Future work will extend these methods to scenarios involving occlusions, ambiguous object identities, and a broader set of manipulation behaviors such as stacking, grouping, or discarding.

8 Team Contributions

- **Group Member:** Keyuan Wu

Changes from Proposal The original project proposed using YOLO for object detection from RGB input, emulating real-world perception. However, MuJoCo provides perfect ground-truth object positions, making YOLO unnecessary for simulation-based training.

Preliminary experiments confirmed that using YOLO significantly slowed training due to preprocessing overhead and introduced noisy observations that degraded learning. As a result, the YOLO module was excluded from this milestone.

The initial plan also included tactile sensing for grasp feedback. Due to MuJoCo's limitations with STL-based contact modeling, tactile feedback was omitted.

References

[1] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. (2018). Soft Actor-Critic Algorithms and Applications. *arXiv preprint arXiv:1812.05905*. https://doi.org/10.48550/arXiv.1812.05905.

[2] Isobel Voysey, Thomas George Thuruthel, and Fumiya Iida. (2021). Autonomous Dishwasher Loading from Cluttered Trays Using Pre-Trained Deep Neural Networks. *Engineering Reports*, 3:e12321. https://doi.org/10.1002/eng2.12321.

[3] Osita Miracle Nwakeze, Ogochukwu C. Okeke, and Ike Joseph Mgbemfulike. (2025). Intelligent Robotic Object Grasping System Using Computer Vision and Deep Reinforcement Learning Techniques. *International Journal of Science and Research Archive*, 14(03), 511–521. https://doi.org/10.30574/ijsra.2025.14.3.0693.

[4] Ya-Ling Chen, Yan-Rou Cai, and Ming-Yang Cheng. (2023). Vision-Based Robotic Object Grasping—A Deep Reinforcement Learning Approach. *Machines*, 11(2), 275. https://doi.org/10.3390/machines11020275.

[5] Shagun Sodhani, Jian Zhang, and Joelle Pineau. (2021). Multi-Task Reinforcement Learning with Context-based Representations. *arXiv preprint arXiv:2102.06177*. https://doi.org/10.48550/arXiv.2102.06177.