DrakeSort: Autonomous Robotic Trash Sorting via VLM Reasoning and Dynamic Throwing

Grace Yuan
Massachusetts Institute of Technology
yuangc@mit.edu

Celinda Zhu
Massachusetts Institute of Technology
celindaz@mit.edu

***Abstract*—Sorting recyclable waste is a critical yet labor-intensive task that poses significant challenges for robotic systems due to the unstructured nature of waste streams and the immense variability in object materials. This project presents a simulation-based robotic sorting system developed in Drake that automates the identification, grasping, and placement of recyclables from a mixed pile. Our perception pipeline integrates YOLO11 for object localization and evaluates several Vision-Language Models (VLMs) for zero-shot material classification, ultimately utilizing GPT-4o for its superior semantic reasoning and ability to generalize to previously unseen items. For manipulation, we combine multi-view point cloud fusion for geometry-aware grasp planning with a dynamic throwing primitive based on analytical ballistic modeling. Experiments demonstrate that the system can successfully classify, grasp, and throw diverse objects into appropriate bins. We also identify key limitations, including occasional post-release arm instability and object slippage during high-acceleration motions. Overall, the results highlight the potential of coupling VLM-driven perception with dynamic manipulation as a promising framework for robotic waste sorting in unstructured environments.**

I. Introduction

The global waste crisis is escalating at an unprecedented rate, with annual production exceeding 2 billion metric tons while U.S. recycling rates remain below 24% [1]. This inefficiency is largely driven by reliance on hazardous, error-prone manual sorting. Consequently, developing high-speed autonomous robotic sorting is a critical industrial need.

However, the biggest challenge for automating this task is the unstructured nature of waste streams, where objects arrive in random configurations. Furthermore, the immense variety of packaging materials creates a "long-tail" problem: while common items are easy to recognize, robots frequently encounter novel or rare packaging designs that do not exist in standard training datasets. Traditional computer vision approaches, which rely on Convolutional Neural Networks (CNNs) trained on fixed categories, are brittle in this context; they often fail to generalize to these unseen objects or distinguish between visually similar but materially different items [2].

We hypothesize that the integration of Visual Language Models (VLMs) into the robotic perception loop can overcome the limitations of fixed-class detectors. To validate this, we benchmarked our pipeline against state-of-the-art open-weights models, specifically Qwen3-VL [11] and SmolVLM [12], finding that GPT-4o offered the most consistent zero-shot material reasoning for our dataset. Unlike traditional classifiers, LMMs like GPT-4o possess broad semantic knowledge and zero-shot reasoning capabilities, allowing a robot to categorize items based on inherent properties rather than mere pattern matching. To address the physical bottleneck of manipulation speed, we further propose a dynamic placement strategy. Rather than traversing the full path to a bin, the system calculates ballistic trajectories to "toss" objects, theoretically decoupling cycle time from the size of the workspace.

This project aims to develop a simulation-based prototype in Drake that achieves three concrete design goals:

1) **Zero-Shot Generalization:** Accurately classify YCB objects into recycling categories (like Metal and Paper) without specific model fine-tuning.
2) **Geometry-Aware Grasping:** Successfully generate grasp poses for multiple objects placed in close proximity by using multi-view point cloud fusion to resolve spatial geometry.
3) **Dynamic Placement:** Demonstrate a reduction in transport time per object via dynamic tossing compared to standard pick-and-place routines.

We implemented a pipeline combining YOLO for localization, GPT-4o for zero-shot classification, and multi-view point cloud fusion for grasping. Results from simulated multi-object trials demonstrate the system's ability to accurately identify diverse waste items and execute efficient ballistic placements, validating the feasibility of VLM-driven sorting.

II. Related Work

A. Vision-Based Classification

The domain of waste classification has historically been dominated by supervised deep learning. Early works, such as Thung and Yang [3], demonstrated that CNNs could classify cropped single-object images with reasonable success. While establishing a baseline, these methods were limited to 2D classification without spatial context. To enable robotic interaction, subsequent approaches adopted real-time object detectors. While widely adopted models like YOLOv5 [4] provide robust bounding boxes, they often include background noise in the depth estimation.

In this work, we utilize the state-of-the-art Ultralytics YOLO11 [5], specifically the instance segmentation variant (`yolo11-seg`). Unlike standard bounding-box detectors,

Special thanks to our supportive TA Sunshine Jiang and Prof Tomas Lozano-Perez

YOLO11 produces pixel-wise masks for each object. This allows us to filter the depth map precisely—projecting only the object's surface into 3D space while discarding background pixels—resulting in significantly cleaner point clouds for grasp planning.

B. Semantic Reasoning and Open-Vocabulary Robotics

However, while models like YOLO11 have achieved real-time detection rates, these "closed-set" classifiers suffer from a critical limitation: they cannot handle the infinite variability of real-world trash and cannot recognize new trash items outside of their training set. To address this, our work aligns with the emerging paradigm of "Open-Vocabulary Robotics". Recent studies have utilized models like CLIP [6] or VLM-based planners [7] to allow robots to interpret natural language commands and recognize novel objects. We apply this specifically to the sorting domain. This approach offers the distinct benefit of semantic reasoning, enabling the system to generalize to novel objects and infer recycling categories based on implicit material properties rather than rigid visual patterns. Commonly adopted models in this domain include GPT-4o [10], Qwen3-VL [11], and SmolVLM [12], which we will evaluate later.

C. Manipulation and Dynamic Placement

Grasping in cluttered environments has been extensively studied, with learning-based systems such as DexNet [8] demonstrating strong robustness through large-scale synthetic training. However, these approaches introduce significant computational and data requirements that are unnecessary for our setting, where grasping primarily serves as a precursor to the throwing action. We therefore adopt a lightweight geometric heuristic to generate grasp poses directly from segmented point clouds. Although simpler than deep models, this approach is fast, deterministic, and sufficiently reliable for the purposes of our dynamic placement pipeline.

Our dynamic throwing component is inspired by prior work, most notably TossingBot [9]. TossingBot uses residual physics learning to refine its throws, enabling high accuracy but requiring substantial model training and engineering overhead. For this project, such complexity would be impractical and would obscure the interpretability of the throwing dynamics. Instead, we implement a simplified analytic formulation based on classical kinematics and ballistic modeling. This choice provides a transparent and computationally efficient alternative that still captures the key benefits of dynamic placement. Overall, these design decisions prioritize modularity, speed, and clarity, making our approach well suited to a simulation-based investigation of open-world waste sorting.

III. Approach

A. Simulation Environment

We developed a high-fidelity simulation in Drake to mimic a recycling sorting station. The workspace is centered around a Kuka LBR IIWA 7 (7-DOF) robotic arm equipped with a Schunk WSG 50 parallel gripper.

To simulate a waste stream, we arranged three elevated pedestals (0.25m $\times$ 0.25m $\times$ 0.30m) linearly along the robot's workspace ($x = 0.25$m). These stands hold a diverse set of YCB test objects—specifically the cracker box, sugar box, and tomato soup can—representing common recyclable materials (paper, cardboard, and metal). A collection bin is positioned at the periphery ($x = 0.6$m, $y = 0.3$m) to serve as the target drop zone for sorted items. This structured layout allows us to isolate perception challenges from occlusion complexities while rigorously testing the system's grasp planning and classification accuracy.

B. Perception Pipeline

Our perception system is designed to translate raw RGB-D sensor data into semantically labeled, actionable 3D grasp targets. As shown in Fig. 1, the pipeline operates in three stages: object localization via instance segmentation, zero-shot semantic classification using Vision-Language Models (VLMs), and multi-view point cloud fusion.

1) Multi-View Sensing Setup: The workspace is monitored by three simulated RGB-D cameras within the Drake environment. To minimize occlusion and maximize grasp accessibility, we position the cameras in a triangular configuration: a primary overhead camera (X_{WC}^{top}) provides a global view, while two oblique cameras ($X_{WC}^{side}, X_{WC}^{left}$) capture lateral geometric features.

2) Instance Segmentation and Localization: The first stage of the pipeline isolates objects from the background. We utilize `YOLO11-seg` (Instance Segmentation) rather than standard bounding box detection. For each incoming RGB frame, the model predicts a binary mask $M \in \{0,1\}^{H \times W}$ for each object. The use of pixel-wise masks is critical for our application; it allows us to filter the depth map D to exclude table surfaces

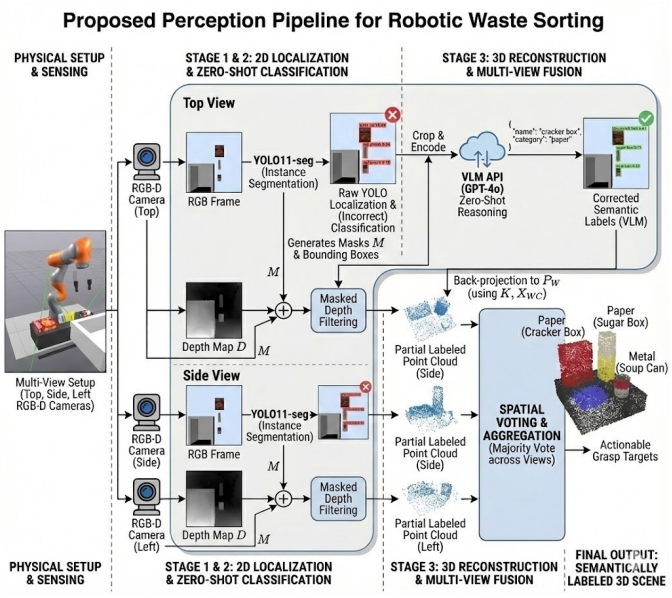


Fig. 1. The perception pipeline

and background noise, ensuring that only object-specific depth values are projected into 3D space.

3) Zero-Shot Semantic Classification: As demonstrated in Fig. 2, standard object detectors often fail to generalize to the specific domain of recyclable waste. While YOLO successfully generates accurate segmentation masks for localization, its pre-trained classification head produces irrelevant or hallucinated labels (e.g., identifying a cracker box as a "traffic light"). To overcome this limitation, we implement a two-stage pipeline that offloads semantic reasoning to a Vision-Language Model (VLM).

Upon detecting a valid object mask, the system extracts a high-resolution RGB crop of the item using the bounding box coordinates. This crop is encoded and transmitted to the LMM via API, where the model utilizes its zero-shot capabilities to analyze the object's visual properties—reading text and identifying material textures. We enforce a structured output by prompting the model to return a strict JSON object containing a descriptive name (e.g., "cracker box") and a recycling category (e.g., "paper"). This approach allows the system to handle the "open-set" nature of waste streams without requiring extensive fine-tuning for every new packaging design.

While we integrated and tested multiple open-weight models including Qwen3-VL [11] and SmolVLM [12], our primary classification pipeline utilizes **GPT-4o** [10] due to its robust adherence to JSON formatting and superior reasoning on low-resolution simulation textures. A detailed performance comparison of these models is presented in Section IV.

4) 3D Reconstruction and Multi-View Fusion: To generate grasp candidates, we transform the 2D perception data into the world frame. For every pixel (u, v) in the validated mask M, we back-project the corresponding depth value $z = D(v, u)$ into a 3D point P_W using the camera intrinsics K and pose X_{WC}:

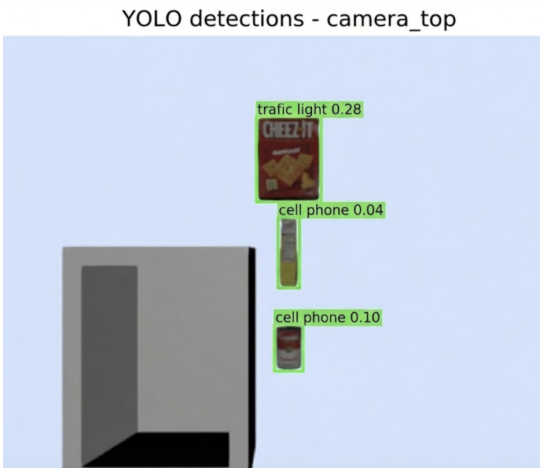


Fig. 2. YOLO Localization and Classification

$$P_W = X_{WC} \cdot \left(zK^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right) \tag{1}$$

This process yields a partial point cloud for each camera view. To resolve spatial inconsistencies and occlusions, we implement a **spatial voting** algorithm. Point clouds from all three camera views are aggregated and snapped to predefined workspace slots, as shown in Fig. 3. We apply a majority voting scheme to fuse semantic labels across views—for example, if the top camera detects "paper" but the side and left cameras detect "plastic," the object is definitively assigned the label "plastic." This redundancy significantly reduces classification noise caused by specular reflections or ambiguous viewing angles.

C. Grasp Generation

We generate valid gripper poses (X_G) via geometric sampling on the object mesh generated from the YOLO-localized point cloud.

The process employs antipodal sampling to identify stable contact points:

- **Ray Casting:** We sample random points (p_1) on the mesh surface and cast rays along their inverted normal vectors ($-n_1$). Intersection points (p_2) on the opposite side of the mesh form candidate pairs.
- **Filtering:** Candidate pairs are rigorously filtered based on three criteria:
  1) *Antipodality:* The surface normals at p_1 and p_2 must be approximately collinear to ensure force closure.
  2) *Gripper Width:* The distance between points must fall within the physical stroke limits of the IIWA gripper.
  3) *Collision:* The resulting grasp pose is checked against the environment to ensure the gripper fingers do not penetrate the table plane.
- **Selection:** From the set of valid candidates, we select the grasp with an approach vector closest to the global vertical ($-z$). This heuristic prioritizes top-down picks, which generally maximize kinematic reachability and minimize collision risks with neighboring clutter.

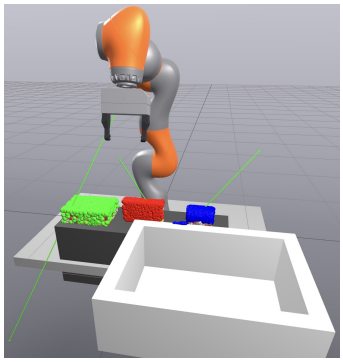


Fig. 3. The resultant 3D point clouds after multi-view fusion and spatial voting

D. Throwing

To extend the manipulation capabilities of the IIWA arm beyond simple pick-and-place operations, we implemented a dynamic throwing primitive. This approach leverages the robot's ability to generate high end-effector velocities to launch objects into a target bin that lies outside the immediate workspace of the pick location.

1) Trajectory Design and Keyframes: The throwing motion is modeled as a sequence of discrete keyframes where, unlike quasi-static pick-and-place operations, precise timing is critical for momentum generation. The sequence transitions through the following poses and is shown in Fig. 4:

- **`throw_start`:** The robot retracts to a "wind-up" pose, orienting the gripper horizontally.
- **`throw_back` (Optional):** An additional retraction pose to increase the acceleration path for targets requiring higher velocities.
- **`throw_mid`:** The target pose for the acceleration phase. The robot moves linearly from the start pose to this pose at high velocity.
- **`throw_end`:** The release pose. Immediately upon reaching the coordinate of *throw_mid* (0.01 seconds), the gripper orientation rotates 90° upward while simultaneously opening. This directs the object forward and slightly upward.

The full keyframe sequence is defined as follows:

```
keyframes = [
    (X_WGinitial, opened),
    (X_WG_pre_pick, opened),
    (X_WG_pick, opened),
    (X_WG_pick, closed),
    (X_WG_throw_start, closed),
    (X_WG_throw_back, closed), # Optional
    (X_WG_throw_mid, closed),
    (X_WG_throw_end, opened),
    (X_WGinitial, opened)
]
```

2) Adaptive Ballistic Planning: A core challenge in robotic throwing is determining the required release velocity and timing to hit a specific target. Instead of hardcoding velocities, we developed an analytical inverse-ballistics formulation to adaptively calculate the optimal throw parameters based on the target location.

We simplify the trajectory generation by treating the throw as a 2D planar problem projected onto the vertical plane connecting the start and target poses. Given a target location p_{target}, we calculate the necessary release state $(p_{release}, v_{release})$ and the time Δt required to achieve it.

1) **Optimal Geometric Parameters:** We first determine the release pose *throw_mid* and the acceleration distance (d_{throw}) adaptively. To ensure the robot operates within its reachable workspace while maximizing power, we constrain the throw distance based on the target distance:

$$d_{throw} = \text{clip}(0.4 \times ||p_{target} - p_{start}||, 0.25, 0.6) \quad (2)$$

The upper bound of 0.6m was selected to restrict the retraction motion to the safe workspace of the IIWA arm, preventing kinematic singularities or self-collisions during the wind-up phase. To ensure a consistent ballistic trajectory, we fix the launch angle θ relative to the throw distance, typically setting the lift height $h_{lift} \approx 0.7 \times d_{throw}$, resulting in a release angle of approximately 35°.

2) **Ballistic Velocity Calculation:** Treating the object as a point mass under gravity, the required release velocity v_{req} to span horizontal distance x and vertical drop z is derived from the projectile motion equation:

$$v_{req} = \frac{x}{\cos\theta\sqrt{\frac{2(x\tan\theta - z)}{g}}} \quad (3)$$

3) **Heuristic "Snap" Adjustments:** Pure physical calculations often underestimate the required velocity due to air resistance, friction during release, and motor tracking errors. We apply a "Force Snap" heuristic:
   - Minimum Velocity Clamp: We enforce a minimum velocity floor (e.g., 2.5 m/s) to ensure the motion remains in the dynamic "throwing" regime rather than a slow "handover."
   - Overkill Factor: We scale the calculated velocity by a safety factor (e.g., $1.2\times$) to compensate for system damping.

4) **Time-Based Trajectory Generation:** Since the trajectory planner accepts time-stamped waypoints, we convert the required velocity back into a duration Δt:

$$\Delta t_{swing} = \frac{d_{throw}}{v_{req}} \quad (4)$$

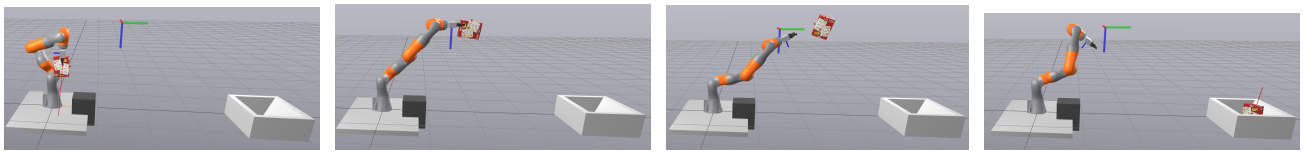
IV. Evaluation and Discussion

A. Perception Pipeline Evaluation

To validate our hypothesis that VLM reasoning is essential for waste sorting, we conducted an ablation study comparing four perception configurations: a baseline YOLO11-seg model, and three VLM-augmented pipelines using SmolVLM, Qwen3-VL-235B-A22B, and GPT-4o. We tested these models on a fixed set of YCB objects (Sugar Box, Cracker Box, Tomato Soup Can) under identical lighting conditions. The results, visualized in Fig. 5, reveal significantly different failure modes for each architecture.

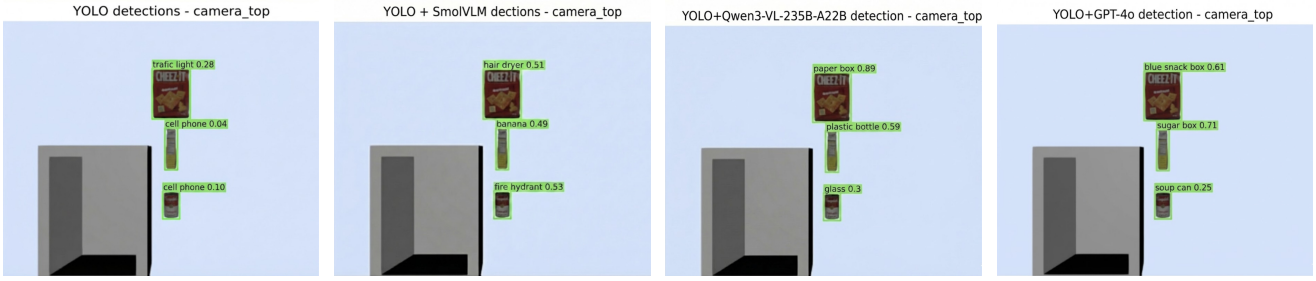
1) Baseline Failure (YOLO Only): The standalone YOLO11 model failed to provide usable semantic labels. As seen in Fig. 5(a), the model relied heavily on simple shape priors, misclassifying the rectangular sugar box as a "cell phone." This confirms that models trained on general datasets (COCO) cannot generalize to waste sorting without extensive retraining.

2) Small VLM Hallucinations (SmolVLM): Surprisingly, the lightweight SmolVLM (Fig. 5(b)) performed worse than the baseline in terms of semantic coherence. Likely driven by the object's yellow color, it hallucinated a "banana" for the sugar box. This suggests that smaller Vision-Language Models



throw_start **throw_mid** **throw_end** **throw_initial**

Fig. 4. Throwing Trajectory Design



(a) YOLO11 Baseline *Label: "Cell Phone"* *Result: Fail*

(b) YOLO + SmolVLM *Label: "Banana"* *Result: Hallucination*

(c) YOLO + Qwen3-VL *Label: "Plastic Bottle"* *Result: Material Error*

(d) YOLO + GPT-4o *Label: "Sugar Box"* *Result: Success*

Fig. 5. **Qualitative Comparison of Perception Pipelines.** We tested four varying levels of semantic intelligence on the same target object (YCB Cracker Box). (a) The baseline YOLO model hallucinates labels based on color blobs. (b) Lightweight VLMs struggle with specific OCR. (c) Qwen3-VL correctly identifies the material but lacks descriptive specificity. (d) GPT-4o successfully reads the text "Cheez-It" and infers the material context.

may suffer from severe grounding issues when presented with low-resolution simulation crops.

3) Material Inference Failure (Qwen3-VL): The Qwen3-VL pipeline (Fig. 5(c)) demonstrated the risks of "hallucinated reasoning." While it correctly identified the cracker box, it misclassified the cardboard sugar box as a "plastic bottle." This is a critical failure for recycling, as placing cardboard into a plastic stream contaminates the batch. This inconsistency highlights that mid-sized open-weight models still lack the robustness required for strict material separation.

4) Semantic Superiority (GPT-4o): The GPT-4o pipeline (Fig. 5(d)) achieved the highest sorting success rate. It successfully performed OCR to read the text "Domino Sugar" and correctly labeled the item as a "sugar box." This semantic classification allows the robot to confidently map the item to the paper/cardboard recycling stream, validating that large-scale commercial models offer the necessary world knowledge to handle the "open-set" diversity of trash.

B. Throwing Evaluation

To assess the robustness and generalizability of our dynamic throwing primitive, we designed a two-stage evaluation protocol. We tested the system using three YCB objects with distinct inertial and geometric properties: the cracker box(baseline), sugar box(small/dense), and tomato soup can (cylindrical).

1) Evaluation Protocol: We define success based on two sequential criteria:

Retention Success (S_1): The robot successfully retains the object during the high-acceleration phase from *throw_start* to *throw_mid*. A failure here indicates slippage or grasp failure due to excessive inertial forces.

Target Accuracy (S_2): Given S_1 is met, the object lands and stays as close to the target position.

2) Evaluation and Results:

Baseline Throwing Performance. With the target bin fixed at $(2.0, 0.0, 0.2)$ (world frame; we fixed z=0.2 for all experiments), the IIWA generated a smooth swing trajectory that reliably delivered the cracker box forward into the bin. The throw remained stable across repeated trials, confirming that the nominal trajectory design is sufficient when the target lies directly forward and within a favorable region of the robot's workspace.

Multi-Direction Throw Experiments. We evaluated the throwing formula across five directions—left, right, forward, left-forward, and right-forward—at multiple distances from 0.0 to 2.0 in each dimension to assess overall performance. Targets beyond approximately $(2.0, 2.0)$ were excluded because they fall outside the IIWA's reachable workspace, and backward throws were omitted due to unstable behavior. Across all feasible settings, objects generally landed close to the target bin, though some failed to drop inside, typically bouncing off the bin edge or falling slightly short. Even among successful throws, certain trials did not pass through the exact *throw_mid* waypoint, likely due to workspace limitations that prevent the robot from reaching the ideal intermediate pose. The forward direction produced the most reliable performance, achieving successful throws in all tested distances, while left-forward and right-forward also yielded consistent results. In contrast, pure left and right throws at short range (e.g., 0.5 m) failed, which aligns with expectations since objects

begin too close to the robot, making the required joint motion infeasible. To investigate how parameters behave for each case, in Table I, a summary table reports the optimized horizontal distance from *throw_start* to *throw_mid* (*opt_dist*), optimized vertical lift (*opt_lift*) from *throw_start* to *throw_mid*, throwing velocity v_{throw}, and swing duration T_{swing}. These values show a clear trend: both *opt_dist* and *opt_lift* increase with target distance, accompanied by a corresponding increase in required throw speed.

Varying Throw_Start Location. We further examined how modifying the *throw_start* position influences the resulting *throw_mid* waypoint, the IIWA joint configuration, and the final target accuracy. Although our baseline fixes *throw_start* at $y = 0$, targets located to the left or right raise the question of whether shifting the starting pose could improve performance. For a leftward target at $(1.8, 1)$, we compared *throw_start* values of $y = 0$ and $y = 0.25$, and found that $y = 0$ yielded slightly better accuracy, likely because it placed the robot farther from the object and closer to a favorable throwing arc. For a symmetric rightward target at $(1.8, -1)$, comparing $y = 0$ and $y = -0.25$, both settings produced similar performance. Although throwing to the right appears more kinematically natural for the IIWA, our results suggest that maintaining *throw_start* at $y = 0$ generally offers a more stable and comfortable configuration for generating the required swing motion.

Adding Throw_Back. To investigate whether introducing a backward wind-up could generate additional momentum and extend the throwing distance, we tested a modified configuration that preserves the baseline setup while adding a *throw_back* motion for 0.3 seconds, implemented as a backward rotation of $\pi/3$ with the same translation as *throw_start*. Although we tuned the relevant parameters in an attempt to make the backward wind-up behave as intended, incorporating this motion generally reduced performance: objects were more likely to slip from the gripper during the wind-up, resulting in noticeably shorter throwing distances. Moreover, decreasing the duration of the *throw_back* phase increased the likelihood of slippage, indicating that rapid backward motion amplifies grasp instability. Overall, the throw-back strategy did not improve throwing range and ultimately remained ineffective due to persistent object slip.

Different Objects. Using the fixed baseline parameters, we further tested the throwing performance on additional YCB objects, including the sugar box and soup can, in comparison to the original cracker box. All three objects were successfully thrown into the bin, demonstrating that the baseline controller generalizes across varied object geometries. However, we observed consistent differences in landing positions: the soup can tended to fall slightly short, the sugar box reached marginally farther, and the cracker box achieved the longest distance. This ordering may relate to differences in mass, volume, or aerodynamic properties.

TABLE I
OPTIMIZED THROW PARAMETERS ACROSS TARGET LOCATIONS

(x,y)	opt_dist (m)	opt_lift (m)	v_throw (m/s)	T_swing (s)
Left				
(0.0, 0.5)	0.250	0.175	0.839	0.3636
(0.0, 1.0)	0.410	0.287	1.545	0.3238
(0.0, 1.5)	0.600	0.420	2.034	0.3601
(0.0, 2.0)	0.600	0.420	2.823	0.2594
Right				
(0.0, -0.5)	0.250	0.175	0.734	0.4157
(0.0, -1.0)	0.394	0.276	1.503	0.3199
(0.0, -1.5)	0.593	0.415	1.979	0.3660
(0.0, -2.0)	0.600	0.420	2.765	0.2649
Forward				
(0.5, 0.0)	0.250	0.175	0.479	0.6366
(1.0, 0.0)	0.360	0.252	1.412	0.3114
(1.5, 0.0)	0.560	0.392	1.906	0.3587
(2.0, 0.0)	0.600	0.420	2.642	0.2772
Left Forward				
(0.5, 0.5)	0.262	0.184	1.122	0.2854
(1.0, 1.0)	0.544	0.381	1.870	0.3552
(1.5, 1.5)	0.600	0.420	2.886	0.2537
(2.0, 2.0)	0.600	0.420	3.807	0.1924
Right Forward				
(0.5, -0.5)	0.250	0.175	1.082	0.2821
(1.0, -1.0)	0.532	0.373	1.843	0.3525
(1.5, -1.5)	0.600	0.420	2.844	0.2575
(2.0, -2.0)	0.600	0.420	3.773	0.1941

3) Limitations: Through experimentation, we identified several limitations.

Post-Release Arm Instability. In some trials, the IIWA entered a twisted or unstable configuration immediately after releasing the object and was unable to return to its initial pose. This behavior is likely related to the pseudo-inverse–based controller, which can produce large joint velocities or unintended null-space motions when the arm is near kinematic singularities. As a result, the current system cannot reliably support sequential multi-object throwing within a single simulation.

Limited Reachability of Throw_Mid. Although we cap the maximum swing magnitude at 0.6 to respect the IIWA's reach, the robot still sometimes fails to reach the specified *throw_mid* pose, reducing accuracy for certain targets. Future improvements may involve tuning the release angle, further restricting the swing bound, or optimizing the tradeoff between kinematic feasibility and maximum reachable throw range.

Object-Dependent Variability. Different objects introduce additional uncertainty due to variations in mass, geometry, contact area, and how securely they can be grasped. Objects with weaker or partial grasps are especially prone to slippage during high-speed motions, which significantly alters the resulting trajectory and is difficult to control with the current setup.

V. Conclusion

In this work, we developed an integrated simulation framework for autonomous waste sorting that combines open-world perception with a dynamic throwing primitive for long-range object placement. Our results demonstrate that both components, semantic understanding of diverse waste items and high-speed manipulation beyond the robot's nominal reach, are essential for scalable, efficient recycling systems.

From the perception perspective, we found that conventional object detectors struggle with the heterogeneity of real-world waste streams. While lightweight vision-language models provide limited improvements, large foundation models such as GPT-4o reliably infer material type from broader visual and contextual cues. This suggests that offloading high-level semantic reasoning to powerful generative models is a promising strategy for handling the "long-tail" distribution of waste categories, though such models introduce notable latency and cost considerations.

On the manipulation side, our analytical ballistic formulation enabled the IIWA arm to execute accurate, high-speed throws across multiple directions and target distances. We observed consistent performance with several object geometries and identified key factors influencing success, including reachable workspace constraints, grasp stability, and release timing. However, the system also exhibited limitations: pseudo-inverse control occasionally caused post-release arm instability, throw_mid poses were not always reachable for distant targets, and object-dependent slippage remained a significant failure mode.

Overall, our study highlights the feasibility of coupling foundation-model-based perception with dynamic robotic throwing to expand the operational envelope of automated recycling. Future work will focus on improving grasp robustness, enhancing controller stability, and reducing reliance on computationally expensive perception models, ultimately bringing the system closer to real-time deployment in industrial settings.

References

[1] Columbia Climate School, "How AI Is Revolutionizing the Recycling Industry," *State of the Planet*, Jun. 18, 2025. [Online]. Available: https://news.climate.columbia.edu/2025/06/18/how-ai-is-revolutionizing-the-recycling-industry/.

[2] H. Abdu and M. H. M. Noor, "A Survey on Waste Detection and Classification Using Deep Learning," *IEEE Access*, vol. 10, pp. 128152–128167, 2022.

[3] G. Thung and M. Yang, "Classification of Trash for Recyclability Status," Stanford University, CS229 Project Report, 2016.

[4] R. Khanam and M. Hussain, "What is YOLOv5: A deep look into the internal features of the popular object detector," *arXiv preprint arXiv:2407.20892*, 2024.

[5] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO11," *Ultralytics*, Sep. 2024. [Online]. Available: https://github.com/ultralytics/ultralytics.

[6] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and Where Pathways for Robotic Manipulation," in *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[7] A. Brohan et al., "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control," in *Proceedings of the 7th Conference on Robot Learning (CoRL)*, 2023.

[8] J. Mahler et al., "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics," in *Proceedings of Robotics: Science and Systems (RSS)*, 2017.

[9] A. Zeng et al., "TossingBot: Learning to Throw Arbitrary Objects with Residual Physics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1307–1319, 2020.

[10] OpenAI, "GPT-4o System Card," *OpenAI*, Oct. 2024. [Online]. Available: https://openai.com/index/gpt-4o-system-card/.

[11] Qwen Team, "Qwen3-VL-235B: A Frontier Multimodal Mixture-of-Experts Model," *Alibaba Cloud*, Oct. 2025. [Online]. Available: https://github.com/QwenLM/Qwen3-VL.

[12] A. Marafioti *et al.*, "SmolVLM: Redefining small and efficient multimodal models," *arXiv preprint arXiv:2504.05299*, 2025.