Enhancing Robotic Vision through Deep Learning Techniques: From Detection to Construction

Yusuf Aykin[1], **Hans Sachs**[1], **Nikolai Gerzen**[1]

[1]Technische Hochschule Ostwestfalen-Lippe, Germany

yusuf.aykin@th-owl.de, hans.sachs@th-owl.de, nikolai.gerzen@th-owl.de

Abstract -

This paper presents a robotic system developed to enhance automation in construction workflows through advanced AI and computer vision technologies. The system integrates a robotic arm with a 3D point cloud camera and state-of-the-art 2D pre-trained Deep Learning models, such as GroundingDINO and SegmentAnything, to detect and segment construction elements in 3D dynamic, unstructured environments. By processing point cloud data from the camera and aligning it with real-world coordinates, the system achieves precise object localization, enabling tasks such as element handling and assembly. Designed to address challenges like clutter, occlusion, and variability in construction sites, this system bridges the gap between controlled laboratory conditions and real-world applications. Experimental evaluations highlight its potential to improve efficiency and adaptability in construction tasks.

Keywords -

Construction Automation; Robotic Vision; Object Detection; 3D Vision; Deep Learning

1 Introduction

The integration of artificial intelligence (AI) with robotics offers transformative potential for the construction industry. Construction workflows, requiring precision and efficiency, are well-suited for robotic automation, which can address critical challenges like labor shortages, high costs, and safety risks by automating repetitive tasks such as element handling, assembly, and inspection [1]. However, deploying robotics effectively in the chaotic and unpredictable environments of real construction sites remains a significant hurdle.

The transition from controlled laboratory settings to dynamic construction sites presents formidable challenges. Real-world construction environments are inherently complex, characterized by clutter, inconsistent lighting, diverse materials, and unforeseen obstacles. Unlike controlled experiments, these sites involve overlapping objects, occlusions, and constantly changing conditions that can severely limit the effectiveness of conventional robotic systems. While robotic automation has advanced, adapting these systems to the visual complexities of real-world construction remains a critical gap.

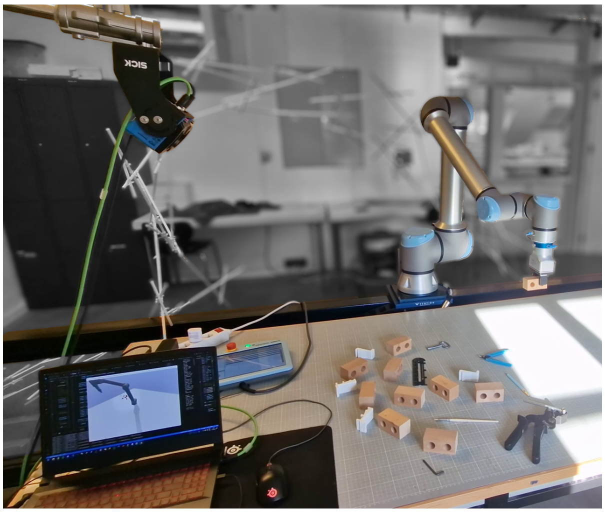

Figure 1. Robotic environment setup for object manipulation in a construction scenario. The system integrates a UR10e robotic arm and a Visionary-S point cloud scanner.

Vision is paramount for robotic automation in these environments. Although 2D image recognition models have advanced significantly [2], they often lack the spatial understanding essential for robots in complex 3D spaces. Trained primarily for 2D analysis, they struggle with depth perception and spatial relationships crucial for manipulation and assembly in cluttered scenes. In contrast, 3D vision systems, capable of capturing detailed spatial data through point clouds, offer a more comprehensive understanding of the environment, enabling robots to perceive and interact with objects more effectively, even under challenging conditions. Our work seeks to leverage the power of these advanced 2D image recognition models and apply them within a 3D context. Figure 1 illustrates our system setup.

Recent AI breakthroughs in object detection and segmentation have bolstered robotic vision. Models such as GroundingDINO and SegmentAnything leverage pre-trained deep learning techniques to detect and segment unseen objects with high accuracy, even in cluttered, complex environments [3, 2]. These models use transformer networks to process image data, achieving high accuracy and robustness. Additionally, real-time advancements like YOLO-World [4] and EfficientSAM [5] make them suitable for dynamic environments. Extensions of such models for 3D point cloud processing [6] enable robust frameworks for detecting, segmenting, and manipulating objects under real-world constraints. However, directly applying these powerful 2D models within a comprehensive 3D robotic perception system for construction remains largely unexplored.

The integration of robotics and AI in construction has seen significant progress in object detection, manipulation, and automation of tasks. Early methods relied on feature-based approaches such as SIFT and HOG descriptors [7], which struggled with clutter and occlusions. The introduction of deep learning architectures, including transformer-based models, revolutionized object detection and segmentation [8]. Joint vision-language models like GroundingDINO and SegmentAnything demonstrated remarkable performance, enhancing applications in cluttered construction settings. Several successful applications of these techniques in construction-related tasks include ROI-based detection systems for equipment [9] and advanced point cloud segmentation using networks like PointRCNN [10]. Despite these gains, a cohesive framework effectively using 2D AI models within a 3D robotic system for construction tasks is still needed.

Robotic manipulation in construction settings also requires robust planning and control mechanisms. Recent research expanded into reinforcement learning and AI-based control strategies, as shown in [1]. Grasp planning with point clouds, advanced by tools like PointNetGPD [11], and tactile feedback integration [12] have enabled precise manipulation in real-world scenarios. Transformer-based architectures, such as RT-1 and RT-2 [13], have further shown promising results in robotic manipulation through vision-language understanding. However, realizing the potential of these manipulation advances in unstructured construction requires enhanced 3D perception, which our work addresses.

3D vision systems play a vital role in enabling robots to understand and interact with their surroundings. Point cloud data provides detailed spatial information for accurate object localization and manipulation. Enhanced registration techniques, such as Colored ICP [6], incorporate color information valuable in construction scenes. Recent depth estimation methods, such as Depth Anything V2 [14], and fusion techniques like BEVFusion [15], continue to advance 3D perception capabilities. Building on these, our system aims for a more integrated and practical approach for construction robotics by leveraging both 2D AI models and 3D point cloud data.

This paper presents a novel system designed to bridge the gap between controlled lab settings and the challenges of real construction sites. Our core novelty is the effective application of state-of-the-art 2D image recognition models within a 3D point cloud framework. This integration enhances robotic automation in construction by combining 2D AI object recognition with 3D spatial understanding. By equipping a robotic arm with a cloud scanner and advanced object detection and segmentation algorithms, the system achieves:

- Accurate detection and segmentation of unseen construction elements and tools using AI-based pretrained models, originally designed for 2D images, but adapted for 3D point cloud understanding, even in cluttered scenes.
- Precise alignment of 3D data with real-world coordinates for effective robotic manipulation.

The experimental setup, shown in Figure 1, demonstrates the system's ability to adapt to the complexities of real-world construction environments. This work is an initial step in a controlled lab setting mimicking construction scenarios. While simplified, our experiments demonstrate the feasibility and potential of our approach to tackle visual challenges in real-world construction. Future work will validate and extend these findings in more complex field conditions. By addressing challenges such as cluttered scenes, occlusions, and dynamic conditions, the proposed system aims to enhance efficiency and adaptability in construction workflows.

2 System Overview

This section provides a comprehensive overview of the hardware and software components used in the proposed robotic system, focusing on its adaptability to dynamic construction tasks. The integration of advanced AI models with a collaborative robotic arm for efficient execution of operations such as element handling, assembly, and inspection.

2.1 Hardware Setup

Our experimental setup, designed to mimic real-world construction scenarios, comprises a UR10e robotic arm, an OnRobot 2FG7 gripper, a Visionary-S point cloud scanner, and a high-performance workstation. The workspace is set on a 1000 mm x 700 mm table where construction elements are placed.

The UR10e (Universal Robots) offers a 10 kg payload capacity, 1300 mm reach, and ±0.03 mm repeatability, making it suitable for tasks like manipulating construction elements and assembling components. Its collaborative

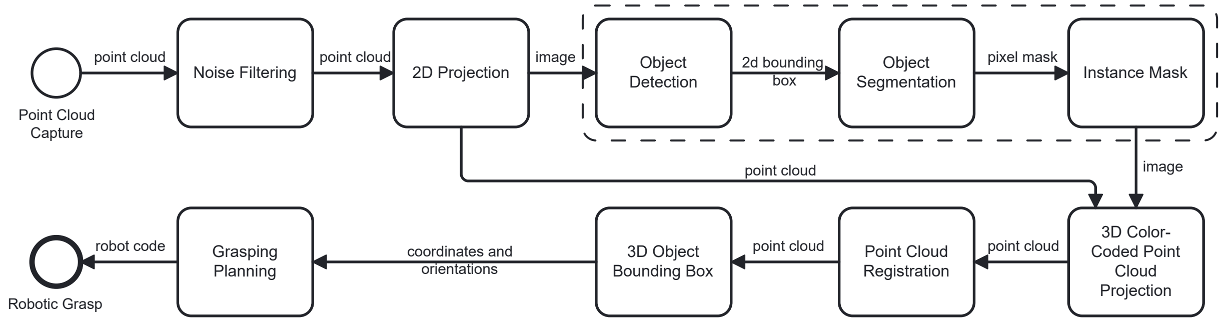

Figure 2. Flowchart of the process: integrating AI-based object detection, point cloud processing, and robotic manipulation.

design, including force sensing, enables safe human-robot interaction. Attached to the UR10e, the OnRobot 2FG7 two-finger gripper provides adjustable gripping force and width for handling various elements, such as bricks and tools.

The SICK Visionary-S point cloud scanner captures high-resolution 3D data of the environment. It boasts a pixel resolution of 640 × 512, depth accuracy of $\leq 0.25\,\mathrm{mm}$, a detection angle of $60° \times 50°$, and operates within a camera distance of 0.5 to 65 meters. Long distance capture allows us to operate in large construction environments. The scanner is positioned at one end of the workspace to capture the scene from a distance of approximately 1 meter from the table. To handle the computational demands of the AI models and point cloud processing, a high-performance workstation is used. This workstation is equipped with an NVIDIA RTX 3070 GPU, an Intel i7-12700H processor, and 64 GB of RAM, providing the necessary processing power for real-time operations.

2.2 Software Architecture

The software architecture integrates advanced AI models with point cloud processing and robotic control to enable seamless operation. The Visionary-S scanner captures 3D point clouds and RGB images, which undergo preprocessing to enhance data quality. Techniques such as noise filtering using DBSCAN and voxel downsampling reduce the computational load while preserving geometric details, ensuring the accuracy and reliability of the data for subsequent stages.

AI models are employed to process the preprocessed data for object detection and segmentation. These models run within Docker containers to provide consistent and isolated environments. GroundingDINO and SegmentAnything models form the backbone of the detection and segmentation pipeline. The processed data, including the detected objects and their segmentation masks, is then used to calculate target positions and orientations for robotic manipulation. The robot executes these commands through a Python-based control system, which translates high-level instructions into precise joint movements.

2.3 AI Models

The system relies on advanced AI models, GroundingDINO and SegmentAnything, to identify and segment construction elements and components accurately. These models are optimized to handle the diverse and cluttered nature of construction environments to provide reliable performance in a wide range of tasks.

GroundingDINO is used for object detection in 2D images derived from the 3D point cloud. Using its language-guided detection capabilities, the model can identify objects based on natural language prompts, such as “brick” or “wooden block.” This adaptability allows the system to handle a wide variety of construction elements. GroundingDINO generates precise 2D bounding boxes for detected objects, even in complex and cluttered environments, as compared with YOLO-World [4] in Figure 3.

SegmentAnything complements GroundingDINO by providing detailed instance masks for the detected objects. The versatility of the model allows it to segment a diverse range of objects without requiring additional training, which makes it suitable for dynamic construction scenarios. The high-quality instance masks generated by SegmentAnything allow the robot to distinguish between overlapping objects and perform precise manipulations.

3 Methodology

The proposed system integrates advanced AI models with robotic control to achieve accurate detection, segmentation, and manipulation of objects in dynamic construction environments. This section outlines the methodological framework shown in Figure 2 for data acquisition, preprocessing, point cloud registration, AI-driven detection and segmentation, object localization, grasp planning,

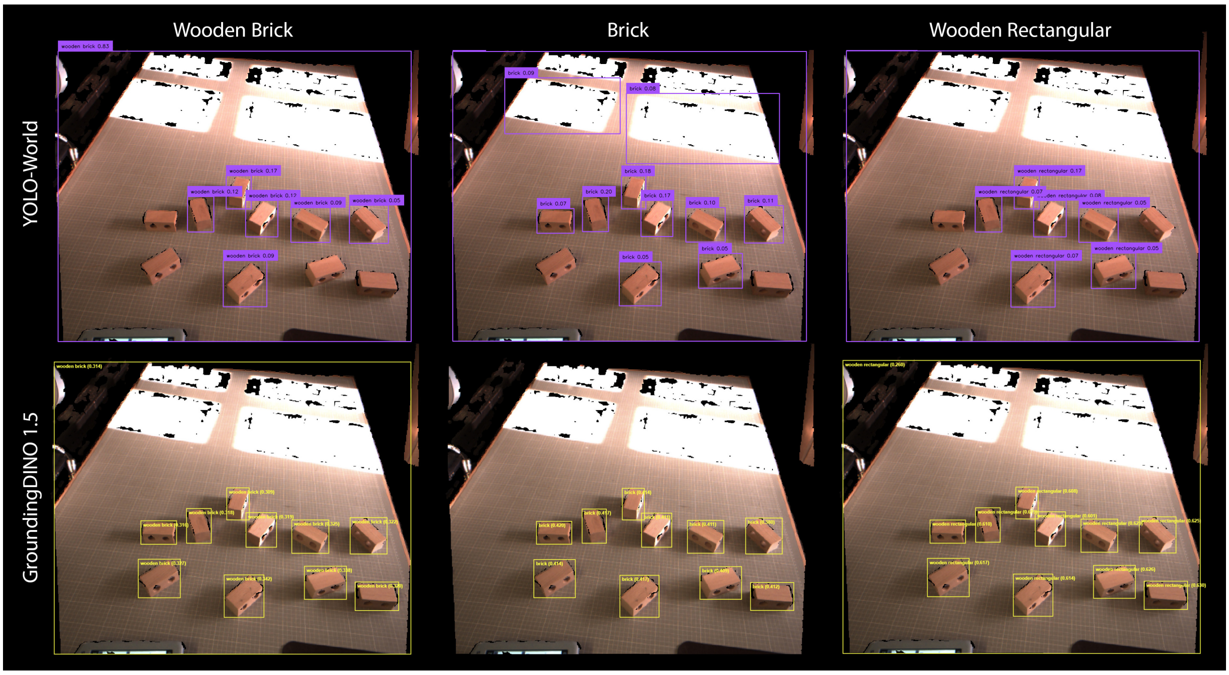

Figure 3. Comparison between GroundingDINO and YOLO-World with different prompts.

and robotic execution.

3.1 Data Acquisition and Preprocessing

The data acquisition process begins with the Visionary-S point cloud scanner capturing 3D data of the construction workspace. The scanner provides a high-resolution depth map and corresponding RGB image. Noise filtering is applied to the raw point cloud $P = \{p_1, p_2, ..., p_n\}$ using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm. DBSCAN classifies points into core, border, and noise points based on their local density. The ϵ-neighborhood of a point p is defined as:

$$N_\epsilon(p) = \{q \in P \mid \mathrm{dist}(p, q) \le \epsilon\} \tag{1}$$

where $\mathrm{dist}(p, q)$ is the Euclidean distance between points p and q, and ϵ is a predefined radius. A point p is a core point if its ϵ-neighborhood contains at least minPts points:

$$|N_\epsilon(p)| \ge \mathrm{minPts} \tag{2}$$

A point q is a border point if it is within the ϵ-neighborhood of a core point p but is not a core point itself. A point that is neither a core point nor a border point is classified as a noise point. The filtered point cloud P' consists of core and border points, excluding noise points. The filtered point cloud is then projected onto a 2D image plane using the intrinsic parameters of the Visionary-S scanner. The camera intrinsic matrix K is defined as:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where f_x and f_y are the focal lengths in the x and y directions, and (c_x, c_y) is the principal point. For a point $p = (x, y, z)$ in the 3D point cloud, its projection $p' = (u, v)$ onto the 2D image plane is given by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix}, \quad u = f_x \frac{x}{z} + c_x, \quad v = f_y \frac{y}{z} + c_y \tag{4}$$

This projection provides image format required for object detection. (Figure 4)

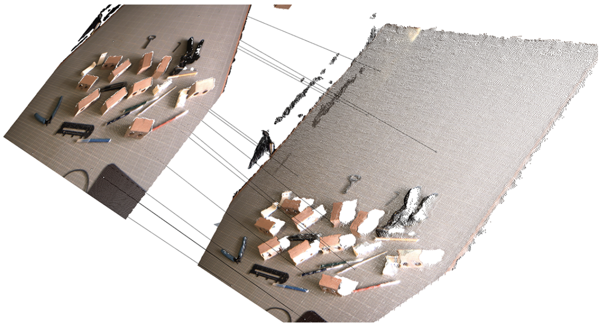

Figure 4. Projection from 3d pointcloud to 2d image.

3.2 Point Cloud Registration

Point cloud registration aligns the captured point cloud with a predefined reference frame, enabling accurate localization of objects within the workspace. The Iterative Closest Point (ICP) algorithm is used to refine the alignment between the source point cloud (captured data) and the target point cloud (reference frame). The ICP algorithm iteratively minimizes the difference between two point clouds. Let P_s be the source point cloud and P_t be the target point cloud. The algorithm aims to find the optimal transformation T that aligns P_s with P_t. The algorithm starts with an initial transformation T_0. For each point $p_i \in P_s$, the closest point $q_i \in P_t$ is found. This forms a set of corresponding pairs $C = \{(p_i, q_i)\}$. The transformation T_k that minimizes the mean squared error between the corresponding pairs is computed:

$$E(T) = \frac{1}{N}\sum_{i=1}^{N} ||q_i - T_k p_i||^2 \tag{5}$$

where N is the number of corresponding pairs. The optimal transformation can be found using Singular Value Decomposition (SVD). The source point cloud P_s is transformed using T_k:

$$P'_s = \{T_k p \mid p \in P_s\} \tag{6}$$

These steps are repeated until convergence. For improved registration accuracy the Colored ICP variant is used. Colored ICP incorporates color information in addition to spatial coordinates. The error function for Colored ICP is modified to include a color difference term:

$$E(T) = \frac{1}{N}\sum_{i=1}^{N} \left(||q_i - T_k p_i||^2 + \lambda ||c(q_i) - c(T_k p_i)||^2 \right) \tag{7}$$

where $c(p_i)$ and $c(q_i)$ are the color vectors of the corresponding points, and λ is a weighting factor that balances the geometric and color terms.

3.3 AI-Driven Detection and Segmentation

The preprocessed and registered data is fed into a pipeline comprising GroundingDINO and SegmentAnything (SAM) models. GroundingDINO processes the 2D image projection of the registered point cloud, generating bounding boxes for objects based on language prompts. The output of GroundingDINO is a set of bounding boxes $B = \{b_1, b_2, ..., b_n\}$, where each bounding box b_i is represented by its coordinates $(x_{min}, y_{min}, x_{max}, y_{max})$ and a confidence score s_i. The bounding boxes generated by GroundingDINO are refined using Non-Maximum Suppression (NMS) to eliminate overlapping and redundant detections. The Intersection over Union (IoU) between two bounding boxes b_i and b_j is computed as:

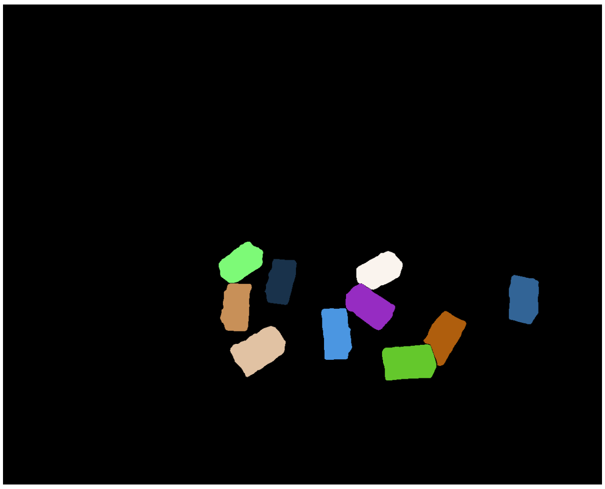

Figure 5. Segmentation mask created with SAM.

$$\text{IoU}(b_i, b_j) = \frac{\text{Area}(b_i \cap b_j)}{\text{Area}(b_i \cup b_j)} \tag{8}$$

If the IoU between two boxes exceeds a predefined threshold, the box with the lower confidence score is suppressed. The refined bounding boxes are passed to the SegmentAnything model, which generates precise instance masks for each detected object (Figure 5). The output of SAM is a set of instance masks $M = \{m_1, m_2, ..., m_n\}$, where each mask m_i corresponds to a detected object and has the same dimensions as the input image. The 2D segmentation masks are mapped back onto the 3D registered point cloud shown in Figure 6. For each point $p = (x, y, z)$ in the point cloud, its corresponding pixel (u, v) in the 2D image is determined using the projection equations derived from the camera intrinsic matrix K. The value of the segmentation mask $m_i(u, v)$ at that pixel is then assigned to the point p as shown in equation (4).

3.4 Object Localization and Grasp Planning

After detection and segmentation, localized objects are analyzed for spatial attributes using Principal Component Analysis (PCA). For a segmented point cloud P_i, the covariance matrix C is computed to determine its principal axes:

$$C = \frac{1}{n-1}\sum_{i=1}^{n}(p_i - \bar{p})(p_i - \bar{p})^T \tag{9}$$

where n is the number of points, p_i is a point, and $\bar{p}$ is the centroid. The eigenvectors of C represent the principal axes, forming the basis for the Oriented Bounding Box (OBB). The extents of the OBB are defined by projecting

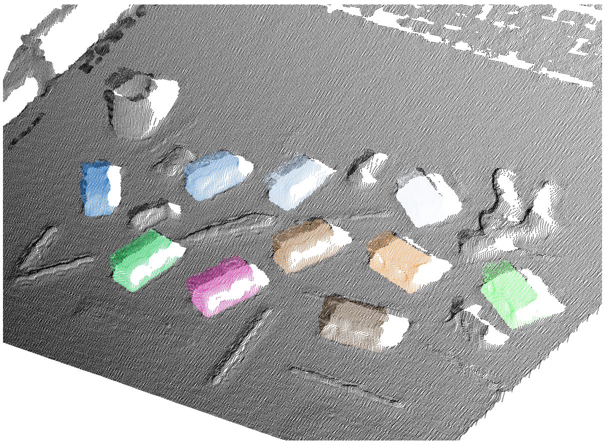

Figure 6. Mapped segmentation mask.

P_i onto the principal axes. Gripping points are selected by evaluating the longest edges of the OBB, with midpoints prioritized as candidate gripping lines:

$$m = \frac{p_j + p_k}{2} \tag{10}$$

where p_j and p_k are endpoints of the edge. Suitability of gripping points is based on gripper dimensions and workspace accessibility. Figure 7 illustrates the OBB and selected gripping locations.

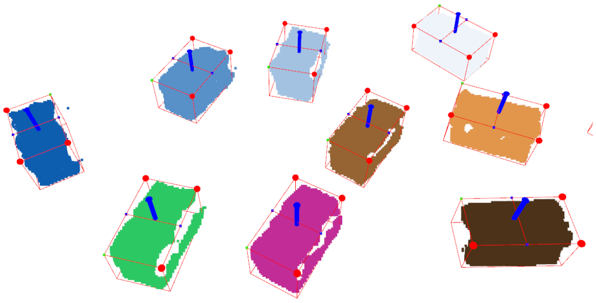

Figure 7. Calculated OBB and grasping locations.

3.5 Robotic Execution

The gripping and placement actions are translated into joint commands for the UR10e robotic arm. Inverse kinematics (IK) computes the joint angles θ required to achieve the desired end-effector pose $\tilde{T}$, ensuring smooth, collision-free movements. The desired pose $\tilde{T}$ is derived from the calculated gripping points and desired placement locations. The relationship between joint angles and end-effector pose is given by:

$$FK(\theta) = \tilde{T} \tag{11}$$

where $FK(\theta)$ represents the forward kinematics function of the robot, mapping joint angles θ to the end-effector pose.

Trajectory planning generates intermediate waypoints between the initial and target poses, optimizing for speed and safety. Smooth joint-space trajectories are interpolated using cubic splines:

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{12}$$

where $\theta(t)$ is the joint angle at time t, and a_0, a_1, a_2, a_3 are the spline coefficients.

The trajectory is validated in a PyBullet simulation and converted into robot-specific motion and gripper control commands. These commands are executed by the UR10e robot to perform the desired tasks, such as picking up a detected object and placing it at the target location.

4 Experimental Evaluation and Results

4.1 Experimental Setup

The experiments were carried out in the setup described in the System Overview section 2, representing typical construction scenarios. A total of 150 test scenarios were executed, with varying object arrangements of items like wooden blocks, cables and other common construction items, and environmental conditions to evaluate the robustness and adaptability of the system.

4.2 Evaluation Metrics

The performance of the system was evaluated using the following metrics:

- **Detection Accuracy:** The percentage of correctly detected objects compared to the ground truth, measured for both bounding boxes and segmentation masks.
- **Segmentation Precision:** The Intersection over Union (IoU) between predicted and ground-truth masks, averaged across all detected objects.
- **Manipulation Success Rate:** The percentage of successful grasp-and-place operations out of the total attempts.
- **Processing Time:** The average time required for data acquisition, detection, segmentation, and command generation.

4.3 Results

The system achieved high performance across all evaluated metrics, demonstrating its robustness and adaptability

Table 1. Performance Metrics of the Robotic System

Metric	Average Value
Detection Accuracy	95.8%
Segmentation Precision (IoU)	89.6%
Manipulation Success Rate	93.2%
Average Processing Time	6.3 s

to diverse scenarios. The results, averaged over the 150 test scenarios, are summarized in Table 1.

The integrated use of GroundingDINO and SegmentAnything resulted in an average detection accuracy of 95.8% and an average segmentation precision (IoU) of 89.6%. These results demonstrate the system's ability to accurately detect and segment objects even in cluttered environments with varying object arrangements. Figure 8 illustrates the detection and segmentation outputs for a sample scene, highlighting the precision of bounding boxes and instance masks generated by the AI models.

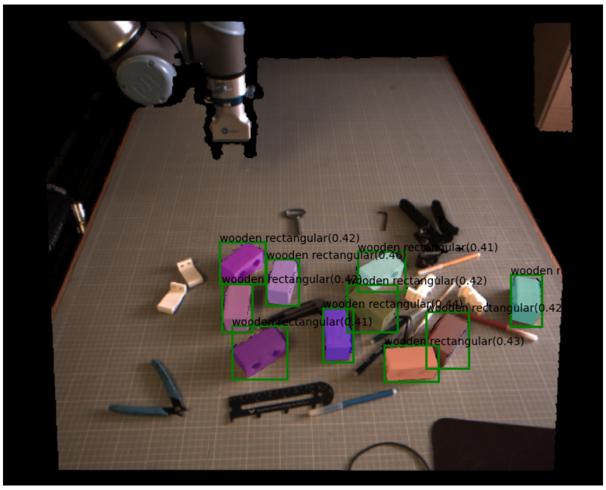

Figure 8. Sample outputs showing object detection and segmentation results.

The robotic arm successfully completed 93.2% of the grasp-and-place tasks across the 150 test scenarios. This high success rate demonstrates the system's capability to handle objects with varying shapes, sizes, and orientations. The few instances of failure were primarily attributed to limitations in the accuracy of the point cloud data for objects located further away from the scanner or those with highly reflective surfaces. These inaccuracies occasionally led to slight errors in the generated bounding boxes, making grasping more challenging.

The average processing time from data acquisition to command execution was 6.3 seconds. This includes 2.1 seconds for point cloud preprocessing and registration, 1.5 seconds for detection and segmentation using GroundingDINO 1.0 and SAM, and 2.7 seconds for object localization, grasp planning, trajectory generation, and command execution. Experiments were also conducted with GroundingDINO 1.5, which resulted in a slightly higher processing time of 9.5 seconds. The increased time is attributed to its API-based object detection approach, which added latency. However, GroundingDINO 1.5 provided a marginal improvement in detection precision (+2.5%).

4.4 Discussion

The lab evaluation demonstrates the system's potential for construction robotics, achieving high accuracy in detection and segmentation using 2D AI models in a 3D context. However, the controlled lab environment differs significantly from complex real-world construction sites, which involve dynamic lighting, dust, and larger scales. Sensor limitations were observed with reflective surfaces and distant objects, suggesting lab performance may not directly translate to real-world scenarios. Processing time, while acceptable for lab demonstrations, could limit real-time applications in dynamic environments. Validation metrics are encouraging for lab experiments, but failure analysis highlights areas for improvement. Manipulation failures were mainly due to perception inaccuracies from degraded point clouds, leading to grasping errors. Even with accurate detection, subtle 3D localization errors impacted grasping, emphasizing manipulation's dependence on robust perception in challenging conditions. Qualitatively, compared to methods like PointNetGPD, [11], RT-2 [13], PointRCNN [10] our 2D AI-in-3D approach, using pre-trained GroundingDINO and SAM, offers a balance of generalization and performance. It leverages 2D vision advancements but involves a pipeline that may introduce latency. Quantitative comparisons are needed to validate our approach against state-of-the-art methods. The successful manipulation rate in the lab confirms perception-to-action feasibility in a controlled setting.

5 Conclusion and Future Works

This work demonstrates a robotic system for object detection, segmentation, and manipulation in a controlled lab, showing promise for construction automation. Leveraging 2D AI models like GroundingDINO [3] and SegmentAnything [2] for 3D vision was effective in simulations, and manipulation success validated grasp planning strategies under these conditions. Future efforts will prioritize robustness, versatility, and real-time performance for real-world deployment. Rigorous virtual testing using game engines and datasets is crucial to assess robustness and bridge the gap to real-world validation. Future research should generalize AI models to diverse construction elements and tools, exploring adaptation strategies.

Improving robustness requires better sensing solutions, including sensor fusion and advanced point cloud processing, to address challenges like reflective surfaces and distance. Future directions include integrating advanced depth estimation models like Depth Anything V2 [14], exploring tactile sensing [16] for dexterous manipulation, and optimizing efficiency for real-time performance. Expanding to multi-robot collaboration using multi-RGBD camera setups [6], addressing occlusion, and investigating intuitive human-robot interaction via AR [17], NLP, and voice commands using reasoning LLMs like OpenAI's o1 [18] are also key. Furthermore, continual learning will be explored for autonomous adaptation. Building upon the findings of this research, our next step involves implementing the system on a larger KUKA robot and deploying it in a real construction site environment to further evaluate its performance and address real-world challenges. Pursuing these paths aims to create a highly adaptable and efficient robotic solution to enhance construction productivity, safety, and flexibility, enabling broader robotic automation and transforming building practices.

References

[1] Sulabh Kumra, Shirin Joshi, and Ferat Sahin. Antipodal robotic grasping using generative residual convolutional neural network. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9626–9633, 2020. doi:10.1109/IROS45743.2020.9340777.

[2] Alexander Kirillov, Eric Mintun, Nikhila Ravi, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023. doi:10.48550/arXiv.2304.02643.

[3] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 38–55, 2024. doi:10.1007/978-3-031-72970-6_3.

[4] Tianheng Cheng, Lin Song, Yixiao Ge, et al. Yolo-world: Real-time open-vocabulary object detection. *arXiv preprint arXiv:2401.17270*, 2024. doi:10.48550/arXiv.2401.17270.

[5] Yunyang Xiong, Bala Varadarajan, Lemeng Wu, et al. Efficientsam: Leveraged masked image pretraining for efficient segment anything. *arXiv preprint arXiv:2312.00863*, 2023. doi:10.48550/arXiv.2312.00863.

[6] Yunhan Yang, Xiaoyang Wu, Tong He, et al. Sam3d: Segment anything in 3d scenes, 2023.

[7] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2nd edition, 2022. doi:10.1007/978-3-030-34372-9.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, et al. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229, 2020. doi:10.1007/978-3-030-58452-8_13.

[9] Hanbo Zhang, Xuguang Lan, Site Bai, et al. Roi-based robotic grasp detection for object overlapping scenes. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019. doi:10.1109/IROS40897.2019.8967869.

[10] Shaoshuai Shi, Xiaogang Wang, Hongsheng Li, et al. Pointrcnn: 3d object proposal generation and detection from point cloud. pages 770–779, 2019. doi:10.1109/CVPR.2019.00086.

[11] Hongzhuo Liang, Xiaojian Ma, Shuang Li, et al. Pointnetgpd: Detecting grasp configurations from point sets. 2019. doi:10.1109/ICRA.2019.8794435.

[12] Yunhai Han, Kelin Yu, Rahul Batra, et al. Learning generalizable vision-tactile robotic grasping strategy for deformable objects via transformer. *IEEE/ASME Transactions on Mechatronics*, 2024. doi:10.1109/TMECH.2024.3400789.

[13] Anthony Brohan, Noah Brown, Justice Carbajal, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. doi:10.48550/arXiv.2307.15818.

[14] Lihe Yang, Bingyi Kang, Zilong Huang, et al. Depth anything v2: Accurate depth maps using a single rgb image. 2024. doi:10.48550/arXiv.2406.09414.

[15] Zhijian Liu, Haotian Tang, Alexander Amini, et al. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. 2023. doi:10.1109/ICRA48891.2023.10161234.

[16] Georgia Chalvatzaki, Nikolaos Gkanatsios, Petros Maragos, et al. Orientation attentive robotic grasp synthesis with augmented grasp map representation. 2020. doi:10.1109/IROS45743.2020.9341200.

[17] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. 2018. doi:10.1109/ICRA.2018.8460875.

[18] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774v6*, 2024.