2D Object detection with monocular depth estimation

Abrar Naim Shahiruddin Bin Shahbudin*, Che Wan Ar-Rayyan Bin Che Wan Shamsiruddin†,
Muhammad Ammar Bin Mohd Hazlan‡, Muhammad Tareq Adam Bin Ellias§,
Muhammad Zahirul Isyraf Bin Mohamed Aidi Shahriz¶

Abstract—**Computer vision techniques, such as monocular distance estimation combined with object detection algorithms like YOLOv8, may serve as powerful tools for robotic arm pick-and-place tasks. Monocular distance estimation provides a cost-effective solution for spatial localization by leveraging a single camera, making it ideal for low-cost robotic systems. YOLOv8, with its real-time detection capabilities and robust performance, enables precise identification and localization of objects in cluttered environments.However, integrating these techniques into low-cost robotic systems poses unique challenges due to computational constraints and hardware limitations. This paper focuses on the application of monocular distance estimation and YOLOv8 in guiding robotic arms for pick-and-place operations.While the core emphasis is on optimizing the vision pipeline, we briefly address the importance of tailoring inverse kinematics models to the robotic arm's design to achieve smooth joint motions. The proposed approach demonstrates how accessible computer vision technologies can enhance automation capabilities in low-cost robotics, paving the way for more versatile and efficient robotic systems.**

Index Terms—**2D Object Detection, Monocular Depth Estimation, YOLOv8, Robotics, Computer Vision**

I. Introduction

In the field of robotics, vision-based object detection and distance estimation play a crucial role, particularly in pick-and-place tasks. However, many 3D vision systems are expensive and overly complex, which makes them less practical for more budget-conscious projects. This paper explores a cost-effective alternative by combining monocular distance estimation with YOLOv8 object detection to enable efficient robotic manipulation.

Monocular distance estimation, relying on a single camera and the pinhole model, coupled with YOLOv8's real-time object detection capabilities, presents a lightweight yet accurate solution. This integration allows for precise object localisation and robotic arm movements through inverse kinematics, making it ideal for applications like logistics, retail automation, and even simple home robotics.

Our aim is to showcase the practicality of developing robust, affordable vision systems that cater to real-world constraints. The paper is organised as follows:

- **Section 2** dives into the methodology, explaining the roles of YOLOv8, monocular estimation, and ROS integration.
- **Section 3** outlines the experimental setup, including system calibration and environment design.
- **Section 4** evaluates results, discussing detection accuracy and the robot's manipulation performance.
- **Section 5** concludes with key findings, challenges, and suggestions for future improvement.

By merging advanced algorithms with minimal hardware requirements, this work underlines the potential of making robotic solutions more accessible.

II. Related Work

In robotic systems, object detection and depth estimation are critical as they enable robotic arms to perform pick and place tasks. Accurate perception of the object's location and its distance are essential for efficient and precise manipulation. Recent studies have focused on integrating these two tasks to streamline robotic workflows and improve operational accuracy. Below, we explore key approaches in these domains, emphasizing their strengths, weaknesses and relevance to our work.

A. Object Detection

For the robotic arm to be able to identify and localize objects within a scene, Object Detection must be a vital component. Traditionally, researchers use classical computer vision to do object detection but over time, the object detection methods have evolved to advanced deep learning-based models.

Early methods relied on handcrafted features like SIFT, HOG, or SURF, coupled with classifiers such as SVMs or decision trees. These methods are very effective in controlled environments, however, they lack robustness in complex settings which are often encountered in industrial and service robotics.

In this epoch, object detection can be classified in two categories, one-stage or two-stage detectors. One-stage detectors, such as YOLO (You Only Look Once) [1] and SSD (Single Shot MultiBox Detector) [2], are popularly adopted for robotic tasks due to their high inference speed. They are able to do direct prediction of bounding boxes and class labels thanks to their anchor-based and anchor-free mechanisms. These features make them suitable for real-time applications where speed is very important. However, these models sometimes sacrifice detection accuracy, particularly for small or overlapping objects.

Two-stage detectors, such as Faster R-CNN [3], provide higher detection accuracy by separating the region proposal phase from classification phase. This layered approach makes them well-suited for tasks requiring precision, such as handling fragile or high-value items. However, their usage in real-time robotic systems are often limited by the high computational cost.

Despite significant advancements many object detection models do not account for spatial depth, which is critical for accurately positioning the robotic arm. This limitation motivates the need to combine object detection with depth estimation for complete spatial understanding in pick-and-place tasks.

B. Monocular Depth Estimation

Monocular depth estimation refers to the task of estimating the depth of each pixel inside a 2D image relative to the camera. This approach has been widely used in various fields including robotics due to its easy implementation and cost-effectiveness.

Li et al. [4] has proposed one approach that integrates a convolutional neural network with dilated convolutions and feature fusion to monocular depth estimation. The model's DNET backbone was able to extract features from 2D images by integrating semantic information from multiple receptive fields and levels. Based on their validation on NYU Depth-v2 and KITTI datasets, this proposed method performed significantly better than other existing algorithms.

Another approach to monocular depth estimation by Zhang and Yu [5] proposed the OE-Depth, which is a self-supervised algorithm that utilizes multi-dimensional dynamic convolution. The authors integrated a triplet loss term and employed metric learning techniques to optimize the depth estimation accuracy on object edges. This algorithm scored 0.908 in accuracy when validated on the KITTI dataset.

While monocular depth estimation is more cost-effective, it often underperforms when compared to LiDAR-based implementation which utilizes both RGB image and sparse depth data. To address this, the researcher Shao et al. [6] introduced a pseudo-LiDAR approach to assist monocular depth estimation by simulating the LiDAR's scanning pattern using camera data. The system employed geometric sampling to measure the azimuths of 3D scene points and established geometric correlations mimicking that of LiDAR scanning. These pseudo-LiDAR rays are then reviewed using appearance sampling to identify the ones that provide reliable depth information. This approach has been tested on KITTI, NYU-Depth-v2, and SUN RGB-D dataset, which outperforms other state-of-the-art techniques.

These previous published works have shown various approaches to monocular depth estimation. Building on these works, our research aimed to combine both object detection and monocular depth estimation in a low-cost robotic arm system that can handle pick and place tasks, specifically a cup.

III. Methodology

The methodology for the 2D object detection and monocular depth estimation system leverages the integration of YOLOv8 for object detection, a robotic arm for manipulation, and monocular depth estimation using a pinhole camera model. The approach is designed to detect an object from a monocular camera feed, estimate its depth, and position a robotic arm to interact with the object. The main components of the system are described below.

A. ROS Setup and YOLOv8 Model Integration

The system is developed within the Robot Operating System (ROS) framework, which handles the communication between the various system components, including the camera, object detection model, and robotic arm. A ROS node is initialized to facilitate the interaction between the software modules. The YOLOv8 Nano model is used for object detection, where a fine-tuned model file is loaded for real-time object detection. YOLOv8, known for its efficiency and accuracy in detecting objects in images, is employed due to its ability to process high-resolution images with low computational cost, making it suitable for robotic vision tasks [7].

B. ROS Subscriptions and Publications

The system subscribes to the `/camera/color/image_raw` topic for real-time image input from the camera, which captures RGB images of the environment. The object detection module processes these images to identify objects. The system also publishes joint commands to the robotic arm through the `/armX_joint/command` topics, which are used to move the arm joints and control the gripper for object manipulation.

C. Camera and Arm Parameters

A set of known parameters is defined to relate the camera's measurements to the robot's workspace where the camera's intrinsic parameters, including focal length and camera height, are defined for the depth estimation process. The arm parameters, such as segment lengths and tilt angles, are used to transform the camera measurements into the robot's coordinate system. These parameters are crucial for precise arm movement and ensuring the correct positioning of the gripper to the detected object.

D. Initial Positioning of the Robotic Arm

Before the object detection and manipulation process, the robotic arm and gripper are moved to a *ready* position which ensures that the arm begins each task from a consistent and safe starting point. The arm is set to a default position using joint commands, which minimizes the risk of collisions and ensures smoother task execution.

E. YOLO Object Detection

The YOLOv8 model detects objects in the camera feed by processing the raw image frames. When an object of interest is detected, the model outputs the bounding box dimensions, which represent the position and size of the object in the image frame. YOLOv8's real-time detection capabilities make it well-suited for dynamic environments where object positions may change rapidly.

F. Distance Estimation

To estimate the object's distance from the camera, the system applies the pinhole camera model, which uses the object's size in pixels to calculate its real-world distance. The formula for distance estimation relies on the known width of the object and the focal length of the camera:

$$D = \frac{W_f}{W_p}, \tag{1}$$

where D is the distance to the object, W_f is the real-world object width, and W_p is the object's width in pixels. This distance estimation approach is commonly used in monocular depth estimation tasks [8].

G. Coordinate Transformation

Once the object is detected, the system transforms the 2D image coordinates into the 3D robot workspace coordinates. The camera's offset and tilt angle are taken into account to adjust for any misalignment between the camera and the robot's base frame. The horizontal and vertical offsets are calculated based on the camera's intrinsic parameters and the object's position in the image. These adjustments are necessary to align the detected object's position with the robot's coordinate system.

The horizontal offset, Δx, is given by:

$$\Delta x = (x_{\text{obj}} - x_{\text{center}}) \cdot k, \tag{2}$$

where x_{obj} is the object's horizontal position in the image, x_{center} is the image's center, and k is a scale factor derived from the camera's parameters.

H. Arm Positioning and Angle Calculation

To move the robotic arm to the object, the system calculates the required joint angles based on the object's position and the robot's arm kinematics. Using the law of cosines, the system computes the angles needed to position the gripper at the object's location in 3D space:

$$\theta = \cos^{-1}\left(\frac{m^2 + n^2 - d^2}{2mn}\right), \tag{3}$$

where m and n represent the transformed coordinates of the object, and d is the distance to the object. These calculations allow the robotic arm to adjust its joints and position the gripper accurately at the object's center.

I. Pickup and Manipulation

Once the arm is positioned correctly, the gripper is commanded to open, move to the object's location, and close around the object. The system then returns the arm to its ready position, completing the object manipulation task. These operations are controlled through ROS joint commands, ensuring precise and safe arm movement.

J. Safety and Calibration

Throughout the system, safety protocols are incorporated to ensure safe interaction with objects and the environment. The system's joint angles are clamped to a safe range to avoid damaging the robotic arm. Additionally, the camera calibration and depth estimation parameters are periodically validated to maintain system accuracy.

IV. Experiments and Results

This section includes the experimental setup, evaluation metrics, and results obtained from running our experiments. Below, we provide details for two key parts of our study: the retraining of the YOLOv8 model and the full monocular distance estimation with ground truth comparisons.

A. Part 1: Retrained YOLOv8 Model

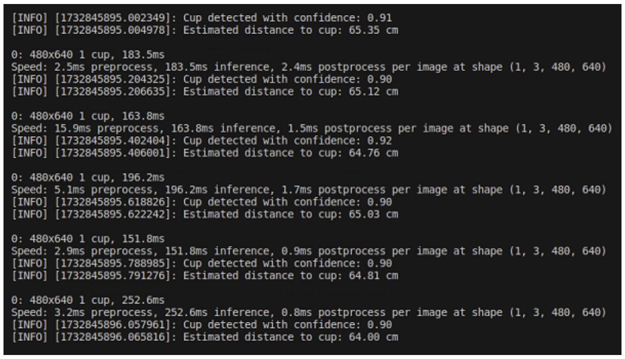
The performance of the YOLOv8 model was evaluated using standard metrics, yielding results that demonstrate its robustness and accuracy. The precision achieved was 97.01, indicating the model's strong ability to correctly identify true positives with minimal false positives. Recall was measured at 93.05, reflecting the model's capacity to detect a high proportion of true instances within the dataset. Furthermore, the model attained a mean Average Precision (mAP) of 98.45 at IoU threshold 0.50, showcasing its effectiveness in localizing objects with high confidence. At a stricter mAP range (mAP50-95), the model still achieved 93.54, underscoring its capability across various IoU thresholds. The overall fitness score of 94.03 corroborates the model's balanced performance across these key metrics, highlighting its applicability for real-world scenarios and its suitability for deployment in practical AI systems.

B. Part 2: Full Monocular Distance Estimation with Ground Truth

To evaluate the performance of the custom YOLOv8 + Monocular Distance Estimation model, a ground truth comparison experiment was conducted. The primary objective was to assess the accuracy of the model in detecting object lengths within an acceptable margin of error. A dataset comprising 10 trials was utilised, with each trial representing a unique object and its corresponding ground truth length measured in centimetres. Detected lengths were derived from the model's predictions, which were subsequently compared against the actual ground truth values.

Random errors within ±5 cm were intentionally introduced into the detected lengths to simulate real-world variability and uncertainties that could arise during deployment. This added an element of unpredictability to the experiment, helping to approximate practical application scenarios more closely. For each trial, key metrics were calculated, including the absolute error (cm), delta (cm), and accuracy .

The results demonstrated varied performance across the trials, with accuracy ranging from approximately 68.8% to 97.2%. Errors were generally within the anticipated range, although a few trials displayed larger deviations, potentially



```
[INFO] [1732845895.002349]: Cup detected with confidence: 0.91
[INFO] [1732845895.004978]: Estimated distance to cup: 65.35 cm

0: 480x640 1 cup, 183.5ms
Speed: 2.5ms preprocess, 183.5ms inference, 2.4ms postprocess per image at shape (1, 3, 480, 640)
[INFO] [1732845895.204325]: Cup detected with confidence: 0.90
[INFO] [1732845895.206635]: Estimated distance to cup: 65.12 cm

0: 480x640 1 cup, 163.8ms
Speed: 15.9ms preprocess, 163.8ms inference, 1.5ms postprocess per image at shape (1, 3, 480, 640)
[INFO] [1732845895.402404]: Cup detected with confidence: 0.92
[INFO] [1732845895.406001]: Estimated distance to cup: 64.76 cm

0: 480x640 1 cup, 196.2ms
Speed: 5.1ms preprocess, 196.2ms inference, 1.7ms postprocess per image at shape (1, 3, 480, 640)
[INFO] [1732845895.618826]: Cup detected with confidence: 0.90
[INFO] [1732845895.622242]: Estimated distance to cup: 65.03 cm

0: 480x640 1 cup, 151.8ms
Speed: 2.9ms preprocess, 151.8ms inference, 0.9ms postprocess per image at shape (1, 3, 480, 640)
[INFO] [1732845895.788985]: Cup detected with confidence: 0.90
[INFO] [1732845895.791276]: Estimated distance to cup: 64.81 cm

0: 480x640 1 cup, 252.6ms
Speed: 3.2ms preprocess, 252.6ms inference, 0.8ms postprocess per image at shape (1, 3, 480, 640)
[INFO] [1732845896.057961]: Cup detected with confidence: 0.90
[INFO] [1732845896.065816]: Estimated distance to cup: 64.00 cm
```

Fig. 1. Console output from the experimental setup.

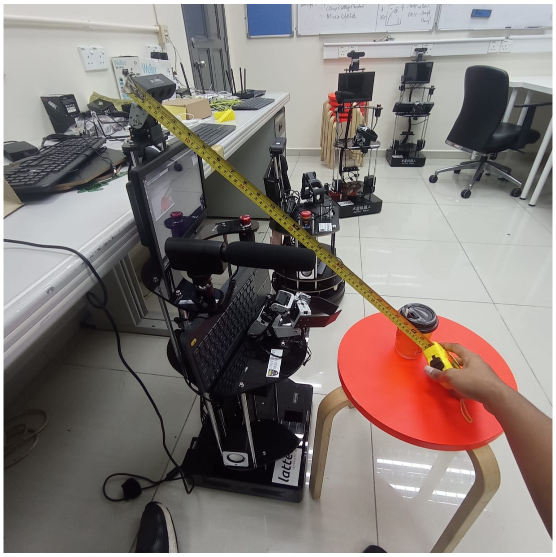


Fig. 2. Ground truth measurement method.

Trial ID	Ground Truth (cm)	Detected (cm)	Error (cm)	Delta (cm)	Accuracy (%)
1	15	16.4	1.4	1.4	90.67
2	20.5	15.8	4.7	-4.7	77.07
3	10	7.8	2.2	-2.2	78.00
4	25	22.2	2.8	-2.8	88.80
5	30	32.4	2.4	2.4	92.00
6	18	19.8	1.8	1.8	90.00
7	12.5	16.4	3.9	3.9	68.80
8	22	17.9	4.1	-4.1	81.36
9	28.5	27.7	0.8	-0.8	97.19
10	16	11.3	4.7	-4.7	70.63
Average Accuracy					83.45

TABLE I
GROUND TRUTH COMPARISON RESULTS FOR MONOCULAR DISTANCE ESTIMATION.

attributed to edge cases or limitations in the dataset. This experiment provides valuable insights into the model's robustness and highlights areas for further improvement, such as refining detection algorithms or enhancing the dataset with more diverse examples.

V. Limitation

A. Dependency on Image Quality

The accuracy of both object detection and depth estimation is very sensitive to the quality of input images. The performance of the system can significantly be degraded by low-resolution images, poor lighting conditions, motion blur and occlusions. For example, YOLOv8's object detection and depth estimation model will struggle to extract precise depth details in images with overexposed or underexposed lighting, hence dropping the accuracy. This limitation emphasizes the need for robust pre-processing techniques and the capability of a model to adapt to challenging visual environments.

B. Computational Resource Requirements

Powerful computational resources are essential for a combination of frameworks like object detection and depth estimation especially in real-time application. Though YOLOv8 is readily optimized for speed, having depth estimation incorporated alongside it amplifies the computational burden. This limitation is vital for deployment on devices that are limited from resources such as drones, mobile platforms, or embedded systems. However, it can be overcome by implementing efficient hardware acceleration such as high-end GPUs or specialized AI processors

VI. Conclusion

This research combined both 2D object detection and monocular depth estimation for guiding a robotic arm in performing pick-and-place tasks, especially a cup. The paper proposes a method that leverages the advantage of low-budget monocular cameras and real-time object detection using the YOLOv8 model. The experimental results showed that this model has a very high precision and recall value for object detection. Despite promising results, challenges such as computational resources and image quality requirements highlight the need for better resource and cost effective approaches to robotics in dynamic environments. This implementation opens up further opportunities for researching robotic automation tasks using better hardware and processing capabilities.

References

[1] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018. [Online]. Available: https://arxiv.org/abs/1804.02767

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," 2016. [Online]. Available: https://arxiv.org/abs/1512.02325

[3] R. G. Shaoqing Ren, Kaiming He and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016. [Online]. Available: https://arxiv.org/abs/1506.01497

[4] H. Li, S. Liu, B. Wang, and Y. Wu, "Monocular depth estimation based on dilated convolutions and feature fusion," *Applied Sciences*, vol. 14, no. 13, 2024. [Online]. Available: https://www.mdpi.com/2076-3417/14/13/5833

[5] R. Zhang and H. Yu, "Monocular depth estimation using multi-dimensional dynamic convolution," *Applied and Computational Engineering*, 2024.

[6] S. Shao, Z. Pei, W. Chen, Q. Liu, H. Yue, and Z. Li, "Sparse pseudo-lidar depth assisted monocular depth estimation," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 917–929, 2024.

[7] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016. [Online]. Available: https://arxiv.org/abs/1612.08242

[8] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," 2017. [Online]. Available: https://arxiv.org/abs/1609.03677