PEER ROBOTICS ASSESSMENT - REPORT

Abhinav Bhamidipati - abhinav7@terpmail.umd.edu

INTRODUCTION

This project is a perception pipeline for real-time pallet detection and ground segmentation. The goal was to develop a ROS2-based pipeline that can run smoothly on edge devices like the NVIDIA Jetson AGX Orin. I used YOLOv8 for detecting pallets, and a UNet model to segment the ground. The project is also deployed through a Dockerized ROS2 node that processes live camera input.

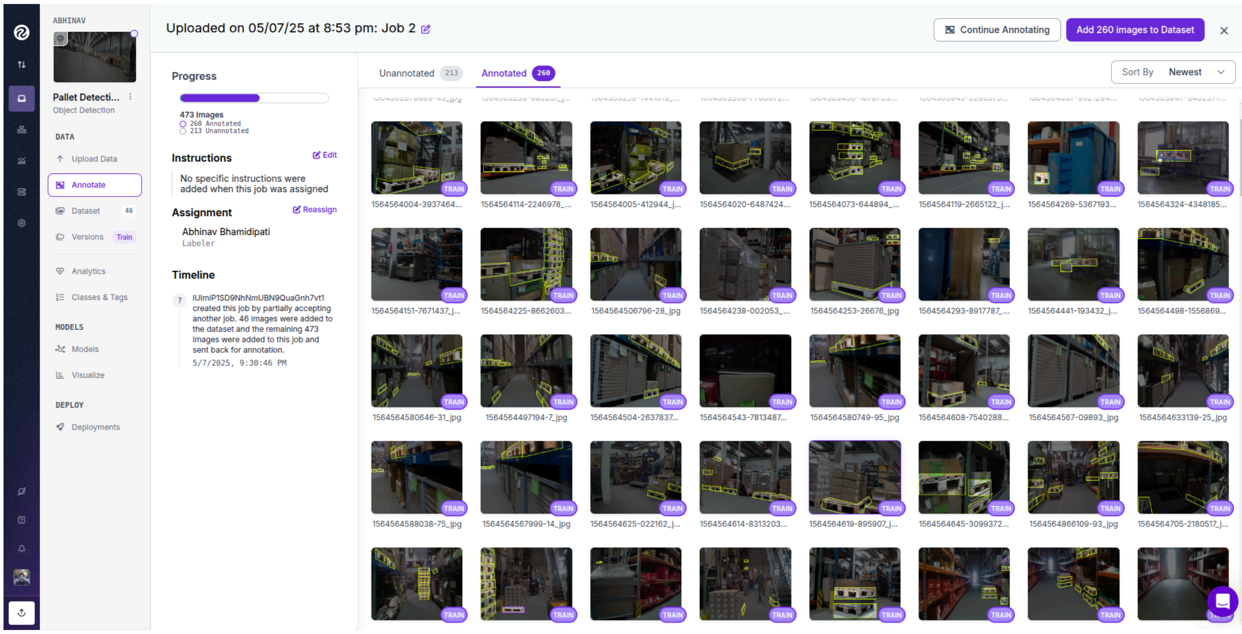
DATASET ACQUISITION AND PREPARATION

The initial dataset provided in the "Pallets" folder consisted of 519 images capturing various manufacturing and warehouse environments. I began with the ground segmentation task by generating ground masks using a combination of GroundingDINO and SAM (Segment Anything Model). This approach yielded reasonably accurate results for isolating ground regions. The segmented images were then organized into training, validation, and test sets following a 75:15:10 split.

Fig. 1. Ground mask created using GroudingDINO+SAM

However, this method was less effective for pallet detection. To address this, I switched to manual annotation using Roboflow, which allowed for precise bounding box labeling of pallets. The annotated dataset was then exported in YOLO format.

To improve model generalization and mimic real-world variation, I applied a range of data augmentation techniques, including horizontal flipping, random rotation, hue shifts, brightness adjustment, Gaussian blur, and noise injection.



Since the initial dataset was limited in size, I supplemented it with a larger public dataset, LOCO: Logistics Objects in Context, which provided around 4,000 additional images of pallets and related warehouse objects. This significantly enhanced the robustness of both detection and segmentation models by exposing them to varied pallet types, layouts, and lighting conditions.

MODEL DEPLOYMENT

Object Detection

For detecting pallets, I used YOLOv8n, a lightweight and fast variant, suitable for edge deployment. I trained the model on a combined dataset that included both the provided images and the LOCO dataset. Training was carried out over 50 epochs with a resolution of 640×640 and a batch size of 8.

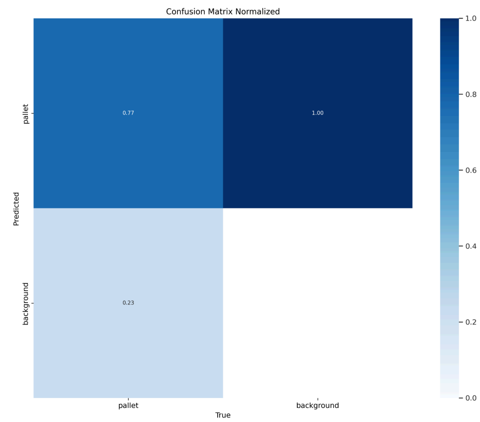
During training, the model achieved:

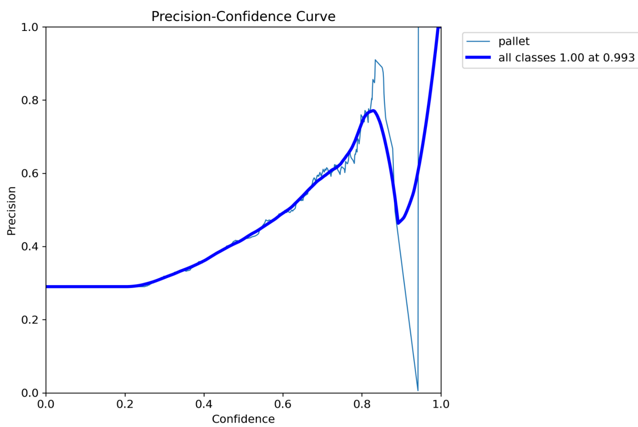
- Precision: 0.691
- Recall: 0.609
- mAP@0.5: 0.656
- mAP@0.5:0.95: 0.289

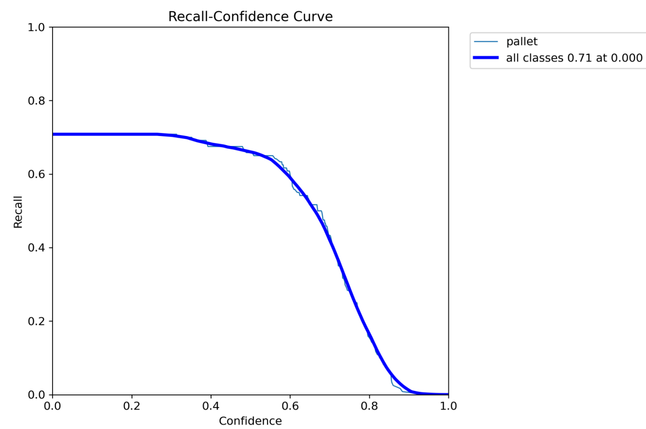
On the test set, the performance was consistent:

- Precision: 0.7218
- Recall: 0.5942
- mAP@0.5: 0.6789
- mAP@0.5:0.95: 0.3310

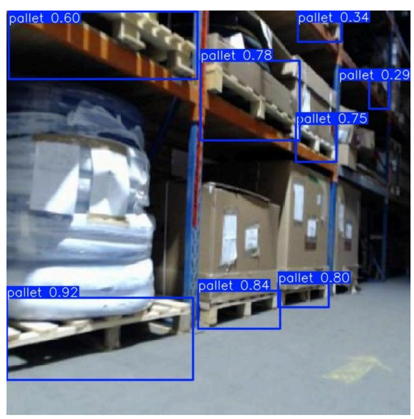
Evaluation curves showed an optimal F1 score of 0.53 at a confidence threshold of 0.595. Precision peaked at 1.00 but with a corresponding drop in recall. The confusion matrix revealed that most pallets were correctly detected.







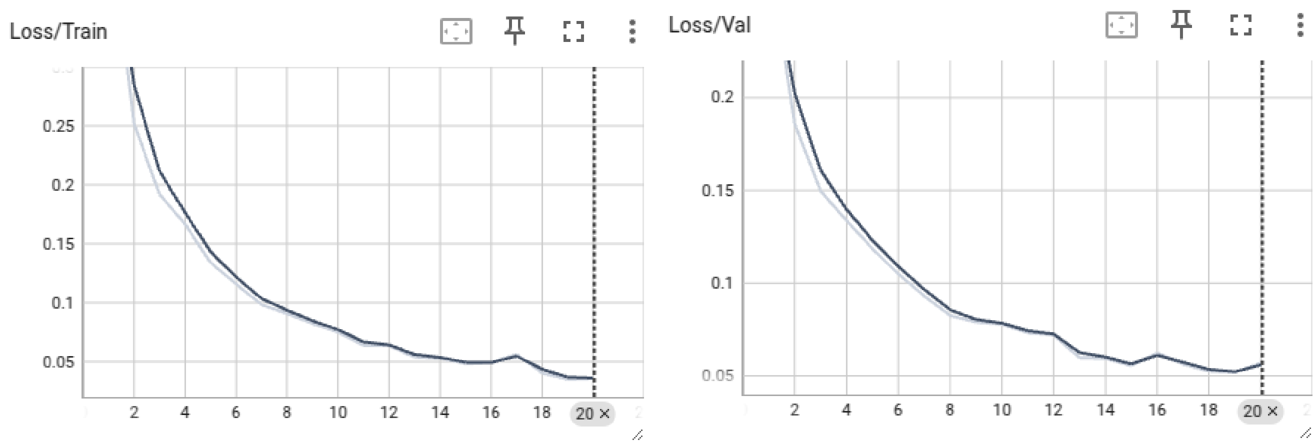
Overall, the model demonstrated robust performance.



Ground Segmentation

For segmenting the ground plane, I used a UNet architecture with a MobileNetV2 encoder, trained using the segmentation_models_pytorch library. This model was selected for its low inference cost and suitability for real-time edge deployment.

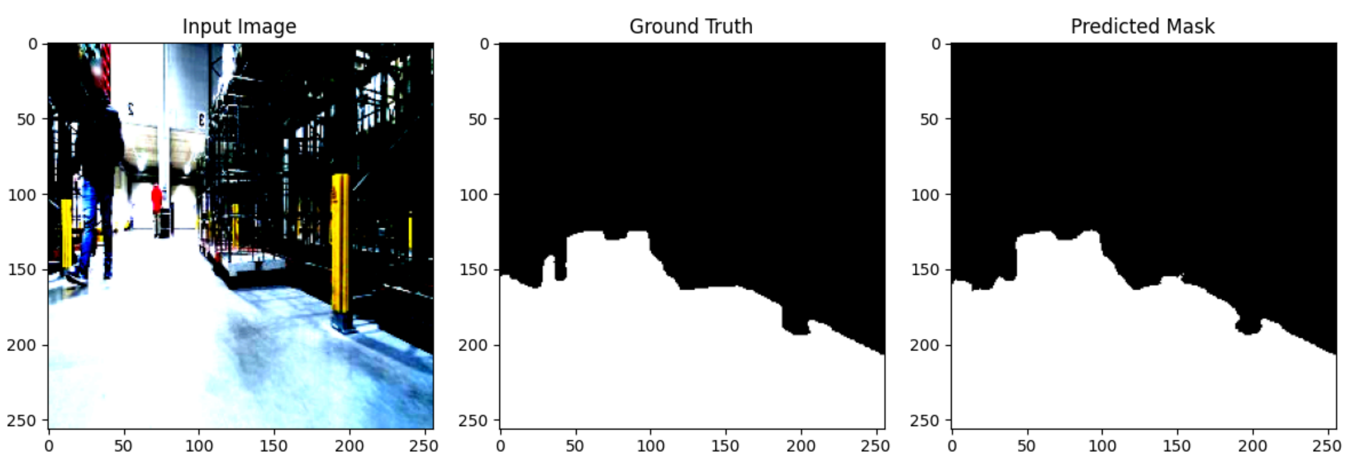
Training was conducted over 20 epochs using binary cross-entropy with logit loss. The training and validation loss curves indicate stable and consistent convergence without overfitting, ending at a validation loss of 0.056.



On the test set, the model achieved strong performance:

- Test IoU: 0.7145
- Pixel Accuracy: 0.9724

A visual comparison between the input image, ground truth, and predicted mask shows the model accurately captures the ground layout, even under clutter and occlusion.



The segmentation model proved to be lightweight, accurate, and reliable.

ROS2 NODE DEVELOPMENT

I developed a ROS2 Python node that performs real-time inference for pallet detection and ground segmentation. The node subscribes to both RGB and depth image topics from a ZED2i camera:

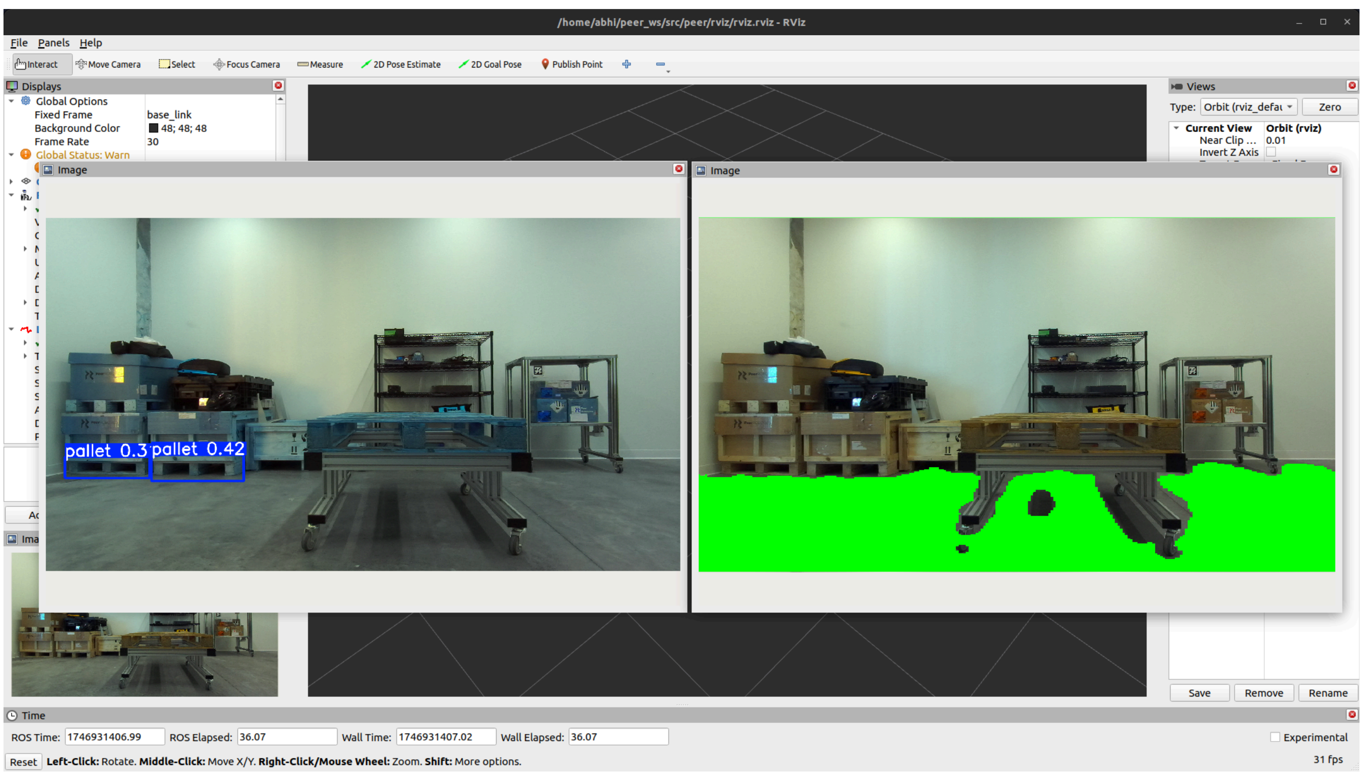
RGB Topic: /robot1/zed2i/left/image_rect_color
Depth Topic: /robot1/zed2i/left/image_depth (currently not used, just subscribed)

The RGB image is processed by two models:

A YOLOv8 model detects pallets and draws bounding boxes.
A UNet model segments the ground region and overlays it in green.

The processed outputs are then published on the following topics:

/pallet_detection: RGB image with pallet bounding boxes.
/ground_segmentation: RGB image with ground segmentation mask overlaid.



These were tested on the given bag file, and visualized using RViz. Both subscriptions use a BEST_EFFORT QoS policy for robustness in lossy environments.

EDGE DEVELOPMENT AND DOCKERIZATIOIN

To prepare the perception pipeline for real-time edge deployment on devices like the NVIDIA Jetson AGX Orin, I applied model optimization techniques where feasible. For ground segmentation, the MobileNetV2-based UNet model was successfully converted from PyTorch to ONNX and finally to a TensorRT engine using FP16 precision. FP16 precision was used during TensorRT conversion to significantly speed up inference and reduce memory usage. This resulted in significantly faster inference and reduced memory usage, making it ideal for low-power embedded systems.

While I intended to optimize the YOLOv8 model as well, I encountered compatibility and versioning issues during the ONNX-to-TensorRT conversion. However, given that the YOLOv8n model was already lightweight and demonstrated fast inference speeds on GPU, I retained the original PyTorch version for detection during deployment. This ensured that overall system latency remained low without compromising detection accuracy.

For portability and ease of deployment, the entire ROS2 inference pipeline was containerized using Docker.The containerized system can run natively on any machine with NVIDIA drivers, allowing for reproducible results across development and deployment setups. With this setup, the node is capable of live inference from a ZED2i camera, publishing processed results through ROS topics without the need for manual reconfiguration.

CONCLUSION

This project delivered a real-time pallet detection and ground segmentation pipeline using YOLOv8 and UNet, integrated into a ROS2 system. The segmentation model was optimized with TensorRT for edge deployment. The entire system was Dockerized for easy deployment on Jetson devices, enabling reliable performance in warehouse and manufacturing environments.