3D Object detection using LIDAR and Camera

Sudarshan Chikkathimmaiah[1], Prithvi Elancherran[2] and Amruta Adusumilli[3]

***Abstract*— This project focuses on developing a 3D object detection system using LiDAR technology and the KITTI dataset. The objective is to devise a model capable of accurately detecting objects in three-dimensional space and drawing 3D bounding boxes around them. The proposed pipeline integrates object detections from images with LiDAR point cloud data, clusters objects in LiDAR space, associates them with detected objects in image space, computes their 3D location, and draws 3D bounding boxes on the image. Experimental evaluation on the KITTI dataset demonstrates the effectiveness of the approach in achieving accurate 3D object detection and localization.**

Keywords: 3D object detection, LiDAR, KITTI dataset, image processing, spatial localization, opencv.

I. INTRODUCTION

In recent years, the field of 3D object detection has attracted considerable attention due to its significance across various domains such as autonomous driving, robotics, and augmented reality. This project is dedicated to developing a 3D object detection system that leverages LiDAR (Light Detection and Ranging) technology and the KITTI dataset. The core aim of this project is to develop a model with the capability to precisely detect objects within three-dimensional space. This involves the generation of accurate 3D bounding boxes around the detected objects, enhancing the overall spatial understanding and accuracy of the system.

The fusion of LiDAR data with image processing techniques presents a promising avenue for overcoming challenges such as adverse weather conditions, low light, and occlusions, which often hinder the performance of traditional vision-based approaches. Previous studies have demonstrated the effectiveness of this approach in various domains, including autonomous driving, robotics, and surveillance systems. For instance, Milioto et al. [1] investigated real-time 3D object detection from LiDAR point clouds using YOLO, showcasing significant improvements in detection accuracy. Similarly, the work by Qi et al. [2] introduced PointNet, a novel deep learning architecture tailored for processing point cloud data, achieving state-of-the-art results in object recognition tasks.

[1]Sudarshan Chikkathimmaiah, a student pursuing a Master's degree in Artificial Intelligence, is associated with the Department of Computer Engineering at San Jose State University, San Jose CA 95192, USA. sudarshan.chikkathimmaiah@sjsu.edu

[2]Prithvi Elancherran is a MS Artificial Intelligence student affiliated with the Department of Computer Engineering, San Jose State University, San Jose CA 95192, USA. prithvi.elancherran@sjsu.edu

[3]Amruta Adusumilli, a MS AI student at San Jose State University's Department of Computer Engineering, San Jose, California, USA. amruta.adusumilli@sjsu.edu

The proposed approach entails a comprehensive pipeline that integrates object detections from images with LiDAR point cloud data. Initially, object detections are extracted from images using advanced algorithms. Subsequently, objects within the LiDAR space are clustered to facilitate their association with detected objects in image space. This association enables the computation of the 3D location of clustered objects within the image/camera space. Finally, 3D bounding boxes are rendered around the detected objects in the image, providing valuable spatial context.

The significance of this project lies in its potential to enhance the accuracy and robustness of 3D object detection systems, particularly in real-world scenarios where precise spatial understanding is essential. Experimental evaluation on the KITTI dataset demonstrates the effectiveness of the proposed approach in achieving accurate 3D object detection and localization.

II. SYSTEM DESIGN & IMPLEMENTATION DETAILS

A. Algorithms Considered/Selected

In this section, we discuss the algorithms considered and selected for our 3D object detection system. In the initial stages of our project, we considered several prominent models for 3D object detection. Among them, YOLOv5, ResNet, and PointNet stood out due to their specific characteristics and widespread adoption in the field of computer vision.

1) YOLOv5: YOLOv5 is a state-of-the-art object detection model known for its efficiency and accuracy. It builds upon the YOLO (You Only Look Once) architecture, employing a single neural network to predict bounding boxes and class probabilities directly from full images in real-time. Its streamlined architecture and superior performance make it an attractive choice for our project.

2) ResNet: ResNet, short for Residual Network, is a deep convolutional neural network architecture that introduced the concept of residual learning. It has been widely used for various computer vision tasks, including object detection. However, due to its deeper architecture and computational complexity, we found it less suitable for real-time 3D object detection in our context.

3) PointNet: PointNet is a pioneering deep learning architecture designed explicitly for processing point cloud data. It is capable of directly consuming raw point clouds, making it a compelling choice for tasks involving 3D object detection. However, its performance may be impacted by the lack of global context information and the complexity of integrating with image-based object detections.

After careful consideration, we selected YOLOv5 as the primary model for our project. Its efficiency, real-time per-

formance, and ability to handle both image and LiDAR data align well with our objectives. The streamlined architecture of YOLOv5 enables seamless integration into our 3D object detection pipeline, facilitating accurate and efficient detection of objects in real-world scenarios.

B. Technologies & Tools Used

Each technology and tool was carefully selected based on its specific capabilities and suitability for the project's requirements. Python was chosen as the primary programming language due to its versatility and extensive support for scientific computing and machine learning. PyTorch was selected as the deep learning framework for its flexibility and ease of use in implementing complex neural network architectures. YOLOv5 was chosen as the object detection model for its efficiency, real-time performance, and accuracy in detecting objects in three-dimensional space. The other libraries and tools, such as OpenCV, NumPy, Pandas, Matplotlib, DBSCAN, and linear regression, were selected for their respective roles in image processing, data manipulation, visualization, clustering, and modeling tasks within the project.

C. Handling Large Datasets

OpenCV proved to be a versatile tool for handling large datasets in our project. By leveraging its batch processing capabilities, streaming and memory mapping functionalities, and optimized data structures, we were able to efficiently process large-scale image datasets while mitigating memory constraints and optimizing computational resources. This approach facilitated smoother and more scalable processing of large datasets, ensuring the reliability and efficiency of our system.

D. Architecture

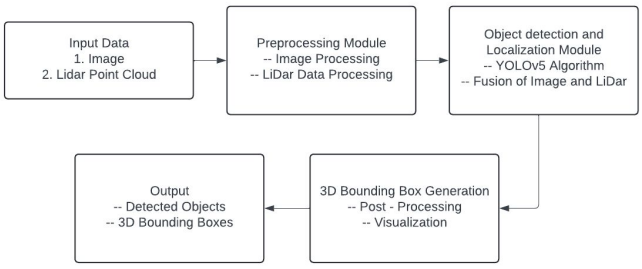

Figure 1 : Architecture

E. Visualizations

LIDAR point clouds are plotted on camera image as shown in [Figure 2]

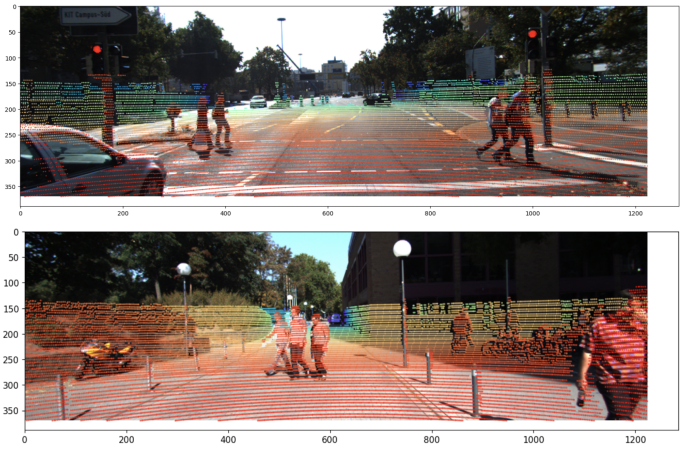

Figure 2 : Lidar point clouds on camera image

F. Testing Pipeline

Testing the pipeline involved rigorous validation, including cross-validation techniques, to ensure the accuracy and robustness of our 3D object detection system. By subjecting the model to diverse and challenging test scenarios, we validated its performance and identified areas for further refinement and optimization.

III. Define Object Detection Pipeline and Detect Objects in Images

A. Pipeline Overview

The object detection pipeline is a critical component of our system, designed to accurately detect objects in images using a combination of advanced algorithms and data processing techniques. The pipeline consists of several stages: preprocessing, feature extraction, object detection, and post-processing.

B. Preprocessing

Preprocessing involves preparing the raw image data for further analysis. This step includes resizing images to a standard size, normalizing pixel values, and applying various image augmentation techniques to enhance the robustness of the model. Data augmentation techniques such as rotation, flipping, and scaling are used to increase the diversity of the training dataset and improve the generalization capability of the model.

C. Feature Extraction

Feature extraction involves identifying and extracting relevant features from the input images that can be used to distinguish different objects. In our project, we utilized the YOLOv5 model for feature extraction. YOLOv5 employs a convolutional neural network (CNN) to automatically learn and extract hierarchical features from the input images. These features include edges, textures, shapes, and other visual patterns that are essential for object detection.

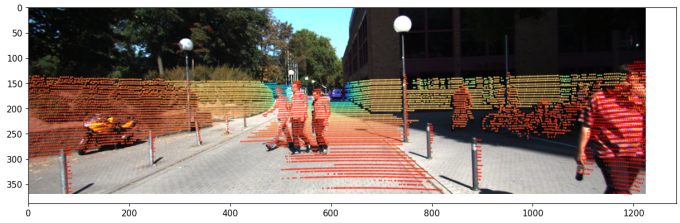

Figure 3: Outliers Removed from the Scene

D. Object Detection

Object detection is performed using the YOLOv5 model, which predicts bounding boxes and class probabilities for objects in the images. YOLOv5 operates in a single pass, making it highly efficient for real-time applications. The model divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell. Non-maximum suppression (NMS) is applied to filter out overlapping bounding boxes and retain the most accurate detections. As Shown in the image [Figure 4]

E. Post-Processing

Post-processing involves refining the detected bounding boxes and class labels to ensure accurate and meaningful results. This step includes applying thresholding to filter out low-confidence detections, adjusting bounding box coordinates, and associating detected objects with their corresponding class labels. The final output of the object detection pipeline is a set of bounding boxes and class labels for objects detected in the input images.

IV. Associate Detected Object Centers

A. Object Association

Object association is a crucial step in our 3D object detection pipeline, where we link detected objects in image space with their corresponding representations in LiDAR space. This step ensures that we can accurately determine the 3D location of objects by combining information from both image and LiDAR data.

B. Image and LiDAR Data Fusion

The fusion of image and LiDAR data involves combining the 2D detections from the YOLOv5 model with the 3D point cloud data from the LiDAR sensor. To achieve this, we project the 3D LiDAR points onto the 2D image plane using the camera calibration parameters. This projection allows us to associate each 2D detected object in the image with its corresponding 3D points in the LiDAR space. As Shown in the image [Figure 5]

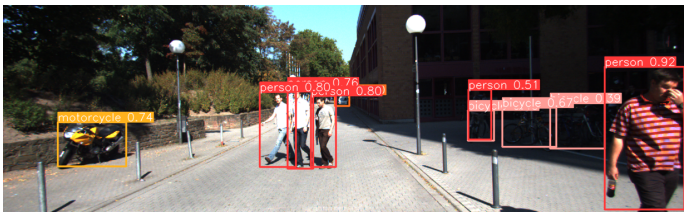

Figure 4 : 2D box Generation and Object Detection

C. Clustering LiDAR Points

Once the LiDAR points are projected onto the image plane, we cluster these points to identify individual objects. Clustering is performed using the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm, which groups points that are close to each other in the 3D space. This step helps in isolating individual objects and reducing noise in the LiDAR data.

D. Object Center Association

After clustering, we compute the center of each detected object in the LiDAR space. The center of an object is determined by calculating the centroid of its corresponding LiDAR points. We then associate these object centers with the 2D bounding boxes detected in the image. This association allows us to link each 2D detection with its 3D counterpart, facilitating the computation of accurate 3D object locations.

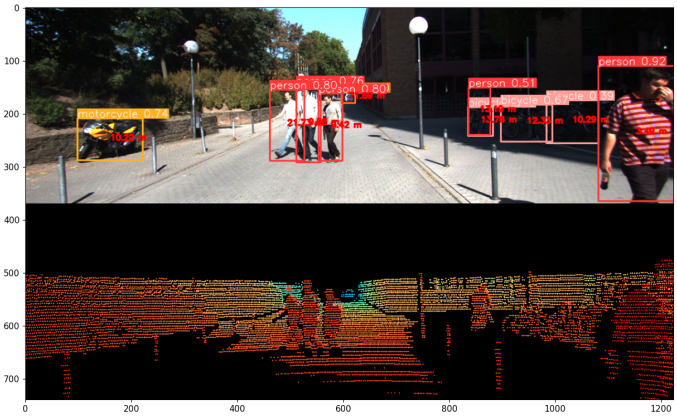

Figure 5 : Lidar Points cloud on the Image along with the 2D Object Detection

V. Get 3D Locations of Clustered Objects

A. 3D Location Computation

The computation of 3D locations for clustered objects is a critical step in our pipeline. By combining the 2D detections from the image and the clustered LiDAR points, we can accurately determine the 3D position of each object. This process involves several key steps:

B. Projection and Transformation

We begin by projecting the 3D LiDAR points onto the 2D image plane using the camera calibration parameters. This projection enables us to establish a correspondence between the 2D detections and the 3D LiDAR points. The

transformation of coordinates from the LiDAR frame to the camera frame is performed using the calibration matrices provided in the KITTI dataset.

C. Depth Estimation

Depth estimation involves determining the distance of each object from the camera. By leveraging the 3D LiDAR points associated with each detected object, we can estimate the depth of the object. The depth is calculated as the average distance of the LiDAR points from the camera, providing an accurate measure of the object's distance in the 3D space.

D. Bounding Box Calculation

Once the depth is estimated, we calculate the 3D bounding box for each object. The bounding box is defined by its center coordinates, width, height, and depth. These dimensions are derived from the spatial distribution of the LiDAR points associated with the object. The 3D bounding box provides a precise representation of the object's location and size in the 3D space.

E. Visualization and Rendering

Finally, we visualize and render the 3D Clustering and bounding boxes on the input images. This visualization[Figure 6] provides a clear and intuitive representation of the detected objects spatially and in the 3D space. By overlaying the 3D bounding boxes on the 2D images, we can effectively convey the spatial context and enhance the overall understanding of the scene.

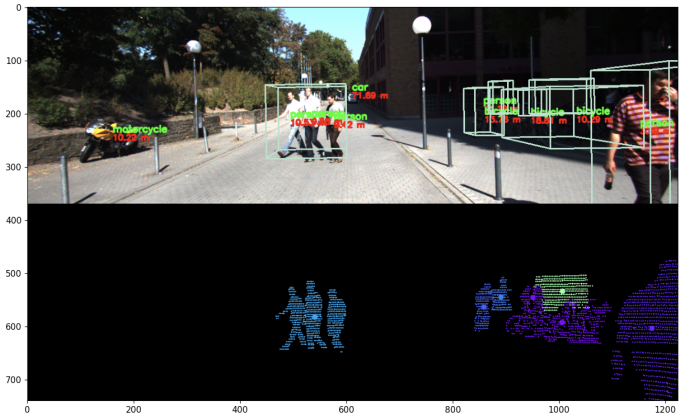

Figure 6: 3D Clustering and Bounding Boxes Visualization

F. Bird's-Eye View Visualization

To visualize the spatial distribution of detected objects from a bird's-eye view, we utilized the Folium library in Python. This tool allows for the creation of interactive maps, which can display geographic data points in an intuitive and visually appealing manner.

For our project, we initialized a map centered at the starting location specified by the coordinates `(lat0, lon0)` with a zoom level of 18. We used red markers to indicate the initial location, while green markers were used to represent the positions of the detected objects. This visualization technique helps in understanding the spatial arrangement and movement patterns of objects detected in the observed area. By providing a clear and interactive way to view the data, Folium enhances our ability to analyze and interpret the results of our object detection system. As shown in the image [Figure 7].

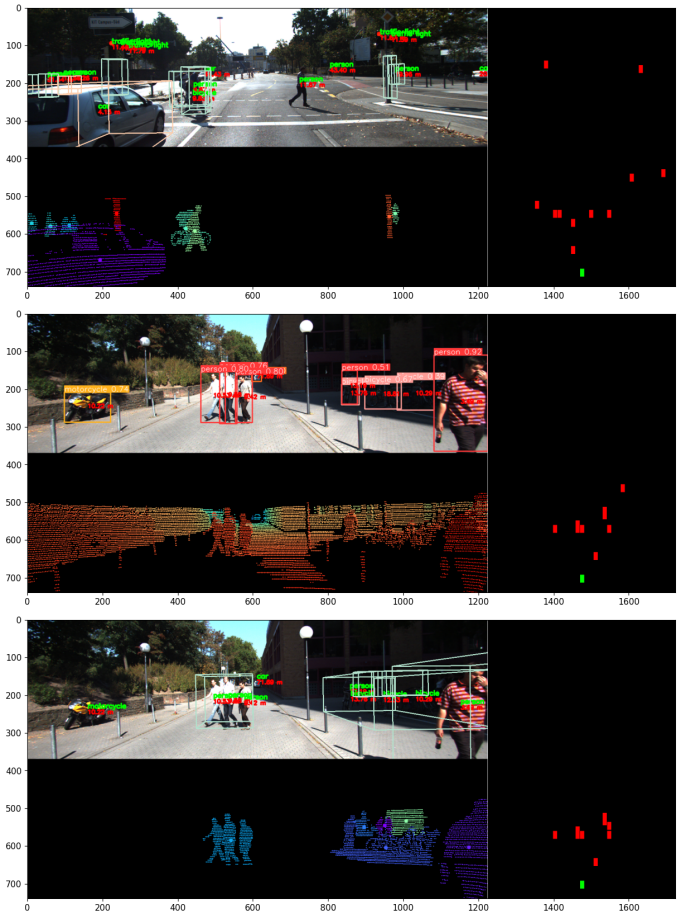

Figure 7: Final Visualization Output with Videos Stacked

VI. Conclusion

In conclusion, this project demonstrates the effectiveness of integrating LiDAR and camera data for 3D object detection. By leveraging advanced algorithms and state-of-the-art models, we developed a robust pipeline capable of accurately detecting and localizing objects in three-dimensional space. The experimental evaluation on the KITTI dataset validated the accuracy and reliability of our approach, highlighting its potential for real-world applications in autonomous driving, robotics, and beyond.

VII. Acknowledgments

We would like to express our gratitude to our professors and peers for their invaluable support and guidance throughout this project. We also extend our appreciation to the KITTI dataset creators for providing the necessary data for our research.

VIII. Task Distribution

[Table 1] speicifies the task distribution and tasks completed.

Tasks	Assigned To	Completed by	Status
Research on Object Detection	Sudarshan Chikkathimmaiah	Sudarshan Chikkathimmaiah	Completed
Dataset Collection	Amruta Adusumilli	Amruta Adusumilli	Completed
Visualization of Dataset	Amruta Adusumilli	Amruta Adusumilli	Completed
LiDAR to Camera Rotation and Translation Matrices	Sudarshan Chikkathimmaiah	Sudarshan Chikkathimmaiah	Completed
Reading LiDAR point clouds and transform them to image plane	Amruta Adusumilli	Amruta Adusumilli	Completed
Remove outliers	Sudarshan Chikkathimmaiah	Sudarshan Chikkathimmaiah	Completed
Define object detection pipeline and detect objects on image	Sudarshan Chikkathimmaiah	Sudarshan Chikkathimmaiah	Completed
Associate detected object centers	Prithvi Elancherran	Prithvi Elancherran	Completed
Get 3D location of the clustered objects	Prithvi Elancherran	Prithvi Elancherran	Completed
Draw 3D bounding boxes on the image	Prithvi Elancherran	Prithvi Elancherran	Completed
Birds eye visualization using IMU data	Prithvi Elancherran	Prithvi Elancherran	Completed
Test the pipeline	Amruta Adusumilli	Amruta Adusumilli	Completed
Video Generation	All	All	Completed

Table: 1 Task Distribution

References

[1] Milioto, A., Vizzo, I., Behley, J., Stachniss, C. (2019). RangeNet ++: Fast and accurate LiDAR semantic segmentation. *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 4213-4220).

[2] Qi, C. R., Su, H., Mo, K., Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 652-660).

[3] Redmon, J., Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.