Optimized and Enhanced Object Recognition Algorithm using RGBD Camera modalities for Improved Accuracy

**Abstract:** In this study, we present an innovative algorithm for detecting objects against a background with a well-defined plane, utilizing RGBD camera data. This algorithm is pivotal for enhancing robotic navigation and interaction in environments where the distinction between objects and their background planes is critical. A key contribution is the introduction of a heuristic algorithm for isometry estimation, designed to perform efficiently even on computers with limited processing capabilities. It features adjustable hyperparameters, enabling a trade-off between computational efficiency and detection accuracy, thereby offering a versatile solution for varying computational environments.

**Keywords:** Object Detection, RGBD Camera Data, Heuristic Algorithm, Isometry Estimation, Planar Backgrounds, Robotic Navigation, Computational Efficiency, Algorithmic Precision, Hyperparameter Tuning, Autonomous Systems.

1 Introduction

An autonomous mobile robot refers to a robot that can operate and navigate in its environment without constant human control or intervention. It is designed to perceive its surroundings using various sensors, process the sensory data, make decisions, and execute actions to accomplish specific tasks or objectives. These robots typically incorporate advanced technologies such as artificial intelligence, computer vision, machine learning, and path planning algorithms to perceive, interpret, and interact with the environment effectively. Autonomous mobile robots are used in a wide range of applications, including logistics, manufacturing, healthcare, agriculture, exploration, and transportation, to perform tasks autonomously and enhance efficiency, productivity, and safety.

Accurate and real-time object recognition, crucial for enhancing robotic autonomy and efficiency in dynamic environments, fundamentally enables autonomous systems to interact with the physical world significantly. By recognizing objects accurately, robots can perform tasks with greater precision and reliability across various applications, such as navigation and manipulation, in both industrial and domestic settings. This capability minimizes the need for human intervention and allows for more nuanced interactions with their surroundings. Moreover, understanding object recognition as the process of finding isometry — identifying objects based on their shape and size regardless of their orientation or position — is pivotal. This aspect underscores the importance of geometric consistency in recognition algorithms, ensuring that robots can identify and interact with objects correctly, even in complex and unfamiliar environments. This advancement is a key step toward achieving truly autonomous robotic systems capable of understanding and navigating the world as effortlessly as humans.

An isometry is a transformation that maintains the distances between all points of a shape unchanged. In other words, it's a way of moving or rotating a shape without altering its size or shape. Isometries are characterized by preserving the geometric properties of figures. Common examples of isometries include translations (moving every point of a shape a constant distance in a specified direction), rotations (spinning a shape around a point without changing its size), reflections (flipping a shape over a line, producing a mirror image), and glide reflections (a combination of a reflection over a line and a translation along that line). Isometries are particularly important in both mathematics and physics because they are associated with symmetries and conservation laws. In computer graphics, isometries are used to manipulate objects in a scene without changing their intrinsic properties, such as when moving or rotating a camera view or an object.

The work [1] introduces a novel approach to real-time 3D pose estimation for unseen objects on mobile devices. It combines lightweight neural networks, MobilePose-Base and MobilePose-Shape, designed for efficiency on mobile platforms. These networks incorporate direct shape prediction into the pose estimation process, enabling the system to learn pose from shape features effectively. The approach is validated through experiments on mixed real and synthetic datasets, showing enhanced accuracy and real-time performance on mobile devices, making significant strides over previous methods in terms of balance between model size, speed, and accuracy. This work addresses enhancing real-time 3D pose estimation of unseen objects using mobile-friendly networks, namely MobilePose-Base and MobilePose-Shape. It aims to facilitate applications across various domains, such as computer vision, augmented reality, autonomous driving, and robotics, by enabling mobile devices to accurately detect and estimate the poses of objects not previously seen during training. The core innovation lies in integrating shape prediction directly into the network architecture, allowing the model to learn poses based on shape features derived from both real and synthetic datasets with weak shape supervision. The authors propose two network variants tailored for different levels of shape supervision availability and demonstrate their effectiveness in improving pose estimation accuracy while maintaining real-time performance on mobile devices. This approach represents a significant departure from traditional methods that rely on instance awareness, where the object must be known beforehand, thereby broadening the scope of objects that can be recognized and their poses estimated in real-world applications.

The authors Yi Li et al. [2] present DeepIM, a novel deep neural network framework for 6D pose estimation through deep iterative matching. This method iteratively refines an initial pose estimation by aligning rendered images of an object against observed images. The refinement process utilizes a network trained to predict relative pose transformations with a disentangled representation of 3D location and orientation. This approach allows for significant improvements in accuracy over state-of-the-art methods on benchmark datasets like LINEMOD and Occlusion LINEMOD. Additionally, DeepIM can refine poses of unseen objects, demonstrating its generalizability and effectiveness in handling textureless objects. Key contributions include the introduction of a network for iterative pose refinement without hand-crafted features, a disentangled representation of the SE(3) transformation for accurate pose estimates, and extensive experimental validation showcasing superior performance on standard benchmarks. The method's ability to refine poses of unseen objects further highlights its potential for various applications in robot manipulation and virtual reality, where accurate object pose estimation is crucial.

In article [19], the authors conducted an assessment of the 6D pose and dimensions of instances of invisible objects in RGB-D images. The problem of assessing the 6D pose "at the instance level" is described, where during training or testing, there are no precise CAD models of the objects. To handle various and invisible instances of objects in a certain category, the authors introduced the Normalized Object Coordinate Space (NOCS) - a shared canonical representation for all possible instances of objects in the category. Consequently, a regional neural network was trained to directly output the correspondence of observed pixels to this shared object representation (NOCS) along with other object information such as class label and instance mask. These predictions were combined with the depth map to jointly estimate the metric 6D pose and dimensions of multiple objects in a cluttered scene. To train the network, the researchers presented a novel context-dependent technique for creating a large amount of fully annotated mixed reality data. For further refinement of the proposed model and evaluation of its performance on real data, an annotated dataset of real-world data with a wide variation of environments and instances was provided. Large-scale experiments in the work [19] demonstrate that the proposed method is capable of reliably estimating the pose and size of instances of invisible objects in a real environment, as well as achieving state-of-the-art performance on standard tests of 6D pose estimation.

The authors of paper [20] propose a one-shot approach to simultaneously detect an object in an RGB image and predict its 6D pose without requiring multiple steps or multiple hypothesis testing. A key component of the proposed method is a novel CNN-inspired architecture that directly estimates the location of 2D images of the projected vertices of the 3D bounding box of the object. The 6D pose of an object is estimated using the PnP algorithm.

In article [21], a stackable Pillar Aware Attention (PAA) module is introduced to enhance pillar feature extraction while suppressing noise in point clouds. By integrating multi-point-channel-pooling, point-wise, channel-wise, and task-aware attention into a simple module, the representation capabilities of pillar features are boosted with minimal additional computational resources. Additionally, Mini-BiFPN, a small yet effective feature network, is presented to create bidirectional information flow and multi-level cross-scale feature fusion for better integration of multi-resolution features. The proposed framework, PiFeNet, is evaluated on three popular large-scale datasets for 3D pedestrian detection, namely KITTI, JRDB, and nuScenes. It achieves state-of-the-art performance on KITTI Bird's-eye-view (BEV) and JRDB, and competitive performance on nuScenes. The approach operates in real-time, achieving 26 frames-per-second (FPS).

The solution proposed by the authors in [22], for Dense Voxel Fusion (DVF), is a sequential fusion method that generates multi-scale dense voxel features, improving salience in regions with low point density. To enhance multimodal learning, the paper demonstrates learning directly with 3D bounding box labels, avoiding noisy 2D predictions associated with detectors. Both the DVF and the multimodal learning approach can be applied to any voxel-based LiDAR backbone. The authors analyzed the DVF method, wherein the reference test of 3D car detection KITTI can be trained without inputting additional parameters, and it does not require stereo images or dense depth labels.

In paper [23], a set abstraction method named Semantics-Augmented Set Abstraction (SASA) is proposed. Technically, a binary segmentation module is first added as a side output to assist in identifying foreground points. Researchers propose a semantically guided point sampling algorithm based on estimated foreground point scores to preserve more important foreground points during down sampling. The paper provides examples of the effective application of the SASA method in practice, demonstrating its effectiveness in determining valuable points associated with foreground objects. It enhances feature learning for point-based 3D detection. Additionally, this module can easily be integrated and is capable of enhancing various point-based detectors, including single-stage and two-stage ones

This work builds upon the method initially introduced by Mochurad et al. for door handle detection [3], which serves as the foundation for the present study. Subsequent optimizations and refinements have been developed to enhance the effectiveness of this method. This paper presents a novel algorithm for enabling autonomous mobile robots to open various types of doors without human assistance. The algorithm utilizes machine learning techniques, including the YOLOv5 [4] object detection model, the RANSAC [5] iterative method for parameter estimation, and the DBSCAN clustering algorithm. Alternative clustering methods are also compared. The algorithm was tested both in simulation and on a real robot, achieving a high success rate of 95% in opening doors during 100 attempts. The door-handle detection algorithm demonstrated an average error of 1.98 mm in 10,000 samples, indicating its precision in locating the actual door handle. These results highlight the algorithm's accuracy and real-time applicability for effectively opening different types of doors autonomously. However, experimental findings have revealed several limitations of this approach. Primarily, the method relies on selecting the optimal cluster and determining its centroid as the door handle's position. This process necessitates a specific angle between the camera and the door handle. Failure to meet this requirement may result in a shifted centroid, potentially leading to unsuccessful door-opening attempts. However, finding a configuration of the robot arm's joints that consistently provides the desired angle between the camera and the door plane is not always feasible.

The contributions of this work are as follows:

1. The algorithm's performance is expected to be enhanced when applied to diverse RGBD camera angles, as opposed to the current limitation where the camera must be positioned at the same height as the door handle. The objective is to achieve improved functionality across a range of camera perspectives.
2. The algorithm's accuracy needs to be enhanced to accommodate doorknobs with varying shapes and starting angles that deviate from the standard ninety-degree orientation. Currently, the algorithm is limited to horizontally installed cylindrical doorknobs. The goal is to ensure improved performance across a wider range of doorknob shapes and orientations.
3. Efforts should be made to improve the algorithm's speed, particularly for extended durations on low-performance computers. Achieving this goal entails implementing optimizations to expedite the computational process. The aim is to enhance the algorithm's efficiency and reduce processing times, even on resource-constrained systems.

The rest of this article is organized as follows: Section 2 presents the proposed algo- rithm. Section 3 describes the test data. Section 4 describes the results in detail. Conclusions and prospects for further research are shown in the last section.

**Table 1.** An overview of related works.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Reference | Sensor | Method | Advantages | Disadvantages |
| Wang et al. [6] | RGB Camera. | YOLOV5 is applied to door-handle recognition. Trust region policy optimization. Proximal policy optimization is used in robotic arm control. | Lots of useful information about the environment | Expansive and height processing time |
| Arduengo et al. [7] | 6-axis force sensor, located on the wrist, and an RGBD camera, located on its head. | Labeling scanned points as either ground or non-ground | Lots of useful information about the environment | Expansive and height processing time |
| Stuede et al. [8] | RGBD Camera. | Divide into either road ground or obstacles based on the average height of each line segment | A method to effectively deal with obstacles of different heights on city roads | Some limitations in distinguishing dynamic obstacles such as pedestrians |
| Quintana et al. [9] | 3D laser scanner and a RGB camera | Quantized digital elevation map and grayscale reconstruction | Data processing by using existing image processing techniques | Not discuss the conversion between different scene |
| Mochurad et al. [3] | Intel Realsense D435 RGBD Camera. | Finding the door plane using RANSAC. Clustering with DBSCAN. Finding a bounding box that has a door handle using YOLOv5. | Exhibits strong reliability when both the door handle and the door meet the specified criteria. | Requires a specific angle between the camera and the door handle / door. Works poorly with non-standard door handles. |
| Wang et al.  [19] | RGBD Camera | The presented method enables category-level 6D pose and size estimation of previously unseen object instances. | A new Normalized Object Coordinate Space (NOCS) has been introduced to establish a unified space with consistent object scaling and orientation. A CNN is proposed to predict NOCS maps, which can be used alongside the depth map to estimate the complete metric 6D pose and size of unseen objects using a pose fitting method. | In the approach, pose estimation is conditioned on the region proposals and category prediction, which could be incorrect and negatively affect the results. The approach relies on the depth image to elevate NOCS prediction to real-world coordinates. |
| Tekin et al.  [20] | RGBD Camera and active depth sensors | A deep CNN architecture is proposed, which takes an image as input and directly determines the 2D projections of the vertices of the 3D bounding box. | A new CNN architecture is proposed for fast and single-shot accurate 6D pose prediction, which naturally extends the single-frame 2D object detection paradigm to a 6D object. The network predicts the 2D location of the 3D bounding box corner projections of the objects, which involves predicting only a few more 2D points than for 2D bounding box regression | Upon a single network invocation, the only computational overhead is an efficient PnP algorithm that operates on just 9 points per object |
| Le et al.  [21] | 3D object detection from LiDAR point cloud | A stackable Pillar Aware Attention (PAA) module the network Mini-BiFPN, proposed framework PiFeNet, | The introduced Pillar Aware Attention Module combines multi-point channel pooling, point-wise, channel-wise, and task-aware attention to better extract pillar features. Next is the Mini-BiFPN module, a lightweight feature network that leverages cross-scale feature fusion and bidirectional connections, enriching information flow in the feature network. | The PAA modules improve accuracy, demonstrating the effectiveness of the PAA module. However, when stacking three or more PAA modules, the performance improvement rate slightly decelerates, and the model requires more time to run. |
| Mahmoud et al.  [22] | Camera and LiDAR sensor | They propose Dense Voxel Fusion (DVF), a sequential fusion method that first assigns voxel centers to the 3D location of the occupied LiDAR voxel features | DVF generates multi-scale multi-modal dense voxel feature representations, improving expressiveness in regions with low point density. DVF can be applied to any voxel-based LiDAR backbone without introducing additional learnable parameters | Experiments with ground truth (GT) 2D labels demonstrate the significant gain of DVF models over sparse methods, showing improvements in 2D object detectors. |
| Chen et al.  [23] | LiDAR sensors | An abstraction method named Semantics-Augmented Set Abstraction (SASA) is proposed | The proposed method offers a promising direction for point-based detection. It can not only be implemented in PointNet-based models but is also compatible with transformer-based networks and graph neural networks for model reduction. | The effectiveness of SASA relies on accurate identification of foreground points, which may be challenging in scenarios with complex backgrounds or occlusions. |

2 Materials and Methods

*2.1 Intel Realsense D435*

The Intel RealSense D435 [10] (see fig. 4) is a depth-sensing camera designed for a wide range of applications, including robotics, augmented reality, 3D scanning, and computer vision tasks. It is part of Intel's RealSense series of depth cameras, which utilize a combination of infrared sensors and cameras to capture depth information.



Figure 1. Intel Realsense D435.

The D435 camera features a stereo vision system with two integrated infrared cameras and an RGB camera. It uses active infrared stereo technology to calculate depth by analyzing the disparity between the images captured by the two infrared cameras. This depth information can be used to create 3D point clouds, detect objects, track motion, and enable various other depth-based applications. The camera has a wide field of view, capturing a large area in a single frame. It can provide depth data with a resolution of up to 1280 x 720 pixels and a frame rate of up to 90 frames per second (FPS). The RGB camera captures color images with a resolution of up to 1920 x 1080 pixels. To work with the Intel RealSense D435 camera, Intel provides the RealSense SDK, which includes a set of libraries, tools, and APIs for developing applications that leverage the camera's capabilities. The SDK provides access to depth data, color images, and other sensor streams, allowing developers to create depth-aware applications and perform tasks such as object recognition, gesture recognition, and 3D reconstruction. Overall, the Intel RealSense D435 is a versatile depth-sensing camera that enables developers and researchers to integrate depth perception into their projects, opening up possibilities for advanced computer vision and augmented reality applications.

Converting a depth image to a point cloud involves associating each depth value in the image with a 3D point in space. This process requires the intrinsic parameters of the camera and the depth information from the depth image. The intrinsic parameters typically include the focal length ( and ), the principal point ( and ), and sometimes distortion coefficients. The depth image provides the distance information for each pixel.

To convert a depth image to a point cloud, you can follow these steps:

1. Obtain Depth Information: Load the depth image and extract the depth values for each pixel. Each pixel in the depth image represents the distance from the camera to the corresponding point in the scene.
2. Calculate 3D Points: For each depth value at pixel , you can calculate the 3D point using the following equations:  
   where  and  are the focal lengths,  and  are the principal points.
3. Apply Camera Transformation: If needed, apply any transformations (e.g., rotation, translation) to align the point cloud with the camera's reference frame.

The Intel RealSense D435 is a depth-sensing camera that provides depth and RGB information. The intrinsic parameters for the RealSense D435 camera are typically provided by the manufacturer, Intel. Here are the common intrinsic parameters for the RealSense D435:

1. Focal Length (in pixels):
   * (Focal length along the x-axis): Approximately 617.3
   * (Focal length along the y-axis): Approximately 617.3
2. Principal Point (in pixels):
   * (Principal point along the x-axis): Approximately 323.0
   * (Principal point along the y-axis): Approximately 237.1

It's important to note that these values are approximate and may vary slightly between different RealSense D435 cameras or firmware versions. For precise calibration and applications, it's recommended to calibrate the camera using a calibration target and appropriate calibration software. Additionally, the RealSense SDK [11] provides a calibration tool that allows for more accurate calibration and extraction of the intrinsic parameters, including distortion coefficients.

*2.1 Camera and object global isometry*

To determine the position of the object within the global coordinate system, knowledge of the camera's spatial placement in said coordinate system is imperative. This necessitates acquaintance with the rotation matrix denoted as and the displacement vector , which collectively define the camera's orientation and position. Consequently, a transformation function is formulated, allowing the translation of a point p from the camera's coordinate system to the overarching global coordinate system.

(2)

*2.2 RANSAC*

RANSAC [5] stands for Random Sample Consensus. It is an iterative algorithm commonly used in computer vision and image analysis for robust parameter estimation. RANSAC is particularly useful when dealing with datasets containing outliers or corrupted data points. The RANSAC algorithm works by randomly selecting subsets of data points, fitting a model to each subset, and then evaluating the goodness-of-fit of each model based on a predefined threshold. The process is repeated for a fixed number of iterations, and the model with the highest number of inliers (data points that fit the model within the threshold) is considered the best estimate. By iteratively selecting and evaluating subsets of data points, RANSAC is able to robustly estimate model parameters even in the presence of outliers. It is commonly used for tasks such as line fitting, plane fitting, and geometric transformation estimation. Overall, RANSAC provides a robust and reliable method for estimating model parameters from datasets containing outliers or noisy data points.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Algorithm 1** RANSAC Algorithm | | | | | |
| **Require:** | | | | | |
| **Ensure:** | | | | | |
|  | **function** | | | | |
| 1: |  | Select random points from and assign them to | | | |
| 2: |  |  | | | |
| 3: |  |  | | | |
| 4: |  | **for**  **to** | | | |
| 5: |  |  | Select 3 random points from set | | |
| 6: |  |  |  | | |
| 7: |  |  |  | | |
| 8: |  |  |  | | |
| 9: |  |  | **for**  **to** | | |
| 10: |  |  |  | Assign the distance to the x s | |
| 11: |  |  |  | **if** | |
| 12: |  |  |  |  |  |
| 13: |  |  | **if** | | |
| 14: |  |  |  |  | |
| 15: |  |  |  |  | |
| 16: |  | **return** | | | |

The time complexity analysis of the RANSAC algorithm centers on the iterations essential for attaining a reliable model fit. The algorithm prominently features a loop indexed by variable (line 4) and a nested loop over (line 9). Consequently, the overall computational complexity is bounded by . Additionally, it's pertinent to acknowledge that the RANSAC algorithm lends itself to effective parallelization [12], presenting an avenue to expedite the overall computational process.

*2.3 DBSCAN*

DBSCAN [13] (Density-Based Spatial Clustering of Applications with Noise) is a popular clustering algorithm used in data mining and machine learning. It is designed to discover clusters of arbitrary shape in a dataset, based on the density of data points. DBSCAN works by defining neighborhoods around each data point and identifying core points, which have a sufficient number of neighboring points within a specified radius. It then expands the clusters by including directly reachable points, which are within the specified radius of a core point, and transitively reachable points that are within the radius of other points in the cluster. One key feature of DBSCAN is its ability to handle noise and outliers. Data points that do not belong to any cluster and do not meet the density criteria are labeled as noise points. This allows DBSCAN to separate clusters from noise in the data. The algorithm requires two parameters: the radius that determines the neighborhood size and the minimum number of points required to form a core point. These parameters influence the sensitivity of the algorithm to density and shape of clusters. DBSCAN has several advantages, including its ability to handle clusters of varying shapes and sizes, its resistance to noise and outliers, and its efficiency in processing large datasets. However, it may struggle with datasets of varying densities or clusters with significantly different densities. Overall, DBSCAN is a powerful clustering algorithm that can effectively identify clusters based on data point density, making it a valuable tool for exploratory data analysis and pattern recognition tasks.

Here are the key parameters of the DBSCAN algorithm [14]:

1. : the radius within which the algorithm searches for neighboring points. Points within this radius are considered neighbors of a given point.
2. : the minimum number of data points required to form a dense region. A dense region is a cluster if there are at least `min\_samples` data points within a distance of `eps` from a core point.
3. Metric: the distance metric used to calculate distances between points. Common choices include Euclidean distance, Manhattan distance, or other distance metrics.

These parameters influence how the DBSCAN algorithm forms clusters based on the density of the data points in the feature space. It's important to experiment with different parameter values to find the most appropriate settings for a specific dataset and clustering task. Euclidean distance was chosen as the metric. In the future, this algorithm will be represented as a function , where is a set of points.

*2.4 Finding the Position of the Object*

Upon determining the plane's normal, our subsequent objective involves identifying the object's orientation adjusted such that the normal aligns with the z-axis. This process entails incrementally rotating the object about the z-axis in discrete steps and subsequently evaluating its projections onto the x-axis and y-axis. The optimal rotation is defined as the one where the projections accurately reflect the object's dimensions, specifically its width and height.

To ascertain the most suitable orientation, it is imperative to employ an algorithm capable of identifying a segment, not exceeding a maximum length , that encompasses the maximal concentration of points. This methodology is crucial for achieving precise alignment and orientation of the object relative to its plane of placement.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm 2** Find Segment Algorithm | | | | |
| **Require:** Array of data , max segment length | | | | |
| **Ensure:** Segment begin and segment end | | | | |
|  | **function** | | | |
| 1: |  | Sort the data from and assign it to | | |
| 2: |  |  | | |
| 3: |  |  | | |
| 4: |  | **while**  **do** | | |
| 5: |  |  | **while**  **and** **do** | |
| 6: |  |  |  |  |
| 7: |  |  | **if** | |
| 8: |  |  |  |  |
| 9: |  |  |  |  |
| 10: |  |  |  |  |
| 11: |  | **return** | | |

The computational complexity of Algorithm 2 is . Despite the presence of nested loops, where ranges from 1 to and iterates through , it's noteworthy that each index in these loops only accesses the elements once.

*2.6 Full Algorithm of The Object Recognition*

|  |  |
| --- | --- |
| **Algorithm 3** Find The Object Isometry Algorithm | |
| **Require:**   * Depth image , where and are the height and width of the image; * Intrinsic parameters ; * Camera in world base rotation and translation ; * RANSAC parameters and ; * Minimum distance to the plane * DBSCAN parameters and * Approximate length and width of the object | |
| **Ensure:** Isometry of the object in world base | |
| 1: | Convert a depth image into a set of points using camera intrinsic parameters |
| 2: |  |
| 3: |  |
| 4: | Remove all points from that are closer to by |
| 5: |  |
| 6: | Set as cluster with maximum points count |
| 7: |  |
| 8: |  |
| 9: | Make where rotation is and translation is center of |
| 10: | **return** |

Algorithm 3 represents an amalgamation of the previously discussed algorithms, culminating in the derivation of the isometric transformation for the object. Specifically, the object's location is ascertained as the midpoint of the segment provided by Algorithm 2, while its orientation is determined by the normal vector of the plane yielded by Algorithm 1.

3. Experiments

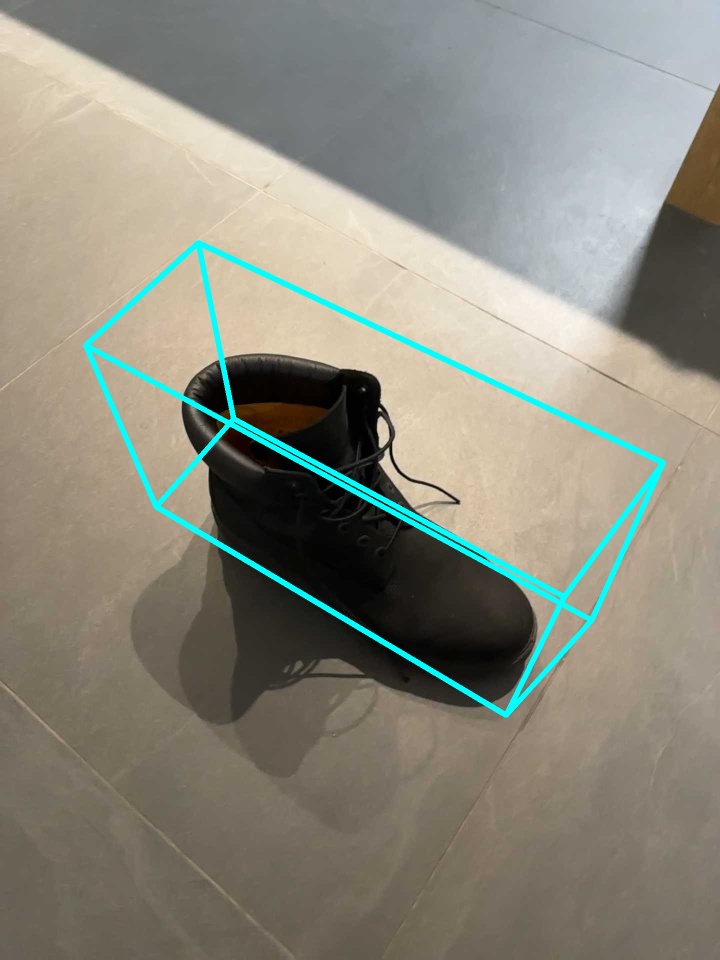


Figure 2. RGB image of the first case with result visualization.

In Figure 2, the depicted object for identification is a shoe, encapsulated within a bounding box delineated in blue, demonstrating the output of the detection algorithm. The presence of the shoe on the floor provides an ideal scenario for the application of the RANSAC algorithm, facilitating the determination of the object's orientation. This environment leverages the plane's visibility on the depth camera, allowing the RANSAC algorithm to efficiently segment the shoe from the planar background, thereby enhancing the accuracy of orientation estimation.

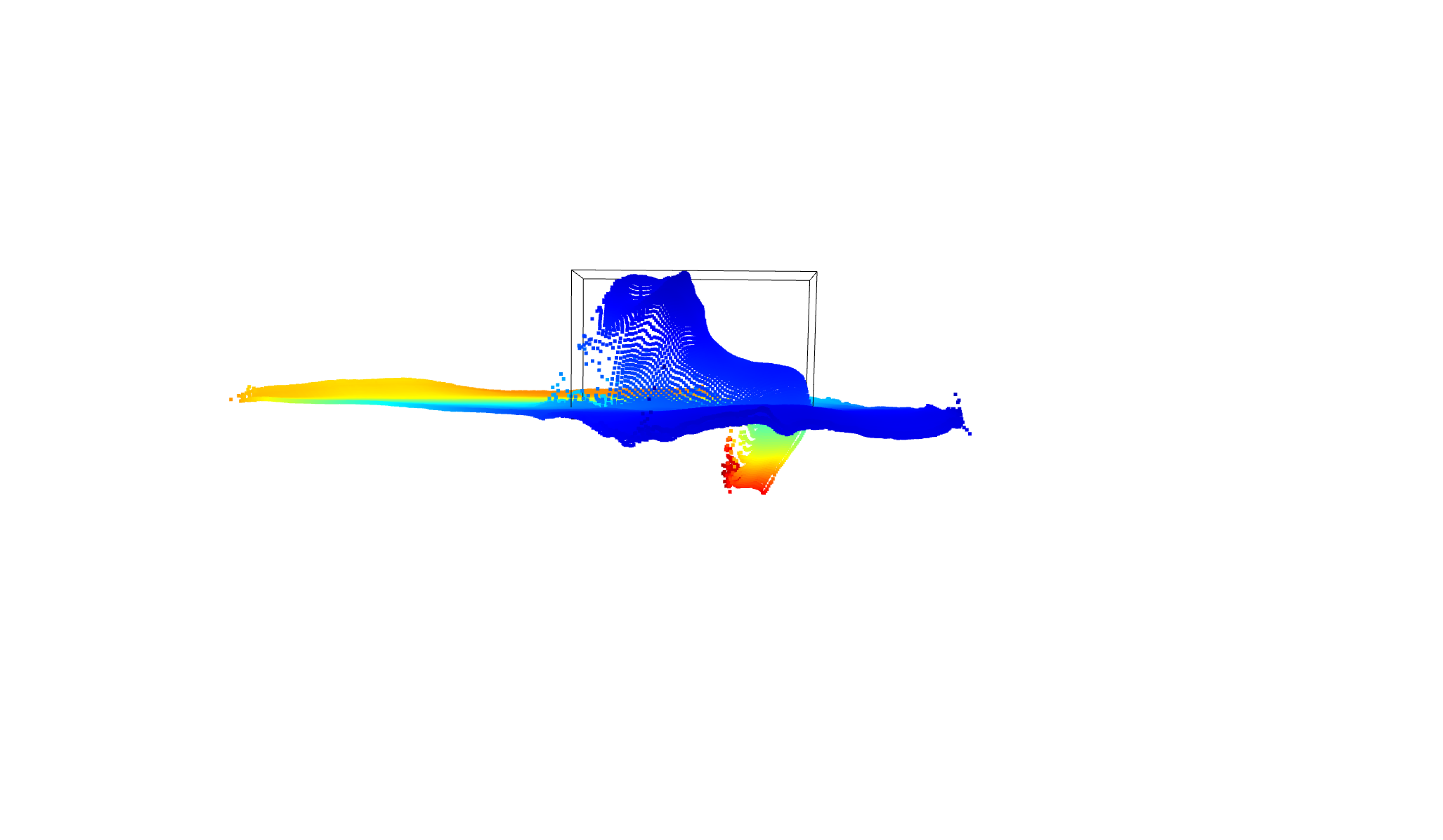


Figure 3. Side view for the detector result in the first case.

Figure 3 illustrates the point cloud representation alongside the detection outcome. The chromatic gradient towards blue signifies proximity to the camera at the capture moment, indicating depth. The algorithm's output is marked in black, delineating the object with a bounding box. This visualization emphasizes the spatial distribution and depth perception of the scene, enabling a precise demarcation of the target object within the three-dimensional space captured by the camera.

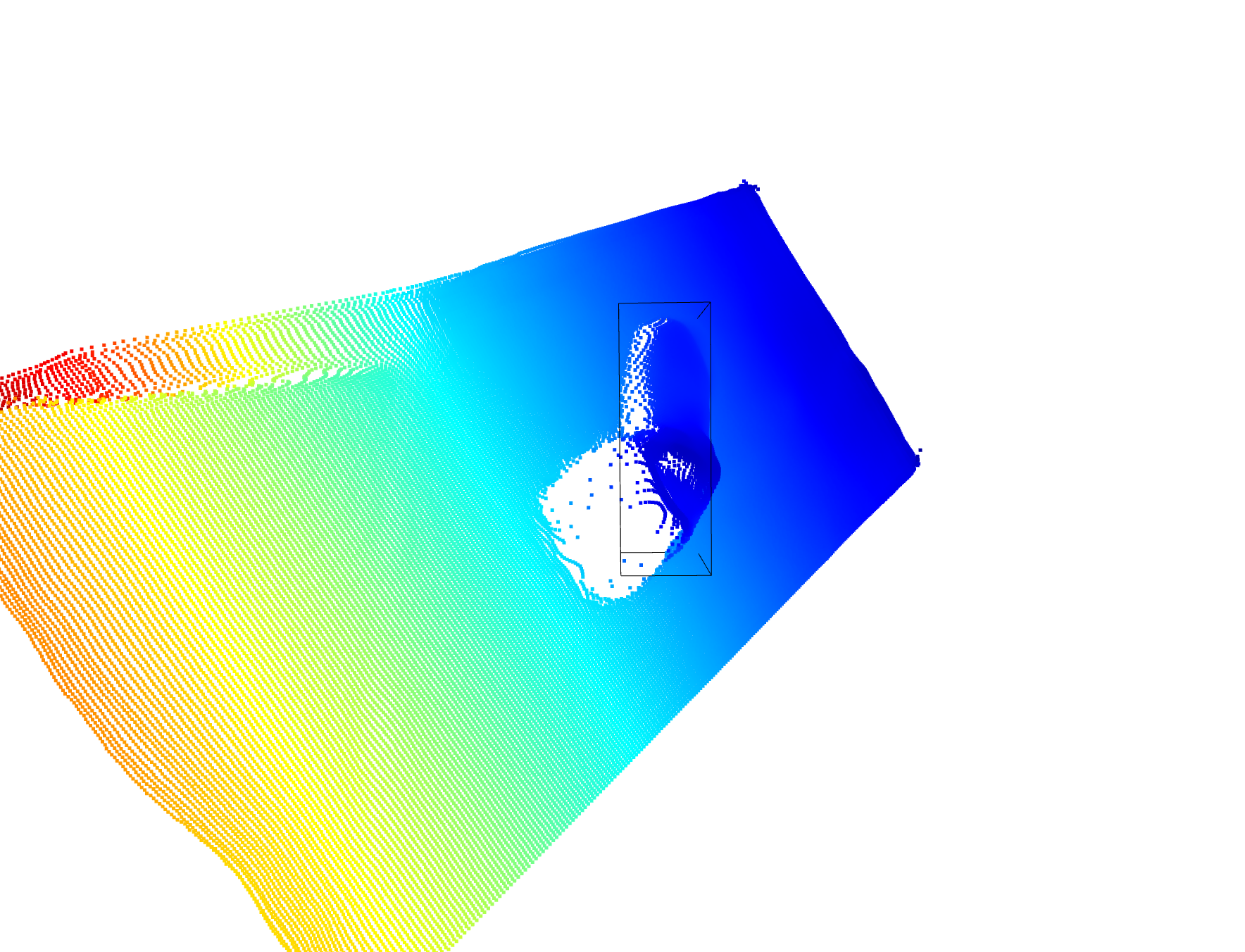


Figure 4. Top view for the detector results in the first case.

On the other hand, Figure 4 serves as a top-view representation of the same scenario, illuminating the algorithm's interplay within the x-y plane for comprehensive comprehension.

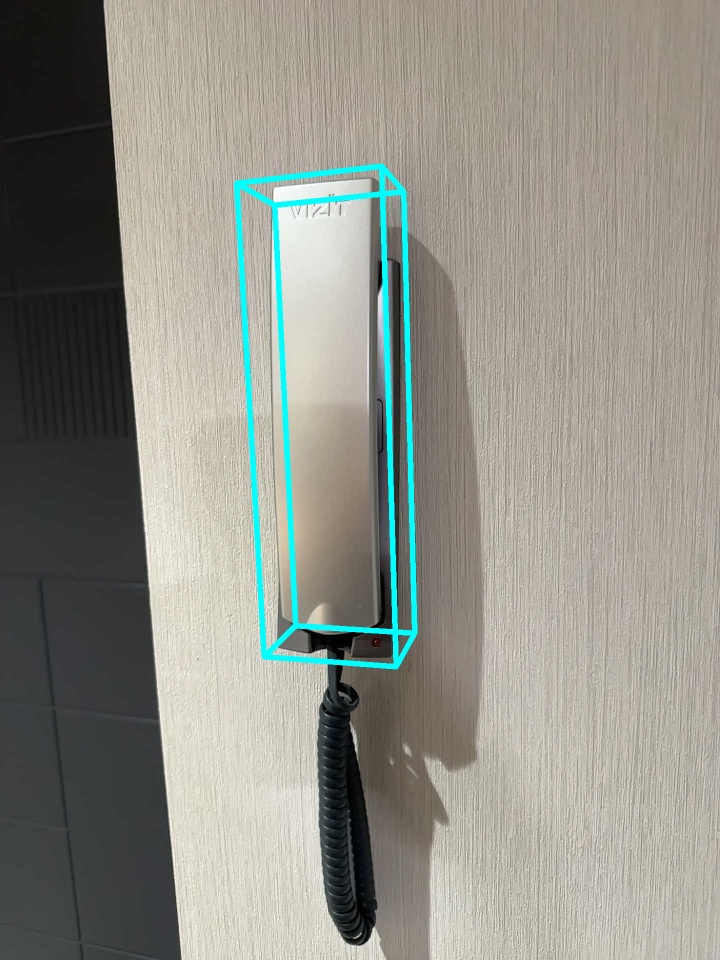


Figure 5. RGB image of the second case.

In Figure 5, the efficacy of the algorithm is demonstrated in Case 2, where an intercom serves as the detection object. The algorithm's successful performance in this scenario is evident, showcasing its capability to accurately identify and localize the intercom within the given environment. This result underscores the algorithm's versatility and effectiveness across diverse object types and detection scenarios.

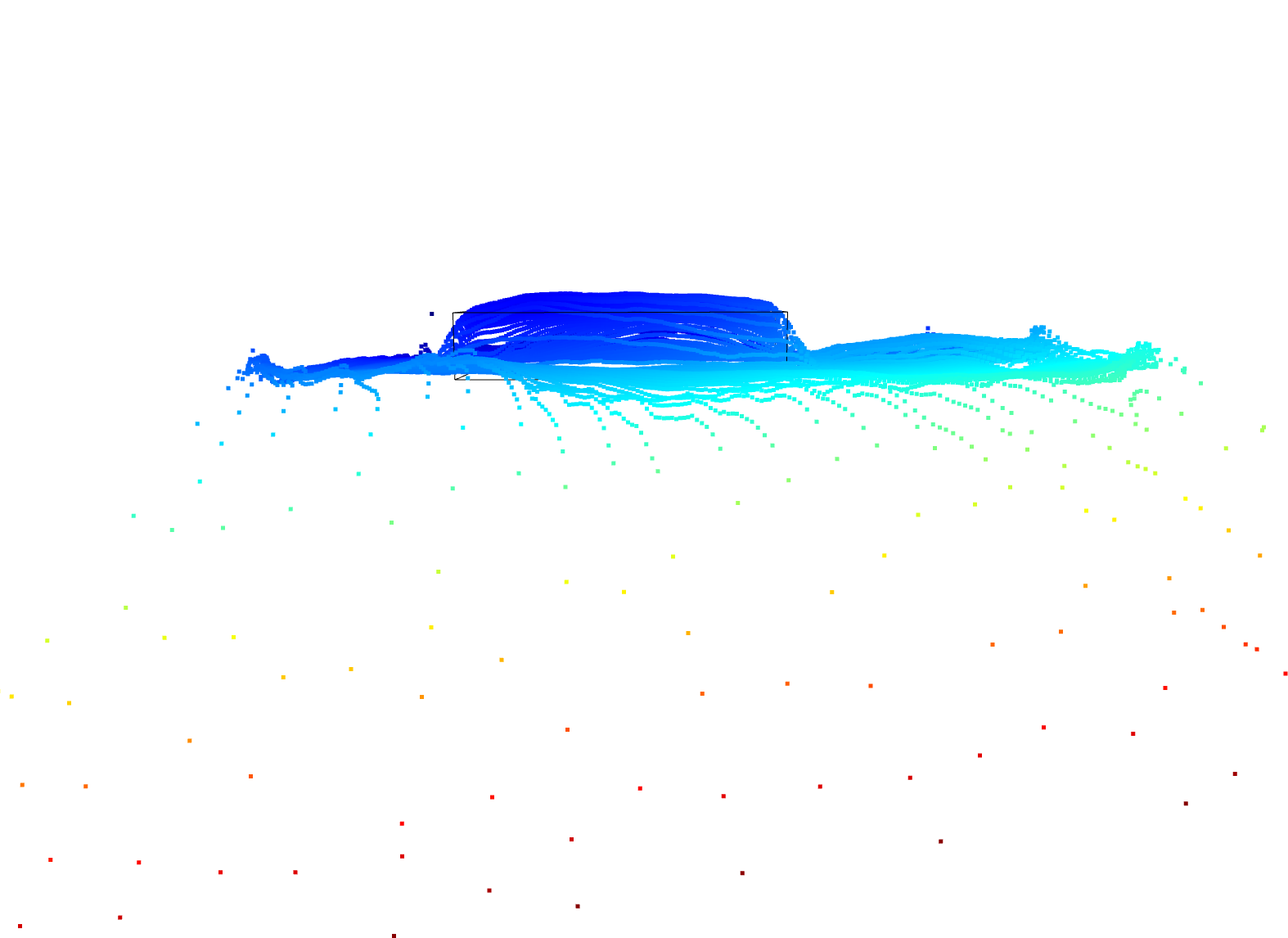


Figure 6. Side view for the detector results in the second case.

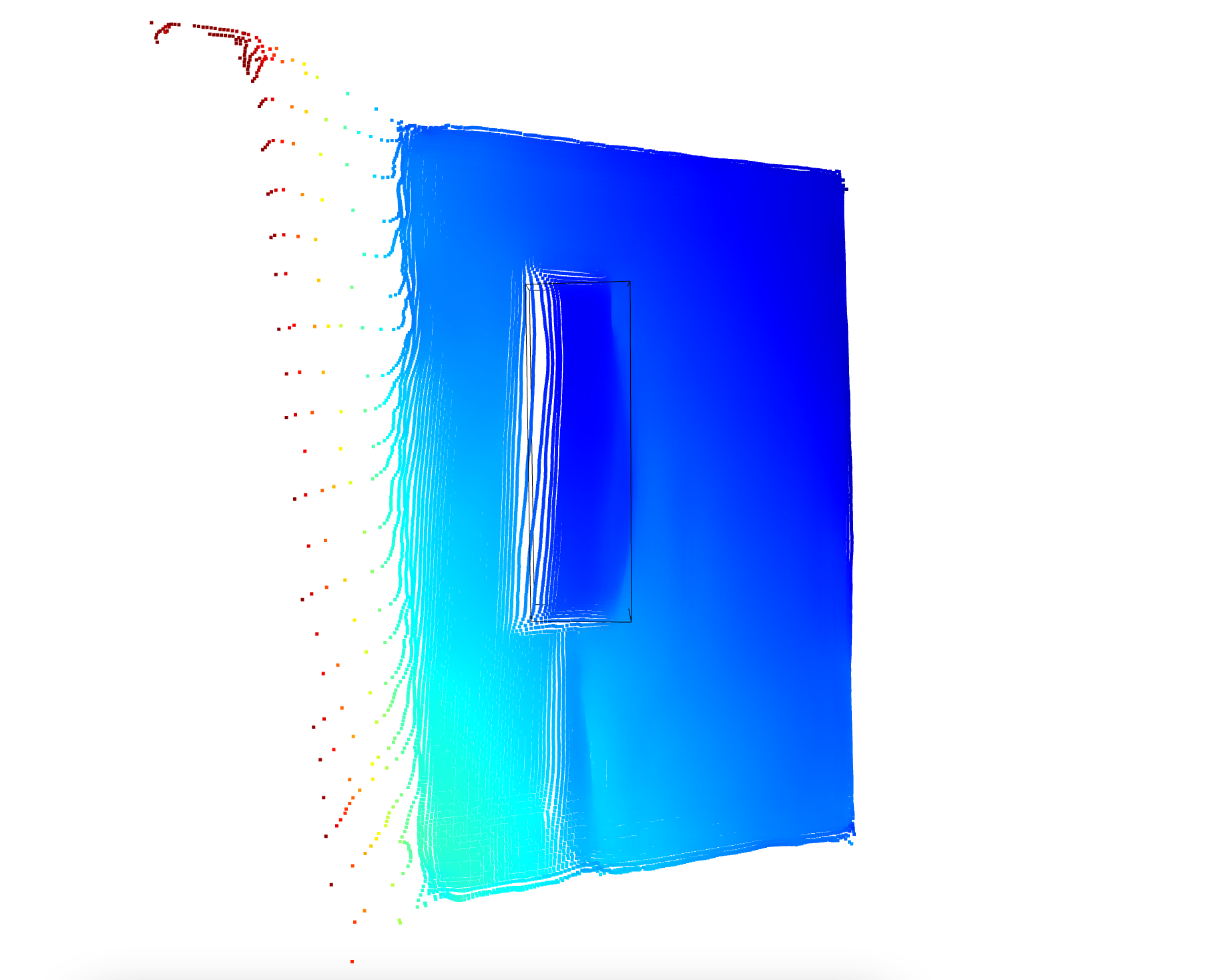


Figure 7. Top view for the detector results in the second case.

Utilizing identical hardware configurations, the performance assessment reveals that the algorithm surpasses the swiftest of the evaluated counterparts in terms of execution speed. Notably, the MobilePoseBase model registers an inference time of 25 milliseconds, whereas the newly proposed algorithm achieves a more rapid processing time of 20 milliseconds. However, this efficiency is accompanied by a notable constraint: the necessity for a depth camera installation. Moreover, for the algorithm to accurately calculate the object's orientation, it is imperative that the object be positioned on a plane. This requirement delineates a significant limitation, indicating a trade-off between enhanced speed and operational constraints related to environmental setup and equipment.  
 Despite the hardware similarities, the algorithm introduces a significant advancement in precision. While MobilePoseBase attained an accuracy level of 4 cm, our algorithm demonstrated a substantial improvement, achieving an accuracy of 50 mm. This comparison highlights the algorithm's superior capability in delivering precise outcomes, marking a notable enhancement in performance metrics. The distinction in accuracy underscores the technological advancement embodied by our algorithm, positioning it as a considerably more accurate solution in applications where precision is paramount.

4. Conclusions

The algorithm's effectiveness has significantly advanced, now accommodating a broader array of RGBD camera angles. This improvement overcomes the previous constraint that necessitated the camera to be aligned with the object's height on the plane background, broadening the range of viable camera positions and enhancing overall performance. Notably, the algorithm's execution time for identifying the object has been optimized, with a remarkable 2-fold speed increase achieved by bypassing complex machine learning models.

Additionally, the algorithm's precision has been refined to identify objects of various shapes and initial positions, moving beyond the earlier limitation to horizontally oriented cylindrical objects. This upgrade allows for superior performance and accuracy across a wider array of object shapes and orientations.

Efforts to accelerate the algorithm, particularly for prolonged use on lower-performance computers, have resulted in a similar 2-fold speed improvement, again due to the exclusion of resource-heavy machine learning frameworks.

These advancements have rendered the object detection algorithm more versatile, precise, and efficient, paving the way for broader application and utility in diverse settings and circumstances.

5. Discussions

Detecting the origin or position of an object using either RGB (color) or depth image involves various approaches depending on the specific context and requirements. Here are some top approaches for object origin detection using either RGB or depth images:

1. Object Center Calculation (RGB Image):

* Approach: Calculate the centroid or center of mass of the object within the 2D bounding box obtained from object detection models (e.g., YOLO, Faster R-CNN) using RGB images.
* Advantages: Simple and fast; provides a quick estimate of the object's position in the image.
* Considerations: May not be accurate if the object is not symmetrical or if the bounding box is imprecise.

1. Depth-based Center of Mass (Depth Image):
   * Approach: Calculate the 3D center of mass of the object by considering depth values within the object's region in the depth image.
   * Advantages: Provides a 3D estimate of the object's position, accounting for depth information.
   * Considerations: Sensitive to noise in depth data, requires accurate depth information.
2. Point Cloud Processing (RGB-D Data Fusion):
   * Approach: Utilize the RGB-D camera to obtain a 3D point cloud of the scene. Process the point cloud to segment and identify the object, then compute the centroid or center of mass.
   * Advantages: Leverages both color and depth information for accurate 3D object localization.
   * Considerations: Processing of point clouds can be computationally intensive, and accuracy depends on the quality of the point cloud.
3. Geometric Shape Fitting (Depth Image):
   * Approach: Fit geometric shapes (e.g., spheres, cylinders) to the object's shape in the depth image to estimate its center and dimensions. [17]
   * Advantages: Can provide accurate estimates of the object's position and shape.
   * Considerations: Requires knowledge or assumptions about the object's shape and may be sensitive to noise or occlusions.
4. Machine Learning-based Approaches (RGB or Depth Image):
   * Approach: Train machine learning models (e.g., CNNs, SVMs) to predict the object's position or center based on RGB or depth images as input. [18]
   * Advantages: Can learn complex features for accurate localization; adaptable to various object types.
   * Considerations: Requires a labeled dataset for training, computational resources for model training, and may need significant data for robust performance.

The strategy for application selection depends on various factors including the specific application requirements, available data (either RGB, depth, or both), computational resources, and the desired accuracy level. Enhanced object detection results can often be achieved by integrating multiple approaches or utilizing both RGB and depth data through fusion-based techniques. This research emphasizes the development of a machine learning algorithm specifically tailored to accurately determine the precise position and orientation of objects against a plane background. This machine learning model is envisioned as a key component within a broader methodological framework, addressing the challenge of objects with relief patterns, which render conventional algorithms like RANSAC ineffective. The algorithm also aims to accommodate a wide variety of object types, overcoming the limitations of prior systems that were restricted to specific shapes or orientations. This requires a carefully considered approach, utilizing machine learning algorithms to achieve a generalized solution that effectively manages the diversity of object types and their orientations.

References

[1] T. Hou, A. Ahmadyan, L. Zhang, J. Wei, and M. Grundmann, “MobilePose: Real-Time Pose Estimation for Unseen Objects with Weak Shape Supervision,” 2020, doi: 10.48550/ARXIV.2003.03522.

[2] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, “DeepIM: Deep Iterative Matching for 6D Pose Estimation,” 2018, doi: 10.48550/ARXIV.1804.00175.

[3] L. Mochurad, Y. Hladun, Y. Zasoba, and M. Gregus, “An Approach for Opening Doors with a Mobile Robot Using Machine Learning Methods,” *BDCC*, vol. 7, no. 2, p. 69, Apr. 2023, doi: 10.3390/bdcc7020069.

[4] G. Jocher *et al.*, “ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation.” Zenodo, Nov. 22, 2022. doi: 10.5281/ZENODO.7347926.

[5] C. Papazov and D. Burschka, “An Efficient RANSAC for 3D Object Recognition in Noisy and Occluded Scenes.,” 2010, doi: 10.13140/2.1.1451.1041.

[6] Y. Wang, L. Wang, and Y. Zhao, “Research on Door Opening Operation of Mobile Robotic Arm Based on Reinforcement Learning,” *Applied Sciences*, vol. 12, no. 10, p. 5204, May 2022, doi: 10.3390/app12105204.

[7] M. Arduengo, C. Torras, and L. Sentis, “Robust and adaptive door operation with a mobile robot,” *Intel Serv Robotics*, vol. 14, no. 3, pp. 409–425, Jul. 2021, doi: 10.1007/s11370-021-00366-7.

[8] M. Stuede, K. Nuelle, S. Tappe, and T. Ortmaier, “Door opening and traversal with an industrial cartesian impedance controlled mobile robot,” in *2019 International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada: IEEE, May 2019, pp. 966–972. doi: 10.1109/ICRA.2019.8793866.

[9] B. Quintana, S. A. Prieto, A. Adan, and F. Bosche, “Door detection in 3D colored laser scans for autonomous indoor navigation,” in *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Alcala de Henares, Spain: IEEE, Oct. 2016, pp. 1–8. doi: 10.1109/IPIN.2016.7743677.

[10] M. S. Ahn, H. Chae, D. Noh, H. Nam, and D. Hong, “Analysis and Noise Modeling of the Intel RealSense D435 for Mobile Robots,” in *2019 16th International Conference on Ubiquitous Robots (UR)*, Jeju, Korea (South): IEEE, Jun. 2019, pp. 707–711. doi: 10.1109/URAI.2019.8768489.

[11] “Intel RealSense Documentation - Get Started,” Intel® RealSenseTM Developer Documentation. Accessed: Sep. 16, 2023. [Online]. Available: https://dev.intelrealsense.com/docs

[12] A. Hidalgo-Paniagua, M. A. Vega-Rodríguez, N. Pavón, and J. Ferruz, “A Comparative Study of Parallel RANSAC Implementations in 3D Space,” *Int J Parallel Prog*, vol. 43, no. 5, pp. 703–720, Oct. 2015, doi: 10.1007/s10766-014-0316-7.

[13] H. Chen, M. Liang, W. Liu, W. Wang, and P. X. Liu, “An approach to boundary detection for 3D point clouds based on DBSCAN clustering,” *Pattern Recognition*, vol. 124, p. 108431, Apr. 2022, doi: 10.1016/j.patcog.2021.108431.

[14] “sklearn.cluster.DBSCAN,” scikit-learn. Accessed: Sep. 17, 2023. [Online]. Available: https://scikit-learn/stable/modules/generated/sklearn.cluster.DBSCAN.html

[15] “Depth Post-Processing for Intel® RealSenseTM Depth Camera D400 Series.” Accessed: Sep. 29, 2023. [Online]. Available: https://dev.intelrealsense.com/docs/depth-post-processing

[16] “librealsense/examples/align at master · IntelRealSense/librealsense · GitHub.” Accessed: Sep. 29, 2023. [Online]. Available: https://github.com/IntelRealSense/librealsense/tree/master/examples/align#overview

[17] S. Chandra, G. G. Chrysos, and I. Kokkinos, “Surface Based Object Detection in RGBD Images,” in *Procedings of the British Machine Vision Conference 2015*, Swansea: British Machine Vision Association, 2015, p. 187.1-187.13. doi: 10.5244/C.29.187.

[18] S. Zia, B. Yuksel, D. Yuret, and Y. Yemez, “RGB-D Object Recognition Using Deep Convolutional Neural Networks,” in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, Venice: IEEE, Oct. 2017, pp. 887–894. doi: 10.1109/ICCVW.2017.109

1. H. Wang, S. Sridhar, J. Huang, J. Valentin, Sh. Song, L. J. Guibas, "Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation," in *arXiv:1901.02970*, 2019. https://doi.org/10.48550/arXiv.1901.02970
2. B. Tekin, S. N. Sinha, P. Fua, "Real-Time Seamless Single Shot 6D Object Pose Prediction," in a*rXiv:1711.08848*, 2017. https://doi.org/10.48550/arXiv.1711.08848
3. D.-T. Le, H. Shi, H. Rezatofighi, J. Cai, "Accurate and Real-time 3D Pedestrian Detection Using an Efficient Attentive Pillar Network," in *arXiv:2112.15458*, 2022. https://doi.org/10.48550/arXiv.2112.15458
4. A. Mahmoud, J. S. K. Hu, S. L. Waslander, "Dense Voxel Fusion for 3D Object Detection," in *arXiv:2203.00871*, 2022. https://doi.org/10.48550/arXiv.2203.00871
5. Ch. Chen, Z. Chen, J. Zhang, D. Tao, "SASA: Semantics-Augmented Set Abstraction for Point-based 3D Object Detection," in *arXiv:2201.01976*, 2022. https://doi.org/10.48550/arXiv.2201.01976