machines

MDPI

Article

Obstacle Detection for Autonomous Guided Vehicles through Point Cloud Clustering Using Depth Data

Micael Pires [1], Pedro Couto [1,2,*], António Santos [3] and Vítor Filipe [1,4]

[1] School of Science and Technology, University of Trás-os-Montes e Alto Douro (UTAD), 5000-801 Vila Real, Portugal; piresmicael@gmail.com (M.P.); vfilipe@utad.pt (V.F.)
[2] CITAB—Centre for the Research and Technology of Agro-Environmental and Biological Sciences, Quinta de Prados, 5000-801 Vila Real, Portugal
[3] Active Space Technologies, Parque Industrial de Taveiro, Lote 12, 3045-508 Coimbra, Portugal; antonio.santos@activespacetech.com
[4] Institute for Systems and Computer Engineering Technology and Science (INESC TEC), 4200-465 Porto, Portugal
* Correspondence: pcouto@utad.pt

Abstract: Autonomous driving is one of the fastest developing fields of robotics. With the ever-growing interest in autonomous driving, the ability to provide robots with both efficient and safe navigation capabilities is of paramount significance. With the continuous development of automation technology, higher levels of autonomous driving can be achieved with vision-based methodologies. Moreover, materials handling in industrial assembly lines can be performed efficiently using automated guided vehicles (AGVs). However, the visual perception of industrial environments is complex due to the existence of many obstacles in pre-defined routes. With the INDTECH 4.0 project, we aim to develop an autonomous navigation system, allowing the AGV to detect and avoid obstacles based in the processing of depth data acquired with a frontal depth camera mounted on the AGV. Applying the RANSAC (random sample consensus) and Euclidean clustering algorithms to the 3D point clouds captured by the camera, we can isolate obstacles from the ground plane and separate them into clusters. The clusters give information about the location of obstacles with respect to the AGV position. In experiments conducted outdoors and indoors, the results revealed that the method is very effective, returning high percentages of detection for most tests.

Keywords: obstacle detection; RANSAC algorithm; Euclidean clustering; 3D point cloud; RGB-D data

Citation: Pires, M.; Couto, P.; Santos, A.; Filipe, V. Obstacle Detection for Autonomous Guided Vehicles through Point Cloud Clustering Using Depth Data. *Machines* **2022**, *10*, 332. https://doi.org/10.3390/machines10050332

Academic Editors: Sergey Y. Yurish and Dan Zhang

Received: 26 March 2022
Accepted: 30 April 2022
Published: 2 May 2022

1. Introduction

Automation is one of the fastest-developing fields of robotics. With the ever-growing interest in automating processes, concerning both efficiency and safety, the ability to provide robots with autonomous navigation capabilities is of paramount significance [1,2].

Several sensing technologies, types of navigation systems and data fusion are available for the autonomous navigation of mobile robots. A key aspect of mobile autonomous robots is the ability to detect objects in their path and allow them to navigate freely with minimum risk of collision. Thus, in industrial environments, AGVs are usually provided with laser scanners to prevent them from running into people or objects. However, the effectiveness of these systems outdoors can be affected by adverse weather conditions such as direct sunlight, rain, snow or fog. Even considering the use of laser scanners prepared to operate outdoors, other vehicles (e.g., forklifts, truck trailers) may not be detected, possibly leading to collisions. To mitigate this problem, 3D point clouds captured with a depth camera could be used in these complex conditions. With 3D point clouds, more than ten thousand points of information, including 3D coordinates and reflection intensities, can be obtained in a short streaming sequence. Unlike RGB images, the significant advantages of point clouds are the obvious 3D spatial structures that also include reflection features about the

surface of objects [3]. A large number of automated reconstruction approaches focus on the unsupervised processing of point clouds. The interpretation of these data is challenging due to the number of points, noise and the complexity of the structure [4]. Moreover, most point clouds are acquired with remote sensing techniques associated to line of sight (LoS). As a result, crucial parts of the structure are occluded due to clutter or inaccessible areas [5].

There are different ways to perform object detection using 3D point clouds. Most solutions tackle this problem with neural networks adapted to this kind of data. Several methods, depending on the type of deep learning network inputs that can be raw point cloud data [6], voxel [7] or image [8], have been proposed for 3D object detection. Shi et al. [6] proposed PointRCNN for 3D object detection from raw point clouds. The whole framework is composed of two stages: stage one for the bottom-up 3D proposal generation and stage two for refining proposals in the canonical coordinates to obtain the final detection results. Those results showed that the proposed architecture outperforms state-of-the-art methods with remarkable margins by using only the point cloud as input. An important study in this type of method by Zhou et al. presented the VoxelNet algorithm [9], which works with sparse 3D point clouds without the need of a manual feature selection. Further studies have led to the development of other algorithms (e.g., Qi et al. [10]) that use RGB images to generate detection proposals for 3D detection. However, ConvNets-based models generally require a large amount of data for effective training.

Ground plane extraction and obstacle detection are essential tasks for collision-free navigation of autonomous mobile robots [11]. One of the most common methods to detect the ground plane in a scene is based in processing depth information. To address the ground plane detection, Choy et al. [12] proposed a robust and fast ground plane detection with an asymmetric kernel and RANSAC (random sample consensus). A probabilistic model of 3D point clouds with the assumption that a point from other objects is always above the ground was derived. The asymmetric kernel is the key approximation for fast computation, which incorporates RANSAC as a score function. Experimental results showed that the method was sufficiently accurate.

After ground plane extraction, objects can be detected in three-dimensional data by using point cloud segmentation, which is the method explored in this work. Segmentation is one of the most important steps for the automatic processing of point clouds. It is a process of classifying and labelling data points into a set of separate groups or regions, each one ideally corresponding to an object. An example of this kind of methodology is discussed by Habermann et al. in [13], where they presented three different point cloud segmentation methods and compared their performance with the k-means segmentation algorithm. Miao et al. proposed a novel point-cloud-based 3D object detection method for achieving higher-accuracy of autonomous driving in [3]. The proposed method employed YOLOv4 as the feature extractor and Normal-map as additional input. This way, the additional input delivered more enhanced shape information and can be associated with other hand-crafted features, achieving higher precision for 3D object detection with lower perturbations from other parameters such as distance. Bizjak et al. proposed a new method for the segmentation of point clouds in [14]. The topological structuring of the input point cloud was realized by a nearest-neighbor graph, while locally fitted surfaces were used to quantify the objects' shapes. Noise was filtered by a graph cut, and the segmentation was performed by linking the nodes to those objects to which they fit best. The presented results showed that the proposed method can improve the accuracy of the segmentation while reducing the computation time in comparison with other methods.

A ground-aware framework that aims to solve the ambiguity caused by data sparsity present in lidar point cloud measurements was proposed by In Wu et al. in [15]. A multi-section plane fitting approach to roughly extract ground points to assist segmentation of objects on the ground was employed. The proposed method captures long-range dependence between the ground and objects, which significantly facilitates the segmentation of small objects that only consist of a few points in extremely sparse point clouds.

Results showed that the proposed method outperforms state-of-the art methods such as PointNet [16] and SPG [17].

Lu et al. [18] proposed a method that proved to be a robust and efficient point cloud segmentation algorithm in a large set of experimental results. A novel hierarchical clustering algorithm named Pairwise Linkage (P-Linkage) was proposed. The method can be used for clustering any dimensional data and then effectively applied on 3D unstructured point cloud segmentation.

From a different perspective, based on the argument that a single segmentation method will typically not provide a satisfactory segmentation for a variety of classes, Vosselman et al. [19] proposed a methodology that explores the combination of various segmentation and post-processing methods to arrive at useful point cloud segmentations. They concluded that by combining different segmentation and post-processing methods, a segmentation result can be obtained for group points of different classes (e.g., vegetation, ground, walls, roofs), hence increasing the quality of a segment-based classification. A 3D point cloud segmentation survey was presented by Nguyen et al. in [20], where a comparative analysis on various state-of-the-art point cloud segmentation methods is made. Furthermore, Liang et al. presented a survey where several 3D object detection methods based on point cloud are described and analyzed [21]; in addition, those methods were compared in terms of advantages and disadvantages for 3D object detection, including image-based and fusion of point cloud and image methods.

In the project INDTECH 4.0, developed by a consortium of companies and universities, one of the work packages aims to study and develop autonomous navigation solutions to be integrated in AGVs operating in the automotive assembly industry. In this context, an Intel RealSense D435i camera was mounted on the front of an AGV to collect depth data. The 3D point clouds acquired are continuously processed to ensure obstacle avoidance, which is an essential requirement for real-time outdoor autonomous navigation of AGVs.

In this paper, we present a method to detect potential obstacles and to measure their relative position with respect to the robot, using point cloud data provided by the depth camera [22]. This way, the AGV is guided through vision technology, without any previous knowledge of the environment that can be static or dynamic, indoor or outdoor, in a process called natural navigation.

The remainder of this paper is structured as follows. Section 2 presents a description of our methodology, detailing the various processing phases of our method. Section 3 discusses the results according to various metrics relevant for this kind of algorithm. Section 4 summarizes the conclusions and suggestions for future work.

2. Proposed Methodology

In this work, we aim to develop a method for obstacle detection in front of an AGV and measure their positions to the camera. To that end, we used a depth camera, which captures point clouds representing the surrounding space. Since these point clouds contain the relevant features of the robot's surrounding environment regarding potential obstacles, it is important to pose the camera in such a way that obstacles in the path of the AGV can be detected. It is concluded that the most interesting placement for the camera is at the front of the AGV, so that we capture the full extent of the obstacles in front and reduce the number of points corresponding to the ground plane. During the tests we also empirically found the optimal height and pitch for the camera. The experimental setup can be seen in Figure 1.

As the point clouds contain all elements of the scene, some processing has to be carried out in order to remove scene elements that are not interesting for obstacle detection, namely the ground plane. By extracting the ground plane, the output point cloud is made of a more detached set of clusters of points that represent various objects in the scene. These sets of points can be grouped through clustering algorithms into groups that we use as evidence to an existent obstacle and for position measuring. To operate in real time, the point clouds

recorded with the camera were subsequently range limited and down sampled to reduce computational effort.

A method to detect obstacles and their locations by means of 3D point clouds provided by the Intel RealSense D435i depth camera is proposed. This point cloud is subjected to three different stages of processing.

Figure 1. Experimental setup used during this work.

2.1. Pass-Through Filter and Voxel Grid

The first processing stage consists of applying a pass-through filter. This filter is used to restrict the point cloud limits to a relevant distance window.

The ultimate use of this algorithm is to apply it in a navigation system of a low-speed AGV for logistics. Thus, it is possible to limit the computational effort while conserving a high relevance of the results since, due to the low speed, we do not need to detect obstacles very far from the robot.

Another processing tool used to save computational time is the voxel grid to down sample the input point cloud. This filter divides the point cloud into voxels with a user-defined size. The points present in each voxel are approximated by their centroid. This process results in a point cloud with fewer points that still describes the original surface accurately.

In our tests, we defined the limits of the distance window in the interval [1.5, 5] m along the z-axis (depth) for the pass-through filter and a voxel edge length of 0.07 m for the voxel grid.

2.2. RANSAC Segmentation

To successfully identify obstacles in the camera's field of view, it is necessary to segment the volume of the point cloud that corresponds to the ground plane. This is carried out to deliver a higher degree of separation between objects, facilitating the process of grouping points into clusters. For this purpose, the RANSAC (random sample consensus) algorithm is used [23].

RANSAC is an iterative algorithm for fitting mathematical models to experimental data. Instead of considering all the data and eliminating invalid data points, as common in most conventional smoothing techniques, RANSAC uses the minimum amount of data possible and adds as consistent an amount of data points as possible [11]. This method takes the minimum number of points that define a particular model from the experimental set and instantiates a model using those points, (in the case of plane detection, RANSAC selects three random points from the set). Having a candidate model defined, the number of points that are within a defined tolerance to the plane is counted (inliers). When the number of inliers exceeds a threshold, the candidate model is chosen. If the number of inliers is too low, the process is repeated until a plane is identified or the maximum number of iterations is achieved.

The PCL (point cloud library) implementation of RANSAC further includes the option of constraining the plane fit to a specified axis within an angular tolerance. This helps in identifying the ground plane by restricting the detection to horizontal planes. We set the angular tolerance as six degrees.

Once the ground plane is identified, inliers are removed, and the resulting point cloud solely contains objects that may represent obstacles.

2.3. Euclidean Clustering

After removing the ground plane, Euclidean clustering is carried out to identify the various clusters of points present in the point cloud. This is a crucial step to discard points that do not belong to a meaningful cluster and thus do not represent a potential obstacle.

Euclidean clustering consists of separating sets of points according to the Euclidean distance between those points. If two points are within a predefined distance threshold, they are considered to be in the same cluster. However, if that distance is higher than the threshold, the points are segregated into different clusters. Mathematically, we can describe the problem of finding clusters as follows. Letting P be our input dataset and d_{th} a defined distance threshold, a particular set of points $O_i = \{p_i \in \mathrm{P}\}$ is a distinct cluster to $O_j = \{p_j \in \mathrm{P}\}$ if $min||p_i - p_j||_2 \geq d_{th}$ [24].

The PCL implementation of Euclidean clustering also contains the ability to define the minimum and maximum cluster sizes to mitigate the detection of undesirable sets of points.

Once all the clusters are defined, we can visualize them and measure the position to each obstacle. For the interest of avoiding AGV collisions, our method returns the coordinates of the closest point of each cluster.

2.4. Implementation

The method processing sequence is shown in Figure 2.

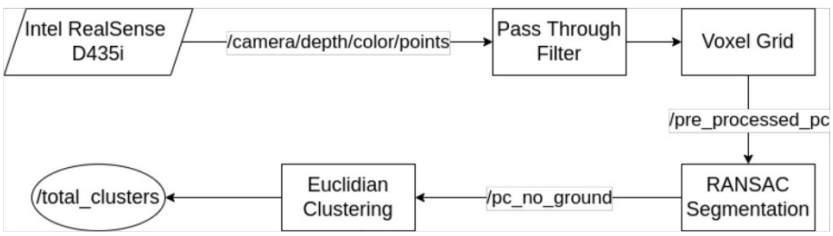

Figure 2. Processing flowchart.

The method consists of an ROS (robot operating system) node that subscribes to the point cloud provided directly by the depth camera in the "camera/depth/color/points" ROS topic. Some intermediate topics are also published for algorithm visualization. These topics include the "pre_processed_pc" topic for the down-sampled and range-limited point cloud and the "pc_no_ground" topic for the point cloud after removal of the ground plane.

The result is the separation of obstacles into point clusters. Each cluster represents a different obstacle, and through the clusters we can obtain the position of each obstacle relative to the AGV. After processing, all detected clusters are merged in a point cloud that is published in the "total_clusters" ROS topic. An additional ROS topic is published for analysis of the algorithm's performance: the "cluster_closest_point" topic that is discussed in the results section.

In addition to ROS, functionalities provided by PCL [25] have been used in all processing steps. These functionalities include the down sampling, plane detection and extraction and clustering algorithms.

3. Results

3.1. Experimental Data

Throughout this work, various tests have been carried out to optimize this algorithm. These tests have been performed on the premises of Active Space Technologies and consisted of recording ROS bags using the D435i depth camera with various types of obstacles located at different distances. We tested the algorithm for both static and mobile obstacles considering indoor and outdoor environments.

To represent static obstacles, chairs placed at 1.8 m, 2.5 m and 4.5 m were used. The distance to the chairs was measured taking the closest part of the chair into account. This setup allowed us to evaluate the algorithm performance for different distances across the defined depth window ([1.5, 5] m). For each of the three distances, an ROS bag was recorded with two chairs side-by-side, indoors and outdoors.

Another test was made with three chairs, one at each distance mentioned above, to assess simultaneous detection of various obstacles at different distances. The final test with chairs consisted of moving the camera perpendicularly to the three chairs, in the direction of the chair furthest away. This test simulates object detection of static obstacles with a moving AGV.

For mobile obstacles, we performed tests in which people moved in front of the camera in various directions. A final test was made where the camera was also moving among wandering people, simulating a more complex scenario that would be common for an industrial AGV.

Every test mentioned above was performed indoors and repeated outdoors. This batch of tests enables us to obtain a sense of the algorithm's performance in different conditions considered relevant for the project. Figures 3 and 4 represent the typical outputs of this algorithm concerning the three major processing stages.

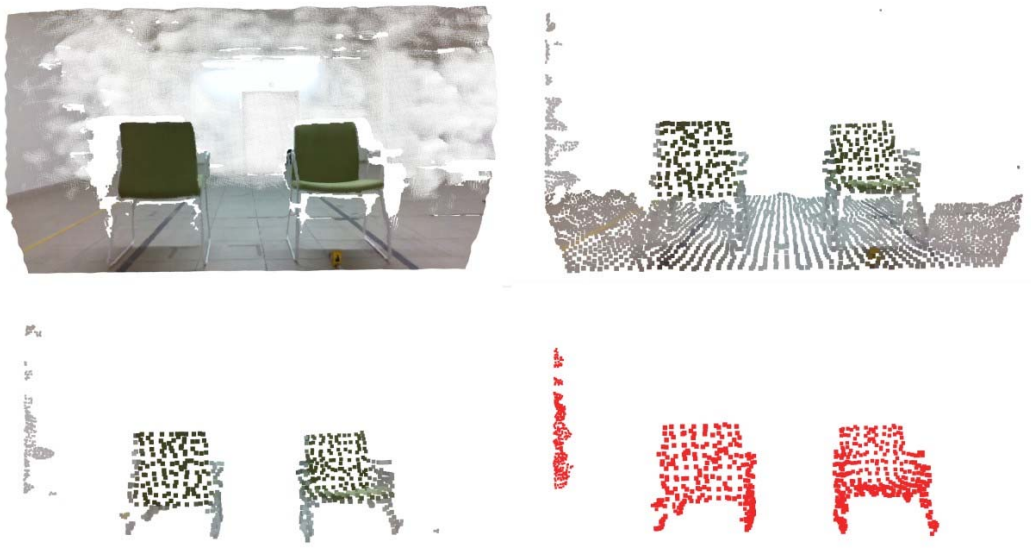

Figure 3. Indoor results for the four stages of processing. From top-left to bottom-right: original point cloud, down-sampled and range-limited point cloud, point cloud after ground plane removal and point cloud with detected clusters (two chairs).

Figure 4. Outdoor results for the four stages of processing.

3.2. Object Detection Effectiveness

There are various 3D object detection frameworks, each one of them using different imaging technologies and methods and operating under different conditions. For these reasons, a direct comparison between the proposed framework and the existing ones could lead to misleading conclusions. To correctly analyze the performance of different frameworks, the same datasets (both indoor and outdoor datasets) under different metrics should be used [21].

To measure the object detection effectiveness, we counted the number of times that each object was detected and compared it with the total number of frames of each test. This was performed using the ROS bags corresponding to the placement of chairs side-by-side at various distances and using indoor and outdoor measurements.

For each ROS bag, the point from each cluster closest to the camera was obtained and published in an ROS topic (cluster_closest_point) for analysis. This topic contains the xyz coordinates of the closest point of each object.

The total number of point clouds analyzed by the algorithm was also counted.

A Python script was used to subscribe to the (cluster_closest_point) topic, and the information was assembled and plotted in a 3D graph. This graph gave us a sense of the position of obstacles in the corresponding test.

In addition to visualizing the data, a clustering algorithm was applied to this set of points, originating clusters of points that correspond to detections of the same object. Using the clusters of closest points, we counted the number of points that represent each obstacle detected by the camera. That value determines how many times each obstacle is detected.

By comparing these values with the number of point clouds processed by our object detection algorithm, we were able to calculate the effectiveness of the object detection.

Each ROS bag has a duration of approximately 15 s, which at ≃30 fps represents a total of 450 frames for each test run. The process described above was repeated 10 times for each ROS bag, and the results are presented in Tables 1 and 2.

Table 1. Indoor detection effectiveness.

Distance (m)	Indoor Detection Effectiveness (%)
1.8	92.72
2.5	70.59
4.5	51.02

Table 2. Outdoor detection effectiveness.

Distance (m)	Outdoor Detection Effectiveness (%)
1.8	99.05
2.5	99.12
4.5	89.22

From these results, we concluded that the method is very effective, returning high percentages of detection for most tests. Moreover, the percentages of detection of both objects were very similar for all distances.

It can be seen that the detection rates drop more significantly for indoor obstacles farther away from the camera. This is due to the poorer luminosity and the fact that for farther objects, the point cloud includes parts of the ceiling, extending the dimensions of the point cloud and making the object detection process more difficult. In this case, the performance bottleneck arises from the increased size of the point cloud, hence explaining why, for outdoor conditions, an efficacy drop is not as drastic.

We can also conclude that, albeit the detection rates varied according to environmental conditions, there was not a variation in detection efficacy between the two obstacles. Thus, detection success decreases with distance to the obstacle. The percentage of obstacle detection is also significantly more stable in outdoor conditions.

3.3. Distance Measurements

A major requirement of this work was not only the detection of obstacles in front of the robot but also measuring their position relative to the camera.

Using the same ROS bags, with two chairs side-by-side at various distances, we analyzed the distance measurements obtained with our algorithm.

The results in Tables 3 and 4 show the mean and standard deviation for the distances obtained in each test, for indoor and outdoor conditions.

Table 3. Indoor detection effectiveness.

Indoor Measurements		
Distance (m)	**Average Distance (m)**	**Standard Deviation (m)**
1.8	1.85	0.03
2.5	2.60	0.09
4.5	5.01	0.11

Table 4. Outdoor detection effectiveness.

Outdoor Measurements		
Distance (m)	**Average Distance (m)**	**Standard Deviation (m)**
1.8	1.91	0.03
2.5	2.56	0.13
4.5	4.95	0.20

For distance measurements, we were able to obtain reasonable values, with an expected drop in accuracy for farther objects. These measurements are entirely dependent on the accuracy of the Intel RealSense D435i depth camera, and as demonstrated, we can rely on the good accuracy provided by the depth sensor for our application.

3.4. Moving Object Detection

For moving obstacles, the performance of the algorithm shows favorable results, although the analysis of these tests was made merely by visual inspection.

The proposed method is able to segment and follow moving obstacles smoothly with a high degree of accuracy and with a high frame rate without apparent failure in detection.

Figure 5 shows two people walking side-by-side in opposite directions in front of the stationary AGV; both people are detected and segmented correctly.

Figure 5. Moving obstacle detection (pedestrians walking in opposite directions). In this test, the AGV was stationary. (**Top**): original point cloud; (**Bottom**): detected clusters.

Figure 6 shows two frames from the mobile AGV/wandering person test where it is clear that the person is segmented accurately. In most frames, there are no false positives. The detection remains accurate during the whole test regardless of the movement of the AGV and of the person.

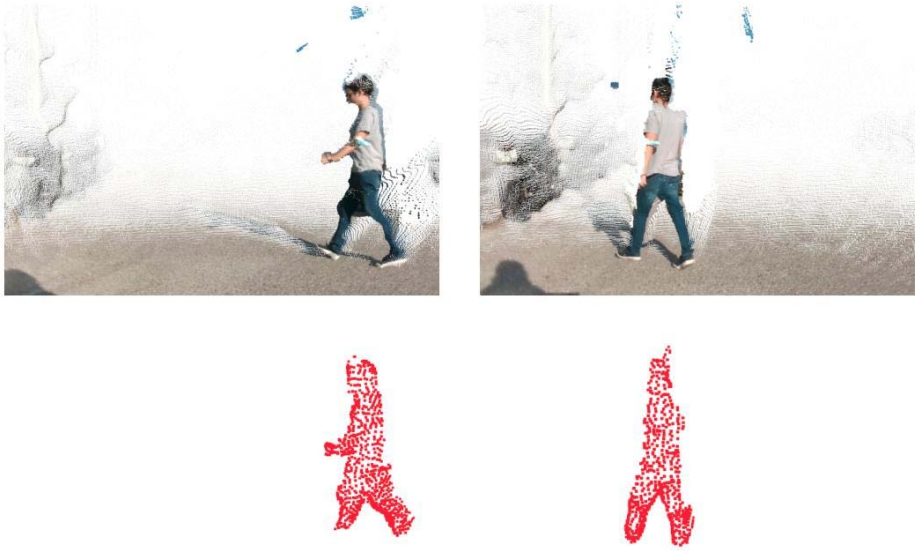

Figure 6. Moving obstacle detection (wandering pedestrian). In this test, both the person and the AGV were moving. (**Top**): original point clouds; (**Bottom**): detected clusters.

3.5. Frame Rates

Since this algorithm is intended for real-time obstacle avoidance, it was also important to analyze the frame rate.

Using a computer with an Intel® Core™ i7-6700HQ CPU @ 2.60 GHz, we did not register significant frame rate drops relative to the native frame rate of the point cloud (≃30 fps), with the maximum frame rate drop being of ≃58% in one of the tests but that results were still in a rate of ≃12 fps for obstacle detection. That being said, most of the tests performed had a frame drop inferior to 10% relative to the native frame rate of the camera, with the average frame drop in all tests being of 18%. These values mean that our method is applicable in a slow moving AGV for real-time object detection.

Table 5 shows data regarding the frame rates from all tests pictured above and two extra tests where we obtained the best and worst results regarding frame rate drops. The columns contain the frame rates for the ROS topics of the original point cloud provided by the camera ("camera/depth/color/points") and of the final topic containing all clusters ("total_clusters").

Table 5. Frame rate drops for different tests.

	Frame Rates		
Test	**Point Cloud**	**Clusters**	**Frame Rate Drop (%)**
Indoor chair detection	30.05	27.12	9.77
Outdoor chair detection	18.37	17.71	3.56
Moving people/stationary AGV	29.94	22.20	25.84
Moving person/moving AGV	29.98	27.34	8.83
Highest frame rate drop	29.96	12.32	58.88
Lowest frame rate drop	30.02	29.40	2.09

4. Discussion

In this work, an object detection algorithm using point cloud data from an Intel® RealSense™ D435i is proposed. This algorithm is meant to be applied in an autonomous navigation system with obstacle avoidance capabilities for AGVs working in dynamic industrial environments.

The presented results show this method can detect obstacles and measure their position relative to the AGV in various environmental conditions, from low luminosity to sunny conditions. This is very important in a SLAM (simultaneous localization and mapping) context to enable the robust autonomous navigation of robots. This method is also able to function in real time, considering the pre-processing applied to the point clouds provided by the camera, which is crucial for autonomous navigation. In this context, real-time conditions mean the algorithm is capable of processing and making decisions quickly enough to ensure the AGV can be safely driven when it is moving with a velocity up to 1.5 m s^{-1}. The latency of the system, i.e., the time delay between the cause and the effect of some physical change (e.g., obstacle motion) being observed, is usually lower than 50 ms.

The algorithm detects objects with high reliability and returns results with inconsequential frame drops from the camera's native frame rate. When the illumination conditions are optimal, no frame rate drop has been observed, at least not to compromise system performance. This applies for the various environmental conditions tested, specifically indoor and outdoor environments, static and mobile obstacles and with the AGV being mobile or stationary. The resulting frame rates of detection are adequate for a slow-moving AGV in an industrial environment, as intended for our application. Supplementary tests have been made to ascertain the impact of direct illumination on the sensor. As long as direct incident radiation from the source (e.g., sunlight) does not hit the optical sensor directly, the algorithm is capable of reliable object detection. The exception has been a camera malfunction due to indirect heating effects rather than illumination issues of the sensor.

The obstacle position measurements returned by our method also present high degrees of precision that improve with illumination and when obstacles get closer to the camera. In some cases, camera pitch was altered to optimize the detection range, but the algorithm performance remained stable. This was shown with the outdoor tests, where the measurements presented lower standard deviation. These are characteristics innate to the Intel RealSense D435i depth camera, and the precision of distance measurements is entirely dependent on the capabilities of the chosen camera to be used along with our obstacle detection method.

Thus, a three-stage methodology using well-established methodologies such as a pass-through filter and voxel grid for data dimensionality reduction, RANSAC algorithm for identification and removal of inliers of the ground plane and Euclidean clustering for point cloud clustering for object detection is proposed. As a result, a framework for real-time autonomous navigation with low computational effort and no need for prior knowledge of the surrounding environment is proposed in this work.

In addition to indoor measurements frequently reported in many studies that consider stable, calibrated illumination conditions, we have conducted a test campaign considering several outdoor experiments to assess algorithm performance and identify possible limitations and blocking points. Although carried out at a time when the factory was not under nominal operation conditions, logistics and environmental concerns were selected to mimic real environments. The most relevant conditions were the following: ram- and wake-facing natural illumination, clear and cloudy sky, flat and inclined paths (slope up to 8%), smooth and irregular grounds, different types of objects and vehicles as well as multiple pedestrians wandering in different directions. The algorithm was able to perform correctly under all these conditions without major issues. Qualitatively, algorithm performance was not degraded during the whole campaign, at least not to a point rendering it useless. As mentioned above, the main issue was caused by a temporary hardware failure (camera overheating) rather than poor performance of the algorithm.

Further developments of this algorithm may provide the capability to create a system that generates odometry measurements using depth camera data, namely by implementing a visual odometry system for SLAM. Visual odometry systems calculate the movement of a robot by taking advantage of information regarding obstacle position over time, which we can obtain through our method. Such algorithms are useful to perform sensor fusion with

an inertial measurement unit or, according to the accuracy of their results, can even replace this kind of sensors.

5. Conclusions

Having achieved a high reliability regarding object detection, the presented methodology can be extended to process RGB images, acquired simultaneously by the Intel camera, to extend the capabilities of our algorithm to not only detect the position of obstacles but also classify them into types of objects. For indoor detection, being a controlled environment, previous knowledge about possible/existing obstacles (e.g., person, forklift, bucket, etc.) can be used to eliminate potential obstacles and improve effectiveness, being that the RGB images can be easily used to classify objects using that prior knowledge. With the information of the type of obstacle in front of the robot, it is possible to program more informed decisions regarding the movement of the AGV. Another interesting topic for further development is the detection of holes in the ground, which is another crucial problem that must be solved for safe AGV movement, and it is a feature that is usually not detected with traditional laser scanners.

Author Contributions: Conceptualization, M.P., A.S., P.C. and V.F.; data curation, M.P.; investigation, M.P.; methodology, M.P., P.C., A.S. and V.F.; supervision, V.F.; validation, M.P., P.C., A.S. and V.F.; writing—original draft preparation, M.P.; review and editing, M.P., P.C., A.S. and V.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by Project "INDTECH 4.0—new technologies for smart manufacturing", POCI-01-0247-FEDER-026653, and by the European Regional Development Fund (ERDF) through the COMPETE 2020—Competitiveness and Internationalization Operational Program (POCI). This work was also funded by national funds from FCT—Portuguese Foundation for Science and Technology, under the project UIDB/04033/2020.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript or in the decision to publish the results.

References

1. Pires, M.A. Natural Navigation Solutions for AMRs and AGVs Using Depth Cameras. Master's Thesis, University of Coimbra, Coimbra, Portugal, 2021.
2. Lynch, L.; Newe, T.; Clifford, J.; Coleman, J.; Walsh, J.; Toal, D. Automated ground vehicle (AGV) and sensor technologies—A review. In Proceedings of the 2018 12th International Conference on Sensing Technology (ICST), Limerick, Ireland, 4–6 December 2018; pp. 347–352.
3. Miao, J.; Hirakawa, T.; Yamashita, T.; Fujiyoshi, H. 3D Object Detection with Normal-map on Point Clouds. In Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, online, 8–10 February 2021; pp. 569–576. [CrossRef]
4. Tang, P.; Huber, D.; Akinci, B.; Lipman, R.; Lytle, A. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Autom. Constr.* **2010**, *19*, 829–843. [CrossRef]
5. Bassier, M.; Vergauwen, M. Clustering of Wall Geometry from Unstructured Point Clouds. In Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Bergamo, Italy, 6–10 May 2019; Volume XLII-2/W9, pp. 101–108. [CrossRef]
6. Shi, S.; Wang, X.; Li, H. PointRCNN: 3d object proposal generation and detection from point cloud. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 770–779.
7. Simon, M.; Amende, K.; Kraus, A.; Honer, J.; Samann, T.; Kaulbersch, H.; Milz, S.; Michael Gross, H. Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 15–20 June 2019.
8. Beltrán, J.; Guindel, C.; Moreno, F.M.; Cruzado, D.; García, F.; Escalera, A.D.L. Birdnet: A 3d object detection framework from lidar information. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 3517–3523.

9. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3D object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4490–4499.
10. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3D object detection from RGB-D data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 918–927.
11. Kırcalı, D.; Tek, F.B. Ground plane detection using an RGB-D sensor. In *Information Sciences and Systems*; Springer: Cham, Switzerland, 2014; pp. 69–77.
12. Choi, S.; Park, J.; Byun, J.; Yu, W. Robust Ground Plane Detection from 3D Point Clouds. In Proceedings of the 2014 14th International Conference on Control, Automation and Systems (ICCAS 2014), Seoul, Korea, 22–25 October 2014; pp. 1076–1081.
13. Habermann, D.; Hata, A.; Wolf, D.; Osório, F.S. 3D point clouds segmentation for autonomous ground vehicle. In Proceedings of the 2013 III Brazilian Symposium on Computing Systems Engineering, Niteroi, Brazil, 4–8 December 2013; pp. 143–148.
14. Bizjak, M. The segmentation of a point cloud using locally fitted surfaces. In Proceedings of the 18th Mediterranean Electrotechnical Conference, Lemesos, Cyprus, 18–20 April 2016.
15. Wu, J.; Jiao, J.; Yang, Q.; Zha, Z.; Chen, X. Ground-Aware Point Cloud Semantic Segmentation for Autonomous Driving. In Proceedings of the 27th ACM International Conference on Multimedia (MM'19), Nice, France, 21–25 October 2019; pp. 971–979. [CrossRef]
16. Charles, R.; Su, H.; Kaichun, M.; Guibas, L. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85.
17. Landrieu, L.; Simonovsky, M. Large-scale point cloud semantic segmentation with superpoint graphs. In Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 4558–4567.
18. Lu, X.; Yao j Tu, J.; Li, K.; Li, L.; Liu, Y. Pairwise Linkage for Point Cloud Segmentation. In Proceedings of the ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Prague, Czech Republic, 12–19 July 2016; Volume III-3, pp. 201–208. [CrossRef]
19. Vosselman, G. Point cloud segmentation for urban scene classification. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2013**, *1*, 257–262. [CrossRef]
20. Nguyen, A.; Le, B. 3D point cloud segmentation: A survey. In Proceedings of the 2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM), Manila, Philippines, 12–15 November 2013; pp. 225–230.
21. Liang, W.; Xu, P.; Guo, L.; Bai, H.; Zhou, Y.; Chen, F. A survey of 3D object detection. *Multimed. Tools Appl.* **2021**, *80*, 29617–29641. [CrossRef]
22. Pires, M.; Couto, P.; Santos, A.; Filipe, V. Obstacle Segmentation for Autonomous Guided Vehicles through Point Cloud Clustering with an RGB D Camera. In Proceedings of the 2nd IFSA Winter Conference on Automation, Robotics and Communications for Industry 4.0 (ARCI'2022), Andorra la Vella, Andorra, 2–4 February 2022; pp. 79–83.
23. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
24. Rusu, R.B. Semantic 3D object maps for everyday manipulation in human living environments. *KI-Künstl. Intell.* **2010**, *24*, 345–348. [CrossRef]
25. Rusu, R.B.; Cousins, S. 3D is here: Point cloud library (PCL). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.