sensors

MDPI

Article

Free Space Detection Algorithm Using Object Tracking for Autonomous Vehicles

Yeongwon Lee and Byungyong You *

Department of Mechanical Engineering, Kyungil University, Gyeongsan 38428, Korea; ywuu1214@gmail.com
* Correspondence: zealot@kiu.kr

Abstract: In this paper, we propose a new free space detection algorithm for autonomous vehicle driving. Previous free space detection algorithms often use only the location information of every frame, without information on the speed of the obstacle. In this case, there is a possibility of creating an inefficient path because the behavior of the obstacle cannot be predicted. In order to compensate for the shortcomings of the previous algorithm, the proposed algorithm uses the speed information of the obstacle. Through object tracking, the dynamic behavior of obstacles around the vehicle is identified and predicted, and free space is detected based on this. In the free space, it is possible to classify an area in which driving is possible and an area in which it is not possible, and a route is created according to the classification result. By comparing and evaluating the path generated by the previous algorithm and the path generated by the proposed algorithm, it is confirmed that the proposed algorithm is more efficient in generating the vehicle driving path.

Keywords: autonomous vehicle; free space detection; object tracking; LiDAR sensor

Citation: Lee, Y.; You, B. Free Space Detection Algorithm Using Object Tracking for Autonomous Vehicles. *Sensors* **2022**, *22*, 315. https://doi.org/10.3390/s22010315

Academic Editor: Arturo de la Escalera Hueso

Received: 24 November 2021
Accepted: 29 December 2021
Published: 31 December 2021

1. Introduction

In order for an autonomous vehicle to travel stably in various environments, it is important to recognize obstacles around the vehicle. In particular, in an environment in which lanes are not recognized while driving or in a road environment in which a precise map is not built, a path must be created by detecting an obstacle-free space, that is, a free space to generate a driving path. Many papers deal with free space detection. In previous papers, there are many cases of detecting free space using only the location information of obstacles. In this case, the behavior of the surrounding obstacles cannot be predicted, meaning there is a high possibility of creating an inefficient path in terms of safety.

Assume that there is an obstacle traveling in the opposite lane of the traveling vehicle. In this environment, if the free space is detected in units of frames using only the location information, it will be detected as a free space that can travel up to the opposite lane area because it does not know which direction the obstacle is traveling in. In this case, there is a possibility of creating a driving route in the opposite lane, and the collision risk is increased. If it is possible to know at what speed and direction the surrounding obstacles are moving with respect to the driving vehicle, then more efficient path generation will be possible.

In this paper, to solve this problem and increase the efficiency of path generation in free space, we propose a new algorithm for detecting free space based on the speed information of obstacles. The data were acquired using a LiDAR (light detection and ranging) sensor mounted on the actual vehicle, the speed of the obstacle was calculated through the object tracking process, and the free space was divided into four areas according to the speed. For comparison with the previous algorithm, a driving route was generated according to the free space detection result. By comparing the path generated by the previous method and the proposed method, it is confirmed that a more efficient path is generated with the proposed method.

Recently, various studies on autonomous vehicles have been conducted in various fields. There are papers on various signal processing methods of LiDAR sensors mainly

used in autonomous vehicles [1–24]. Among them, there are many papers that presented research on free space sensing. Free space detection is being studied not only for the study of autonomous vehicle driving, but also in various fields such as the recognition of a parking space or the driving of an aircraft [25–27]. Among them, the research trends in the detection of free space in autonomous vehicles are as follows [1–12]. Often, a camera or LiDAR sensor is used, or both sensors are used, to recognize the space and obstacles around the vehicle. These sensors recognize the surrounding space and detect a drivable free space free from obstacles. In this process, existing papers often detected empty space using only the location data of obstacles. H. M. Eraqi et al. [1] used a laser scanner to recognize the surroundings and process signals in units of frames to understand the surrounding environment of a driving vehicle. Using these static data, the ground and obstacles are recognized, and the free space is expressed in a simplified polygonal form. C. Fernandez et al. [2] used LiDAR and camera sensors to detect free space and speed bumps. This method detects based on static data and does not take into account the speed of obstacles. J. Tao [3] used LiDAR sensor data to detect free space, in which free space was also detected without information about the speed of the obstacle. Many previous studies used static data without considering the speed of obstacles [1–12]. Free space detection for autonomous driving must also be considered in dynamic environments. To supplement this, in this paper, we propose a method to detect free space by reflecting the velocity data of obstacles.

Table 1 summarizes existing papers [1–12]. In previous papers, sensors were used in the FSD process, and the speed information of obstacles was summarized. In [1–12], obstacles were mainly recognized using LiDAR or camera sensors, and all of them detected free space without information on the velocity of the obstacles.

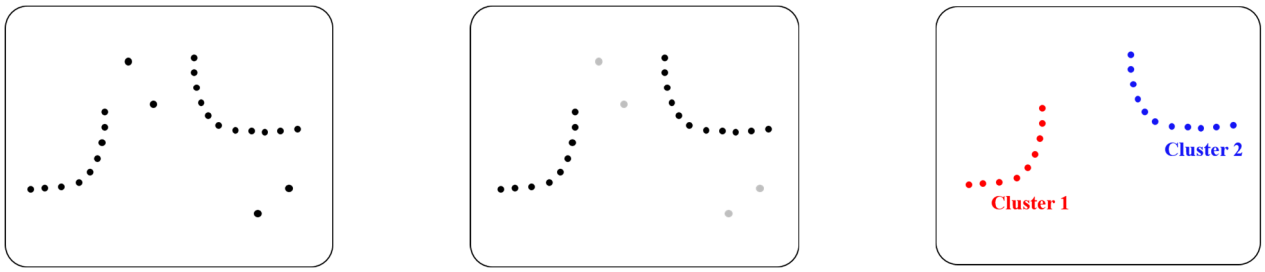
Table 1. Summary of previous papers [1–12].

Previous Paper	Sensor	FSD with/without Obstacle Speed Information
[1]	Laser Scanner	without
[2]	LiDAR, Camera	
[3]	LiDAR	
[4]	LiDAR, Camera	
[5]	Camera	
[6]	Camera	
[7]	LiDAR, Camera	
[8]	Camera	
[9]	Camera	
[10]	Radar	
[11]	Camera	
[12]	Camera	

2. Materials and Methods

2.1. Clustering and Object Tracking

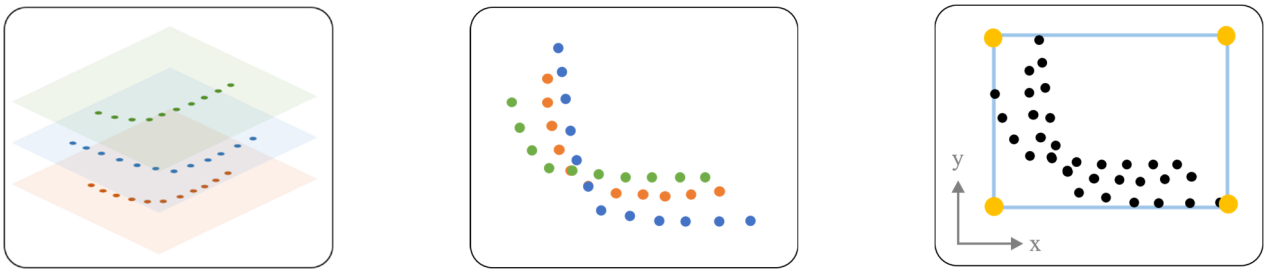
This is the process of identifying the behavior of obstacles around the vehicle using sensor data. LiDAR is an abbreviation of light detection and ranging. It is a radar system that measures the positional coordinates of a reflector by measuring the time it takes to emit a laser pulse, reflect it, and return it, which are output as point data. The point data output by the LiDAR sensor are clustered using the DBSCAN (density-based spatial clustering of applications with noise) algorithm [28–30]. DBSCAN is one of the clustering algorithms based on density and is a widely used algorithm for clustering point data. Figure 1 is a pictorial representation of the DBSCAN clustering process. Figure 1a shows the single-layer LiDAR raw data. Figure 1b is the result of judging data having a number of points greater than or equal to the MinPts (minimum number of points) in the eps (epsilon) radius as cluster data according to the DBSCAN algorithm, and separating the rest into noise data. Figure 1c is the result of clustering into Cluster 1 and Cluster 2 after removing noise.



(**a**) (**b**) (**c**)

Figure 1. DBSCAN clustering: (**a**) raw data of LidAR sensor that recognized two obstacles; (**b**) data with noise removed; (**c**) data of two clustered obstacles.

The boxing process is executed in multiple layers using the results of clustering in a single layer. This is to integrate and simplify multi-layered data. This process is shown in Figure 2. Figure 2a is a classification of data that recognize the same obstacle in each layer, and 3 single layers that recognize the same obstacle are shown in 3D. For integration, the data are integrated and displayed in 2D on one plane as in Figure 2b. For the simplification of the integrated data, square data are created with four points, as shown in Figure 2c. The rectangle is calculated using the x max, x min, y max, and y min values of the integrated point data. As a result, cluster data of multiple layers are represented by four-dot square data.



(**a**) (**b**) (**c**)

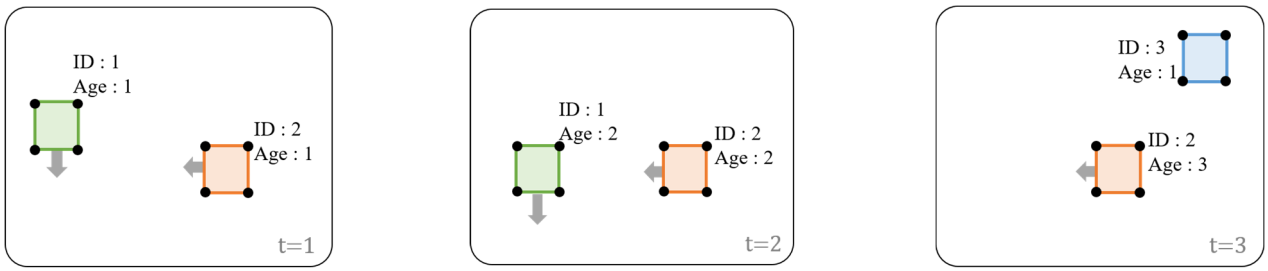
Figure 2. Multi-layer clustering: (**a**) classification of data that recognize the same obstacle in each layer, and 3 single layers that recognize the same obstacle are shown in 3D; (**b**) representation of all 3D data on a 2D plane; (**c**) representation as square data for data simplification.

An object tracking process is required to determine the speed of the obstacles [31]. Figure 3 shows the object tracking process. By comparing the previous and current positions of obstacles, it is judged whether they are the same obstacles, and an ID and Age are assigned to each obstacle. If it is judged to be the same obstacle, the same ID as before is assigned and the Age is increased. If it is determined that it is a newly measured obstacle, a new ID is assigned. Through this process, the speed can be calculated using the displacement according to the time the obstacle moves. In this case, when the position of the obstacle is changed, the data of the rectangle may also be changed. Therefore, the minimum and maximum sizes of obstacles (mainly vehicles) are reflected and tracked.

2.2. Grid Map

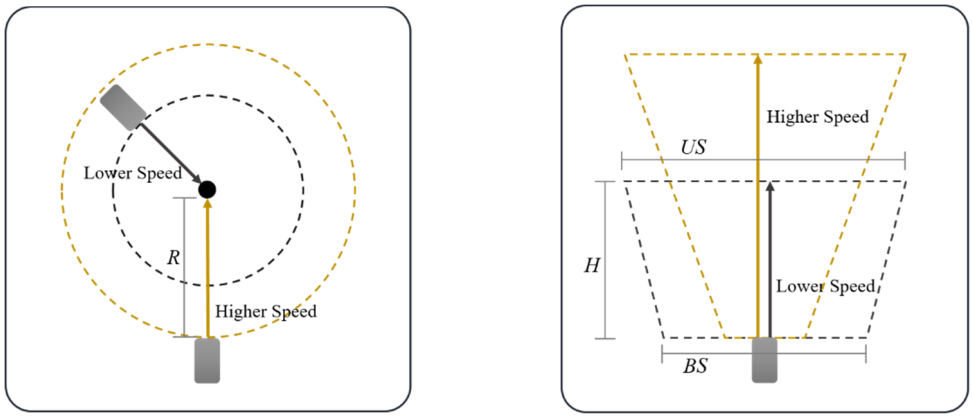
Figure 4a is the circular grid map method used in [1], and Figure 4b is the trapezoidal grid map method proposed in this paper. A grid map is applied to recognize the surrounding environment of the vehicle and detect free space. As shown in Figure 4a, the grid map used in the previous paper is in the shape of a circle. In the circular grid map, the center of the circle is fixed in absolute coordinates, the radius of the circle is determined according to the speed of the driving vehicle, and the position of the vehicle on the circle is determined according to the heading of the vehicle. As the speed of the vehicle increases, the radius

increases, and the vehicle's position on the circle becomes opposite when moving forward and backward.



(**a**) (**b**) (**c**)

Figure 3. Object tracking. Obstacle position change with time (t) change: (**a**) examples of obstacles when t = 1; (**b**) examples of obstacles when t = 2; (**c**) examples of obstacles when t = 3.



(**a**) (**b**)

Figure 4. (**a**) Circular grid map of previous paper [1]. (**b**) Proposed trapezoidal grid map. R is the radius of the circle. US is the upper side of the trapezoid. BS is the base of the trapezoid. H is the height of the trapezoid.

The newly proposed trapezoidal grid map is shown in Figure 4b. The height of the trapezoid is determined according to the speed of the vehicle, and the lower and upper sides of the trapezoid are determined according to the vehicle heading. As the speed is high and the heading is small, the trapezoid has a long and narrow shape, and if the speed is low and the heading angle is large, the trapezoid shows a short and wide shape. The trapezoidal shape is used because it can reduce the use of unnecessary cells and reflects the vehicle yaw angle better than the circular shape.

$$US = |HD| \times a + b_1 \quad (1)$$

$$BS = |HD| \times a + b_2 \quad (2)$$

$$H = V/c + d \quad (3)$$

Equations (1)–(3) are equations for the proposed trapezoidal grid map generation. *US* is the upper side of the trapezoid(m), *BS* is the base of the trapezoid(m), and *H* is the height of the trapezoid(m). *HD* is the vehicle's heading (deg), and *V* is the vehicle's speed (kph).

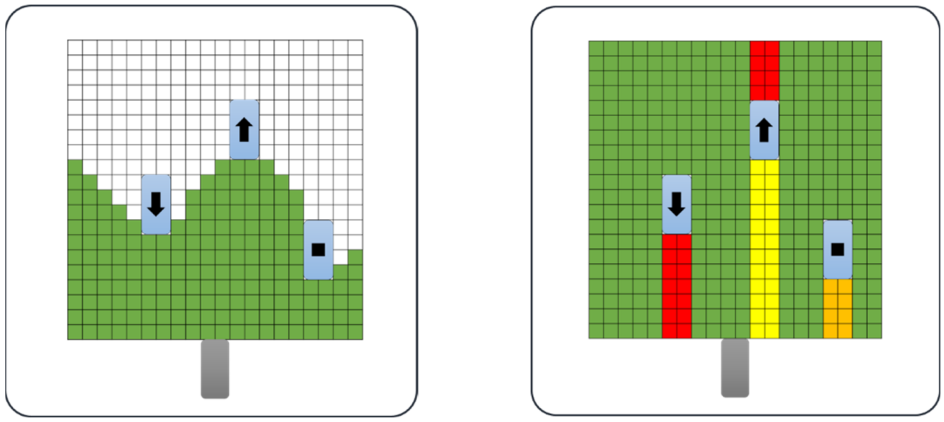
a, b_1, b_2, *c*, and *d* are constant values. a is a constant value for converting the heading value expressed in deg units to the base of the trapezoid in m units, b_1 is the minimum value of the upper side of the trapezoid (*US*), and b_2 is the minimum value of the base of the trapezoid (*BS*). When the heading of the vehicle is zero, the length of the upper side *US* of the trapezoid is b_1, and the length of the base side *BS* is b_2.

$$R = V \times e + f \quad (4)$$

Equation (4) is a simplified formula for generating a prototype grid map used in previous papers. R is the radius of the circle (m) and V is the speed of the vehicle (kph). e and f are constant values. e is a constant value for converting the vehicle speed value expressed in kph units to the radius of the circle in m units, and f is the minimum value of the radius (R). The length of the R is f when the vehicle speed is zero.

2.3. Free Space Detection (FSD)

As shown in Figure 5, the free space is represented by segmentation. The authors of [1] used a method to represent free space by extracting free space boundary cells from the grid map, as shown in Figure 5a, and simplifying the polygons connecting the cells. Since this paper focuses on classifying the free space area according to the behavior of obstacles, it would be more appropriate to express it in a divided way rather than a polygon, as shown in Figure 5b.



(**a**) (**b**)

Figure 5. Free space detection with polygon and free space detection with segments: (**a**) polygon method of previous paper; (**b**) proposed segmentation method.

Figure 5 shows the different free space detection methods. In the figure, the gray squares represent a vehicle in motion, and the three blue squares represent obstacles. The direction of movement of the obstacle is indicated by an arrow, and the obstacle drawn with a black rectangle instead of an arrow indicates that it is stationary. Figure 5a shows a free space detection method used in previous papers, representing free space with polygons. Figure 5b is the method proposed in this paper, which represents free space as a segment.

Figure 5b shows the separated free space area in color. It has a different meaning depending on the segment color. Green is the space where you can drive because there are no obstacles in front, yellow is the space where you can drive with obstacles, orange is where the obstacle is stationary or you are driving at a lower speed than the ego vehicle, meaning it is impossible to drive, and red is the direction the obstacle is moving. Conversely, if the vehicle is traveling at a high speed, it is impossible to drive, and there is a collision risk. Even if it is detected as a free space, the free space is detected and classified to reflect that there is a space where driving is impossible.

Segments are classified according to the perceived speed of the obstacle. If an obstacle is driving in the opposite direction to the vehicle being driven, there is a collision risk with the obstacle when creating a path and driving in this space, even if there is a free space in the direction of the obstacle. Conversely, if there is not enough free space due to the obstacle, but the obstacle is traveling in the same direction as the currently traveling vehicle and at a higher speed, this space will become a drivable space. As shown in Table 2 the free space is classified according to the driving possibility by dividing the cases where the behavior of the obstacles is different into four categories. In Table 2, the divided free space is represented by a total of four expressions: CA (completely able to drive), A (able to drive), UA (unable to drive), and CUA (completely unable to drive).

Table 2. Driving possibility according to object speed.

Relative Speed of the Obstacle (S)	Possibility of Driving
No obstacles	CA
S $\geq$ 0	A
-(vehicle speed) < S < 0	UA
S < -(vehicle speed)	CUA

In Table 2, the relative speed of the obstacle (S) is the relative speed between the driving vehicle and the obstacle. First, if there are no obstacles in the segmented segment, the segment is divided into a completely drivable area (CA; completely able to drive). Second, if there is an obstacle in the segment, but the relative speed of the obstacle is greater than or equal to 0, it is classified as an able to drive area (A; able to drive). Even if there is an obstacle near the front, if the relative speed is greater than or equal to 0, the collision risk is low even if the vehicle continues driving without avoiding the obstacle. Third, when the relative speed of the obstacle is less than 0 and greater than the value obtained by multiplying the vehicle speed by -1, the area is classified as an unable to drive area (UA; unable to drive). This means that the obstacle and the vehicle are driving in the same direction, and the vehicle is driving faster than the obstacle. In this case, when driving in this segment, there is a collision risk with obstacles, meaning it is classified as an impossible area. Fourth, if the relative speed is less than the vehicle speed multiplied by -1, it is classified as a completely unable to drive area (CUA; completely unable to drive). This means that there is an obstacle moving in the opposite direction to the vehicle. That is, it represents an obstacle in a lane opposite to the vehicle driving lane. When driving in this segment, the vehicle runs in the opposite direction, meaning that even if there is a free space between the vehicle and an obstacle, it is divided into an area where driving is impossible.

In the pseudocode below (Algorithm 1), P denotes an obstacle, RS denotes the relative speed of the obstacle and the vehicle, VS denotes the speed of the vehicle, and Sp denotes the separation of free space.

Algorithm 1. Free Space Detection (P, RS, VS, Sp)

```
for each obstacle P
    if P exist
        calculation RS
        if RS ≥ 0
            mark Sp as able to drive
        else if -VS < RS < 0
            mark Sp as unable to drive
        else if RS < -VS
            mark Sp as completely unable to drive
    else
        mark Sp as completely able to drive
```

2.4. Path Generation

In order to reach a set target point, it is necessary to create an optimal path. The A* algorithm is used for path generation [32,33]. A weight is assigned to each cell of the grid according to the result of classifying the free space in which driving is possible and the space which cannot be driven in, and according to the distance from the target point. The weights of cells adjacent to the current location of the vehicle are calculated, and a path is generated to the cell with the smallest weight, that is, the lowest cost. By repeatedly executing this process, it is possible to generate the shortest optimal path considering the collision risk with obstacles.

As shown in Figure 6, when the weight of the cell $P_1 \cdots P_8$ is calculated at the current position P_0, if the cell with the lowest weight is P_3, the path is created in the direction from

P_0 to P_3. In the next step, the weights of adjacent cells are calculated based on P_3, and then a path is created.

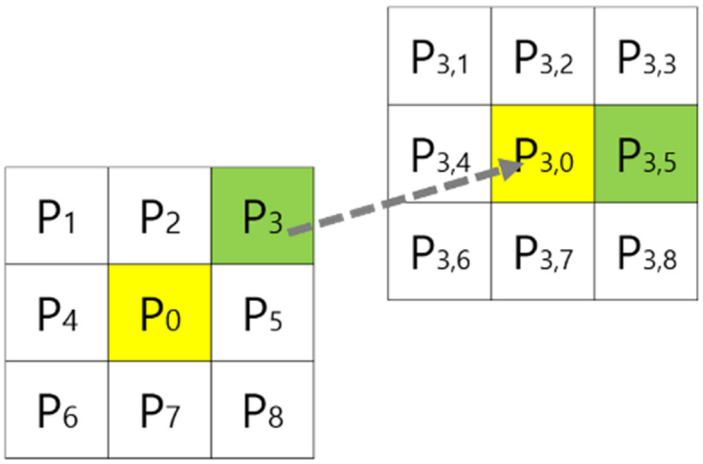


Figure 6. Selection of the cell with the higher weight among neighboring cells.

The weight for each divided area is calculated according to the calculation constant (u) in Table 3. The weight is the product of the linear distance to the target point multiplied by the calculation constant u. In the region where there is a high possibility of driving without obstacles, the smallest value is 1, and when the relative speed of the obstacle is greater than or equal to 0, the weight is 2. In the area where the relative speed of the obstacle is less than 0, but it is running in the same direction as the vehicle, the calculated constant value is 3, and the area where the obstacle is running in the opposite direction to the vehicle is set to 4.

Table 3. Weight according to free space classification.

Possibility of Driving in a Demarcated Area	***u***
CA	1
A	2
UA	3
CUA	4

2.5. Time to Collision and Collision Risk

To confirm that the proposed method (free space detection using speed information) is an improvement of the previous method (free space detection without speed information), the CR (collision risk) was calculated using TTC (time to collision) [34,35]. Two methods were used to create a path in the detected free space and calculate the CR when the path is driven. After taking the reciprocal of TTC, the reciprocal of TTC with all obstacles was added to the ego vehicle position and speed. Then, this value was added at all points in the generated path. The result of this calculation is the CR value.

In Equation (5), d_{rel} is the distance between the position of the vehicle and the obstacle, and v_{rel} is the relative speed of the vehicle and the obstacle [34]. In Equation (6), $o_1 \ldots o_n$ represent an obstacle, n is the number of obstacles, and the reciprocal of TTC corresponding to all obstacles is added. pt represents each point of the path, and it is the sum of all corresponding values from the starting point pt_0 of the path to the last point pt_m of the path.

$$TTC = \frac{\left|\vec{d}_{rel}\right|}{\left|\vec{v}_{rel}\right| cos\theta} \tag{5}$$

$$CR = \sum_{pt_0}^{pt_m}\left(\sum_{o_1}^{o_n}\frac{1}{TTC}\right) \tag{6}$$

2.6. Experiment Environments

A LiDAR sensor was attached to a small electric vehicle, and the sensor data were measured while driving on a real road. Sensor data were acquired in various environments, such as static obstacles, dynamic obstacles, and obstacles in real road driving. The model was a Velodyne VLP-16. Table 4 shows the technical specifications of the LiDAR sensor used. Figure 7 shows the LiDAR sensor and vehicle used for data measurement. The vehicle used was an electric vehicle, D2, owned by Kyungil University, which has obtained an autonomous driving license from the Ministry of Land, Infrastructure and Transport, Korea.

Table 4. Velodyne VLP-16 specifications.

Specifications	Value
Channel	16 channels
Measuring range	100 m
Accuracy	Max ±3 cm
Field of view (vertical)	+15.0° to −15.0°
Field of view (horizontal)	360°
Angular resolution (vertical)	2.0°
Angular resolution (horizontal/azimuth)	0.1–0.4°
Rotation speed	5–20 Hz

Figure 7. LiDAR sensor on the vehicle.

Table 5 shows the constant values (a, b_1, b_2, c, d, e, f) of Equations (1)–(4).

Table 5. Constant values of Equations (1)–(4).

Constant	Value
a	0.46
b_1	6
b_2	2
c	4.3
d	2
e	0.1
f	1

3. Results

Figure 8 is a pictorial representation of the results of Figure 9a. In Figure 8, green is the free space that the vehicle can drive in. In the figure, red is a space in which the vehicle is completely unable to drive, and orange is a space in which the vehicle is unable to drive. The dotted line represents the grid map range, and the yellow represents the obstacle. The direction in which the obstacle moves is represented by an arrow, and the static obstacle is represented by a square dot. The driving target point is indicated by a gray circle. Paths are generated only in free space cells, and the generated paths are indicated by blue lines. By referring to this pictogram, the execution result screen of Figure 9 can be grasped.

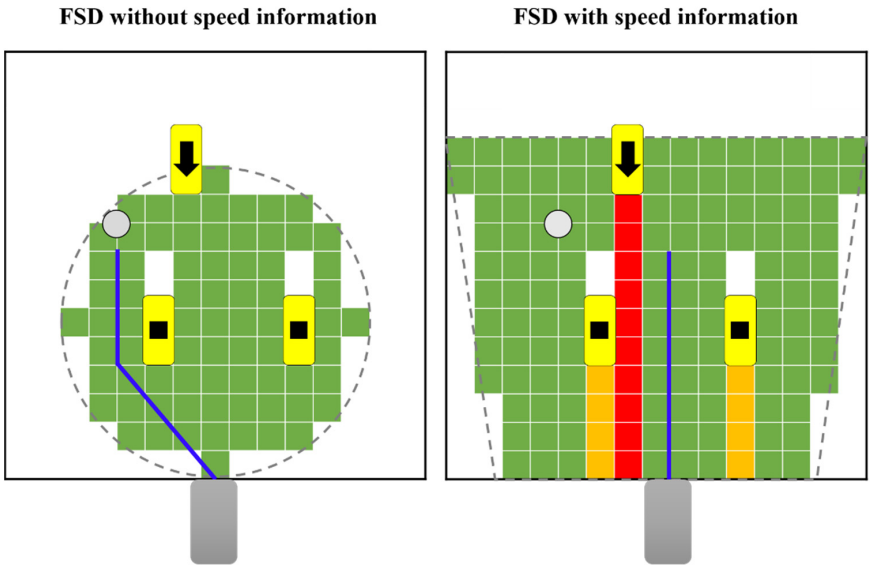


Figure 8. Pictorial result of path generation based on FSD of two methods in case Figure 9a.

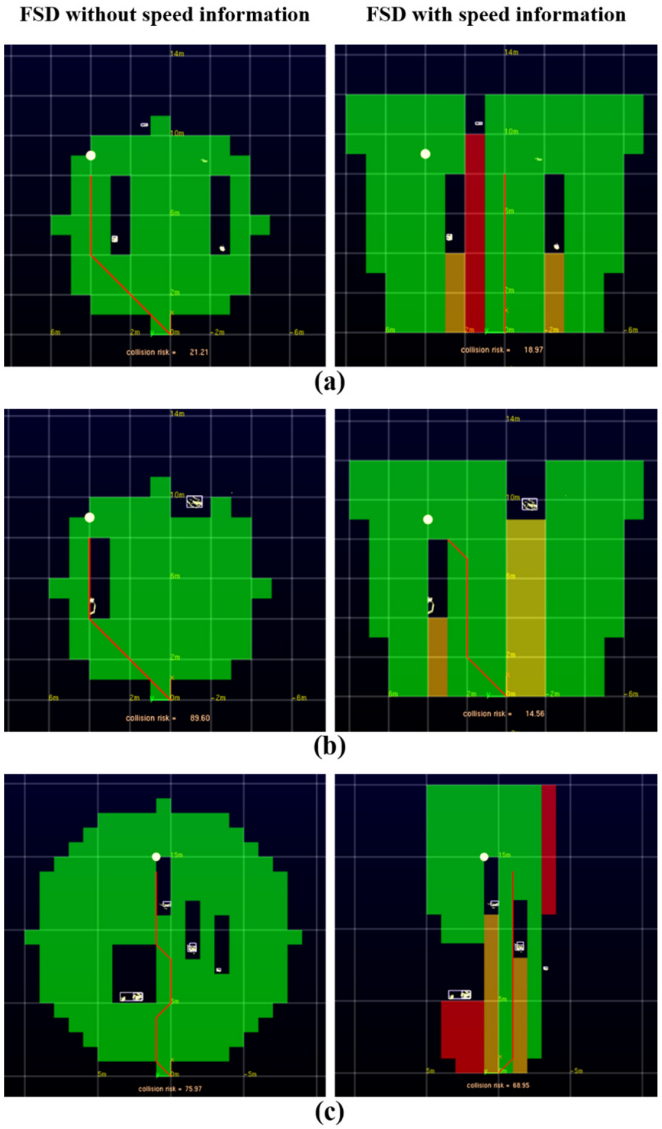


Figure 9. *Cont.*

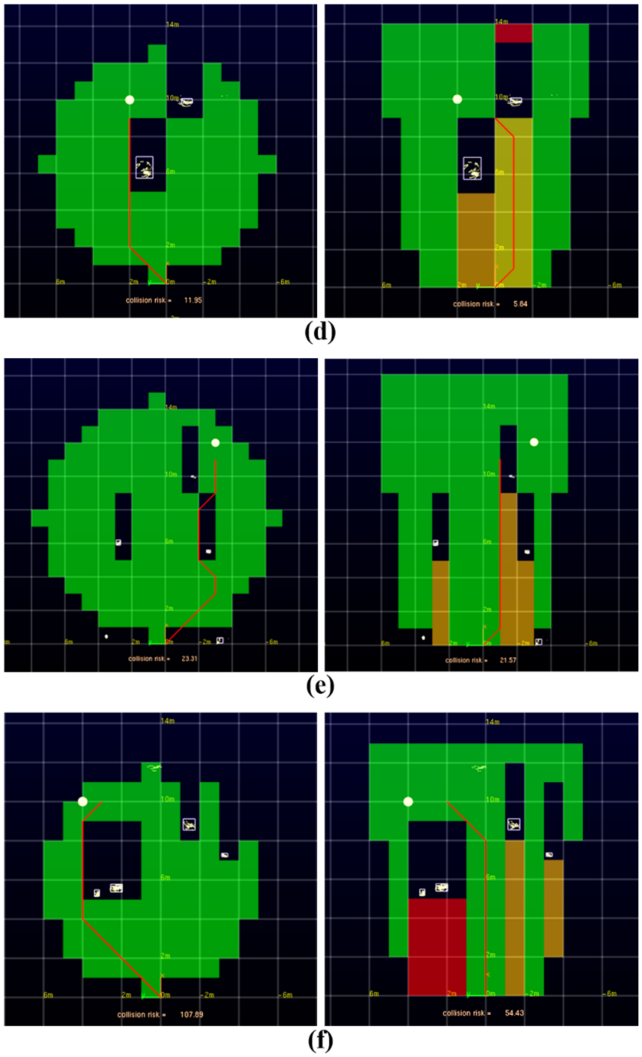


Figure 9. Generated path comparison of FSD without speed information in circular grid map and FSD with speed information in trapezoid grid map in cases (**a**)–(**f**).

Figure 9 show the execution screen of generating a path based on the FSD of the two methods in six cases. Figure 9 and Table 6 are the FSD results in cases (a)–(f). Case (a) shows one dynamic obstacle moving in opposite directions and two static obstacles. Case (b) has one dynamic obstacle moving in the same direction and one static obstacle. Case (c) is a case with two dynamic obstacles moving in opposite directions and two static obstacles. Case (d) has one dynamic obstacle moving in the same direction and one static obstacle. Case (e) is a case with three static obstacles. Case (f) is a case with two dynamic obstacles moving in opposite directions and two static obstacles.

Table 6. Collision risk (CR) in cases (**a**)–(**f**).

Case	CR	
	FSD without Speed Information	FSD with Speed Information
(a)	21.21	18.97
(b)	89.60	14.56
(c)	75.97	68.95
(d)	11.95	5.84
(e)	23.31	21.57
(f)	107.89	54.43

Figure 9 are the results of the comparison between the cases of FSD without obstacle speed information in the circular grid map and the case of FSD by dividing areas with

speed information of obstacles in the trapezoidal grid map. In Figure 9, the white point is the vehicle driving target point, and the red line represents the generated path according to the free space. As a result of FSD without speed information, it can be seen that a path with a collision risk is created even though there is an obstacle moving in the direction opposite to the vehicle direction. In the result of FSD with speed information, it is determined that the area where the obstacle is moving in the opposite direction to the vehicle's driving direction is an impossible area, and it is divided into red areas to indicate that a safe path is created in the space, not in the driving direction of the obstacle.

Table 6 is a quantitative representation of the comparison results of the two cases. As a result of the calculation, the results shown in Table 6 were obtained. A smaller CR value means that there is less risk of colliding with obstacles around the vehicle when traveling on that path. When the free space is detected using the speed information, the CR value is smaller than when the speed information is not used, and more secure driving is possible.

4. Discussion

The purpose of this study was to ensure the stable operation of autonomous vehicles. While it is important for autonomous vehicles to travel on the shortest route, it is also important to drive on a safe and optimal route with less collision risk. Most obstacles surrounding a moving vehicle are dynamic obstacles. Therefore, speed information must be taken into account when detecting free space. By excluding the free space in which driving is impossible in consideration of the speed information, unnecessary calculations can be reduced, and the risk of a collision can be further reduced. This can be confirmed by the difference between the result path creation screen and the collision risk. If this algorithm is applied to an autonomous vehicle, a safe route can be generated in various environments where route creation is impossible, such as an environment where lanes are not recognized, or where it is difficult to build an accurate map. Therefore, safer driving is possible through this algorithm.

5. Conclusions

In autonomous driving, generating the fastest path is important, but creating a safe path with less collision risk with surrounding obstacles and driving safely are also two of the most important aspects. As a result of the path generation comparison, when the FSD algorithm including the speed information proposed in this paper was used, it was possible to generate a path with a low collision risk. In the future, we plan to conduct research on generating a path in real time in a more complex driving environment, avoiding obstacles by controlling the actual vehicle according to the path, and autonomously driving to the destination.

Author Contributions: Conceptualization, Y.L. and B.Y.; methodology, Y.L. and B.Y.; software, Y.L. and B.Y.; validation, Y.L. and B.Y.; formal analysis, Y.L.; investigation, Y.L.; resources, Y.L.; data curation, Y.L.; writing—original draft preparation, Y.L.; writing—review and editing, Y.L. and B.Y.; visualization, Y.L.; supervision, B.Y.; project administration, B.Y.; funding acquisition, B.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding or This research was funded by MOTIE grant number P0018599.

Acknowledgments: This research was financially supported by the Ministry of Trade, Industry and Energy (MOTIE), Korea, under the "Regional Industry Infrastructure and R&D Support Program" (P0018599) supervised by the Korea Institute for Advancement of Technology (KIAT).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Eraqi, H.M.; Honer, J.; Zuther, S. Static free space detection with laser scanner using occupancy grid maps. *arXiv* **2018**, arXiv:1801.00600.
2. Fernández, C.; Gavilán, M.; Llorca, D.F.; Parra, I.; Quintero, R.; Lorente, A.G.; Vlacic, L.; Sotelo, M. Free space and speed humps detection using lidar and vision for urban autonomous navigation. In Proceedings of the 2012 IEEE Intelligent Vehicles Symposium, Alcala de Henares, Spain, 3–7 June 2012; pp. 698–703.
3. Tao, J. 3D LiDAR based Drivable Road Region Detection for Autonomous Vehicles. Master's Thesis, KTH Royal Institute of Technology, School of Electrical Engineering and Computer Science, Stockholm, Sweden, 2020.
4. De Silva, V.; Roche, J.; Kondoz, A. Robust fusion of LiDAR and wide-angle camera data for autonomous mobile robots. *Sensors* **2018**, *18*, 2730. [CrossRef] [PubMed]
5. Neumann, L.; Vanholme, B.; Gressmann, M.; Bachmann, A.; Kählke, L.; Schüle, F. Free space detection: A corner stone of automated driving. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Las Palmas de Gran Canaria, Spain, 15–18 September 2015; pp. 1280–1285.
6. Pizzati, F.; García, F. Enhanced free space detection in multiple lanes based on single CNN with scene identification. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 2536–2541.
7. Patra, S.; Maheshwari, P.; Yadav, S.; Banerjee, S.; Arora, C. A joint 3d-2d based method for free space detection on roads. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 643–652.
8. Pazhayampallil, J. *Free Space Detection with Deep Nets for Autonomous Driving*; Report; Stanford University: Stanford, CA, USA, 2014.
9. Hänisch, S.; Evangelio, R.H.; Tadjine, H.H.; Pätzold, M. Free-space detection with fish-eye cameras. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Redondo Beach, CA, USA, 11–14 June 2017; pp. 135–140.
10. Li, M.; Feng, Z.; Stolz, M.; Kunert, M.; Henze, R.; Küçükay, F. High Resolution Radar-based Occupancy Grid Mapping and Free Space Detection. In Proceedings of the VEHITS, Madeira, Portugal, 16–18 March 2018; pp. 70–81.
11. Hamandi, M.; Asmar, D.; Shammas, E. Ground segmentation and free space estimation in off-road terrain. *Pattern Recognit. Lett.* **2018**, *108*, 1–7. [CrossRef]
12. Haltakov, V.; Belzner, H.; Ilic, S. Scene understanding from a moving camera for object detection and free space estimation. In Proceedings of the 2012 IEEE Intelligent Vehicles Symposium, Alcala de Henares, Spain, 3–7 June 2012; pp. 105–110.
13. Kraemer, S.; Stiller, C.; Bouzouraa, M.E. LiDAR-based object tracking and shape estimation using polylines and free-space information. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4515–4522.
14. De Silva, V.; Roche, J.; Kondoz, A. Fusion of LiDAR and camera sensor data for environment sensing in driverless vehicles. *arXiv* **2017**, arXiv:1710.06230v2.
15. Yahya, M.A.; Abdul-Rahman, S.; Mutalib, S. Object detection for autonomous vehicle with LiDAR using deep learning. In Proceedings of the 2020 IEEE 10th International Conference on System Engineering and Technology (ICSET), Virtual Conference, Shah Alam, Malaysia, 9 November 2020; pp. 207–212.
16. Lee, S.; Lee, D.; Choi, P.; Park, D. Accuracy–Power Controllable LiDAR Sensor System with 3D Object Recognition for Autonomous Vehicle. *Sensors* **2020**, *20*, 5706. [CrossRef]
17. Zhao, X.; Sun, P.; Xu, Z.; Min, H.; Yu, H. Fusion of 3D LIDAR and camera data for object detection in autonomous vehicle applications. *IEEE Sens. J.* **2020**, *20*, 4901–4913. [CrossRef]
18. Tang, L.; Shi, Y.; He, Q.; Sadek, A.W.; Qiao, C. Performance test of autonomous vehicle lidar sensors under different weather conditions. *Transp. Res. Rec.* **2020**, *2674*, 319–329. [CrossRef]
19. Verucchi, M.; Bartoli, L.; Bagni, F.; Gatti, F.; Burgio, P.; Bertogna, M. Real-Time clustering and LiDAR-camera fusion on embedded platforms for self-driving cars. In Proceedings of the 2020 Fourth IEEE International Conference on Robotic Computing (IRC), Taichung, Taiwan, 9–11 November 2020; pp. 398–405.
20. John, V.; Karunakaran, N.M.; Guo, C.; Kidono, K.; Mita, S. Free space, visible and missing lane marker estimation using the PsiNet and extra trees regression. In Proceedings of the 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018; pp. 189–194.
21. Xu, F.; Chen, L.; Lou, J.; Ren, M. A real-time road detection method based on reorganized lidar data. *PLoS ONE* **2019**, *14*, e0215159. [CrossRef]
22. Yao, J.; Ramalingam, S.; Taguchi, Y.; Miki, Y.; Urtasun, R. Estimating drivable collision-free space from monocular video. In Proceedings of the 2015 IEEE Winter Conference on Applications of Computer Vision, Waikoloa Beach, HI, USA, 5–9 January 2015; pp. 420–427.
23. Kessler, T.; Minnerup, P.; Esterle, K.; Feist, C.; Mickler, F.; Roth, E.; Knoll, A. Roadgraph Generation and Free-Space Estimation in Unknown Structured Environments for Autonomous Vehicle Motion Planning. In Proceedings of the IEEE 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2831–2838.
24. Aihara, R.; Fujmoto, Y. Free-space estimation for self-driving system using millimeter wave radar and convolutional neural network. In Proceedings of the 2019 IEEE International Conference on Mechatronics (ICM), Ilmenau, Germany, 18–20 March 2019; pp. 467–470.

25. Jang, C.; Sunwoo, M. Semantic segmentation-based parking space detection with standalone around view monitoring system. *Mach. Vis. Appl.* **2019**, *30*, 309–319. [CrossRef]
26. Gkolias, K.; Vlahogianni, E. Convolutional neural networks for on-street parking space detection in urban networks. *IEEE Trans. Intell. Transp. Syst.* **2018**, *20*, 4318–4327. [CrossRef]
27. Trejo, S.; Martinez, K.; Flores, G. Depth map estimation methodology for detecting free-obstacle navigation areas. In Proceedings of the 2019 IEEE International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 11–14 June 2019; pp. 916–922.
28. Chen, Y.; Zhou, L.; Bouguila, N.; Wang, C.; Chen, Y.; Du, J. BLOCK-DBSCAN: Fast clustering for large scale data. *Pattern Recognit.* **2021**, *109*, 107624. [CrossRef]
29. Hasan, M.; Hanawa, J.; Goto, R.; Fukuda, H.; Kuno, Y.; Kobayashi, Y. Person Tracking Using Ankle-Level LiDAR Based on Enhanced DBSCAN and OPTICS. *IEEJ Trans. Electr. Electron. Eng.* **2021**, *16*, 778–786. [CrossRef]
30. Munho, N. Development of Moving Object Detection Method for Automotive 3D LiDAR Sensor using Multi-Resolution Clustering Algorithm. Master's Thesis, Graduate School, Kookmin Unuiversity, Seoul, Korea, 2019.
31. Zhao, L.; Wang, M.; Su, S.; Liu, T.; Yang, Y. Dynamic Object Tracking for Self-Driving Cars Using Monocular Camera and LIDAR. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 10865–10872.
32. Hong, Z.; Sun, P.; Tong, X.; Pan, H.; Zhou, R.; Zhang, Y.; Han, Y.; Wang, J.; Yang, S.; Xu, L. Improved A-Star Algorithm for Long-Distance Off-Road Path Planning Using Terrain Data Map. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 785. [CrossRef]
33. Zheng, T.; Xu, Y.; Zheng, D. AGV path planning based on improved A-star algorithm. In Proceedings of the 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 11–13 October 2019; pp. 1534–1538.
34. Jeong, S.; Gim, J.; Ahn, C. V2V Based Vehicle Detection and Collision Avoidance Algorithm. *Trans. Korean Soc. Automot. Eng.* **2018**, *26*, 773–782. [CrossRef]
35. Li, Y.; Wu, D.; Lee, J.; Yang, M.; Shi, Y. Analysis of the transition condition of rear-end collisions using time-to-collision index and vehicle trajectory data. *Accid. Anal. Prev.* **2020**, *144*, 105676. [CrossRef] [PubMed]