

ORIGINAL ARTICLE

A robust collision prediction and detection method based on neural network for autonomous delivery robots

Seonghun Seo¹  | Hoon Jung²

¹Cognition & Transportation ICT
Research Section, Electronics and
Telecommunications Research Institute,
Daejeon, Republic of Korea

²Postal and Logistics Technology
Research Center, Electronics and
Telecommunications Research Institute,
Daejeon, Republic of Korea

Correspondence

Seonghun Seo, Cognition &
Transportation ICT Research Section,
Electronics and Telecommunications
Research Institute, Daejeon, Republic of
Korea.

Email: ssh@etri.re.kr

Funding information

Institute of Information and
Communications Technology Planning
and Evaluation, Grant/Award Number:
2020-0-00297-001

Abstract

For safe last-mile autonomous robot delivery services in complex environments, rapid and accurate collision prediction and detection is vital. This study proposes a suitable neural network model that relies on multiple navigation sensors. A light detection and ranging technique is used to measure the relative distances to potential collision obstacles along the robot's path of motion, and an accelerometer is used to detect impacts. The proposed method tightly couples relative distance and acceleration time-series data in a complementary fashion to minimize errors. A long short-term memory, fully connected layer, and SoftMax function are integrated to train and classify the rapidly changing collision countermeasure state during robot motion. Simulation results show that the proposed method effectively performs collision prediction and detection for various obstacles.

KEYWORDS

autonomous delivery robot, classification, collision prediction and detection, light detection and ranging, long short-term memory, neural network

1 | INTRODUCTION

With the development of robotic logistics services, the demand for autonomous delivery services is increasing [1]. Several process automation techniques have been applied to indoor environments, such as inside factories and warehouses. For example, a robot palletization method was recently proposed to simplify supply-chain logistics and reduce initial robot setting times [2]. In similar studies, robotic palletization sensing, control, and operational techniques have been advanced [3, 4]. In-factory logistics have been actively pursued for many practical applications, including an autonomous mobile robot that transports boxes of materials to

workstations to reduce human workloads [5]. Other studies have promoted frameworks for automating logistics in other types of indoor environments [6, 7].

Recently, several studies have focused on the last-mile delivery aspects of these technologies, in which customer deliveries are the focus. These and similar capabilities have received increased attention owing to COVID-19 pandemic restrictions. To ensure the safe operation of robots, collision prevention technologies are needed based on the recognition of obstacles, and rapid response capabilities are needed for accidents when they occur. Historically, disparity maps of depth cameras have been used for robotic arm collision prevention in manufacturing [8]. However, these are only good for simple,

repetitive, and stationary robotic movements. The scope of human–robot interactions has now expanded to mobile transportation capabilities, and the need for expanded safety measures has dramatically increased.

Contemporary intuitive collision detection methods rely on tactile sensors that directly measure contact and impact pressures [9]. However, stand-off methods are urgently needed to sense collisions before they happen. Thus, light detection and ranging (LiDAR), inertial measurement unit (IMU), and Global Navigation Satellite System (GNSS) devices have been used to assess proximities during motion, especially in autonomous driving fields [10, 11]. Some studies [12–14] have proposed collision avoidance methods using these sensors and their combinations. Additionally, machine-learning path-planning techniques have been investigated for collision avoidance [15]. Based on promising autonomous navigation results, effective collision avoidance has been demonstrated. However, existing techniques rely on localization results that are optimized for abnormality detection. Unfortunately, this technique allows for significant blind spots in which hazardous obstacles are not recognized as threats [16]. Hence, although collision detection can be performed quite effectively, its prediction is not very mature.

Because the autonomous robots that are envisioned for our logistical scenario will be required to traverse non-roadway infrastructures (e.g., sidewalks, passageways, alleys, trails), a new and more dynamic type of collision recognition is needed. To move toward this goal, we provide robot training that accounts for a wide variety of operational scenarios using continuous time-series data.

Our robust collision prediction and detection method uses LiDAR to obtain the relative distances to obstacles and an accelerometer to measure the impulse when a collision occurs. The sensor data are tightly coupled and input to a neural network model comprising a long short-term memory (LSTM), fully connected layer, and appropriate SoftMax functions. Although gate recurrent units (GRUs) are often applied to help avoid vanishing gradient problems with a minimum number of gates, an LSTM can cope with the vanishing gradient more effectively using a larger number of gates [17], which is more appropriate for our dynamic needs. Additionally, when the features and sequences of the input data are large, the characteristics of the time series can be better reflected using an LSTM in a cellular state.

The contributions of this study are as follows:

- A simultaneous collision prediction and detection system that does not depend on navigation results, even when navigation sensors are used.

- An optimized neural network that robustly classifies collision states using tightly coupled real-time time-series sensor data.

The remainder of this paper is organized as follows. In Section 2, the concept of our approach is described, and in Section 3, the collision prediction and detection method is detailed. Section 4 presents simulation test results, and we conclude in Section 5.

2 | CONCEPTUAL APPROACH

Because delivery robots are generally quite smaller than autonomous vehicles, which can be equipped with many sensors (e.g., GNSS, LiDAR, and IMUs), delivery robots are normally equipped with only the most essential sensors. Hence, because our robot must predict and detect collisions by rapidly recognizing obstacles along the robot's movement path, we employ LiDAR for accurate relative distance measurements. We also use an accelerometer to rapidly detect collisions that will inevitably occur. Because rapid changes in acceleration can be caused by events other than collisions, the proposed method provides complementary services that use tightly coupled measurements. As a result, fixed-threshold countermeasures are neither appropriate nor effective. Hence, we apply a novel neural network model for dynamic, universal prediction and detection.

3 | COLLISION PREDICTION AND DETECTION METHOD

3.1 | System overview

The proposed method is illustrated in Figure 1. The algorithm is divided into three parts: LiDAR distance extraction, LSTM model training, and collision detection. First, three-dimensional (3D) point-cloud data (PCD) are converted to the relative distance measurement to the front obstacle along the robot's movement path using LiDAR distance extractions. The relative distance measurements are then combined with three-axis acceleration measurements before being fed to the LSTM model. The acceleration measurements are processed using a low-pass filtering method to alleviate sensor noise. The model is then trained by updating the changes in relative distance and acceleration. Finally, the training results are provided to a SoftMax function for collision state classification via the fully connected layer. By adopting various obstacle types, the proposed network optimizes training and classification performance.

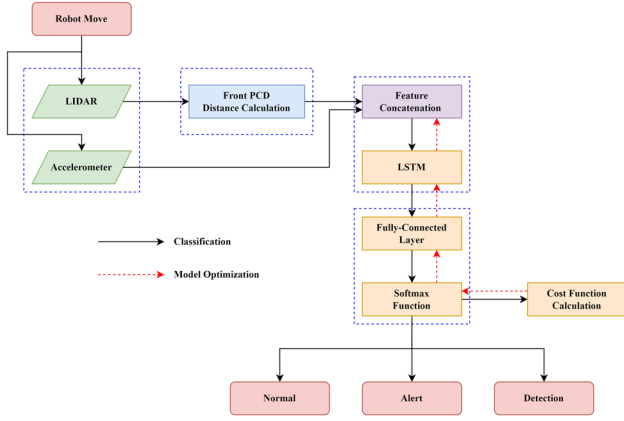


FIGURE 1 Proposed collision prediction and detection algorithm.

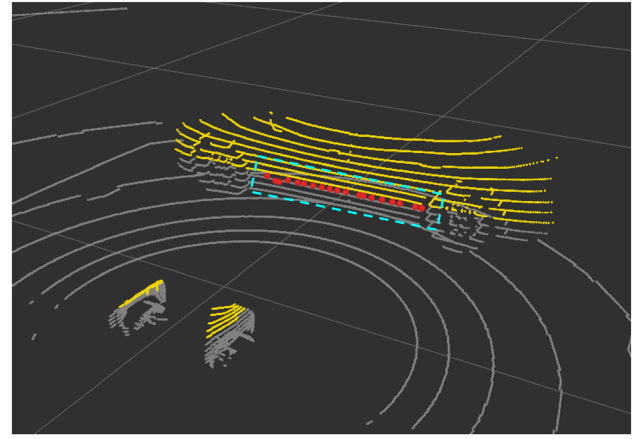


FIGURE 2 LiDAR point cloud of the front obstacle. Yellow points indicate a greater height than the z-axis. Gray points indicate a lower height than the z-axis. Red points represent projected two-dimensional points in each lateral section.

3.2 | LiDAR distance extraction

LiDAR provides PCD that reflects the objects in the environment around the robot. We first voxelize the PCD. Then, by considering the lateral size of the robot, only the front objects along the trajectory are extracted [18] using the front-area filter. The voxelized PCD contains the relative 3D position information of each point measured around the origin of the LiDAR sensor axis. These 3D positions are then projected onto a plane comprising the two horizontal LiDAR axes (Figure 2). The projected points are divided into consecutive sections based on their lateral positions.

Using two-dimensional (2D) PCD data, the relative distance, ρ_t^n , between the robot and the objects along the path can be calculated as

$$\rho_t^n = \sqrt{(x_t^n)^2 + (y_t^n)^2} \quad (n = 1, 2, \dots, N), \quad (1)$$

where N denotes the total number of projected frontal 2D PCDs, t represents time, and (x, y) denotes the relative positions of each point in the sensor frame. Then, the closest relative distance, $\rho_{\min,t}^j$, in each lateral section is selected as the set of relative distances between the obstacle and the robot, as represented in (2).

$$\rho_{\min,t}^j = \underset{k}{\operatorname{argmin}}(\{\rho_t^k | k \in N_j\}) \quad (j = 1, 2, \dots, J), \quad (2)$$

where J denotes the number of lateral sections and N_j represents the number of points in the j th section. The sum of the lengths of the lateral sections is determined by considering the lateral size of the robot.

3.3 | LSTM model training

Before training the neural network, the measurements obtained by each sensor are concatenated to generate input data for the LSTM. The concatenated measurement, M_t , at each time step is continuously stored in chronological order and used for time-series model training:

$$M_t = [\rho_{\min,t}^1, \rho_{\min,t}^2, \dots, \rho_{\min,t}^J, a_t^x, a_t^y, a_t^z], \quad (3)$$

$$M = [M_0^T, M_1^T, \dots, M_t^T, M_{t+1}^T, \dots], \quad (4)$$

where (a_t^x, a_t^y, a_t^z) denotes the acceleration of each accelerometer axis and T represents the transposition. All measurements are normalized before input to the LSTM so that it can be trained to determine how M changes over time as the robot approaches an obstacle.

Figure 3 presents a time-series raw measurement example of the three-axis acceleration and minimum relative distance measurements in J sections. When moving without obstacles, acceleration measurement is affected by the movement of the robot in terms of rotation, acceleration, or road surface conditions. When an obstacle is detected, the acceleration trend persists, but the relative distance decreases. As the robot gradually approaches and collides with the obstacle, the acceleration increases dramatically between 120 and 330 time steps. Thereafter, the acceleration is similar to that of the normal state, but the relative distance between the vehicle and the object remains near zero. Owing to the continuity of these measurements, the LSTM model can achieve robust

performance against instantaneous errors from the LiDAR and accelerometer. For example, if the relative distance is suddenly detected erroneously and shows a value near zero while the robot is driving normally, the error can be safely filtered using the previous state information. Unlike traditional methods that use a fixed specific collision threshold [9], the LSTM model learns the changes in acceleration and relative distance that occur during collisions in a variety of cases, enabling the model to be more versatile.

Figure 4 shows the structure of the LSTM model [19], whose principal processes are represented in (5) through (10).

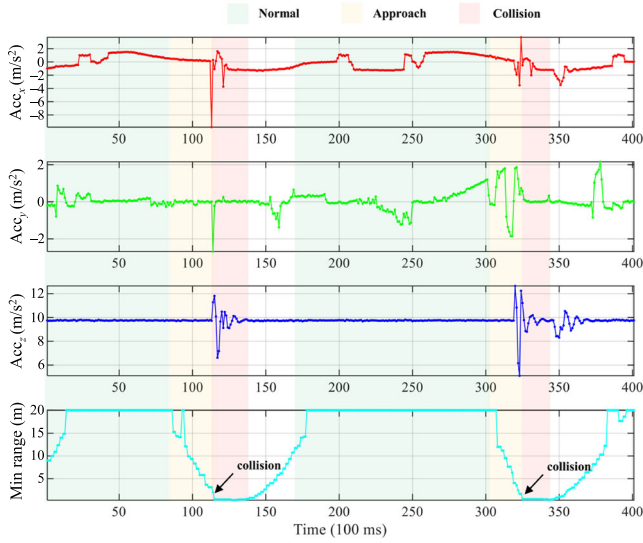


FIGURE 3 Example of the concatenated measurement. As the robot approaches an obstacle, the minimum relative distance decreases. During a collision, the acceleration changes abruptly, and the minimum relative distance approaches zero.

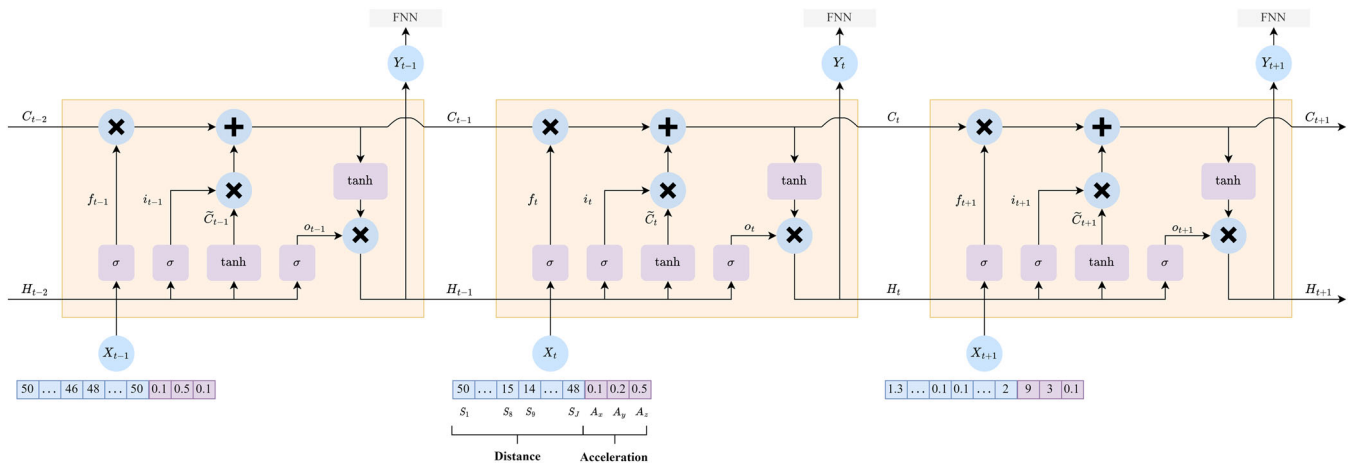


FIGURE 4 Structure of the long short-term memory (LSTM) model. The concatenated measurements are input during each stage. The training result at the current stage influence the training of the next stage based on the LSTM cell.

$$f_t = \sigma(W_f \cdot [H_{t-1}, X_t] + b_f), \quad (5)$$

$$i_t = \sigma(W_i \cdot [H_{t-1}, X_t] + b_i), \quad (6)$$

$$\tilde{C}_t = \tanh(W_C \cdot [H_{t-1}, X_t] + b_C), \quad (7)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t, \quad (8)$$

$$o_t = \sigma(W_o \cdot [H_{t-1}, X_t] + b_o), \quad (9)$$

$$Y_t = o_t * \tanh(C_t), \quad (10)$$

where W represents the weight of the neuron in each layer and b represents a bias of the same neuron. A key element of LSTM is the cell state, C_t , which can store prior training results and reflect them on the current results. First, the forget-gate layer, f_t , determines the trained information of the previous stage that must be excluded from the current training cycle. Second, input gate layer i_t decides which of the new data must be stored in C_t . Third, the tanh layer generates a new candidate value, \tilde{C}_t , for application in C_t . Finally, the output of the current stage is generated through the input from the sigmoid layer, o_t , and tanh layers with C_t .

3.4 | Collision classification

The output of the LSTM model at each time step is input to the SoftMax function at the end of the fully connected layer, producing three outputs corresponding to normal, collision prediction, and detection states in order. Figure 5 depicts the collision classification process. The

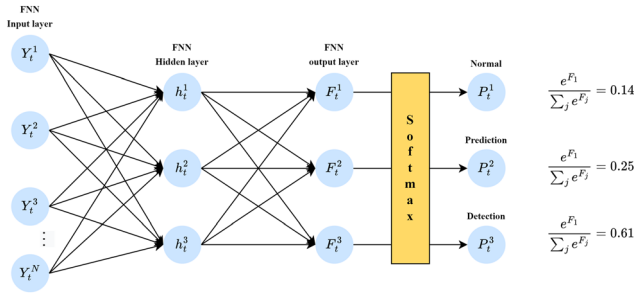


FIGURE 5 Collision classification algorithm. In the fully connected layer, the long short-term training results are organized into three outputs that are provided to a SoftMax function to provide probabilities with respect to normal, prediction and collision states.

LSTM training results are organized into three outputs using the fully connected layer. The outputs are then fed into the SoftMax function, which generates three probability values with the sum of probabilities equaling one. For precise classification, the labels are encoded using a one-hot vector.

4 | SIMULATION TEST

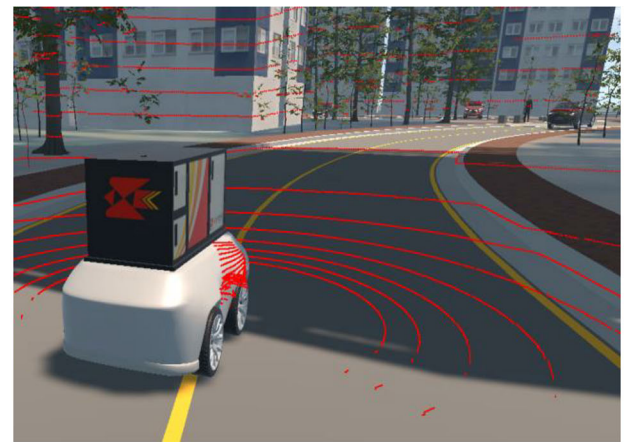
4.1 | Testing environment

A test was conducted to evaluate model performance when employed by an autonomous delivery robot in a complex simulated apartment complex environment (Figure 6). Velodyne's VLP-16 LiDAR model was used with a Bosch BMI055 IMU accelerometer. The latter comprised a built-in Intel D435i camera. The simulator used the Unity physics engine [20] equipped with Nvidia's Physx engine to ensure that the physical properties of the surrounding objects well-reflected real-world sensor data.

Figure 7 presents examples of several collision opportunities. We created scenarios for each and collected data that were serially fused to train the time-series LSTM. Table 1 lists the specifications of the collected data. From a total of 11 671 time steps, the data contained 24 feature columns, including the three-axis acceleration and minimum relative distance measurements of 21 lateral sections. Data from 8169 time steps were used for training, and the remaining 3502 were used for validation. Additionally, three labels for output were generated for normal, collision prediction, and detection states in the form of one-hot vectors. We trained the LSTM via post-processing and used the pre-trained model for validation.



(A)



(B)

FIGURE 6 Simulation environment for collecting collision data: (A) referenced real driving environment and (B) simulation environment.

The model was configured in a many-to-many format to confirm sequential training at each time step, as noted in Table 2. Fully connected layer outputs were reduced to three for SoftMax operation. Overall, the activations were not used to reflect the characteristics of time-series input in the first three layers.

4.2 | Test results

Using the simulator, data were acquired for various obstacles, and collision predictions and detections were performed. Each input batch data sequence was 100 time steps long, corresponding to 10 s, and 10 randomly selected sequences were repeated at each iteration. Figure 8 shows the training and validation losses. The training loss was initially large, but it decreased to ~ 0.1 as the number of iterations increased. Conversely, the

validation loss initially showed a low value, but it converged near the 15th iteration. These findings suggest that the model was slightly overfitted to the training data.



(A)



(B)



(C)

FIGURE 7 Various types of obstacles: (A) car, (B), barrier, (C) wooden box.

TABLE 1 Input data specifications.

Input data	Description	Time step(100 ms)	Column order
Acceleration	x-axis acceleration	11 671	1
	y-axis acceleration	11 671	2
	z-axis acceleration	11 671	3
Distance	1st section	11 671	4
	⋮	11 671	⋮
	21th section	11 671	24

However, the targeted training loss and classification performance were satisfactory. Consequently, the trained model was suitable for use with the remaining analyses.

Figure 9 shows the collision prediction and detection classification results obtained using the proposed method. The true values represent one-hot vector labels in Figure 9A–C, and the classified values represent the model output. Overall, the classified values correspond to the true values. However, classification was not performed nominally in certain areas.

Figure 10 shows the test results from 4100 to 4500 time steps. At steps 4312 and 4330, the collision recognition state labels were positive based on relative distances and acceleration values. However, these events were deemed as cases in which the LiDAR raw measurements were incorrectly provided during normal driving. However, from the classification results, we confirmed that, although there was a change in probability caused by incorrect labels, each state was effectively maintained.

For the one-hot encoding labels, it was difficult to intuitively assess success or failure as each case was alternately expressed as a zero or one. Figure 11 presents a comparison of true and classified values, as represented in Figures 9 and 10, respectively, which were converted to integer encodings. Each label and classification state was encoded as one, two, or three for normal, collision prediction, and detection states, respectively, as integers. The zero value in Figure 11 indicates that the label and classification results were the same. A value of one indicates a classification delay as the state changes. Moreover, a value of two indicates that the proposed method failed to classify the collision state. In reality, these

TABLE 2 Layer specifications.

NN model	Layer	Activation	No. of output
LSTM	LSTM	None	64
Fully connected	Dense-1	tanh	32
	Dropout	None	32
	Dense-2	tanh	8
	Dense-3	Softmax	3

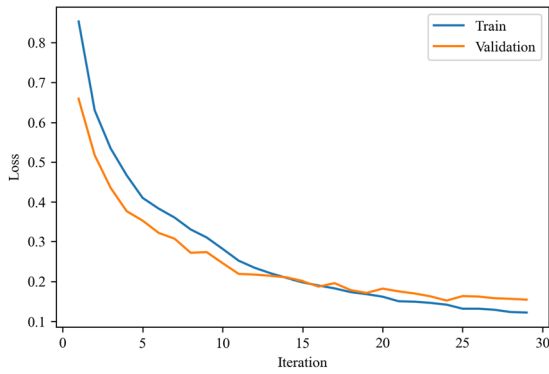


FIGURE 8 Neural network model loss trend.

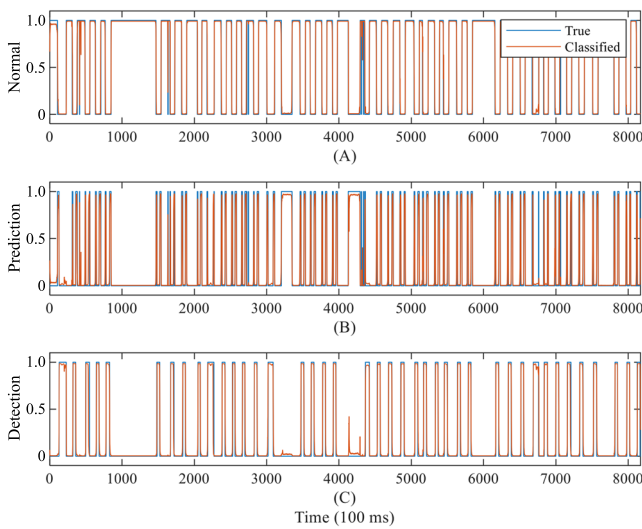


FIGURE 9 Collision state classification results: (A) normal state, (B) collision prediction state, and (C) detection state. True values represent one-hot vector labels, and classified values represent neural network model output.

situations most commonly occurred when there was no actual collision; however, there was a sudden acceleration change related to other factors. For example, small and thin obstacles, such as plastic lane markers, may not be detected at long distances by LiDAR; however, they are registered at close positions. To address this shortcoming, more training data are needed. However, as shown in Figure 11, the classification success rate was approximately 96%, confirming that the proposed method achieves very high performance. The delivery robot targeted in the proposed method moved at a maximum speed of 6 km/h with a response speed of less than 1 ms from the control command. Additionally, the proposed method uses a pretrained classification model; hence, the classification results can be output within 10 ms, including the LiDAR sensor processing time, for actual

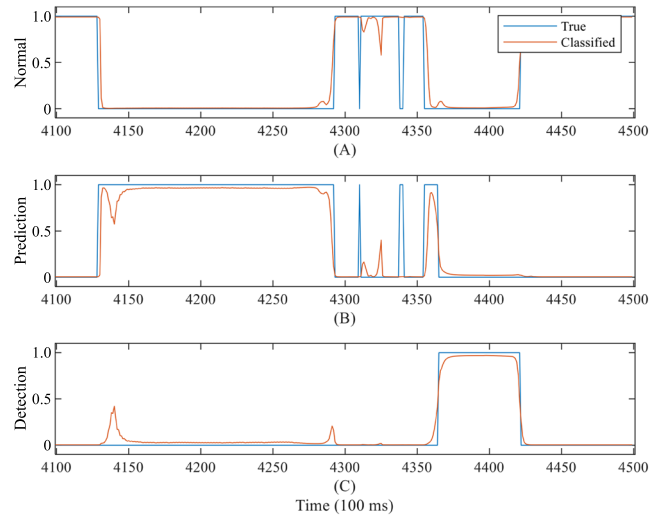


FIGURE 10 Enlarged collision state classification results: (A) normal state, (B) collision prediction state, and (C) detection state.

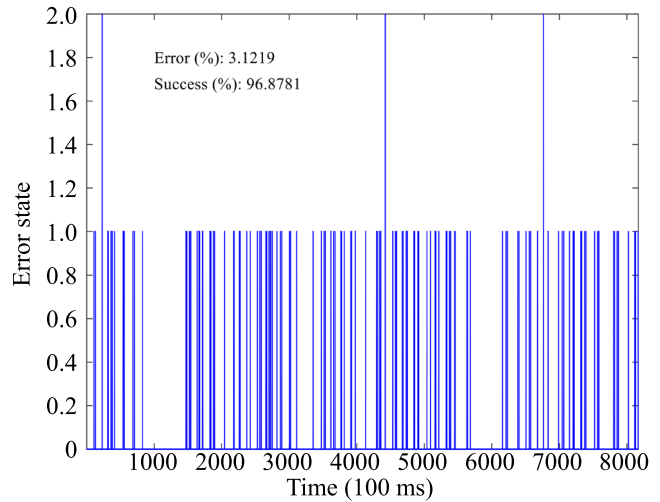


FIGURE 11 Error state based on integer encodings of labels and classification results. 0, 1, and 2 represent successful, delayed, and misclassified results, respectively.

applications. When the classification results are output at 10 Hz, the relative distance to an obstacle can be distinguished with a high resolution of approximately 16 cm.

In conclusion, we confirmed that the proposed method adequately handles instantaneous misclassifications when applied to an autonomous delivery robot that primarily moves at slow speeds.

5 | CONCLUSION

Because a delivery robot operates in a complex mobility environment with a variety of obstacles, a method for

predicting and detecting collisions is vital. In this study, we proposed a robust collision prediction and detection method based on a time-series classification LSTM model. With the proposed method, navigation sensors were used without the need for additional sensors, and tightly coupled sensor measurements were processed by the fully connected layer and SoftMax function to achieve robust and precise collision state classification performance. Furthermore, because the relative distance was used for measurement, the method could be utilized universally without restrictions with static and dynamic obstacles. The performance of the proposed algorithm could be improved by learning about various surrounding environments and obstacle shapes using additional machine-learning options. Our future studies will focus on acquiring and applying various measurements to a wider range of obstacles so that we can improve our algorithm.

ACKNOWLEDGMENTS

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-00297-001, Development of delivery assistant technology and commercialization field test for high weight movable delivery service based on 5G).

CONFLICT OF INTEREST STATEMENT

The authors declare that there are no conflicts of interest.

ORCID

Seonghun Seo  <https://orcid.org/0000-0003-0582-5514>

REFERENCES

1. A. Dekhne, G. Hastings, J. Murnane, and F. Neuhaus, Mckinsey, 2019.
2. F. M. Moura and M. F. Silva, *Application for automatic programming of palletizing robots*, (Proc. IEEE International Conference on Autonomous Robot Systems and Competitions, Torres Vedras, Portugal), 2018, pp. 48–53.
3. S. Smys and G. Ranganathan, *Robot assisted sensing, control and manufacture in automobile industry*, J ISMAC **1** (2019), no. 3, 180–187.
4. G. A. Fontanelli, G. Paduano, R. Caccavale, P. Arpentì, V. Lippiello, L. Villani, and B. Siciliano, *A reconfigurable gripper for robotic autonomous depalletizing in supermarket logistics*, IEEE Robot Autom Lett **5** (2020), no. 3, 4612–4617.
5. A. Lourenco, F. Marques, R. Mendonca, E. Pinto, and J. Barata, *On the design of the ROBO-PARTNER intra-factory logistics autonomous robot*, (Proc. IEEE Int. Conf. Syst. Man Cybern., Budapest, Hungary), 2016, pp. 2647–2652.
6. D. Wahrmann, A. Hildebrandt, C. Schuetz, R. Wittmann, and D. Rixen, *An autonomous and flexible robotic framework for logistics applications*, J. Intel. Robot. Syst. **93** (2019), no. 3, 419–431.
7. R. Krug, T. Stoyanov, V. Tincani, H. Andreasson, R. Mosberger, G. Fantoni, and A. J. Lilienthal, *The next step in robot commissioning: Autonomous picking and palletizing*, IEEE Robot Autom Lett **1** (2016), no. 1, 546–553.
8. A. Mohammed, B. Schmidt, and L. Wang, *Active collision avoidance for human-robot collaboration driven by vision sensors*, Int J Comp **30** (2017), no. 9, 970–980.
9. M. Fritzsche, J. Saenz, and F. Penzlin, *A large scale tactile sensor for safe mobile robot manipulation*, (Proc. ACM/IEEE Int. Conf. Human-Robot Interaction, Christchurch, New Zealand), 2016, pp. 427–428.
10. Y. Sun, L. Guan, Z. Chang, C. Li, and Y. Gao, *Design of a low-cost indoor navigation system for food delivery robot based on multi-sensor information fusion*, Sensors **19** (2019), no. 22, 4980.
11. N. Kawarazaki, L. T. Kuwae, and T. Yoshidome, *Development of human following mobile robot system using laser range scanner*, Procedia Comp Sci **76** (2015), 455–460.
12. A. Asvadi, C. Premebida, P. Peixoto, and U. Nunes, *3D lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes*, Rob Autom Sys **83** (2016), 299–311.
13. A. Ravankar, A. A. Ravankar, A. Rawankar, and Y. Hoshino, *Autonomous and safe navigation of mobile robots in vineyard with smooth collision avoidance*, Agriculture **11** (2021), no. 10, 954.
14. J.-H. Cho, D.-S. Pae, M.-T. Lim, and T.-K. Kang, *A real-time obstacle avoidance method for autonomous vehicles using an obstacle-dependent Gaussian potential field*, J Adv Transp **2018** (2018), 15.
15. X. Xue, Z. Li, D. Zhang, and Y. Yan, *A deep reinforcement learning method for mobile robot collision avoidance based on double DQN*, (Proc. IEEE Int. Symp. Industrial Electronics, Vancouver, Canada), 2019, pp. 2131–2136.
16. F. Becker and M. Ebner, *Collision detection for a mobile robot using logistic regression*, (Proc. Int. Conf. Informatics in Control, Automation and Robotics, Lisbon, Portugal), 2019, pp. 167–173.
17. S. Yang, X. Yu, Y. Zhou, *LSTM and GRU neural network performance comparison study*, (Proc. Int. Workshop on Electronic Communication and Artificial Intelligence, Shanghai, China), 2020, pp. 98–101.
18. G. Chan, *LidarPerception/segmenters_lib*, 2020. Available from: http://github.com/LidarPerception/segmenters_lib/ [last accessed October 2021].
19. S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Comput **9** (1997), no. 8, 1735–1780.
20. Unity Technologies, *Unity documentation*, 2021, Available from: <https://docs.unity3d.com/kr/2021.2/Manual/PhysicsSection.html/> [last accessed October 2021].

AUTHOR BIOGRAPHIES



Seonghun Seo received BS and PhD degrees in electronics engineering from Konkuk University, Seoul, Rep. of Korea, in 2014 and 2020, respectively. Since 2020, he has worked for ETRI in Daejeon, Rep. of Korea, where he is a researcher. His

research interests are precise positioning and localization based on multiple sensor fusion for UAV and UGV, GPS signal processing, GNSS receivers, and anti-spoofing.



Hoon Jung received the PhD in industrial engineering from the University of Missouri, MO, USA, in 2001. Since then, he has worked for ETRI in Daejeon, Rep. of Korea, where he leads the Postal and Logistics Technology Research Center. His

research interests are drone delivery research and postal logistics technology development.

How to cite this article: S. Seo and H. Jung, *A robust collision prediction and detection method based on neural network for autonomous delivery robots*, ETRI Journal **45** (2023), 329–337. <https://doi.org/10.4218/etrij.2021-0397>