Received January 5, 2021, accepted February 21, 2021, date of publication February 24, 2021, date of current version March 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3062220

3D-LIDAR Based Object Detection and Tracking on the Edge of IoT for Railway Level Crossing

CRISTIAN WISULTSCHEW[1], GABRIEL MUJICA[1], (Member, IEEE), JOSE MANUEL LANZA-GUTIERREZ[2], AND JORGE PORTILLA[1], (Senior Member, IEEE)

[1]Centro de Electrónica Industrial, Universidad Politécnica de Madrid, 28006 Madrid, Spain
[2]Computer Science Department, Universidad de Alcalá, 28805 Alcalá de Henares, Spain

Corresponding author: Cristian Wisultschew (cristian.wpuigdellivol@upm.es)

This work was supported by the European Commission through the EU Horizon 2020 SCOTT Project under Grant 737422.

ABSTRACT Object detection is an essential technology for surveillance systems, particularly in areas with a high risk of accidents such as railway level crossings. To prevent future collisions, the system must detect and track any object that passes through the monitored area with high accuracy, and this process must be performed fulfilling real-time specifications. In this work, an edge IoT HW platform implementation capable of detecting and tracking objects in a railway level crossing scenario is proposed. The response of the system has to be calculated and sent from the proposed IoT platform to the train, so as to trigger a warning action to avoid a possible collision. The system uses a low-resolution 3D 16-channel LIDAR as a sensor that provides an accurate point cloud map with a large amount of data. The element used to process the information is a custom embedded edge platform with low computing resources and low-power consumption. This processing element is located as close as possible to the sensor, where data is generated to improve latency, privacy, and avoid bandwidth limitations, compared to performing processing in the cloud. Additionally, lightweight object detection and tracking algorithm is proposed in this work to process a large amount of information provided by the LIDAR, allowing to reach real-time specifications. The proposed method is validated quantitatively by carrying out implementation on a car road, emulating a railway level crossing.

INDEX TERMS Edge computing, embedded software, energy efficiency, Internet of Things, LIDAR, object detection, object tracking, railway level crossing, sensor systems and applications.

I. INTRODUCTION

The current situation regarding the high number of accidents at railway level crossings in Europe is one of the main motivations for this work. According to the annual report "Railway Safety in the European Union" [1] performed by the European Union Agency for Railways, 28.13% of the total fatalities recorded in Europe between 2010-2016 took place at level crossings. However, the improvements in reducing accidents at level crossings are limited at this time [2], [3]. As a way to address this fact and based on the growing advances in sensor and embedded processing technology, the robust and accurate data captured at level crossings could be used to feed a detection and tracking algorithm to avoid accidents or reduce the damage caused.

The associate editor coordinating the review of this manuscript and approving it for publication was Chin-Feng Lai.

In this regard, sensors are evolving to complex sensing systems, increasing the quality and the amount of information provided about the scenario to be analyzed. In recent literature, some sensors have shown to be useful in identifying usual targets as pedestrians in railway level crossing [4]. These sensors include conventional Red Green Blue (RGB) cameras, RGB-Depth cameras, radar, as well as ultrasonic and Light Detection And Ranging (LIDAR) sensors. Focusing on one of the most challenging areas in tracking and object detection, the autonomous vehicle research field, there is a growing tendency to fuse the information provided by different sensors, as the ones introduced before. The goal is to improve the detection capacity, although this approach includes additional complexity to the system [5], [6].

The type of detection to be performed in autonomous vehicles and level crossing scenarios share some aspects. First, due to the criticality of the system, it is required to detect the targets with the highest accuracy, reliability, and robustness to reduce the risk of an accident. It will involve using complex sensors, combined or alone, which should be able to work under any weather condition and day or night [7]. The second aspect is a consequence of the former, the use of complex sensors usually generates a large amount of data, which should be processed within a time constraint, besides applying computational demanding artificial intelligence algorithms. The third aspect is related to hardware constraints. Thus, in autonomous vehicles, the hardware should be embedded, whereas, in level crossing scenarios, the hardware could be deployed at any place unattended and with any weather condition. That implies that the hardware to be used should be embedded, implying hard restrictions in both computing capacity and energy cost, increasing the challenge of the system [8].

This embedded hardware approach fits with the edge computing concept in the Internet of Things (IoT), whose aim is to process the information as close to the sensing system as possible. This edge computing conception results in increasing reliability, privacy, and scalability, whereas latency and communications are reduced and simplified, respectively [9]–[11].

In this work, an embedded implementation for real-time object detection and tracking algorithm to be used within level crossing scenarios is proposed. The goal of this proposal is to reduce the number of accidents in this type of high-risk areas by monitoring the railway level crossing to inform the train driver about the existence of possible obstacles, warrantying a fast response to avoid potential accidents.

According to the previous discussion, the system proposed to be deployed at level crossing locations should be autonomous, both in terms of energy and decision making. This is because there may be no power supply at level crossing and the decision about a possible obstacle in the railway should be taken without further communication to avoid problems, such as latency, disconnection, security, and privacy. The limitation in energy and computing capacity implies that it could be interesting to reduce the number of sensors considered, due to further processing for data fusion.

In this regard, this work opts for considering a single sensor, which stands out for its high reliability and robustness, outperforming image-based systems. This sensor is a three-dimensional LIDAR (3D-LIDAR), which is characterized by providing a 3D point cloud with distance measurement in the surrounding. This sensor works in the absence of light and most weather conditions. However, the point cloud provided is known to be dense, meaning a large amount of data. This fact could imply an important limitation when managing the data in constrained embedded hardware. Based on this limitation, this work provides the following contributions:

- An embedded implementation of the data management for the 3D-LIDAR sensor is proposed, meeting real-time specifications for the application.
- The LIDAR data management feeds a lightweight object detection and tracking algorithm, avoiding data classification to keep low the computational load.
- The performance of the system is experimentally evaluated in a real-world deployment for a custom IoT node. The railway level crossing is emulated by a car road along with a pedestrian sidewalk.

The rest of this article is structured as follows. In Section II, related works within object detection and tracking applications are exposed. In Section III, the technical background about the LIDAR sensor, the embedded platforms, and the algorithms for object detection and tracking are discussed. In Section IV, a detailed description of the embedded implementation is presented. In Section V, a detailed description of the object detection and tracking algorithm considered is provided. The experimental setup used to perform the experimentation is presented in Section VI. Experimental results are discussed in Section VII. Finally, conclusions and future lines of work are provided in Section VIII.

II. RELATED WORK

In the recent literature about object detection and tracking systems applied to railway scenarios, it is common to find image-based approaches, such as in [12]–[14], getting accurate results for proper light conditions. However, this type of image-based solution is not robust in the absence of light, getting poor results. Thermal cameras were also considered in the field, getting acceptable results working in the absence of light [15]. However, distances were estimated using homography-based methods, which is not as accurate as the ones provided by laser-based systems.

Other sensing technologies were also considered. For instance, the work in [16] compares the usage of optical beams, ultrasounds, 3D-LIDARs, and RGB cameras (single and stereo ones) to detect obstacles at level crossings. This study resulted in that optical beams are easy to install and cheap, but weather-dependent. For ultrasonic detectors, the technology works for stationary and moving vehicles, but it has an extreme sensitivity to environmental conditions. Single RGB cameras perform well in adverse weather conditions, but it is limited in low illumination. Stereo RGB cameras performed better than single ones, but the system is extremely sensitive to adverse weather conditions. For 3D-LIDAR, the performance is robust in most weather conditions, but there are disadvantages related to operating conditions (e.g., vibration due to wind). However, these disadvantages can be solved with a higher investment in the LIDAR sensor. The main disadvantage of this technology is the cost. Nevertheless, the price of LIDAR sensors has decreased in recent years, and a further reduction is expected shortly.

Other works also stated that the LIDAR performance could be reduced in foggy environments [17]. Although, the robustness of the information provided can be increased significantly by using specifics lasers wavelengths.

Moreover, LIDARs performance is not considerably affected when working in rainy environments, as was demonstrated in [18]. Thus, LIDAR seems to be an adequate technology for the detection and tracking problem in crossing lines if sufficient investment is provided. In this regard, the LIDAR-based surveillance system in [19] stands out accurately detecting both large and small objects (with a volume of less than 10 dm^3) on the crossing lines.

LIDAR technology has been successfully applied in other critical use cases, such as autonomous vehicles, where LIDAR usually appears in combination with other sensors. The need for a combination of sensors relies on the high reliability and robustness in this field. For instance, the work in [20] combines ultrasound sensors, RGB cameras, and LIDAR sensors. The authors in [21] combined an RGB camera with a high-resolution 3D-LIDAR to feed a deep neural network, achieving state-of-the-art performance in the KITTI autonomous driving benchmark. The authors in [22] merged data from five low-resolution LIDARs, three millimeter-wave radars, an RGB camera, an inertial measurement unit, and a GPS.

In recent years, there is a growing tendency in working directly with the LIDAR point cloud, instead of applying a preprocessing stage for getting features of interest. Note that processing the point cloud is costly due to a large amount of data. In this regard, the authors may cite the works in [23]–[25], where deep neural networks are feed by the raw point cloud, also resulting in high computational effort. On the other hand, the work in [26] focuses on reducing the computation effort considering only a reduced set of points in the cloud to feed the deep neural network.

As stated before, this work focuses on building a robust surveillance system to be deployed in railway crossing lines. To this end, the authors propose a single sensor low-resolution 3D-LIDAR-based object detection and tracking system adapted to be embedded in a custom IoT edge node. The main novelty in this work relies on the embedded implementation of the data management of the 3D-LIDAR, which feeds the object detection and tracking algorithm, meeting real-time specifications. To this end, the data management of the point cloud is stated following the trend for directly considering the raw LIDAR data. However, instead of feeding a deep neural network with the raw cloud, the authors project the 3D-points on a 2D-plane, next applying vision computing methods to maintain computational load low. The authors should note that the results presented are from a real implementation for the system and then, they are not from simulations, which is valuable. As far as the authors know, there are no related works for 3D-LIDAR-based detection and tracking surveillance systems implemented to be executed in the IoT edge layer.

III. TECHNICAL BACKGROUND

In this section, an overview of the LIDAR sensor technology considered and the object detection and tracking image-based algorithms are provided.

A. LIDAR SENSOR TECHNOLOGY

LIDAR is a technology similar to radar, employing laser instead of radio waves and providing a point cloud with distance measurements in the surrounding. The operation of a LIDAR starts with the emission of a laser pulse, followed by the capture of the reflected laser by measuring the time the laser takes. The distance *dist* to a given object from the LIDAR is given by

$$dist = \frac{c * t}{2}, \tag{1}$$

where c is the speed of light and t is the time taking the laser from its emission to its reception.

For the specific case of a 3D-LIDAR, it has several lasers placed in a column configuration. The column rotates to generate a 360° point cloud map, with the reflectivity value of each point. 3D-LIDARs can be found with a different number of lasers from 1 to 128 in a single vertical column. The more the lasers in a column, the more the points generated in the point cloud (and also the price). The column rotates up to 1200 rpm producing a 3D point cloud map with up to 2.4 million points per second (using 128 lasers per column). The range reached by the lasers is usually up to 100 m. However, some versions extend this distance to 300 m [20].

B. OBJECT DETECTION AND TRACKING IMAGE-BASED TECHNOLOGIES

Object detection technology consists of identifying the presence and location of multiple classes of objects in digital images or videos. There are two significant trends in this field [27]. The first one employs machine learning and deep learning algorithms to perform the task, getting accurate results but usually requiring a medium-high computational effort. The second one consists of applying traditional image processing algorithms from the computer vision field, usually getting accurate results with a low computational effort, but without obtaining classification information.

As stated before, the surveillance system in this use case should identify the presence of obstacles in the railway, but it is not focused on knowing the class of the object. Moreover, the computing platform is a constrained IoT edge node, and then computation effort should also be constrained. Thus, the authors focus on considering traditional image processing algorithms.

The object detection algorithm proposed in this work is composed of three steps: background subtraction, data segmentation, and output generation. Background subtraction is an essential stage in this task. It allows removing the background from the image in a fixed setup camera, only keeping the moving zones. An overview of different strategies for background subtraction is in [28]. Once moving zones in the image are identified, the next step consists of segmenting or splitting the different moving zones into regions, where candidate obstacles could be. To this end, morphological operations and contour calculations are usually applied. Next, the algorithm calculates the coordinates for as many boxes

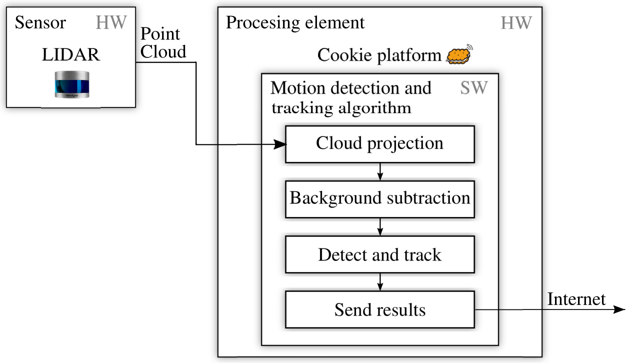


FIGURE 1. Flowchart for the edge surveillance system.

as regions were detected. Finally, a tracking algorithm follows the trajectory of each of the boxes. An overview of lightweight tracking strategies is in [29].

IV. EDGE HARDWARE ARCHITECTURE

In this section, the hardware architecture for the surveillance system in this work is described. It includes two main parts: the 3D-LIDAR, providing the point cloud with distance measurements about the environment through an Ethernet connection, and the IoT edge node, processing the data based on the object detection and tracking algorithm in Section V. Fig. 1 shows the flowchart for this architecture, including the hardware and software components. Note that the outputs (location, speed, and volume for each detected object) are sent via the Internet.

A. LIDAR VELODYNE VLP-16

The LIDAR used in this work is the Velodyne VLP-16 [1], which is composed of an array of 16 Infra-Red (IR) lasers paired with IR detectors to measure distance objects. The 16 lasers are organized in a single, vertical column and are oriented from $-15°$ to $15°$. The azimuth resolution is related to the rotation speed. Thus, the azimuth resolution is $0.1°$ with the minimum rotation speed (300 rpm) and $0.4°$ with the maximum rotation speed (1200 rpm). Moreover, the sensor has an IP67 category, supporting protection against dust and water immersion at 1 meter for 30 minutes without any filtration. The rest of the specifications are shown in Table 1.

The output data provided by this LIDAR is up to 0.3 million points per second using a 100 Mbps Ethernet connection. The output data are UDP packages providing information about the reflectivity and the spherical coordinates of each point, being the coordinate origin at the center of the LIDAR. It also provides the synchronized timestamps with μs resolution to merge data with other sensors.

B. PROCESSING ELEMENT: THE IoT COOKIE NODE

The embedded computing device used in this work to process the data coming from the LIDAR and provide a response about the scenario is a modular IoT platform called

TABLE 1. Velodyne VLP-16 LIDAR Specifications.

Sensor	Measurement range: 1 to 100 m Accuracy: +/- 3 cm (typical) Horizontal field of view: 360º Horizontal resolution: 0.1º - 0.4º Vertical field of view: 30º (-15º to 15º) Vertical resolution: 2º Rotation rate: 5-20 Hz
Laser	Class 1 eye safe from IEC 60825-1:2014 903 nm wavelength (min/max is 896/910 nm) Firing sequence repetition rate: 18.2 kHz Pulse duration: 6 ns Maximum output energy: 31 W
Mechanical and electrical specifications	Power consumption: 8 W Operating voltage: 9-32 VDC Weight: 830 g (without cabling) Dimensions: 103 mm diameter x 72 mm height Vibration: 5 Hz to 2000 Hz, 3G rms Operating temperature: -10 ºC to +60 ºC

Cookie [30], [31]. This platform was designed at *Centro de Electrónica Idunstrial* (CEI) of Universidad Politécnica de Madrid and is composed of four layers. Each one fulfills a specific purpose: communication, processing, power supply, and sensing/actuation. This structure allows including different layers according to the application. To this end, layers are bonded through vertical connectors, which are common to all the layers. Fig. 2 shows a Cookie connected to the LIDAR considered in this work.

Three Cookie layers are required in this work to process the information from the LIDAR and provide a response about the scenario, i.e., power-supply, processing, and communication layers. The power-supply layer needs 5 V to power the other layers. The processing layer is selected according to power consumption and computing resources specifications. The processing layer designed to perform edge computing includes a SAMA5D3 processor with an external RAM memory of 256 MB and an SD card reader. The SAMA5D3 processor is a 32-bit medium-performance, low-power ARM hard-float Cortex-A5 core with 2×32 kB cache memory working with a clock speed of 536 MHz and consuming less than 150 mW. The communication layer is composed of two USB connectors. One of those is connected with a USB-RJ-45 adapter to receive the information coming from the LIDAR Ethernet cable, whereas the other one is used to communicate the edge node to the Internet by using a USB-WiFi adapter.

An embedded Linux-based Operating System (OS) located in the flash SD card is run in the processing layer. The OS is a Debian 9 Stretch selected because of its compatibility with the known Open Source Computer Vision Library (OpenCV) in its version 3.4.8. [32], which is required for the application.

V. EDGE SOFTWARE ARCHITECTURE

This section describes the software implemented in the edge device for LIDAR data management and object detection and tracking. The process starts by creating a point cloud projection according to the point cloud provided by the

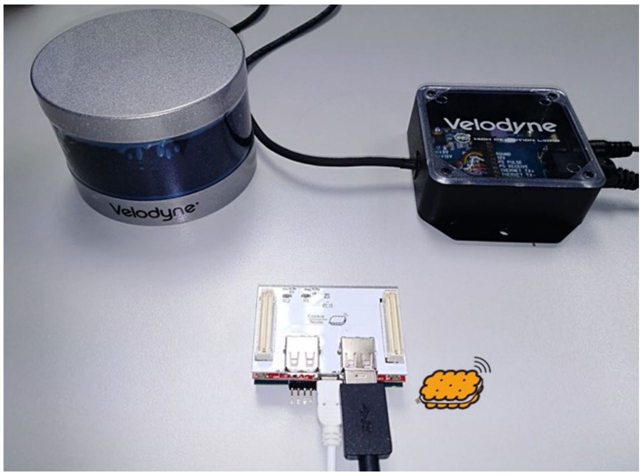


FIGURE 2. LIDAR connected to a Cookie.

LIDAR. Next, the background is subtracted, so that the objects are detected and tracked. Finally, the methodology to obtain the output parameters from the object detection algorithm is reported.

A. POINT CLOUD MAP PROJECTION

The algorithm in the processing layer starts by opening a communication socket to receive the raw data coming from the LIDAR via Ethernet in the form of UDP packets. Once a complete 360° point cloud map frame is received from the LIDAR, such as the one shown in Fig. 3a, a lightweight computation is required to convert the spherical data to Cartesian coordinates. Let r, α, and ω be the radius, the elevation, and the azimuth for a cloud point, respectively, then its Cartesian coordinates (x, y, z) are given by

$$\begin{aligned} x &= r * \cos\omega * \sin\alpha, \\ y &= r * \cos\omega * \cos\alpha, \\ z &= r * \sin\omega, \end{aligned} \tag{2}$$

where the Z-axis is perpendicular to the ground, passing through the center of the LIDAR.

Once the Cartesian coordinate point cloud is generated, the next step is to remove those points which are outside the Critical Region (CR) to be monitored to avoid waste computational resources in detecting objects in zones not required. Fig. 3b shows a photography of the real scenario used during the experimentation. In this figure, the CR is marked with a box, which includes a car road (emulating the railway) and a pedestrian sidewalk (the closest transit area to the railway). The identification of the CR should be manually performed the first time the algorithm is run, as a calibration process in the real deployment. This calibration could be automatized. However, as this process it is only required the first time, the authors opted for following a manual approach to ensure the best definition for the CR as possible.

After removing the points outside the CR, the next step is to generate an XZ plane projection of the LIDAR coordinates

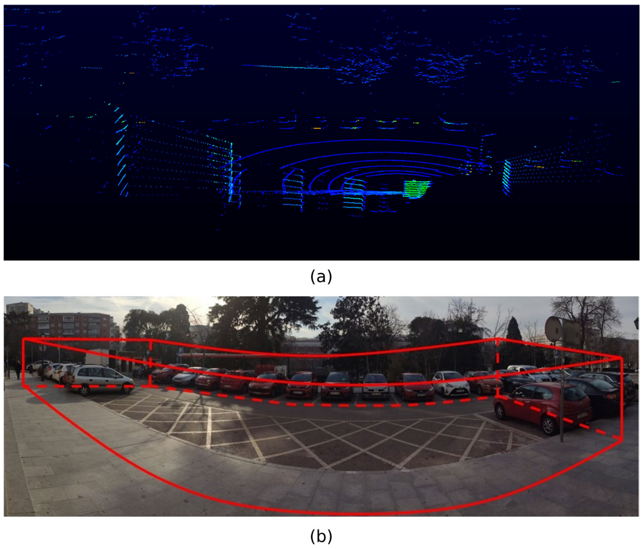


FIGURE 3. Point cloud and image of the CR to be analyzed. (a) Complete point cloud map frame received by the LIDAR, and (b) point cloud and image of the CR to be analyzed.

as a plot (JPEG image). The XY plane projection was not considered because of the reduced number of points provided due to overlapping. The idea of using JPEG images is that in further steps, the detection and tracking algorithms to be applied are within the computing vision field, instead of using the raw point cloud to reduce the computational requirements. Each point received by the LIDAR is represented with a fixed size of 5 pixels, the first one is located in the center, and the remaining four pixels are placed at the top, bottom, left, and right sides of the center pixel, in the form of a squared cross. A greyscale value between 0 and 255 is assigned to these 5 pixels, shown as white and black, respectively. If two or more points are overlapped in the same pixel, their values will be added, producing a darker pixel. As an example of the process, Fig. 4a shows an XZ projection plot with 14474 data points from the map frame in Fig. 3a. After removing the points outside the CR, the XZ projection results in 508 points, as Fig. 4b shows.

B. OBJECT DETECTION ALGORITHM

Starting from the image plane projections within the CR, the next step is to remove the point cloud background from subsequent frames to identify moving objects. To this end, it is necessary to first model the background of the scene by capturing a point cloud within the CR. In this initial capture, no object must be present within the CR, meaning that the point cloud captured can be considered as the ground truth background (also called golden background). This golden background point cloud could be updated periodically to accurately represent the fixed scene, e.g., to cover changes in the scene due to vegetation grown in the surrounding.

Thus, for each point cloud captured and projected during the regular operation, the background subtraction is performed as follows. A Gaussian smoothing of 11 × 11 pixels operation is applied to both images, i.e., the background and

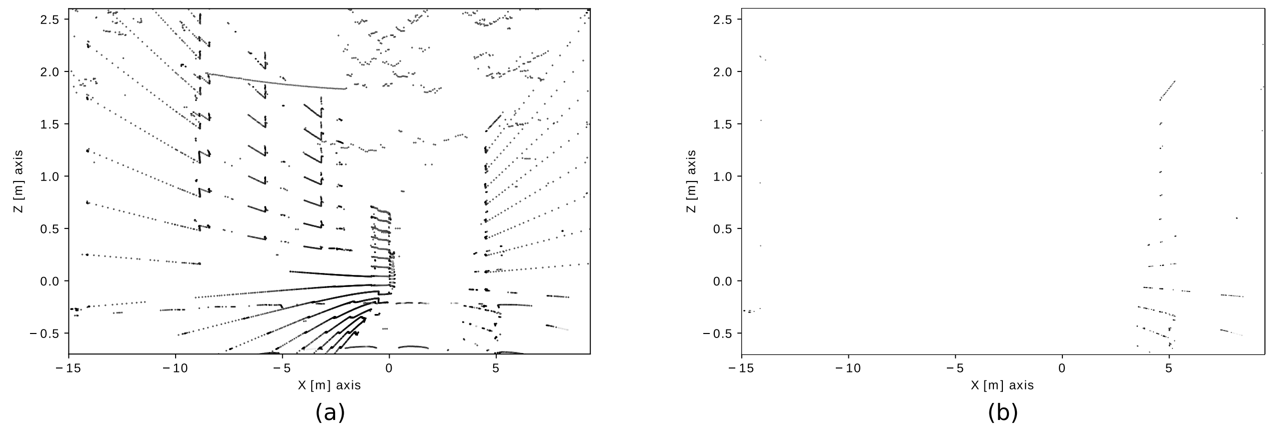


FIGURE 4. Removal of points outside the CR. (a) XZ projection point cloud frame with 14474 points, and (b) XZ projection point cloud frame within of the CR with 508 points.

the recently captured point cloud. The background smooth is performed only once and saved as a golden background point cloud avoiding waste of computer resources. Next, a simple subtraction is performed, where the absolute value of their corresponding pixel differences in the smoothed images is taken, resulting in the removal of the background. Note that the smooth operation acts as a low-pass filter reducing the high-frequency noise. In the point cloud provided by the LIDAR, there is a small scattering of the points between two consecutive frames. The smooth operation helps to reduce this scattering effect.

As a result of the background subtraction process discussed before, a frame is generated, as shown in Fig. 5a. In this figure, the background appears in black, whereas areas with movement appear with some illumination on the grayscale. Next, a threshold operation is applied to reveal frame regions with significant variations in pixel values. To this end, each pixel is compared with a threshold generating a binary figure as the one shown in Fig. 5b. The threshold value requires to be fixed during the deployment stage, depending on the environmental luminosity. This value will vary from 0 (black) to 255 (white).

Once the binary image is generated, a dilatation operation is applied to identify more accurately the contour area of the detected moving areas (see Fig. 5c). The dilation process performs a convolution that uses a filter of a 3×3 pixels size. This process is repeated a number of times equal to the iteration parameter provoking that white regions areas grow in size. The detection of single objects may fail when several objects are very close due to this dilatation operator. However, this situation is not of concern because the capability to detect any object passing through the CR remains, which is the main goal of this system. Then, the contour of each dilated group of points is calculated and evaluated with a minimum contour box limit parameter. If this contour is higher than the limit, it is classified as an object, as shown in Fig. 5d. The units of the contour box limit are pixels^2.

Next, the information provided by the contours is considered to calculate the height, weight, and length in meters of the objects detected (see Fig. 5d). To this end, the Cartesian point with the highest Z value (Z_{high}) located inside the contour rectangle is used along with the point with the lowest Z value (Z_{low}) to calculate the object height value as $height = Z_{high} - Z_{low}$. For the length, it is calculated as $length = Y_{high} - Y_{low}$, where Y_{high} and Y_{low} are the highest and lowest Y values within the contour, respectively. For the width, it is calculated as $width = X_{high} - X_{low}$, where X_{high} and X_{low} are the highest and lowest X values within the contour, respectively.

The height and width metrics obtained are expected to be accurate due to the high resolution provided by the LIDAR. However, it is not possible to accurately calculate the length value of the object because there are points hidden by the object itself. The speed of the object is also calculated by measuring the distance traveled by the centroid point of the contour divided by the time it takes to travel through it. The speed value will be positive when the object moves to the right and negative when moving to the left.

C. OBJECT TRACKING ALGORITHM

Once one or more objects are detected in the scene, it is necessary to track them to estimate the location and speed of each of them. These parameters are required to decide whether the train must be stopped for safety reasons. The tracking process starts by using the contour boxes provided by the object detection algorithm and assigning to them a unique IDentification number (ID) in frame f_t at time t. Next, the centroid for each contour box is calculated taking the high h and width w box values along with the origin box coordinates (X_o, Y_o), which are located in the top left corner of the box, values provided by the contour boxes. Thus, the centroid (X_c, Y_c) is calculated as given by

$$(X_c, Y_c) = (\frac{X_o - h}{2}, \frac{Y_o - w}{2}). \tag{3}$$

In the next frame f_{t+1}, the centroid computation is performed in the same way for each detected object. The object IDs in frame f_{t+1} must be associated with those in frame f_t. To this end, the Euclidean distance between every pair of centroids corresponding to existing objects in both frames f_t

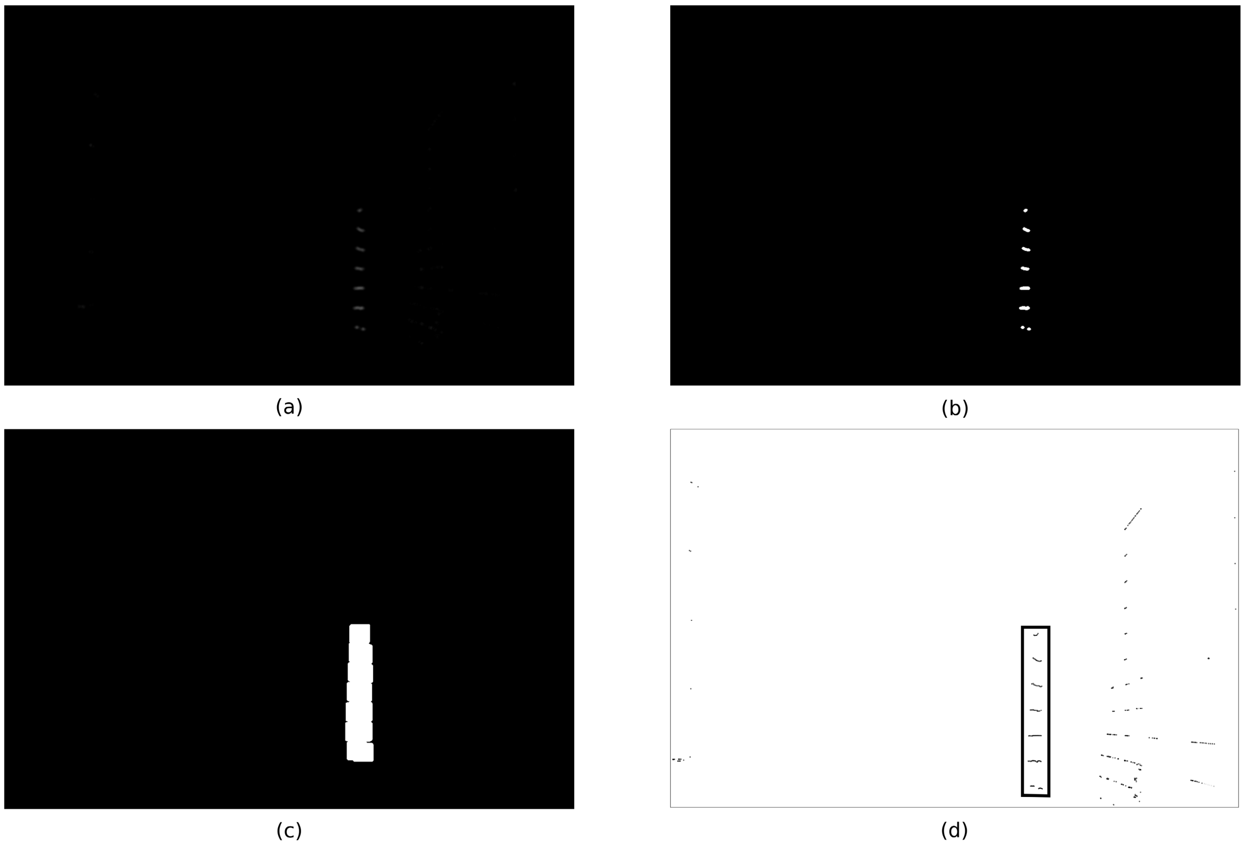


FIGURE 5. Object detection procedure: (a) Image with background subtracted, (b) binary image after applying the threshold operation, (c) binary image dilated, and (d) contour detection for the moving object.

and f_{t+1} is calculated. The tracking algorithm associates the IDs by minimizing the Euclidean distance. The tracking algorithm also considers the appearance of previously undetected objects, as well as the possibility of losing objects (IDs), mainly due to overlapping.

Tracking and detecting objects in overlapping situations are challenging when considering LIDAR sensors. Thus, if two objects are overlapped, the one which is furthest away from the LIDAR will not be detected nor tracked. This fact is a limitation for LIDAR-based systems, which could be addressed by deploying sensors based on other working principles. However, for the specific railway use case in this work, this limitation is not of relevance because it is required to generate an alert when at least one object within the CR is detected. Thus, the functionality is still guaranteed under this limitation.

VI. EXPERIMENTAL METHODOLOGY

In this section, the performance metrics to evaluate the proposal quantitatively are described. Next, the testing setup to perform the experimentation is discussed.

A. OBJECT DETECTION PERFORMANCE METRICS

In this subsection, the usual performance metrics considered to evaluate the proposal [33] is presented. The first group of metrics is as follows:

- True Positive (TP): Number of frames in which all the targets were correctly detected, fitting in with reality.
- True Negative (TN): Number of frames in which no target was detected, fitting in with reality.
- False Positive (FP): Number of frames in which at least a non-existent target was detected.
- False Negative (FN): Number of frames in which at least a target was not detected, but the target exists in reality.

The second group of metrics, based on the first one, includes sensitivity, specificity, precision, and F1 metrics. Sensitivity is the true positive rate and measures the proportion of positive cases correctly identified, calculated as

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \tag{4}$$

Specificity is the true negative rate and measures the proportion of negative cases correctly identified, calculated as

$$\text{Specificity} = \frac{TN}{TN + FN}, \tag{5}$$

Precision is the measure of the correctly identified positive cases from all the predicted positive cases, calculated as

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{6}$$

F1 rate is calculated as the harmonic mean between sensitivity and precision. F1 is an overall measure of the model accuracy that combines the precision and sensitivity ratios in a balanced way, where an F1 rate reaches its best value at 1 and worst at 0. The F1 rate is used when the detection of false negatives and false positives cases requires priority. It is calculated as

$$\mathrm{F1} = 2 * \frac{Pr * Sen}{Pr + Sen}. \tag{7}$$

Among these metrics, the FN rate is especially relevant for the application, meaning that real risk was not detected. A first way to reduce the FN rate is to adequately configure the threshold value used in the background subtraction stage. A low value of the threshold will reduce the FN rate by increasing the probability of detecting a moving element in the background. However, the FP rate will increase if the threshold value is too low, which is also an inadmissible situation. The second way to reduce the FN rate is to pay special attention to the LIDAR positioning and its environment. Thus, the LIDAR should be deployed in an area with the minimum number of obstacles between the LIDAR and the CR because if an object is hidden behind an obstacle, it can become undetectable.

The golden background frame must contain as few points as possible to reduce the FP rate. To this end, the CR must be positioned where the least number of background points fall within the CR. The CR is fixed, as it is calculated according to the location of the railway level crossing. However, it can be slightly tuned, e.g., the CR can be elevated until the points located in the ground are no longer inside the CR. This fact implies an accurate background subtraction stage, which can reduce the FP ratio.

B. TESTING SETUP

The testing methodology consists of a real deployment of the LIDAR-IoT solution presented in this work. The scenario selected was a section of the street in the main entrance of one of the University buildings, where the CEI research center is located, in Madrid. The CR in the scenario is composed of a car road and a pedestrian sidewalk. As discussed before for Fig. 3b, this scenario is similar to a railway cross-level. The size of the CR is 25 m (length) × 11 m (weight) × 4 m (height).

The LIDAR data captured in this real deployment were acquired in the afternoon on a sunny autumn day with few clouds and no rain nor fog. As a result, a total of 23417 frames were collected in 1.3 hours. During the experiment, it can be found from none to several pedestrians and vehicles crossing the CR at the same time. These experimental data will be considered to validate the proposal and the parameter selection performed during the design stage.

VII. EXPERIMENTAL RESULTS

In this section, the results obtained with the proposed object detection and tracking algorithm running on the Cookie edge IoT platform are presented. The first subsection discusses the parameter selection for the algorithm proposed. To this end, the performance metrics are provided for the different configurations evaluated. The second subsection presents results in terms of processing times and power consumption provided by the implemented system. Finally, the results obtained are compared to the state-of-the-art.

TABLE 2. Fine Exploration Parameters Limits.

Parameter	Limits	Interval
Golden background frame	[1-23]	1
Threshold	[5-11]	1
Dilate iterations	[8 - 13]	1
Min. box limit (pixels^2)	[900 - 1400]	100
Point color	[32 - 42]	2
Image size X (pixel)	[150 - 200]	10
Image size Y (pixel)	[150 - 200]	10

A. PARAMETER SELECTION AND PERFORMANCE METRICS

This subsection describes the exploration performed to select the parameters of the proposed implementation. This procedure was performed by evaluating a dataset of 1800 frames because of the large amount of processing time needed to employ the complete dataset, which is composed of 23417 frames. The processing time to perform the exploration with the whole dataset on an Intel i7-8700 processor is about 392 hours. However, the exploration was carried out in 51 hours by using the reduced dataset. The dataset used to calculate the performance metrics was generated by capturing point cloud data with the Velodyne VLP-16 LIDAR in the CR understudy shown in Fig. 3b. A total amount of 23417 frames were captured and then manually labeled with each object crossing the CR. The labeled objects correspond to cars, vans, motorbikes, cyclists, and pedestrians. The performance metrics rates were then calculated using this labeled data.

The parameter selection stage may be divided into two parts. The first one is a coarse exploration with the limitation of the image size, set to 200 × 200 pixels due to the larger size affects the final performance of the system. The coarse exploration starts by varying one parameter with large intervals and fixing the remains. The limits which maximize the F1, sensitivity, and specificity values for each parameter are selected and presented in Table 2. The second stage consists of a fine exploration using the limits calculated in the first stage with the intervals shown in Table 2. It begins with the definition of the golden background frame. Then, three pairs of related parameters are evaluated. These are related to each other, such as threshold with dilate iterations, box limit with point color, and horizontal size with the vertical size of the image.

The fine exploration starts by calculating the golden background point cloud employing the cases with the highest results in terms of the F1 value obtained during the coarse exploration stage. In Table 3 the F1 value of each case by using the first 23 frames of the database as the golden background frame is presented. Best results in terms of the mean value of F1 were obtained by using frames 4 and 5,

TABLE 3. F1 Values of The Golden Background Frame Fine Exploration.

	Golden background frame																						
Case	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	96.92	97.37	97.22	97.32	97.31	96.95	97.14	96.75	97.04	97.20	92.86	97.36	97.18	97.23	96.49	97.41	97.35	95.06	74.22	78.72	88.87	95.78	92.45
2	96.37	97.07	94.41	98.20	97.64	97.80	96.45	91.10	96.75	95.71	96.37	97.07	94.21	98.20	97.64	97.14	96.45	91.10	96.75	96.18	94.87	94.79	94.74
3	96.63	96.30	96.48	98.31	97.09	95.97	95.53	96.60	95.92	96.57	96.18	96.37	96.86	97.40	96.50	96.68	96.66	96.45	48.99	80.04	84.88	95.75	90.63
4	94.51	97.41	97.47	98.09	97.75	87.50	94.83	94.81	94.81	97.07	96.97	97.36	97.64	92.04	97.20	96.41	96.37	97.08	47.04	77.84	81.39	95.12	89.97
5	95.73	97.35	96.04	97.76	98.26	96.58	97.01	98.14	97.48	95.71	95.29	96.16	97.64	94.18	95.04	97.09	94.96	95.97	37.51	61.79	88.20	96.81	89.99
6	95.97	95.69	96.33	98.20	98.14	96.46	94.93	94.75	95.55	96.57	96.56	97.12	97.46	97.08	97.29	96.59	97.01	96.37	57.42	78.54	84.72	97.96	93.17
7	94.52	94.98	95.29	98.43	98.20	93.30	96.05	96.28	96.97	97.07	93.11	96.75	97.19	96.91	93.14	97.27	97.30	96.24	48.14	82.28	86.80	97.34	92.43
8	94.60	96.73	93.99	97.69	98.03	88.22	96.00	92.66	97.02	96.95	95.84	96.80	97.30	97.23	97.30	97.08	97.23	96.79	59.06	73.39	86.32	98.16	92.89
Mean	95.65	96.61	95.90	**98.00**	**97.80**	94.09	95.99	95.13	96.44	96.60	95.39	96.87	96.93	96.28	96.32	96.95	96.66	95.63	58.64	78.59	87.14	96.46	92.04

TABLE 4. Fine Exploration to Select Threshold and Point Dilate Iterations Parameters.

	Dilate iterations																							
	Background = 4												Background = 5											
	8		9		10		11		12		13		8		9		10		11		12		13	
Threshold	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**	**F1**	**Sen**
5	95.45	92.94	92.26	95.73	86.13	97.92	77.70	99.08	70.50	99.16	64.78	99.40	94.89	93.66	90.78	96.18	84.06	98.06	74.85	99.08	67.92	99.16	62.51	99.25
6	94.96	90.71	95.90	94.18	94.32	96.91	89.57	98.56	82.28	98.98	76.01	99.32	94.59	90.45	95.30	93.80	93.31	96.68	89.06	98.34	83.70	98.90	78.11	98.84
7	93.82	88.57	95.88	92.71	96.53	95.07	96.42	97.18	94.14	98.74	88.83	98.79	93.69	88.23	95.75	92.26	96.28	94.49	96.51	96.84	94.18	98.85	91.18	98.82
8	93.05	87.11	95.44	91.48	96.25	93.51	97.40	96.41	97.93	98.31	95.62	98.51	92.41	85.99	95.25	91.03	95.83	92.61	97.67	96.42	**98.14**	98.20	97.25	98.41
9	90.90	83.41	94.63	98.91	95.69	92.05	97.32	95.52	**98.25**	97.53	97.97	98.19	90.36	82.51	94.13	89.01	95.33	91.38	97.31	95.30	**98.42**	97.54	**98.20**	**98.20**
10	89.27	80.72	93.75	88.34	94.65	90.25	97.07	94.74	**98.36**	97.31	**98.13**	97.86	88.72	79.82	93.63	88.12	94.53	90.02	96.90	94.40	**98.24**	97.09	**98.02**	97.63
11	88.10	78.81	92.80	86.66	94.02	89.01	96.54	93.62	97.89	96.19	**98.07**	97.52	88.10	78.81	92.80	86.66	94.02	89.01	96.54	93.62	97.89	96.19	**98.07**	97.52

as shown in Table 3. For this reason, they were selected for the following explorations.

Then, the exploration to define the threshold together with dilate iterations parameters employing the previously selected golden background frames was performed. Table 4 shows the results in terms of F1 and sensitivity (Sen field) obtained by the exploration. The selection of the optimal parameters is the same in the next three explorations. It starts by selecting the results of the F1 performance metric which are greater than 98 (98.5 for the last exploration), as F1 represents a good relationship between FP and FN. Among the selected values, the one with the highest sensitivity is selected, aiming to minimize the FN value whereas keeping a high value of F1. The parameters values used in the following explorations, which are the best according to the terms described above, are a threshold value of 9 with a dilate iterations value of 13 and employed as a golden background frame the number 5, as shown in Table 4.

Then, the pixel color parameter, along with the minimum contour box limit selection, is performed using the limits shown in Table 2. The exploration shown in Table 5 demonstrates that the optimal solution employed as a golden background frame number 5 containing a pixel color value of 34 with a contour box limit of 1100 pixels^2, values used in the following explorations. Finally, the last exploration to select the size of the image with the intervals presented in Table 2 is performed. The image optimal size is 160×160 pixels employed as a golden background frame the number 5, as shown in Table 6. The fine exploration ends with the performance metrics presented in Table 7. With the parameters selected above, the proposed methodology was evaluated over 23417 frames, as shown in Table 7, under the complete evaluation column. In the results presented, it can be seen that the sensitivity value of 99.16% is the highest, which implies the minimum number of FN as it is the most critical rate in the proposed use case. A reduced rate of FN must be matched by a high value of F1, since it correlates the TP, FP, and FN performance metrics. F1 value shown in Table 7 is 98.93%. Finally, the value of the specificity, which is 98.85% has to be considered because it relates the TN with FP rates.

Note that the above parameters should be recalculated for each deployment of the LIDAR-based solution proposed. Part of the coarse exploration and the complete fine exploration stage must be recalculated to obtain the optimal results in terms of performance metrics. The limits provided by the coarse exploration shown in Table 2 are fixed, except for the dilate iterations and the minimum box limit parameters, which have to be recalculated manually according to the distance from the LIDAR to the CR. However, fine exploration is a fully automated process. The system calibration stage with the calculation of the parameters is a critical step due to the severe consequences produced if an object passing through a train level crossing is not detected.

B. PROCESSING TIME AND POWER CONSUMPTION

The LIDAR configuration parameters such as rotation speed and field of view are related to the algorithm processing time, as shown in Fig. 6. Additionally, the rotation speed is related to the point cloud density. With a rotation speed of 600 rpm, half of the points are obtained than at 300 rpm for the same frame. Applying the lowest rotation speed, the highest point cloud density provided by the LIDAR is obtained. The higher the point density of the frame, the better the detection results in terms of performance metrics. Fine exploration and complete evaluation were performed with a rotation speed of 300 rpm, which is the lowest rotation speed of the LIDAR, and a field of view of 180°, due to it is enough to cover the entire CR according to the location

TABLE 5. Fine Exploration to Select Point Color and Minimum Box Limit Parameters.

	Minimum box limit (pixels^2)																							
	Background = 4												Background = 5											
	900		1000		1100		1200		1300		1400		900		1000		1100		1200		1300		1400	
Point color	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen
32	93.87	99.15	96.45	98.85	97.97	98.19	**98.31**	97.76	97.96	97.09	97.15	95.52	94.04	99.15	96.61	98.85	**98.14**	98.20	**98.53**	97.76	**98.19**	97.09	97.32	95.41
34	93.88	99.15	96.29	98.86	97.86	98.53	**98.48**	98.20	**98.14**	97.53	97.32	95.85	93.33	99.15	96.44	98.96	**98.14**	**98.42**	**98.65**	98.09	**98.30**	97.42	97.67	96.08
36	93.26	99.27	95.51	98.95	97.25	98.52	**98.04**	98.31	97.80	97.53	97.11	96.07	93.14	99.27	95.66	98.95	97.81	98.42	**98.48**	98.32	**98.03**	97.42	97.39	96.08
38	91.77	99.26	94.38	98.95	96.05	98.52	97.70	98.31	97.69	97.64	97.00	96.18	92.60	99.27	95.49	98.95	97.81	98.42	**98.37**	98.32	**98.03**	97.53	97.50	96.19
40	91.69	99.26	94.81	99.07	96.70	98.63	97.82	98.53	97.75	97.64	97.17	96.41	90.96	99.26	94.77	98.94	97.13	98.29	97.82	98.31	97.8	97.53	97.33	96.40
42	91.20	99.26	93.93	99.06	95.56	98.74	96.82	98.75	97.08	98.07	97.23	96.63	90.45	99.26	93.78	99.05	96.25	98.51	96.93	98.52	97.14	97.85	97.40	96.41

TABLE 6. Fine Exploration to Select Horizontal and Vertical Image Size Parameters.

	Vertical size (pixel)																							
	Background = 4												Background = 5											
	150		160		170		180		190		200		150		160		170		180		190		200	
H. size (pixel)	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen	F1	Sen
150	97.76	97.87	98.26	97.98	98.15	98.20	98.14	97.87	98.20	97.87	97.43	97.98	98.43	98.09	**98.54**	98.10	98.09	98.32	98.20	98.09	98.43	98.09	98.43	98.31
160	98.43	98.54	97.99	98.65	97.82	98.54	98.04	98.54	**98.65**	98.43	98.43	98.65	98.20	98.20	**98.65**	**98.65**	98.48	98.65	97.98	98.42	98.03	98.08	98.31	98.53
170	98.14	98.42	97.81	98.31	98.25	98.53	95.97	98.06	96.27	97.96	96.36	98.40	97.92	98.31	98.37	98.54	98.31	98.53	98.03	98.31	97.92	98.19	98.09	98.76
180	98.49	98.54	**98.60**	98.54	98.05	98.76	97.44	98.42	97.88	98.32	**98.54**	98.43	97.15	98.19	98.03	98.42	98.20	98.53	**98.54**	98.32	98.31	98.20	98.03	98.42
190	98.43	98.54	98.31	98.65	97.70	98.53	98.20	98.43	98.37	98.20	98.26	98.54	96.93	98.41	97.82	98.64	97.42	98.53	97.64	98.31	96.97	98.07	97.76	98.42
200	97.86	98.53	98.37	98.65	98.19	98.42	96.26	98.40	98.09	98.31	95.88	98.40	98.14	98.42	98.20	98.53	97.97	98.30	97.81	98.42	97.75	98.19	97.65	98.42

TABLE 7. Performance Metrics.

	Fine exploration	Complete evaluation
Evaluated frames	1800	23417
TP	879	13405
TN	896	9722
FP	12	177
FN	12	113
Sensitivity	98.65%	99.16%
Specificity	98.68%	98.85%
Precision	98.68%	98.70%
F1	98.65%	98.93%

of the LIDAR. All configurations shown in Fig. 6, meet the real-time specifications, which are to process a complete frame in less than 0.5 s.

The power consumption of the Cookie edge IoT platform executing the object detection and tracking algorithm at maximum performance and sending the information through the USB-WiFi adapter via the Internet is 1.33 W. The power consumption of the LIDAR is 8 W. However, the system goes into standby mode when there are no trains in the level crossing proximity. In standby mode, the LIDAR remains disconnected, and the power consumption of the Cookie edge IoT platform is 0.29 W.

C. COMPARISON TO THE STATE-OF-THE-ART

In recent literature, there is no common to find object detection works using LIDARs to compare it with, as the current trend is to perform detection along with classification. To provide a system with classification capabilities, the use of neural networks is widespread [21], [23]–[25], [34]. However, providing a system with classification capabilities requires a processing platform with high computing resources, generally using GPUs [21], [35], which are highly power-consuming. The proposed system has to work with reduced power consumption, and GPUs are not acceptable.

Nowadays, databases with high-definition 3D-LIDARs, which usually have 64 channels are widely used

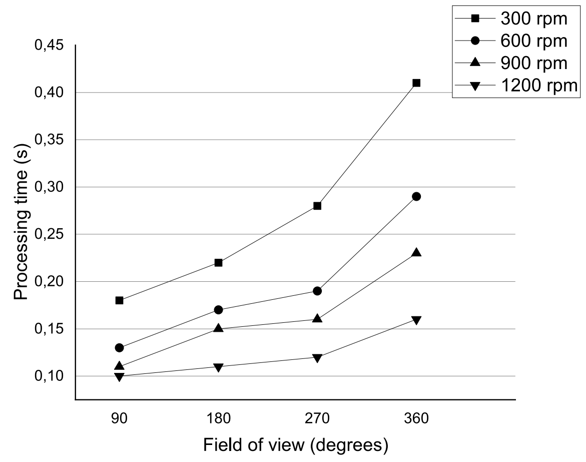


FIGURE 6. LIDAR configurations and processing time.

[21], [23], [26]. However, the LIDAR used in this work is low-resolution containing 16 channels. Low-resolution LIDARs are not commonly used in object detection applications because it is more challenging to detect objects, this implies that there are not many works to compare with.

The authors in [36], use their database generated with 64-channels LIDAR to perform object detection and classification between 5 classes (pedestrian, vehicle, street clutter, facade, and ground). The best results for detecting and classifying vehicle class were obtained, providing values of both F1 and sensitivity of 99%. In this work, similar values using a low-resolution LIDAR are obtained.

In the state-of-the-art, there is, on the one side, a trend that uses lightweight object detection and tracking algorithms with 2D sensors such as RGB cameras [29], [35]. On the other hand, some works use high-resolution 3D sensors to feed DL algorithms, which demand a great amount of computing resources [21], [24], [25]. However, there are not works that use high-resolution 3D sensors to execute lightweight object detection and tracking algorithms in low-power consumption edge IoT node to compare with. The performance metrics values provided in this work will be used in future comparisons.

VIII. CONCLUSION AND FUTURE WORK

In this work, real-time and accurate object detection and tracking implementation using raw 3D point cloud data provided by a low-resolution LIDAR is presented. High accuracy in the detected object spatial location is achieved by using this complex sensor. Besides, other improvements are obtained compared to object detection methods based on RGB cameras, since LIDARs work fine in the absence of light and are more robust in adverse weather conditions.

The parameters selection used in the algorithm was carried out with design space explorations selecting the parameters that maximize F1 and sensitivity values. Applying the parameters provided by the exploration, the proposed implementation achieves a sensitivity value of 99.16%, which means that the system is robust against FN prediction. FN is an essential rate in object detection tasks, particularly in railway level crossing applications. The F1 and specificity values provided are 98.93% and 98.85%, respectively, which implies a reduced value of FN and FP together with a high value of TP and TN.

The lightweight object detection and tracking algorithm is implemented in a low computing resources and low-power consumption edge IoT node meeting the real-time specifications. When running the object detection algorithm, the edge IoT node power consumption is 1.33 W, along with the 8 W power consumption of the LIDAR. However, most of the time, the edge IoT node remains in standby mode with the LIDAR disconnected consuming 0.29 W. The processing time depends on the LIDAR rotation speed, together with the field of view. The explorations were carried out with a field of view of 180°, as this is enough to cover the entire CR according to the location of the LIDAR. The rotation speed was 300 rpm, to obtain the maximum points density. With this LIDAR configuration, the algorithm processes a full-frame in 0.23 s. The results shown in this work prove that by employing this implementation, it is possible to achieve an increase in the railway level crossing security with a reduction in accidents and damages taking advantage of the massive amount of information provided by this type of complex sensors.

Extracting as much spatial information as possible from the LIDAR by performing 3D object detection, and using this information to feed an object classification algorithm whereas keeping power consumption low and meeting real-time specifications is one of the future line of work. Improve the accuracy of both object detection and classification tasks with information coming from several sensors of different types is another future lines of work. There is a current trend to perform sensor fusion using LIDAR, radar, and RGB cameras, as the weak features of each one are covered by the others.

ACKNOWLEDGMENT

The Joint undertaking receives support from the European Union's Horizon 2020 research and innovation programme and several countries such as Austria, Spain, Finland, Ireland, Sweden, Germany, Poland, Portugal, Netherlands, Belgium, and Norway.

REFERENCES

[1] European Union Agency for Railways. (Jun. 2018). *Report on Railway Safety and Interoperability in the EU*. [Online]. Available: https://www.era.europa.eu/library/corporate-publications_en

[2] G. S. Larue, R. A. Blackman, and J. Freeman, "Frustration at congested railway level crossings: How long before extended closures result in risky behaviours?" *Appl. Ergonom.*, vol. 82, Jan. 2020, Art. no. 102943. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003687018303776

[3] G. Larue, C. Watling, A. Black, J. Wood, and M. Khakzar, "Pedestrian distraction at railway level crossings: Can illuminated in-ground LEDs attract their attention back?" *World Congr. Railway Res.*, Dec. 2019. [Online]. Available: https://eprints.qut.edu.au/135066/

[4] M. M. Rahman, Y. Tan, J. Xue, and K. Lu, "Recent advances in 3D object detection in the era of deep neural networks: A survey," *IEEE Trans. Image Process.*, vol. 29, pp. 2947–2962, Nov. 2019.

[5] H. Cho, Y.-W. Seo, B. V. K. V. Kumar, and R. R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 1836–1843.

[6] R. O. Chavez-Garcia and O. Aycard, "Multiple sensor fusion and classification for moving object detection and tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 525–534, Feb. 2016.

[7] I. Yaqoob, L. U. Khan, S. M. A. Kazmi, M. Imran, N. Guizani, and C. S. Hong, "Autonomous driving cars in smart cities: Recent advances, requirements, and challenges," *IEEE Netw.*, vol. 34, no. 1, pp. 174–181, Jan. 2020.

[8] F. Karray, M. W. Jmal, A. Garcia-Ortiz, M. Abid, and A. M. Obeid, "A comprehensive survey on wireless sensor node hardware platforms," *Comput. Netw.*, vol. 144, pp. 89–110, Oct. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128618302202

[9] E. Ahmed and M. H. Rehmani, "Mobile edge computing: Opportunities, solutions, and challenges," *Future Gener. Comput. Syst.*, vol. 70, pp. 59–63, May 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X16303260

[10] R. Roman, J. Lopez, and M. Mambo "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," *Future Gener. Comput. Syst.*, vol. 78, pp. 680–698, Jan. 2018, doi: 10.1016/j.future.2016.11.009.

[11] P. G. López, A. Montresor, D. H. J. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. P. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *Comput. Commun. Rev.*, vol. 45, pp. 37–42, Sep. 2015.

[12] C. Tastimur, M. A. Karakose, and E. Akin, "A vision based condition monitoring approach for rail switch and level crossing using hierarchical SVM in railways," *Int. J. Appl. Math. Electron. Comput.*, vol. 2, pp. 319–325, Sep. 2016.

[13] R. Manikandan, M. Balasubramanian, and S. Palanivel, "Vision based obstacle detection on railway track," *Int. J. Pure Appl. Math.*, vol. 116, no. 24, pp. 567–576, 2017.

[14] C. Tastimur, M. Karakose, and E. Akän, "Image processing based level crossing detection and foreign objects recognition approach in railways," *Int. J. Appl. Math., Electron. Comput.*, vol. 1, pp. 19–23, Aug. 2017.

[15] M. Pavlovic, I. Ciric, D. Ristic-Durrant, V. Nikolic, M. Simonovic, M. Ciric, and M. Banic, "Advanced thermal camera based system for object detection on rail tracks," *Thermal Sci.*, vol. 22, no. 5, pp. 1551–1561, 2018.

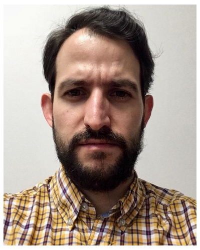
[16] V. P. Milan Pavlovic and T. Nenad Pavlovic, "Methods for detection of obstacles on railway level crossings," in *Proc. Int. Sci.-Expert Conf. Railways*, Oct. 2016, pp. 1–4.

[17] M. Bijelic, T. Gruber, and W. Ritter, "A benchmark for lidar sensors in fog: Is detection breaking down?" in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 760–767.

[18] A. Filgueira, H. González-Jorge, S. Lagüela, L. Díaz-Vilariño, and P. Arias, "Quantifying the influence of rain in LiDAR performance," *Measurement*, vol. 95, pp. 143–148, Jan. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0263224116305577
[19] V. Amaral, F. Marques, A. Lourenço, J. Barata, and P. Santana, "Laser-based obstacle detection at railway level crossings," *J. Sensors*, vol. 2016, pp. 1–11, 2016.
[20] J. Hecht, "Lidar for self-driving cars," *Opt. Photon. News*, vol. 29, no. 1, pp. 26–33, 2018. [Online]. Available: http://www.osa-opn.org/abstract.cfm?URI=opn-29-1-26
[21] K. Burnett, S. Samavi, S. Waslander, T. Barfoot, and A. Schoellig, "AUToTrack: A lightweight object detection and tracking system for the SAE autodrive challenge," in *Proc. 16th Conf. Comput. Robot Vis.*, May 2019, pp. 209–216.
[22] J.-R. Xue, D. Wang, S.-Y. Du, D.-X. Cui, Y. Huang, and N.-N. Zheng, "A vision-centered multi-sensor fusing approach to self-localization and obstacle perception for robotic cars," *Frontiers Inf. Technol. Electron. Eng.*, vol. 18, no. 1, pp. 122–138, Jan. 2017.
[23] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3D lidar using fully convolutional network," 2016, *arXiv:1608.07916*. [Online]. Available: http://arxiv.org/abs/1608.07916
[24] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.
[25] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 922–928.
[26] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7652–7660.
[27] J. Han, D. Zhang, G. Cheng, N. Liu, and D. Xu, "Advanced deep-learning techniques for salient and category-specific object detection: A survey," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 84–100, Jan. 2018.
[28] H. Lee, H. Kim, and J.-I. Kim, "Background subtraction using background sets with Image- and color-space reduction," *IEEE Trans. Multimedia*, vol. 18, no. 10, pp. 2093–2103, Oct. 2016.
[29] S. Ojha and S. Sakhare, "Image processing techniques for object tracking in video surveillance- a survey," in *Proc. Int. Conf. Pervas. Comput. (ICPC)*, Jan. 2015, pp. 1–6.
[30] J. Portilla, A. de Castro, E. de la Torre, and T. Riesgo, "A modular architecture for nodes in wireless sensor networks," *J. Universal Comput. Sci.*, vol. 12, no. 3, pp. 328–339, 2006.
[31] P. Merino, G. Mujica, J. Señor, and J. Portilla, "A modular IoT hardware platform for distributed and secured extreme edge computing," *Electronics*, vol. 9, no. 3, p. 538, Mar. 2020.
[32] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to OpenCV," in *Proc. 35th Int. Conv. MIPRO*, May 2012, pp. 1725–1730.
[33] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection. net: A new change detection benchmark dataset," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2012, pp. 1–8.
[34] P. N. Druzhkov and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognit. Image Anal.*, vol. 26, no. 1, pp. 9–15, Jan. 2016, doi: 10.1134/S1054661816010065.
[35] M. Braham, S. Pierard, and M. Van Droogenbroeck, "Semantic background subtraction," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4552–4556.
[36] A. Borcs, B. Nagy, and C. Benedek, "Instant object detection in lidar point clouds," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 7, pp. 992–996, Jul. 2017.

CRISTIAN WISULTSCHEW received the B.S. and M.Sc. degrees in industrial electronics from the Universidad Politécnica de Madrid, Madrid, Spain, in 2017 and 2018, respectively, where he is currently pursuing the Ph.D. degree. He carries out his research activity within the Centro de Electrónica Industrial, UPM. He is participating in an H2020 project, SCOTT, related to the object detection and tracking system used in railway level crossing surveillance systems. He is also participating in a national research project, PLATINO, related to machine learning techniques applied to embedded systems using specific DL HW accelerators applied to the agro-food industry. His research interests include sensor system integration, digital embedded systems, embedded machine learning, deep learning HW accelerators, and the Internet of Things.

GABRIEL MUJICA (Member, IEEE) received the Ph.D. degree in industrial electronics engineering from the Universidad Politécnica de Madrid. His visiting research stay at Trinity College Dublin strengthened the vision and applicability of IoT technologies for smart and sustainable cities and leveraging collaborations in the area of distributed systems within such contexts. He is currently an Assistant Professor and also a Research Member with the Center of Industrial Electronics, Universidad Politécnica de Madrid, where he is also involved in the area of networked embedded systems and wireless sensor networks (WSN). He has participated in different national and European research projects (including Horizon 2020 projects), related to the development and optimization of WSN, and the integration of heterogeneous hardware, software, and communication technologies for wireless distributed systems, with a particular focus on the performance evaluation of sensor platforms under real deployment and commissioning. In this way, he has authored several contributions in high-impact conferences and journals. He has collaborated in the organization of research tutorials and seminars, and as a reviewer for several international conferences and journals (IEEE and Springer). His current research interests include multi-hop distributed networks, and hardware-software co-design and protocols for embedded systems in smart urban application scenarios.



JOSE MANUEL LANZA-GUTIERREZ received the B.S. and M.S. degrees in computer science, the master's degree in grid computing and parallelism, and the Ph.D. degree, under the guidance of Prof. Dr. Juan A. Gomez-Pulido, in computer science from the University of Extremadura, Caceres, Spain, in 2008, 2009, 2010, and 2015, respectively.

He is currently an Assistant Professor with the Universidad de Alcalá, Alcalá de Henares, Madrid, Spain. He has authored or coauthored more than 40 publications, including Journal Citation Report (JCR) articles in journals, such as *Applied Soft Computing*, *Expert Systems with Application*, *BMC Bioinformatics*, *Soft Computing*, *Reliability Engineering and System Safety*, and the *Journal of Heuristics*. His current research interests include metaheuristics, wireless sensor networks, digital embedded systems, and machine learning.

JORGE PORTILLA (Senior Member, IEEE) received the M.Sc. degree in physics from the Universidad Complutense de Madrid, Madrid, Spain, in 2003, and the Ph.D. degree in electronic engineering from the Universidad Politécnica de Madrid (UPM), Madrid, in 2010.

In 2008, he was a Visiting Researcher with the Industrial Technology Research Institute, Hsinchu, Taiwan, and also with the National Taipei University of Technology (Taipei Tech), Taipei, Taiwan, in 2018, where he was working on wireless sensor networks hardware platforms and network clustering techniques. He is currently a tenured Assistant Professor with the Universidad Politécnica de Madrid. He carries out his research activity within the Centro de Electrónica Industrial, UPM. He has participated in more than 30 funded research projects, including European Union FP7 and H2020 projects, and Spain Government funded projects, and private industry funded projects, mainly related to wireless sensor networks and Internet of Things. He has numerous publications in prestigious international conferences and in journals with impact factor. His research interests include wireless sensor networks, Internet of Things, digital embedded systems, and reconfigurable FPGA-based embedded systems.