


Article

Pedestrian Detection with LiDAR Technology in Smart-City Deployments—Challenges and Recommendations

Pedro Torres ^{1,2,*} , Hugo Marques ^{1,*}  and Paulo Marques ^{1,3}¹ Instituto Politécnico de Castelo Branco, Av. Pedro Álvares Cabral, n°12, 6000-084 Castelo Branco, Portugal² Research Center for Systems and Technologies (SYSTEC)—Advanced Production and Intelligent Systems Associated Laboratory (ARISE), FEUP, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal³ Allbesmart LDA, Avenida do Empresário, Centro de Empresas Inovadoras 1, 6000-767 Castelo Branco, Portugal

* Correspondence: pedrotorres@ipcb.pt (P.T.); hugo@ipcb.pt (H.M.)

Abstract: This paper describes a real case implementation of an automatic pedestrian-detection solution, implemented in the city of Aveiro, Portugal, using affordable LiDAR technology and open, publicly available, pedestrian-detection frameworks based on machine-learning algorithms. The presented solution makes it possible to anonymously identify pedestrians, and extract associated information such as position, walking velocity and direction in certain areas of interest such as pedestrian crossings or other points of interest in a smart-city context. All data computation (3D point-cloud processing) is performed at edge nodes, consisting of NVIDIA Jetson Nano and Xavier platforms, which ingest 3D point clouds from Velodyne VLP-16 LiDARs. High-performance real-time computation is possible at these edge nodes through CUDA-enabled GPU-accelerated computations. The MQTT protocol is used to interconnect publishers (edge nodes) with consumers (the smart-city platform). The results show that using currently affordable LiDAR sensors in a smart-city context, despite the advertising characteristics referring to having a range of up to 100 m, presents great challenges for the automatic detection of objects at these distances. The authors were able to efficiently detect pedestrians up to 15 m away, depending on the sensor height and tilt. Based on the implementation challenges, the authors present usage recommendations to get the most out of the used technologies.

Keywords: pedestrian detection; LiDAR; 3D point clouds; ROS; smart cities; traffic mobility



Citation: Torres, P.; Marques, H.; Marques, P. Pedestrian Detection with LiDAR Technology in Smart-City Deployments—Challenges and Recommendations. *Computers* **2023**, *12*, 65. <https://doi.org/10.3390/computers12030065>

Academic Editor: Paolo Bellavista

Received: 25 January 2023

Revised: 14 March 2023

Accepted: 16 March 2023

Published: 17 March 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Street life and pedestrian activities are of major interest for smart-city planning. Having up-to-date data about pedestrian mobility in cities has traditionally been difficult. City planners are open to incorporating automatic perception systems to better understand human and traffic mobility within the city. Pedestrian detection has been historically obtained through video-processing algorithms that process footage captured by video cameras; however, low-light conditions and privacy concerns are known limitations of this technology.

Current state-of-the-art solutions for pedestrian detection and behavior make use of data acquired by LiDAR technology, which does not have the same above-mentioned limitations. The cost of this technology has dropped and is currently regarded as a key sensing element of Intelligent Transport System (ITS) infrastructures. LiDAR sensors transmit multiple laser beams (each in a different vertical angle), sweeping a particular horizontal field of view. The amount and granularity of the sweep is specific to each LiDAR sensor. When one laser beam hits an object and is reflected back to the LiDAR sensor, based on how long it takes the light to return to the sensor, it is possible to measure the distance to a reflected object. By acquiring reflections in different horizontal and vertical angles, it is possible to compute a three-dimensional scan of an observed area. These scans

are commonly referred as “point clouds” and, with proper object recognition algorithms and sufficient training, it is possible to identify and classify different types of objects (pedestrians, bicycles, cars, etc.), depending on their distance and pose with respect to the LiDAR sensor.

The major difficulty in dealing with point clouds is the sheer volume of data they contain. For example, an entry-level sensor with 16 laser beams will typically generate between 300,000 and 600,000 points per second (in a 360° sweep) and may generate throughputs close to 40 Mbit/s, depending on its configuration (single/dual return mode). This poses a computational and network transmission challenge that can be minimized when placing a capable computing unit and the LiDAR sensor in close network proximity. This approach limits LiDAR data to traversing a single- or few-layer three-network segments, and follows the multi-access edge-computing paradigm, where large volumes of data are processed at the edge of the network (where the sensors are). Hence, smart-city applications are not required to process these data; they can simply query for a particular data insight already extracted by the edge-computing unit. There are a few features required for a suitable edge-computing unit to process LiDAR data, where high performance per watt, weight and compact form-factor may have different influence in the selection process. They should also be capable of running different applications of deep neural networks (DNNs) trained on high-performance GPUs. At the current time, support for CUDA, the parallel-computing platform and programming model developed by NVIDIA, is also welcome, since object detection and recognition/classification benefits from running in multiple GPU cores in parallel.

The remainder of this paper is organized as follows: Section 2 describes the state of the art and related works. Section 3 describes the material and methods for the system architecture. Section 4 presents the achieved experimental results and, finally, Section 5 presents the conclusions and future work.

2. State of the Art

Object detection, tracking and recognition are key challenges in computer vision for both the industry and academia. These methods have been widely used with 2D images; however, systems based on 2D images cannot provide accurate 3D location information. On the other hand, 3D systems such as RGB-D (RGB-Depth), RGB-LiDAR and LiDAR can only provide solutions that significantly improve RGB-only approaches. In [1], the object classification/detection systems theory based on 2D/3D images are presented as a survey to better understand the key differences in both approaches.

Digitization and the emergence of new types of sensors, computing and communications technologies have boosted the development of projects that make cities smarter and in permanent communication with those who inhabit them. The scientific and commercial advances that have emerged in the area of autonomous driving make 3D object detection an increasingly necessary area and with great evolution potential, specifically due to advances in deep-learning architectures, but also to LiDAR sensors. Although this work focuses on object detection in a smart-city context, the main technological advances in this area are in fact due to autonomous driving. Several research works, focused on the detection of objects based on LiDAR technology, have been written in recent years. In [2], a survey of 3D object-detection methods, based on LiDAR, is presented and the performance achieved by each method is discussed. In [3], the PointRCNN method was proposed for the detection of 3D objects in the raw point cloud without the need for RGB images. The method works in two steps, first segmenting the point-cloud scene into foreground and background points, and then refining the proposal into the canonical coordinates to obtain the final detection results. The authors used the KITTI dataset [4] to validate the architecture, considering only the point cloud as input. The same authors extended PointRCNN to a new point cloud-based 3D object-detection framework, later described in [5]. The performance improvements of each component of the initial proposal were demonstrated, and a new state of the art was achieved in the KITTI 3D object-detection dataset using only the LiDAR point-cloud data.

LiDAR has proved to be an extremely useful sensor in 3D object-detection scenarios; however, due to the variation of the density of the points along the distance caused by the angular displacements between the LiDAR lasers, it has an inherent limitation. Thus, objects located at greater distances return fewer points than objects located closer to LiDAR sensor. The density of acquired points is directly related to the successful detection of an object. Smaller objects, or objects farther away from the sensor have less surface area to cross the LiDAR laser beams, which affects detection efficiency. To mitigate these problems, the authors in [6] propose the voxel network with the point-density recognition (PDS) method, taking advantage of the location of the voxel point centroid and feature encodings that directly account for the density of points in the detection of multiclass 3D objects. This limitation is also addressed in [7], where the authors propose a method that fuses point clouds and RGB images to build a 3D object-detection architecture to improve detection accuracy of small objects represented by point clouds. The proposed architecture uses key points to successfully combine point-based, voxel-based and image-based methods. In this way, complementary fusion of the geometric information collected by LiDAR and semantic information collected by the camera is performed.

Deep-learning approaches have also significantly contributed to recent advances in 3D object detection. The authors in [8] perform a review of recent progress in state-of-the-art deep-learning methods for 3D understanding, including 3D shape classification, 3D object detection and tracking, and 3D scene and object segmentation. In [9], a hybrid 2D and 3D Hough net is proposed that combines 3D and 2D local Hough features with a classification deep-learning network for 3D object classification in LiDAR point clouds.

It is also important to note that the characteristics of LiDAR point clouds, such as unstructured distribution, disordered arrangement, and large amounts of data, typically result in high computational complexity and make it very difficult to classify 3D objects. In [10], the authors propose a convolutional neural network (CNN)-based 3D object classification method using the Hough space of LiDAR point clouds to overcome these problems. First, object point clouds are transformed into Hough space using a Hough transform algorithm, and then the Hough space is rasterized into a series of uniformly sized grids. The authors in [11] introduce a common object-detection pipeline and taxonomy to facilitate a thorough comparison between different techniques and, departing from it, critically examine the representation of data, feature extraction and finally object-detection models. A comparison between the performance results of the different models is included, alongside some future research challenges.

In summary, most of the existing 3D object-detection methods based on point-cloud representation are mainly divided into two categories—LiDAR with image fusion and LiDAR-only methods. For LiDAR-only methods, they are divided into voxel-based methods or point-based methods. Both approaches have their advantages and disadvantages; it depends on the type of application. LiDAR provides a more accurate 3D geometry structure, and the camera captures more scene context and semantic information. The fusion of the two types of data can provide more contextualized information, but everything will depend on the processing capacity available and what is required to be analyzed. The authors in [12] present a comprehensive survey of deep learning on LiDAR-only and LiDAR-fusion 3D perception tasks. This paper gives a review according to four key tasks in the field of LiDAR-based perception: object classification, object detection, object tracking, and segmentation.

For 3D object detection based on multiple sensors, LiDAR and cameras, multi-view 3D object-detection network (MV3D) [13], F-PointNet (Frustum PointNet) [14], and MMF [15] stand out, among others. Focusing on LiDAR-only, the methods can be grouped in point-based plus voxel-based, such as PV-RCNN (PointVoxel-RCNN) [16], voxel-based where VoxelNet is included [17], SECOND [18], PointPillars [19], Part-A2 [5] and Point-based, where PointRCNN is included [3], and STD [20], among others. The following section describes the methodology and the methods used in the implementation of a pedestrian-

detection system in a real scenario context, through edge computing, inserted in an architecture of a smart city that includes other sensors and other mechanisms for city monitoring.

3. Materials and Methods

This section describes the methodology followed by the implementation of a pedestrian-detection solution based on point clouds acquired in real time from a LiDAR sensor.

3.1. System Architecture

The high-level system architecture of the deployed system is depicted in Figure 1. Each LiDAR sensor, placed in strategic locations in the city (pre-identified by the municipality as areas of interest, given traffic and pedestrian volume at crosswalks, close to the University of Aveiro), sends point clouds to an edge node, composed of an NVIDIA Jetson Nano or Jetson Xavier. Each edge node runs a mobile edge-computing (MEC) application “Neuron process”, developed over ROS Melodic [21] distribution on Ubuntu 18.04, which is the edge application responsible for pedestrian detection based on the received point clouds. Each edge node also exposes, via the “Neuron API”, the information of each pedestrian detected via Message Queuing Telemetry Transport (MQTT). The real-time data can, therefore, exist on a central database accessed by the city’s monitoring dashboards, available online and in open access.

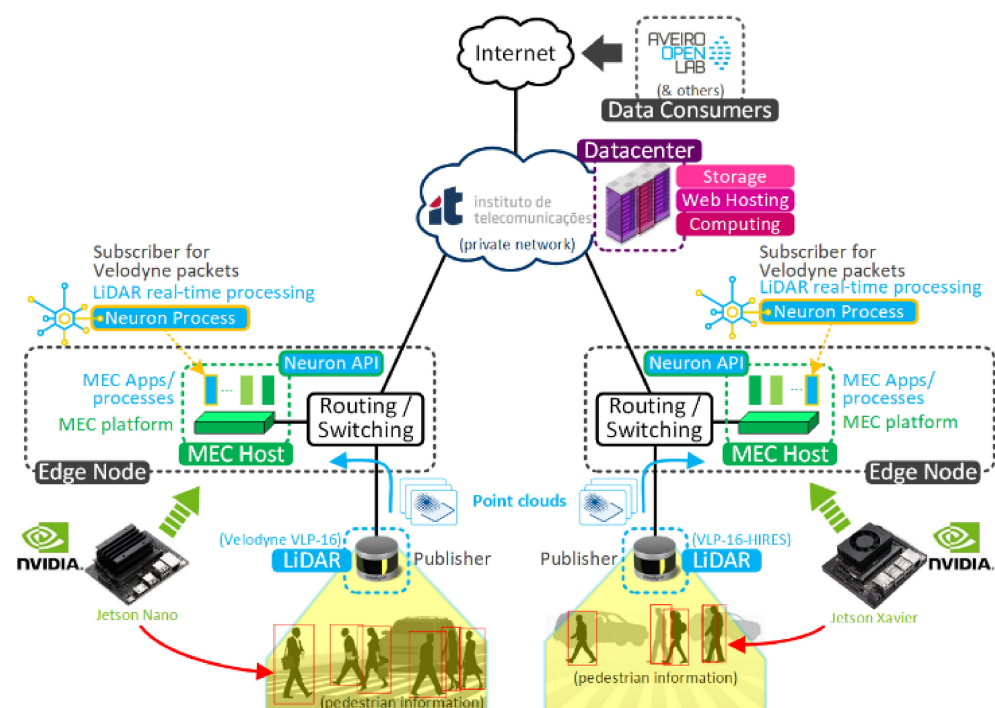


Figure 1. System architecture (high-level).

Briefly, LiDAR sensors capture point clouds and send them to the edge nodes for real-time processing. The edge nodes use a dedicated process (Neuron) to detect pedestrians and retrieve associated information (location, speed, direction, etc.). These results are made available to third-party applications (data consumers) through a dedicated API (the Neuron API). Additional information on the Neuron API is provided in Section 3.5. The Aveiro Tech City Living Lab [22] is the city platform consuming and exposing these results to third-party applications. It is composed of an advanced communications infrastructure and an urban platform for data management and innovative analytics that, together, provide an open and large-scale technology laboratory in the city at the service of researchers, digital industries, start-ups, scaleups, R & D centers and other stakeholders interested in developing, testing or demonstrating concepts, products or services.

3.2. Hardware

Implementation relied on already pre-acquired hardware (LiDAR sensors and edge nodes) by the Aveiro Tech City Living Lab. The reasons for the selected hardware resulted from an equilibrium between hardware capabilities, hardware availability and investment in a pilot project. The LiDAR sensors used were the Velodyne Puck (VLP-16) and Velodyne Puck Hi-Res (VLP-16-HIRES); their main relevant characteristics can be seen in Table 1:

Table 1. Relevant technical specifications for the LiDAR sensors used.

Specifications	Velodyne Puck (VLP-16)	Velodyne Puck Hi-Res (VLP-16-HIRES)
Number of channels		16
Measurement range		100 m
Horizontal FoV		360°
Horizontal angular resolution		0.1°–0.4°
Vertical FoV	30.0° (−15.0° to +15.0°)	20° (−10° to +10°)
Vertical angular resolution	2.0°	1.33°
Rotation rate		300 RPM–1200 RPM
3D LiDAR points generated (per second)		300 K (single return mode) 600 K (dual return mode)

NOTE-1: data extracted from the official Velodyne datasheets (<https://velodynelidar.com/>, accessed on 17 March 2023) NOTE-2: according to Velodyne [23]: “Even if a viewer intentionally stares at a Velodyne sensor, the combination of low power and rapid rotation results in a class I rating. Therefore, the Puck’s roughly 300,000 beam points per second are harmless to the very humans the sensors are designed to protect”.

The edge-computing nodes (MEC Hosts) are the NVIDIA Jetson Nano and Jetson Xavier NX. Their main relevant characteristics can be seen in Table 2:

Table 2. Relevant technical specifications for the edge-computing nodes used.

Specifications	Jetson Nano	Jetson Xavier NX
AI Performance	472 GFLOPs	21 TOPs
GPU	128-core NVIDIA Maxwell™ GPU	384-core NVIDIA Volta™ GPU with 48 Tensor Cores
CPU	4-Core Arm Cortex-A57 MPCore processor	6-core NVIDIA Carmel Arm® v8.2 64-bit CPU (6MB L2 + 4MB L3)
Deep-Learning Accelerator	—	2x NVDLA v1
Programmable Vision Accelerator	—	2x PVA v1
Memory	4 GB 64-bit LPDDR4 (25.6 GB/s)	8 GB 128-bit LPDDR4x (59.7 GB/s)
Companion software	JetPack SDK, which provides Ubuntu 18.04 environment with accelerated graphics support for NVIDIA CUDA Toolkit (v10.0 for Nano and 10.2 for Xavier NX)	

NOTE: Data extracted from the Nvidia (<https://developer.nvidia.com/embedded/jetson-modules>, accessed on 17 March 2023).

The Jetson Xavier NX was a later addition to the setup and was only integrated at the final steps of the pilot.

3.3. Pilot Setup

Edge nodes are housed in a protective shell on top of a “smart pole”, while the LiDAR sensors are laterally attached, at an approximate height of 3m and with a slight tilt in the vertical axis. These smart poles are typically installed close to a sidewalk and road, as depicted by Figure 2.

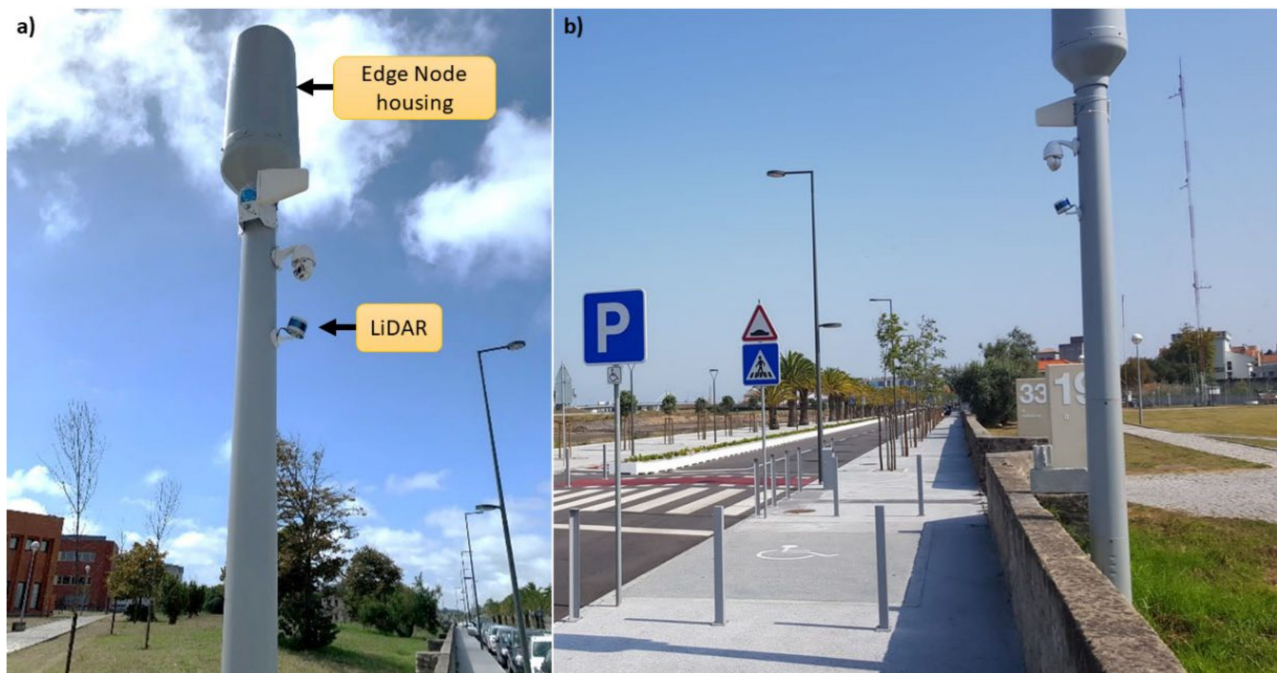


Figure 2. Aveiro Tech City Living Lab smart poles. (a) LiDAR and edge node location in the smart posts. (b) Panoramic view of the road and location of the pole next to a zebra crossing.

The LiDAR sensor parametrization can be performed through their own dedicated web interface, where various parameters can be configured, the most relevant are: laser status (on, off); laser return mode (strongest, last or dual), also known as laser detection of a reflection; motor rotation speed (between 300 and 1200 revolutions per minute (RPM), equivalent to 5 and 20 Hz, respectively), for safety reasons, the laser is automatically turned off if the motor is stopped; the sensor's horizontal field of view (FoV); the host/destination IP address (the receiver of the point clouds), and; the sensor's IP address (static or dynamic, via DHCP). Other parameters are available—for additional information, including programmatic configuration through curl, the sensor datasheet provides further details.

Communication with the edge-computing nodes (MEC hosts) is made through a command-line interface via a protected SSH session. These nodes use Linux Ubuntu 18.04 where the object-detection algorithm (Neuron process) has been deployed and executed. The network topology is represented in Figure 3:

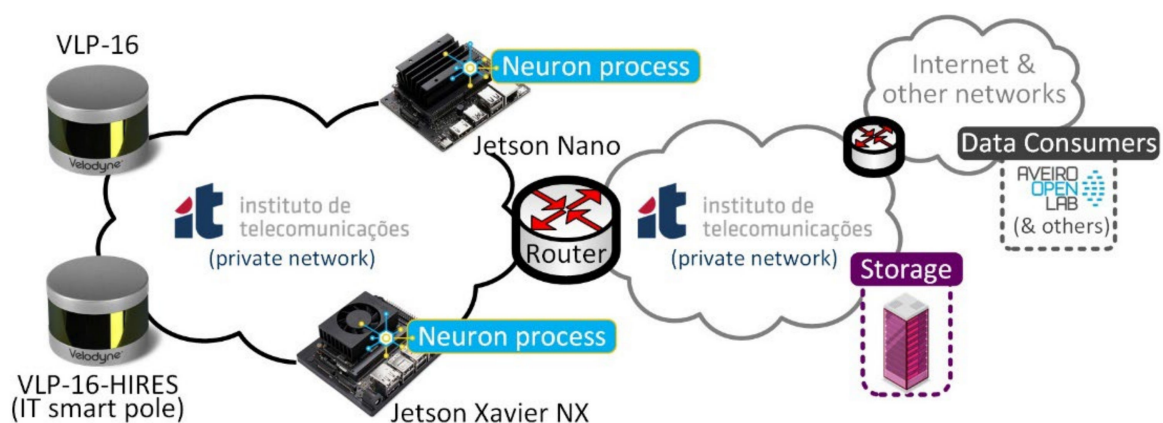


Figure 3. Network topology.

The LiDAR sensors periodically send, through the IP network, point-cloud data to the edge-computing nodes (at approximately 40 Mbps) for processing. These data are processed in quasi-real time by the Neuron process running at the edge-computing nodes. The data-processing results can be remotely accessed by any data consumer through the Neuron API.

When using multiple LiDAR sensors in a network, each sensor's destination IP address should be set to a non-broadcast IP address. Even though two sensors could send to the same destination IP address (reporting to the same MEC host), this is still not the recommended approach, as the ethernet medium contention algorithm will limit transmission cycles from the sensors and may lead to a degradation in performance. The recommended setup is to have each sensor reporting to a different destination IP address (one MEC host per sensor or a single MEC host with two network interface cards).

3.4. Pilot Challenges

Both the VLP-16 and VLP-16-HIRES LiDAR sensors use a mechanical moving mechanism, which may degrade its lifetime. To the best of our knowledge, Velodyne does not publicly advertise the life expectancy of these sensors; nonetheless, it is reasonable to assume these were not manufactured for extensive 24/operation. Therefore, during the pilot setup, the usage of the sensors was deliberately limited to a few hours per day.

These sensors provide somewhat long-range and wide-angle laser scans. Velodyne advertises a range up to 100 m and, while it is true—they can accurately identify the existence of a reflected surface (object) at those distances—a quite different indicator is how much of that information can effectively be used for object classification, since point clouds become considerably sparser as the distance to the sensor increases. This is inherited by the specific sensor's horizontal and vertical angle resolutions (see Table 1). This means that objects closer to the sensor (up to a minimum distance) will produce considerably more laser reflections (3D points in the point cloud) than objects farther away. For stationary instalments (fixed in a single position), this brings an important aspect for sensor placement and data analysis.

For the specific case of this pilot, possible targets for detection were humans and vehicles. Therefore, the sensor should be placed to maximize:

- the number of laser beams that hit a target;
- the distance at which the target can be detected and correctly classified;
- the area of interest for detection and classification.

On the other hand, for security reasons, and given their cost value, LiDAR sensors need to be protected against theft and vandalism. This may lead to non-optimal placement with respect to the above-mentioned detection goals. Figure 4 highlights how these two conflicting goals interact.

Figure 5 depicts a sample point cloud obtained by the LiDAR sensor when applying the configuration depicted in Figure 4d. It can be seen that human detection in 3D point clouds is very challenging, as the number of reflections and their intensity will vary depending on the human pose and how close, or far away, the human is from the sensor.

Figure 6 highlights the problem statement—the clusters of points that can be used to recognize a human (pedestrian) will depend on the human pose and how far, or how close, it is with respect to the LiDAR sensor. To ensure high precision of the cluster detector and considering just one of the setups in Figure 4, this would imply a considerable effort in training, involving a very large dataset of manually annotated data for all possible distances and human poses. Manually labeling 3D LiDAR point clouds is not only inefficient but also very tedious, which leads to human errors due to the sheer number of possible variations of human pose, shape, and size.

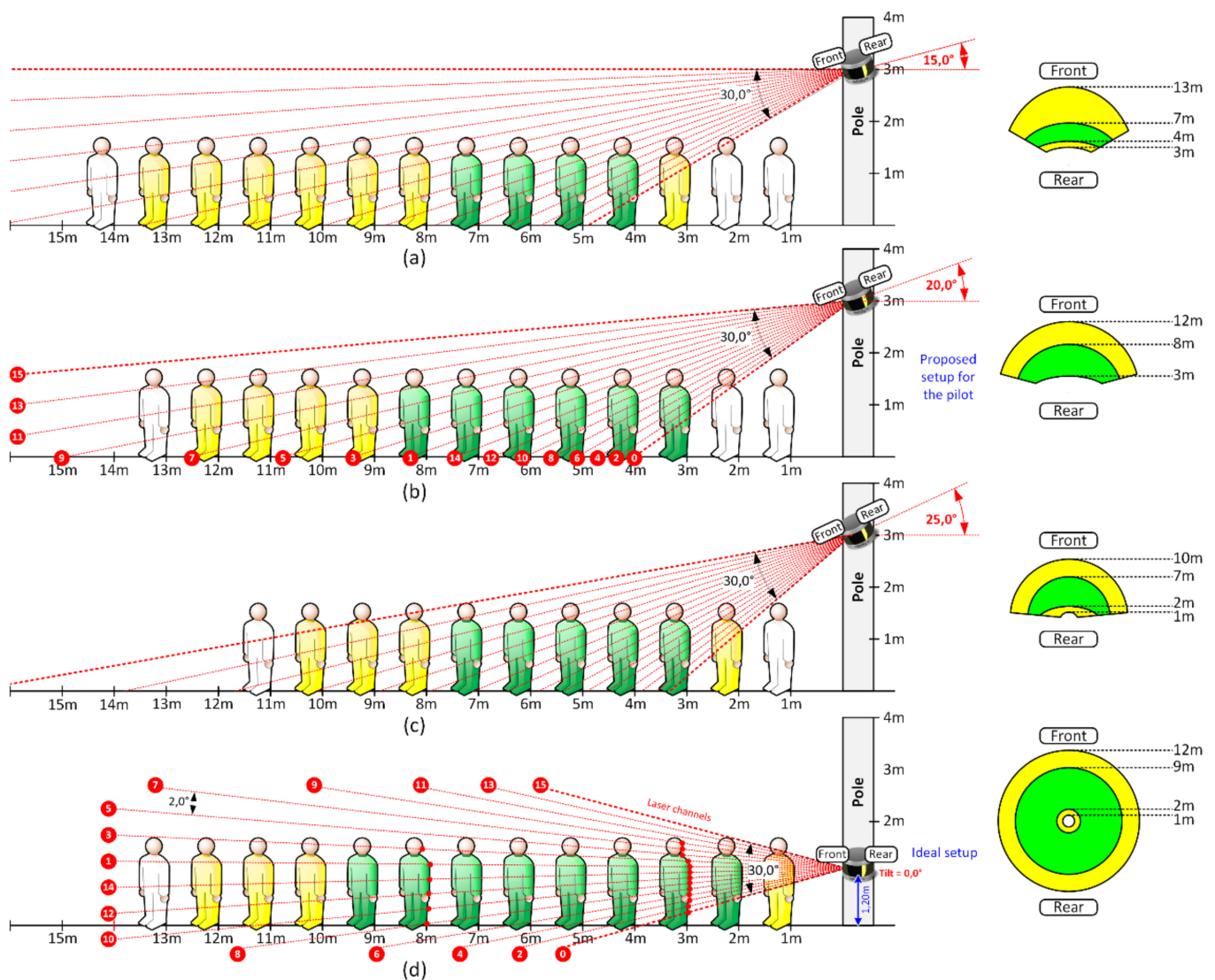


Figure 4. LiDAR sensor position and its impact on human (pedestrian) classification and area of detection. Case study for a 16-channel LiDAR sensor with a vertical resolution of 2.0° . Yellow means the object has a slight chance of being classified, depending on its pose in relation to the sensor; green means the object has a very strong chance to be accurately classified; white means the object does not generate sufficient reflections to be accurately classified by the sensor. (a) LiDAR at 3 meters height with 15° degree tilt. (b) LiDAR at 3 meters height with 20° degree tilt. (c) LiDAR at 3 meters height with 25° degree tilt. (d) LiDAR at 1.2 meters height with 0° degree tilt—the ideal setup.

For this challenge, and considering (i) the decoupling of video footage and the LiDAR 3D point clouds, and (ii) the given time frame for the pilot (approx. 6 months), the work had to rely on already existing training datasets, performed by online methods. Unfortunately, the publicly available datasets mainly considered point clouds obtained with a LiDAR sensor placed at lower heights (0.8–1.20 m). This means the available training datasets are not optimized for a LiDAR sensor placement at the desired height (3 m) and tilt (Figure 4a–c).

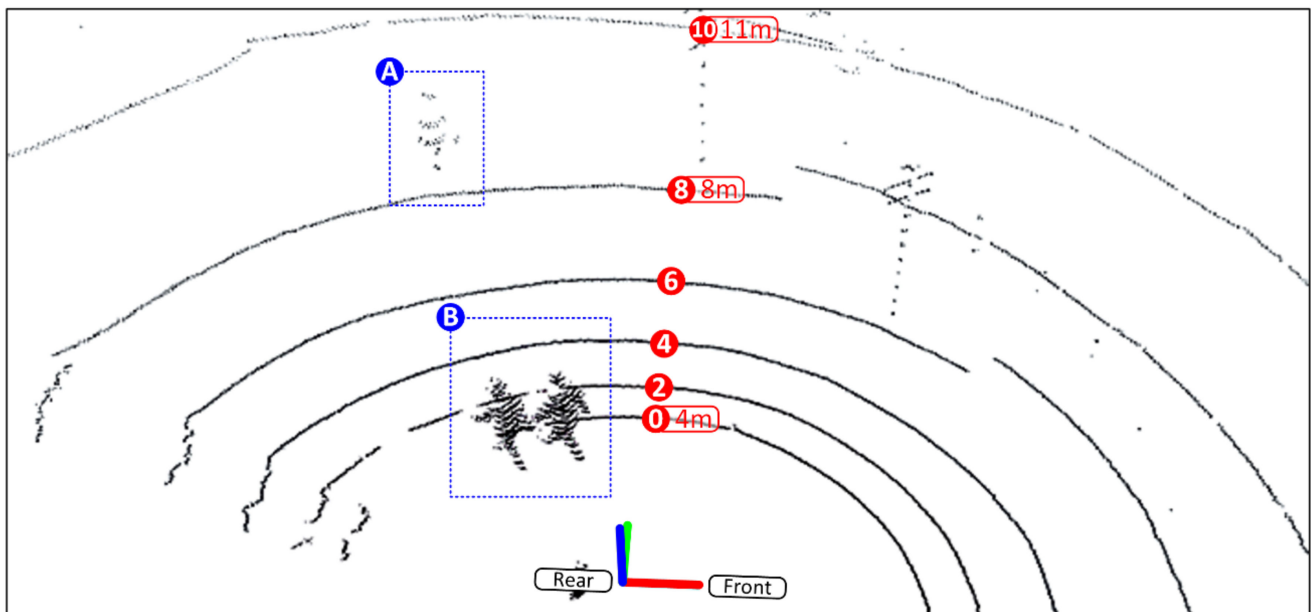


Figure 5. Generated point cloud (VLP-16) for an height of 1.20 m and a tilt of 0° (scenario Figure 4d). (A) A pedestrian just over 8 meters away from the sensor. (B) Two pedestrians within 4 meters of the LiDAR.

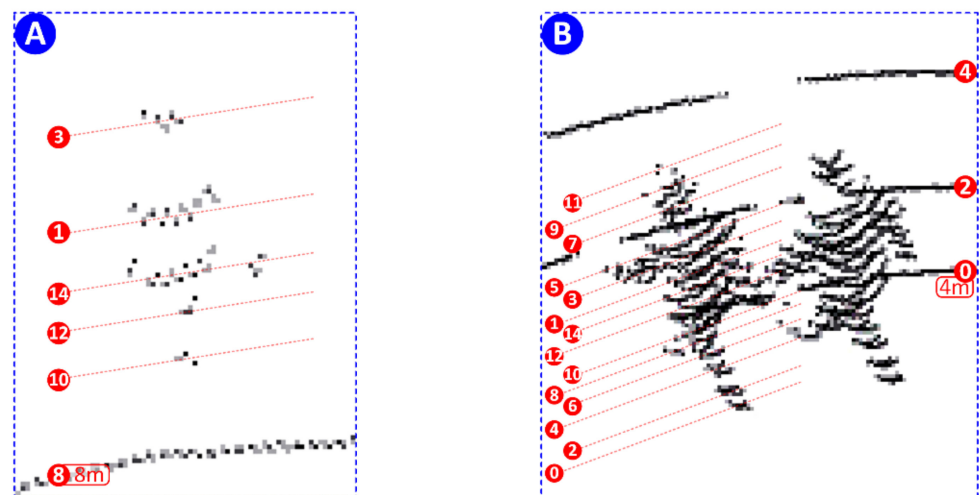


Figure 6. Zoomed-in point-cloud clusters extracted from Figure 5—the closer the human is to the sensor, the denser the cluster of points. The circled numbers reflect the laser channel, as specified by the sensor's datasheet. (A) A pedestrian just over 8 meters away from the sensor. (B) Two pedestrians within 4 meters of the LiDAR.

3.5. Neuron Process and API

The Neuron process follows the typical LiDAR-based detection pipeline, divided into five stages: pre-processing, also known as voxelization, ground segmentation, clustering, feature extraction, and human classifier [24]. This data pipeline is illustrated in Figure 7.

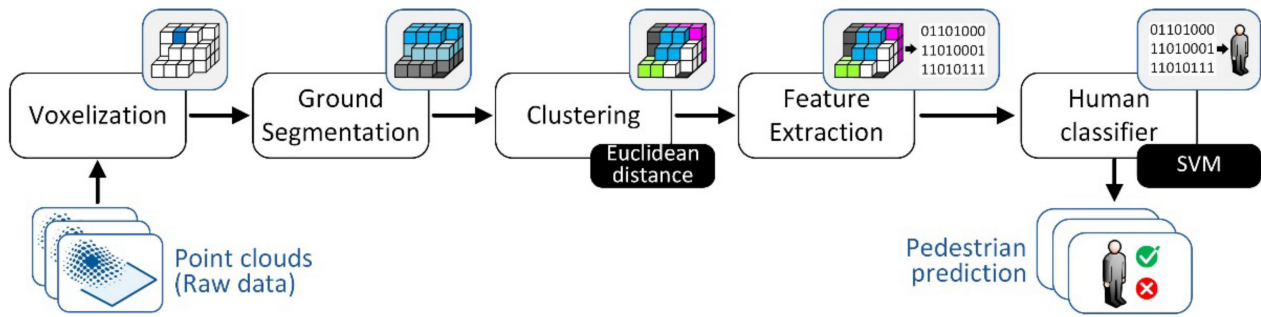


Figure 7. The Neuron process detection pipeline.

The *Voxelization* phase mainly includes downsampling, for processing efficiency and region-of-interest (ROI) definition. The ground segmentation phase is used to remove points from the ground plane and preserve point clouds that have relevant content to analyze. In this work, we are only keeping points $p_i = (x_i, y_i, z_i) \in R^3$, $i = 1, \dots, n$, where $z_i \geq z_{min}$. z_{min} can be configured and adjusted for each scenario.

The next step is to perform *Clustering* of points according to their spatial relationships. In this work, the Euclidean distance metric (d) was used to carry out this clustering process (see Equation (1)), which has proved to provide good results in different contexts.

$$d = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2} \quad (1)$$

In the *Feature Extraction* phase, the following features are included: feature 1 (f_1)—number of points included in the cluster; feature 2 (f_2)—minimum cluster distance from the sensor; feature 3 (f_3)—3D covariance matrix of the cluster; feature 4 (f_4)—normalized moment of inertia tensor; feature 5 (f_5)—slice feature for the cluster; and feature 6 (f_6)—reflection intensity distribution (mean, standard deviation, and normalized 1D histogram). Additional information on these features can be found in [25].

The last phase of the pipeline is the *Human Classifier*, where we use the Support Vector Machines (SVM) learning method as a binary classifier, trained for human classification (i.e., human and non-human) at each iteration, based on the six features previously referenced. The SVM method uses discriminant functions (see Equation (2)) to define a hyperplane that can linearly separate a dataset.

$$\begin{aligned} \hat{y}_n &= W^T x_n + b \\ \hat{y}_n &\in \{1, -1\} \end{aligned} \quad (2)$$

where $n \in \{1, \dots, N\}$, x_n is the data point, W is a vector of weights, b is the bias and \hat{y}_n is the model prediction. Each prediction can be either 1 or -1 for points that lie on both sides of the hyperplane and 0 for points that lie on the hyperplane itself. Given a dataset that contains several x with their corresponding y , the main objective is to find values for W and b that give the best possible margin.

The Neuron process uses the online-learning framework for human classification in 3D LiDAR scans [25]. An updated version of this work is presented in [26]. The original source code is open source, protected by the GNU General Public License v3.0, and can be found here [27]. The following permissions have been enabled:

- Commercial use
- Modification
- Distribution
- Patent use
- Private use

The pedestrian classifier was implemented with SVM [28], as referenced above, since it can produce good results in non-linear datasets such as the point clouds of pedestrians, also demonstrated in [29] and [30].

The Neuron process modifies the original source code to reflect the context of the pilot, including the provision of the following features through a new API:

- Pedestrian counter.
- For each detected pedestrian:
 - an **objectID**, which identifies this pedestrian in the current point cloud—it enables basic tracking functionality with previous point-cloud snapshots;
 - **LiDAR distance** (in meters) measures the pedestrian distance to the sensor;
 - **travel distance** (in meters) measures how much distance this pedestrian has travelled since the last snapshot (uses the objectID parameter for tracking);
 - **velocity** (in meters/second) measures the pedestrian velocity when comparing its position with the previous snapshot (uses the objectID parameter for tracking);
 - the **pedestrian coordinates**, based on its centroid X, Y, Z position;
 - the **elevation angle** (in degrees) measures the pedestrian cluster's centroid relative to the LiDAR sensor's horizontal plane—this parameter is defined as ω (lowercase Greek letter **omega**) in the VLP-16 datasheet;
 - the **displacement angle** (in degrees) provides the walking path angle with respect to the LiDAR sensor's X and Y axis;

With the above-mentioned parameters, it is possible to plot each pedestrian and its corresponding movement vector in space. The interpretation of the walking direction, based on the measured displacement angle, is explained in Figure 8. A positive displacement angle, with respect to the X axis, means the walking path should use a counterclockwise angle and the direction, with respect to the walking path, is from the rear to the front of the sensor. The opposite is true in the case that the measured displacement angle is negative.

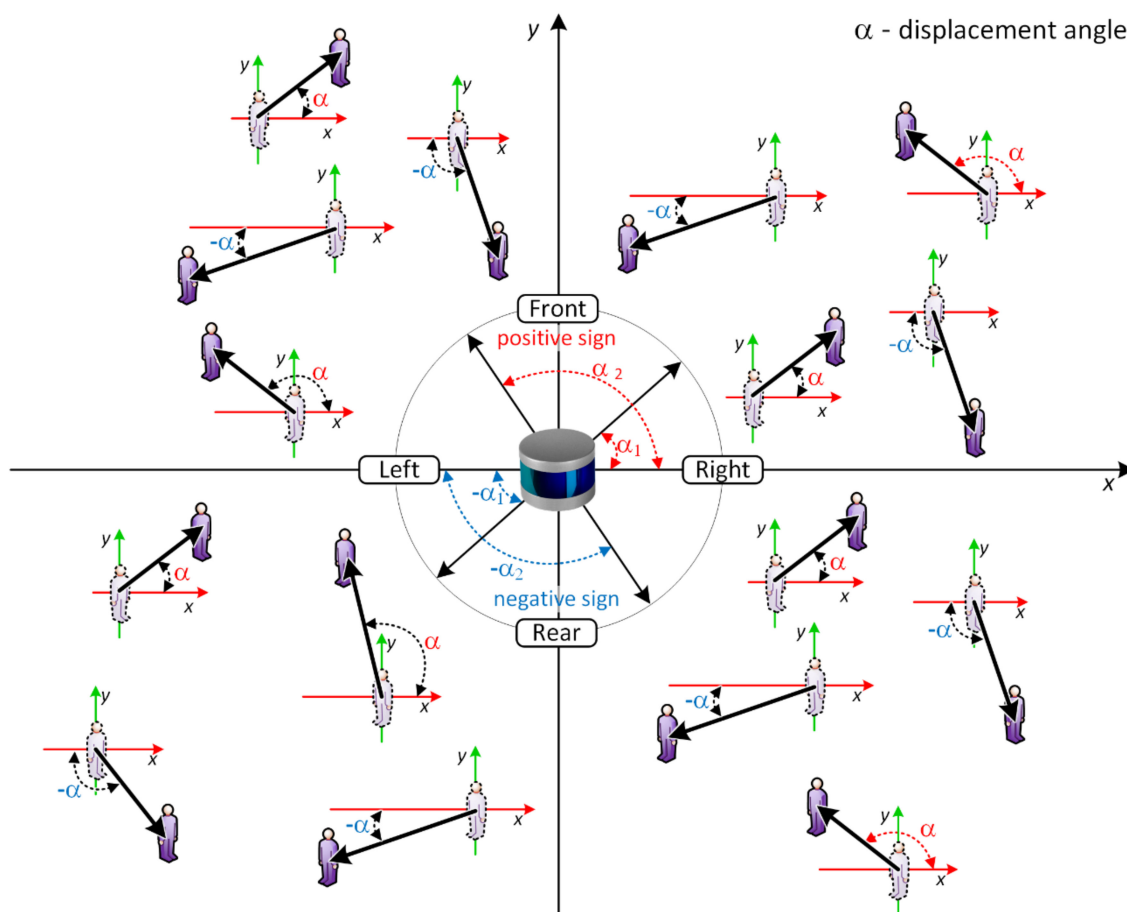


Figure 8. Relationship between the displacement angle (α) and the pedestrian walking direction.

4. Experimental Results

This section presents the experimental results achieved in a real deployment in the city of Aveiro, Portugal.

When placing the sensor (LiDAR) horizontally (no tilt) and at a height of approx. 1.20 m, the detection and classification of pedestrians are within the expectations derived from the analysis provided in Section 3, specifically in what is depicted in Figure 4d. Pedestrians and their associated information (position, velocity, direction, etc.) were successfully extracted from the received point clouds in quasi-real time. Figure 9 depicts a visual interpretation of the detected objects (all pedestrians), confirming the multi-tracking capability of the implemented solution. It is also possible to confirm that pedestrians were detected beyond the fifth ring, corresponding to laser channel 8 (using Figure 4 for context) and therefore up to 9 m distance from the LiDAR sensor. When placing the sensor at a height of 3 m with a 20° tilt (approx.), the detection and successful classification of pedestrians are also within the expectations derived from the analysis provided in Section 3, specifically in what is depicted in Figure 4b.

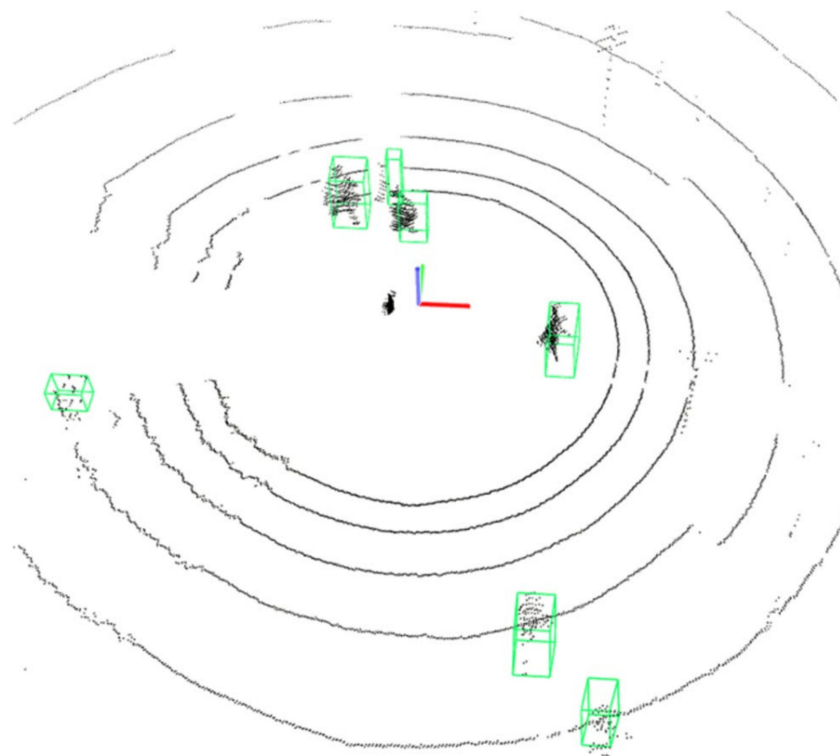


Figure 9. Detection of multiple pedestrians in a point cloud obtained with LiDAR sensor (VLP-16) positioned at a height of 1.20 m and with no tilt.

Figure 10 depicts a point cloud obtained with this setup. The laser reflection pattern clearly identifies the existence of horizontal tilt and a limitation in the field of view. Furthermore, the laser reflections are notably becoming sparser as we move away from the central position. Therefore, pedestrians will not be correctly identified by the implemented algorithm as they move left or right (away) from the sensor's central position, as seen in Figure 11.

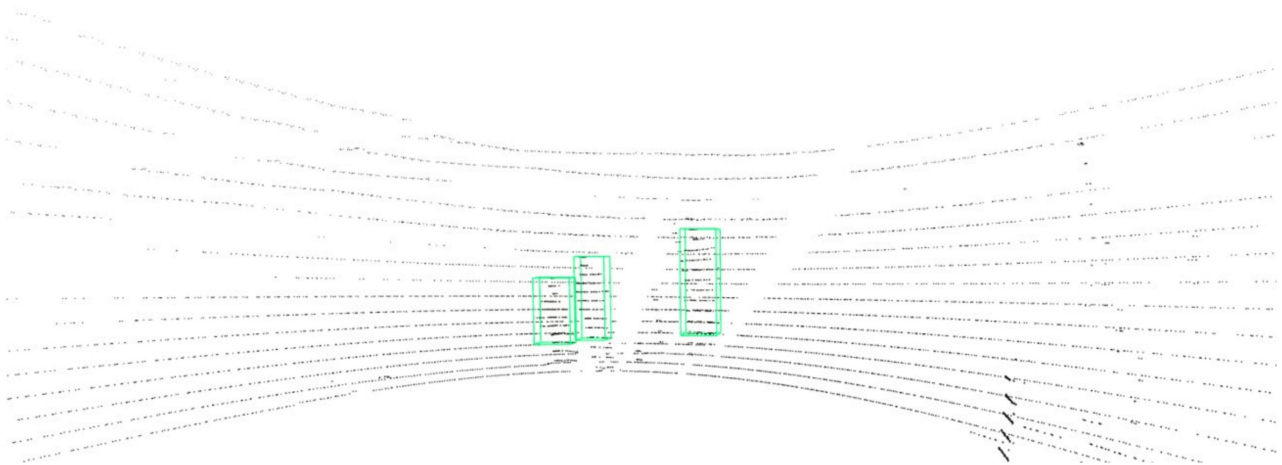


Figure 10. Detection of multiple pedestrians in a point cloud obtained with LiDAR sensor (VLP-16) positioned at a height of 3 m and with a tilt of approx. 20° .

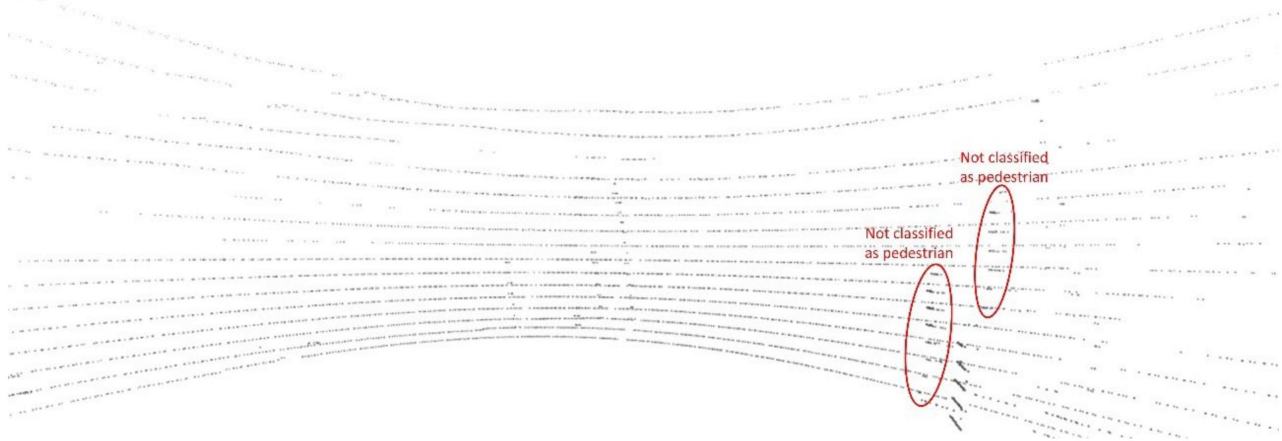


Figure 11. Failure of successfully detecting pedestrians due to insufficient training data for this setup (height of 3 m and tilt of approx. 20°).

Figure 12 depicts the deployed scenario, where, in this case, we can see a successful detection of four pedestrians, very close to each other, and the information extracted from each of these objects, such as its ID, distance to the LiDAR, distance travelled since the first detection, speed, direction of movement and other information. Table 3 presents the movement information related to each pedestrian detected.

Table 3. Pedestrian data information.

Object ID	Lidar Distance [m]	Travel Distance [m]	Velocity [m/s]	Displacement Angle (α) [$^\circ$]
1	2.18	0.14	1.29	118.72
2	2.99	0.15	1.31	120.86
3	3.74	0.15	1.31	−104.87
4	3.04	0.22	1.97	126.60

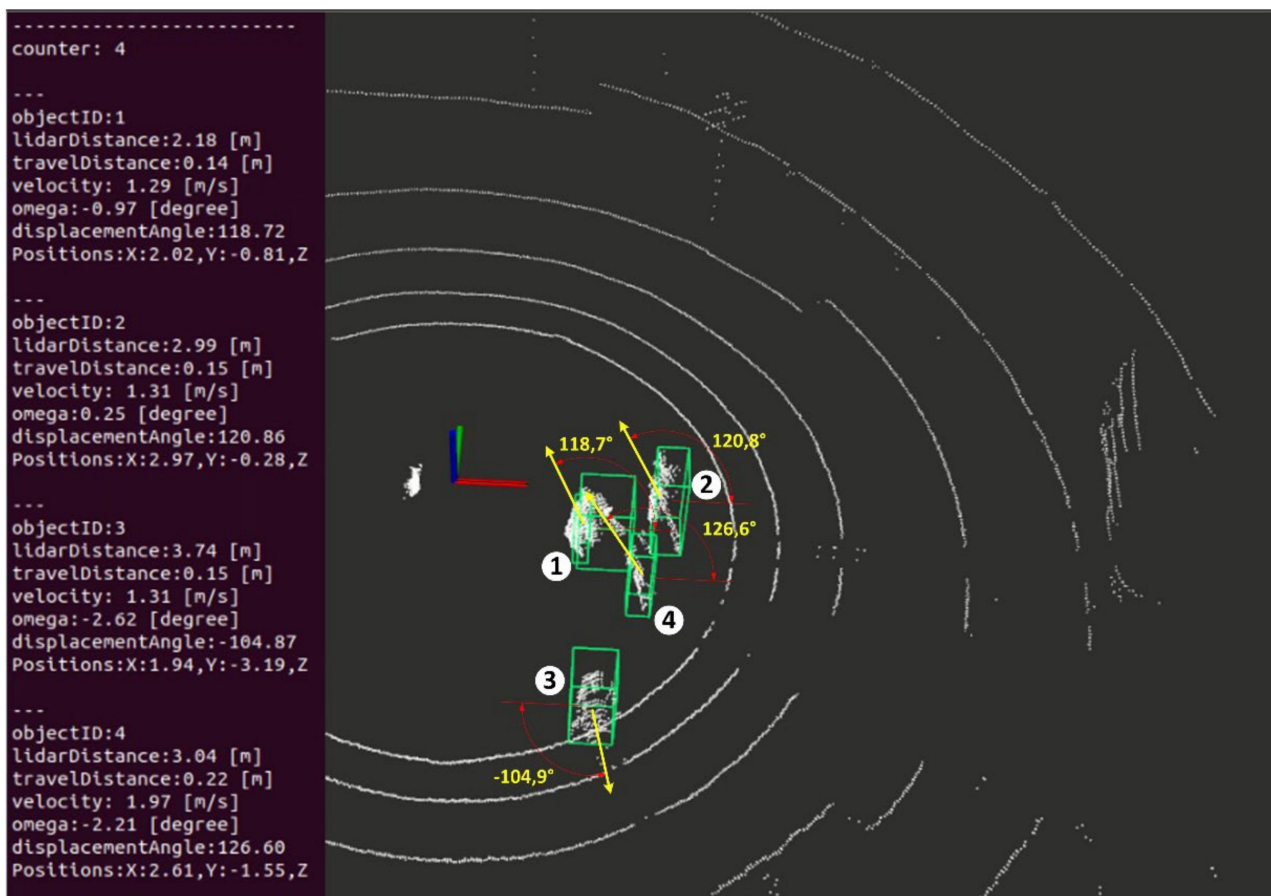


Figure 12. Screenshot of a pedestrian's detection with the counters and movement information.

5. Conclusions

This paper discusses the challenges inherent with pedestrian detection in a smart-city context, using data provided by LiDAR sensors only. It also describes a solution for automatic pedestrian detection and feature extraction using available open source code. Systems such as this are an important asset for municipality management structures to monitor and better understand the flow of pedestrians in their city, mainly for safety monitoring but also to understand city livelihood. The presented solution was developed over an SVM classifier, pretrained for human detection and adapted for outdoor context.

Based on the results, we conclude that installing LiDAR sensors on poles, despite the characteristics referring to having a range of up to 100 m, presents great challenges for the automatic detection of objects at those distances. For an efficient detection in an outdoor scenario in an urban environment, as presented in this work, it is possible to detect pedestrians up to 15 m away, and performance depends on sensor height and inclination, the vibration produced by the motor of the LiDAR, and wind conditions (due to pole movement). Despite these constraints, it was possible to implement a solution capable of detecting pedestrians and produce credible results that can be published by the city monitoring platform. The solution is currently up and running.

The next steps are the inclusion of a real-time visualization tool that plots the most common paths taken by pedestrians and include the detection and classification of cars, bicycles and boats (the city of Aveiro has an inner river where boats/moliceiros travel), and the inclusion of a set of tests to determine the maximum velocity detected by the algorithm (e.g., detecting fast runners).

Author Contributions: Conceptualization, P.M. and H.M.; methodology, H.M.; software and hardware, P.T.; validation, P.T. and H.M.; investigation, P.T. and H.M.; writing—original draft preparation, P.T.; writing—review and editing, P.T., H.M. and P.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by Fundo Europeu de Desenvolvimento Regional (FEDER) through the POCI-01-0247-FEDER-046962 (5GAUTO).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

- Shen, X. A survey of Object Classification and Detection based on 2D/3D data. *arXiv* **2019**, arXiv:1905.12683.
- Zamanakos, G.; Tsochatzidis, L.; Amanatiadis, A.; Pratikakis, I. A comprehensive survey of LIDAR-based 3D object detection methods with deep learning for autonomous driving. *Comput. Graph.* **2021**, *99*, 153–181. [[CrossRef](#)]
- Shi, S.; Wang, X.; Li, H. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 770–779. [[CrossRef](#)]
- Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3354–3361. [[CrossRef](#)]
- Shi, S.; Wang, Z.; Shi, J.; Wang, X.; Li, H. From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 2647–2664. [[CrossRef](#)] [[PubMed](#)]
- Hu, J.K.; Kuai, T.; Waslander, S. Point Density-Aware Voxels for LiDAR 3D Object Detection. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 8459–8468. [[CrossRef](#)]
- Jiang, H.; Lu, Y.; Chen, S. Research on 3D Point Cloud Object Detection Algorithm for Autonomous Driving. *Math. Probl. Eng.* **2022**, *2022*, 8151805. [[CrossRef](#)]
- Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 4338–4364. [[CrossRef](#)] [[PubMed](#)]
- Song, W.; Li, D.; Sun, S.; Zhang, L.; Xin, Y.; Sung, Y.; Choi, R. 2D&3DNet for 3D Object Classification in LiDAR Point Cloud. *Remote Sens.* **2022**, *14*, 3146. [[CrossRef](#)]
- Song, W.; Zhang, L.; Tian, Y.; Fong, S.; Liu, J.; Gozho, A. CNN-based 3D object classification using Hough space of LiDAR point clouds. *Hum. Cent. Comput. Inf. Sci.* **2020**, *10*, 161–191. [[CrossRef](#)]
- Fernandes, D.; Silva, A.; Névoa, R.; Simões, C.; Gonzalez, D.; Guevara, M.; Novais, P.; Monteiro, J.; Melo-Pinto, P. Point-cloud based 3D object detection and classification methods for self-driving applications: A survey and taxonomy. *Inf. Fusion* **2021**, *68*, 161–191. [[CrossRef](#)]
- Wu, D.; Liang, Z.; Chen, G. Deep learning for LiDAR-only and LiDAR-fusion 3D perception: A survey. *Intell. Robot.* **2022**, *2*, 105–129. [[CrossRef](#)]
- Chen, X.; Ma, H.; Wan, J.; Li, B.; Xia, T. Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1907–1915.
- Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3d object detection from rgb-d data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 918–927.
- Liang, M.; Yang, B.; Chen, Y.; Hu, R.; Urtasun, R. Multi-task multi-sensor fusion for 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7345–7353.
- Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10526–10535. [[CrossRef](#)]
- Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
- Yan, Y.; Mao, Y.; Li, B. SECOND: Sparsely Embedded Convolutional Detection. *Sensors* **2018**, *18*, 3337. [[CrossRef](#)] [[PubMed](#)]
- Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705.
- Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; Jia, J. Std: Sparse-to-dense 3d object detector for point cloud. In Proceedings of the IEEE International Conference on Computer Vision, Cambridge, MA, USA, 20–23 June 1995.

21. ROS Melodic. Available online: <http://wiki.ros.org/melodic> (accessed on 1 December 2022).
22. Aveiro Tech City Living Lab. Available online: <https://www.aveirotechcity.pt/en/activities/aveiro-tech-city-living-lab> (accessed on 1 March 2023).
23. Blog, V. Laser Safety in a Lidar World. Available online: <https://velodynelidar.com/blog/laser-safety-in-a-lidar-world/#:~:text=Even%20if%20a%20viewer%20intentionally,sensors%20are%20designed%20to%20protect> (accessed on 1 March 2023).
24. Wu, B. Efficient Deep Neural Networks. Ph.D. Thesis, University of California, Berkeley, CA, USA, 2019.
25. Yan, Z.; Duckett, T.; Bellotto, N. Online learning for human classification in 3D LiDAR-based tracking. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 864–871. [CrossRef]
26. Yan, Z.; Duckett, T.; Bellotto, N. Online learning for 3D LiDAR-based human detection: Experimental analysis of point cloud clustering and classification methods. *Auton. Robot.* **2020**, *44*, 147–164. [CrossRef]
27. GitHub: “Online Learning for Human Classification in 3D LiDAR-Based Tracking”. Available online: https://github.com/yzrobot/online_learning (accessed on 1 March 2023).
28. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
29. Navarro-Serment, L.E.; Mertz, C.; Hebert, M. Pedestrian detection and tracking using three-dimensional lidar data. In Proceedings of the 7th Conference on Field and Service Robotics (FSR), Cambridge, MA, USA, 14–16 July 2009; pp. 103–112.
30. Kidono, K.; Miyasaka, T.; Watanabe, A.; Naito, T.; Miura, J. Pedestrian recognition using high-definition LIDAR. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011; pp. 405–410.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.