Fall 12-22-2020

LiDAR Object Detection Utilizing Existing CNNs for Smart Cities

Vinay Ponnaganti
San Jose State University

LiDAR Object Detection Utilizing Existing CNNs for Smart Cities

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Vinay Ponnaganti

October 2020

The Designated Project Committee Approves the Project Titled

LiDAR Object Detection Utilizing Existing CNNs for Smart Cities

By

Vinay Ponnaganti

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

October 2020

Dr. Teng-Sheng Moh	Department of Computer Science
Dr. Melody Moh	Department of Computer Science
Brad Hoffert	LocoLabs LLC

ABSTRACT

As governments and private companies alike race to achieve the vision of a smart city — where artificial intelligence (AI) technology is used to enable self-driving cars, cashier-less shopping experiences and connected home devices from thermostats to robot vacuum cleaners — advancements are being made in both software and hardware to enable increasingly real-time, accurate inference at the edge. One hardware solution adopted for this purpose is the LiDAR sensor, which utilizes infrared lasers to accurately detect and map its surroundings in 3D. On the software side, developers have turned to artificial neural networks to make predictions and recommendations with high accuracy. These neural networks have the potential, particularly run on purpose-built hardware such as GPUs and TPUs, to make inferences in near real-time, allowing the AI models to serve as a usable interface for real-world interactions with other AI-powered devices, or with human users. This paper aims to example the joint use of LiDAR sensors and AI to understand its importance in smart city environments.

ACKNOWLEDGEMENTS

I would like to thank my project advisors Dr. Teng-Sheng Moh and Dr. Melody Moh, as well as Brad Hoffert from LocoLabs LLC, for their guidance, patience, and expertise in this developing field.

TABLE OF CONTENTS

1. INTRODUCTION

Smart city applications are becoming increasingly popular as advances in artificial intelligence and hardware technology improve over time. Applications range from autonomous navigation to pedestrian detection and traffic congestion management, utilizing a multitude of different sensors. LiDAR sensors, specifically, have been used in more applications as the sensors become readily available and reduce in cost. They provide inherent benefits that other sensing modes do not provide, namely accurate distance measurements, and in many cases, 3-dimensional point clouds of their surroundings. These sensors leverage infrared light to detect distance using the time of flight, and can be used in many different environments. An example of a scan of an intersection can be seen in Figure 1.

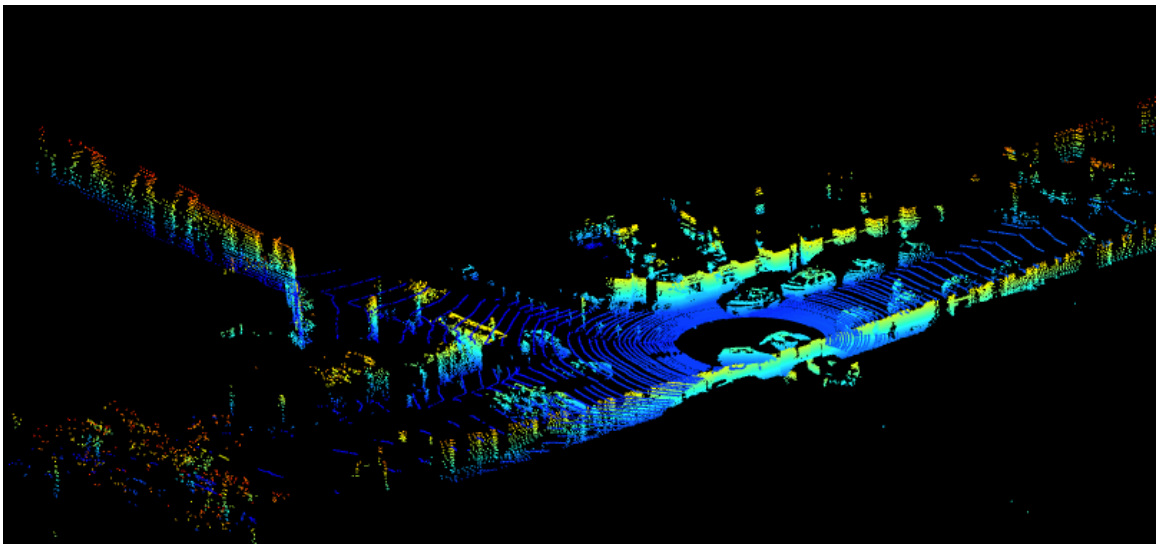

Figure 1: This figure depicts a single frame of LiDAR data collected at an intersection; showing the raw information available for use in smart city inference applications.

As smart city applications begin to develop, there are also growing concerns regarding individuals' privacy. Many current applications propose the use of camera-

based artificial intelligence solutions that leave cities concerned about the use of facial detection algorithms that can cause privacy concerns for residents. LiDAR sensors contain significantly less detail about facial features and color that can be used to distinguish individual people. The sensors targeted in this project are only able to detect distance to a point, and the reflectivity of that point [17][19]. For this reason, among others, several cities are switching to using LiDAR based implementations for smart city applications.

Smart cities tend to require object detection AI in order for their intelligent devices to understand the city's environment, tracking variables such as traffic flow, pedestrian safety, and even social distancing compliance during a pandemic. Many existing solutions leverage artificial intelligence, and specifically Convolutional Neural Networks (CNNs) with camera images, for detection, as these have become a proven solution in recent years. This research aims to leverage the use of existing Convolutional Neural Networks with LiDAR sensors in place of cameras, and evaluate how their performance may vary. The project aims to understand if object detection is feasible with LiDAR sensor data, as more cities begin to transition away from using cameras.

The runtime of these solutions is also critical, as pedestrian and vehicle safety can require near real-time processing in order to make decisions around traffic signals and crosswalks. The runtime and performance tradeoffs will need to be studied on potential edge devices that would be suitable targets for such smart city applications to understand if LiDAR based inference could be useful in such situations.

2. BACKGROUND AND RELATED STUDIES

To understand the state of LiDAR research, and its application to smart city object detection use cases, it is essential to survey both LiDAR studies and smart city neural network projects to identify the opportunities for the two to intersect. This review begins by examining types of LiDAR sensors and use cases of LiDAR data, followed by an exploration of simultaneous localization and mapping (SLAM) technology, as well as existing research using LiDAR as a dataset for convolutional neural networks. The section concludes by providing an overview of anonymity concerns in smart city applications, and the potential of LiDAR data to be applied in smart city solutions.

2.1. Use Cases of LiDAR Technology

Studying use cases plays a key role in understanding relevant applications of LiDAR and how future technologies may benefit from further research using these sensors. It also provides useful insight into different types of LiDAR sensors and how the sensor data could be augmented or used for perception.

One of the earlier papers, provided by the University of Beijing Information & Science Technology, surveyed time-of-flight sensors that were available and compared performance for autonomous vehicle applications. This research compared the data density, or number of output points, and the relative accuracy of several sensors. At the time of this paper, sensors produced by a company called Velodyne dominated the survey results for 3D scanning LiDAR results. The paper also compares single-line scanning and non-scanning LiDAR sensors. The results of this comparison was that single-line scanning LiDARs do not provide enough data output points for SLAM, and non-scanning

LiDARs require technology that is not suited for outdoor autonomous vehicles [9]. Non-scanning LiDARs, also referred to as Flash LiDARs, act on a similar basis of a pinhole camera, where light that travels through the hole can be projected on an imager. This would provide high-density results, however, is highly sensitive to a visible light spectrum [9]. While having the ability to have high output and densely packed points, the high sensitivity to visible light, found outdoors, makes this technology limiting for most real world applications.

Using the research found in this paper, it was clear that a scanning LiDAR, similar to the ones from Velodyne were a target platform for further research and development. Upon surveying this topic, a few papers were studied to try and understand the range of existing use cases of LiDAR. The first was a paper from the LiDAR Group at the Institut für Physik der Atmosphäre in Germany, where airborne water vapor was studied using LiDAR technology mounted on an aircraft. This research allowed scientists to better understand and map water vapor densities in geographic areas with higher resolution. Previously, data was only recorded from ground level weather stations and did not allow for the level of detail and depth scientists needed to understand water vapor distribution throughout the atmosphere [4]. With the high resolution information, scientists could detect weather patterns and precipitation trends in regions. The data was collected using a LiDAR that emitted a wavelength that was distorted by the amount of water vapor present in the air. The LiDAR was mounted underneath an aircraft which then scanned several regions for data.

Another paper worth noting is “Dual IR Spectral Video Inspection of a Concealed Live Animal.” While surveying LiDAR technology, it became clear that LiDAR technology primarily operates in the infrared spectrum of light. It was worth noting the different applications of IR and detection in the real world, to understand how different research projects selected sensors based on their use case. This paper aimed to detect animals, even while concealed in boxes, or behind reflective barriers. This paper was one of the older research papers studied and did not utilize deep learning methods for detection. The paper leveraged an IR camera and common pixel filtering techniques to enhance heat signatures of living creatures. This was achieved through an Adaptive Neighbor Contrast Enhancement algorithm that grouped similar pixels based on an acceptance threshold [12]. By doing so, researchers were able to create a filtered image. By further refining these results, and conducting centroid calculations on similarly grouped pixels, researchers were able to better visualize the animal. Given the development of deep learning methodologies and libraries since the conception of this paper, the stages used to filter the image would serve as useful preprocessor steps in image detection CNNs.

The 2019 study “Pedestrian Detection and Tracking Based on 2D LiDAR” took the use case of a sidewalk robot, using a machine learning algorithm for feature extraction to detect the legs of pedestrians from LiDAR data and avoid collisions. In this paper, the authors relied on the less expensive single-line LiDAR sensors rather than the multi-line LiDAR sensors usually used in autonomous driving applications. This

produced mixed results due to the higher degree of noise in the data, but demonstrated a low-cost LiDAR application that was also low in computational demand [8].

2.2. LiDAR SLAM Technology

SLAM, a navigation technology for autonomous devices like robots, uses a combination of sensors — including traditional cameras and LiDAR. Exploring SLAM technology sheds light on the relative benefits and shortcomings of LiDAR data for perceiving a machine's surroundings. For one, LiDAR sensors do not contain densely packed pixels of information the way visual camera images do. This poses a concern that LiDAR data frames would not contain enough information for a neural network to detect or classify types of objects.

The first relevant paper, "Road-feature extraction using point cloud and 3D LiDAR sensor for vehicle localization" attempts to solve road feature extraction using LiDAR SLAM algorithms to create higher density point clouds. In doing so, the researchers were able to successfully filter the information to extract features such as curbs and lane markings. These two features may seem similar, however, there are two modes of detection. In the case of curbs, the depth information is received from the sensor and then stitched using SLAM to provide shape information. For lane markings, the reflectivity of the paint, in contrast with the road, is used to detect the lane line in a point cloud [6]. While the algorithm may not have relied on deep learning, the researchers were able to create dense point clouds that utilized two forms of data that LiDAR can sense: depth and reflectivity.

Another SLAM paper, “Comparison of ROS-based visual SLAM methods in homogeneous indoor environment,” explores various methodologies of SLAM technology — primarily focused on stereoscopic cameras and 2D LiDAR. Stereoscopic cameras, also called RGB-D, Red, Green, Blue, and Depth, perceive depth information similar to the way humans do. Using two fixed camera points, the depth information of a pixel can be calculated using triangulation, based on where the same pixel appears in each camera. In order to classify depth to a pixel, camera images can be preprocessed to find different objects on screen. This can be done using a CNN, or image segmentation [7]. Then by comparing image segment location between the two images, a perceived depth can be calculated. An issue mentioned in this research was loop closure issues regarding existing SLAM programs/algorithms. Current SLAM algorithms regularly fail to track location over time, and will create drift in the point clouds that are created. When doing so, the resulting image can look nothing like the real world.

2.3. LiDAR and Deep Learning Applications

LiDAR technology has been used in various applications and continues to make inroads in several industries — today most commonly in the robotics and autonomous driving sectors. LiDAR companies have been marketing their sensors to numerous self-driving companies and research initiatives, in order for autonomous vehicles (cameras, radars, LiDAR) to detect objects and obstacles on the road. One study, titled “CNN for Very Fast Ground Segmentation in Velodyne LiDAR Data,” used neural networks to segment the ground, a useful tool for self-driving systems to detect roadways. The researches encoded the 3D LiDAR data into a 2D channel for the task, and achieved high

performance accuracy and speed — essential metrics for successful deployment in autonomous driving use cases [13].

In many cases, researchers have experimented with neural networks and LiDAR for pedestrian detection. For example, "Pedestrian-Detection Method based on 1D-CNN during LiDAR Rotation" took 3D LiDAR data and converted it to 1D waveform data in order to run it as a 1D CNN — making it so that pedestrians were classified by individual points instead of bounding boxes on a 2D image or 3D point cloud. Using a process not unlike semantic segmentation, the researchers were able to achieve 20% improvement over prior methods while shrinking computation time using the 1D data approach [23]. On the other side of the spectrum lies 3D CNNs, which were used by Tatebe et. al in "Pedestrian detection from sparse point-cloud using 3DCNN." Despite the low-resolution dataset acquired by low-budget LiDAR sensors, the researchers demonstrated high accuracy in pedestrian detection from 3D point clouds using the 3DCNN architecture. Targeting towards a smart city use case of advanced driver-assistance systems, or ADAS, this neural network architecture leveraged a 3D object detection algorithm [24]. To enhance the performance of the deep learning model, the researchers also relied on data augmentation techniques to enhance their training dataset.

Other research has focused on CNNs that detect other cars from LiDAR data. The paper, "Online Camera LiDAR Fusion and Object Detection on Hybrid Data for Autonomous Driving," leverages data fusion, specifically LiDAR and camera data, to create accurate predictions using CNNs. The data and results observed in this research were aimed at proving the validity of data fusion and the importance of calibration

between the fused sensor data sources. The LiDAR data has been overlaid on top of camera data to provide reflectivity and depth information that the ordinary camera images do not provide. This gives trained CNNs more information to detect and locate objects on a 2D image [11]. It is clear that the CNN was able to detect objects that were not seen in the raw visual image alone. While the results of this effort was impressive, the LiDAR data was used as an enhancement, rather than a primary source of data. However, this project treated the LiDAR data as a 2D image instead of a 3D depth map — like in the current project. Another study by Baidu Research, "3D fully convolutional network for vehicle detection in point cloud", developed a custom fully convolutional network (FCN) to detect vehicles from 3D LiDAR data, focusing primarily on accuracy but without an analysis of speed, which is critical for use in real world autonomous driving situations [1].

3. METHOD AND EXPERIMENT

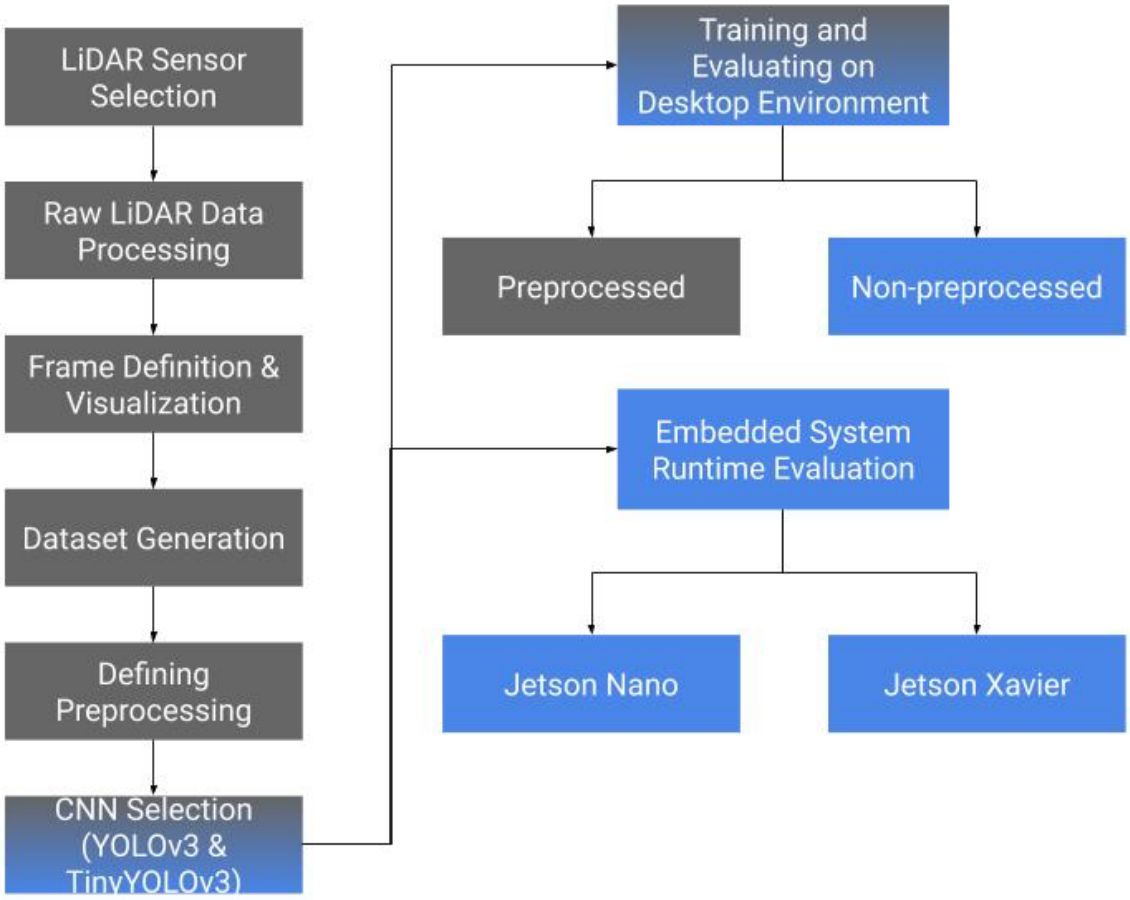

Figure 2: Block diagram representing work done as part of this research [22].

The block diagram shown in Figure 2 depicts the path and design chosen for this project. Since LiDAR based CNN detection is a relatively new topic, there were many steps required in order to process raw LiDAR data and present it in a 2D frame that can be used as input to an existing CNN originally intended for camera images. The solutions proposed in this method are then evaluated and compared with the current state of camera based inference to understand if LiDAR based inference using the same CNNs is a viable solution to transition to for smart city applications. This method was also evaluated on

different embedded platforms to understand the runtime performance to provide readers with an understanding of potential runtime environments, as these devices, among others, are being used in existing AI and CNN applications.

For reference, the grey sections of the block diagram were conducted by the same authors of this paper, and were included as part of the chapter, “Deep Learning for LiDAR-based Autonomous Vehicles in Smart Cities.” The blue sections indicate the work conducted as part of a conference submission, “Utilizing CNN for Object Detection in LiDAR Data for Autonomous Driving,” also written by the same authors as this paper [21][22].

3.1. LiDAR Sensor Selection

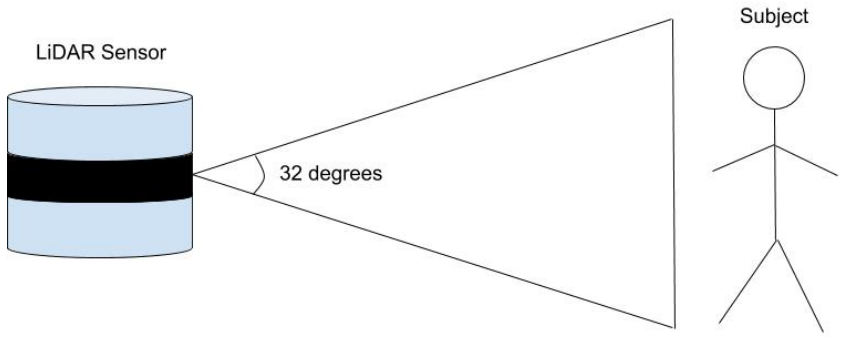

Figure 3: Vertical field of view of a Velodyne LiDAR sensor.

In order to begin research, a LiDAR sensor needed to be selected to process raw data to create a dataset for training and validation. LiDAR sensors vary greatly and should be selected based on the environment they are used in. After some searching, the Velodyne VLP-16 sensor was selected based on its popularity and type of data it was capable of producing. This sensor is able to create 3D point clouds with a field of view of 360 degrees horizontally and 32 degrees vertically in a single frame. This is a spinning

sensor that has 16 lasers mounted horizontally that can measure distance up to 100 meters.

For the purposes of object detection, it was important to select a sensor that had enough lasers and a vertical field of view that could capture and detect objects within it. This was evaluated by observing the data within the Velodyne provided visualization tool, VeloView [20], and seeing if objects were captured within the sensor, and if they could be distinguished by a human. VeloView, paired with some sample data provided from the sensor showed that the sensor was capable of capturing cars, trees, and other surrounding objects within a single 360 degree scan of data. With some difficulty, these objects were distinguishable by a human within the data.

Since this was a rather objective measurement, this could also be verified quantifiably by using formula 1 to determine if the height of an object would fit within the LiDAR scans, and the distance the sensor needs to be in order to capture the entire object.

$$Distance = \frac{Y}{2 \tan 16°}$$

Formula 1: derived to find the minimum required distance to an object to capture it in frame.

Using formula 1, we can see that a vehicle with a height of 6 feet would need to be a minimum of 10.46 feet away in order to fit within the LiDAR scan. This is well within the scope of a smart city application, as common use cases include city roads and intersections, spanning several hundred feet at times.

3.2. Raw LiDAR Data Processing

While there were applications, such as VeloView and several other tools available to parse and view raw LiDAR data in 3D, there were not many options to label this 3D point cloud data. In order to present the data in the 2D image that was later decided on, the raw binary data from the LiDAR sensor needed to be processed.

Some LiDAR sensors, including the Velodyne VLP-16, transmit packets of binary sensor data over a local network. By understanding the programming guide provided by Velodyne, the packet format could be deciphered and parsed to collect the sensor data before being converted into a 3D set of points in a cartesian coordinate system. This is important since the raw data from the sensor is actually presented in polar coordinates. Maintaining a polar coordinate system makes it possible to present the data into a space efficient 2D image. This will be discussed in further detail later.

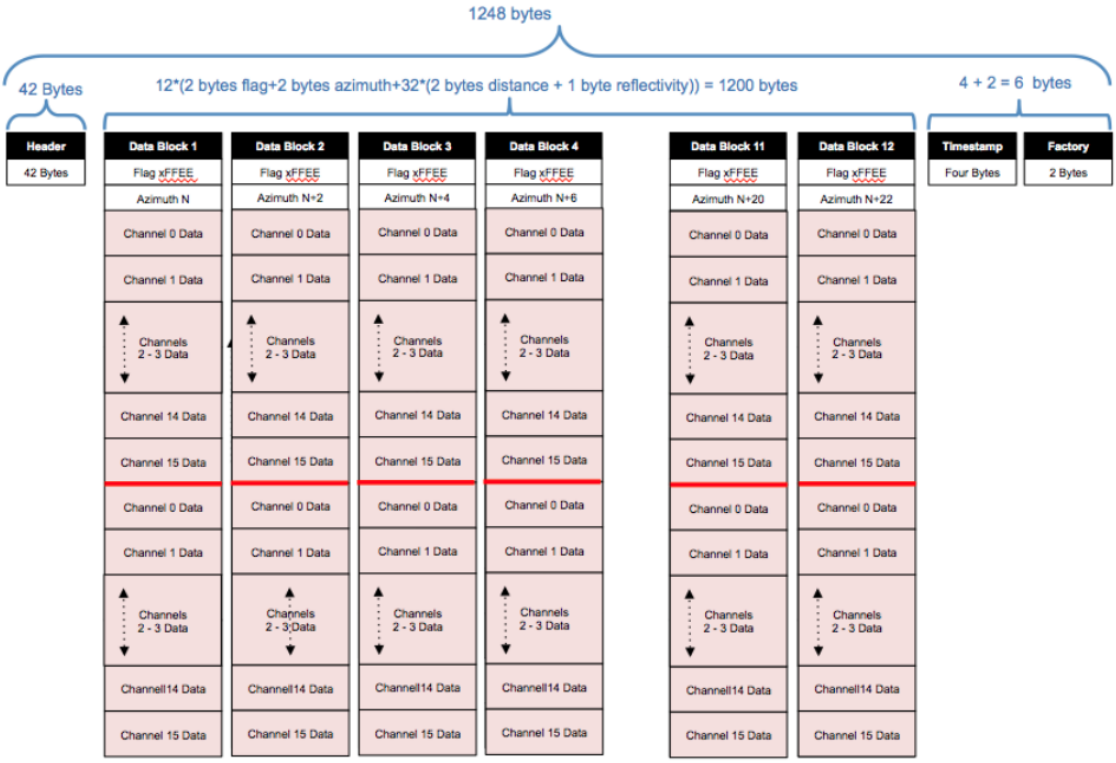

Figure 4: This image shows the Velodyne VLP-16 packet structure [18].

An outline of the packet data can be seen in Figure 4, where each channel represents a fixed spinning laser, at a given azimuth, or horizontal angle. Each channel contains both a reflectivity value of the object the laser bounced off of, as well as the distance. The sensor broadcasts this raw data utilizing the User Datagram Protocol (UDP), and sometimes results in some packets being lost, requiring the parser to be robust to handle partial information from the capture, and fill in missing information. For the purposes of this research, the header, timestamp, and factory bytes were ignored.

3.3. Frame Definition and Visualization

In order to create a dataset, a frame needed to be defined to create images from the raw data. Since the horizontal field of view of the sensor was 360 degrees, a new

frame was created every time the azimuth, or horizontal angle, collected from the data packet looped back to zero, or the next closest azimuth if a packet was lost. This would create frames of 360 degrees horizontal by 16 lasers vertical. The vertical lasers spanned a vertical field of view of 32 degrees.

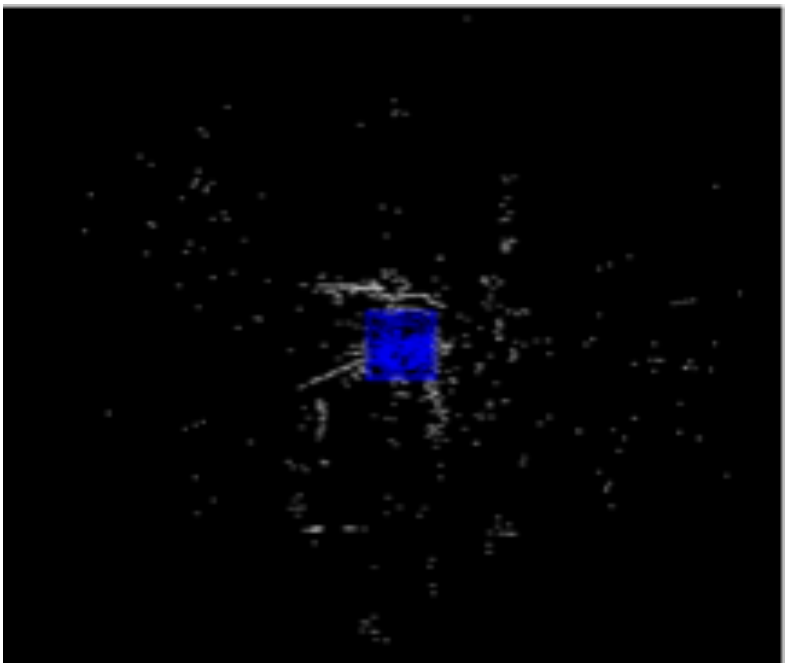

Figure 5: Birds eye view of 3D point cloud in 3D data structure.

With this raw data processed into frames, an actual visualization method needed to be determined. Initially, the data was visualized in 3D, converting the polar coordinate system to cartesian, and displaying them using an open source library called OpenFrameworks [16]. However, this method created a highly inefficient dataset that would only utilize 0.5% of the total structure at maximum capacity. A sample of a LiDAR frame in a 3D cartesian coordinate space can be viewed in Figure 5, where a majority of the structure is not populated.

After understanding the inefficiencies associated with the 3D data structure, a 2D representation was explored. In the polar coordinates provided in raw form, the data can be seen as two dimensional, where the horizontal, or X-axis, could be considered the azimuth, and the vertical angle, or Y-axis, could be considered the channel. This would create a two-dimensional data structure that had two values associated with each point: distance and reflectivity. For convenience and code reuse, this could fit a structure of a 3-channel RGB image that had one channel left empty. An example of this can be seen in Figure 6, where the green channel represents the distance, and the red represents the reflectivity. These frames were populated using an open source image processing library called OpenCV [15].

The final data structure that was selected, in this case a 2D image, also was intended to preserve contiguous data. In the case of CNNs, the algorithm relies on relevant data to be near each other as the convolution layers process neighboring batches of pixels. This was important to preserve when using CNNs and was maintained by using this 2-dimensional structure.

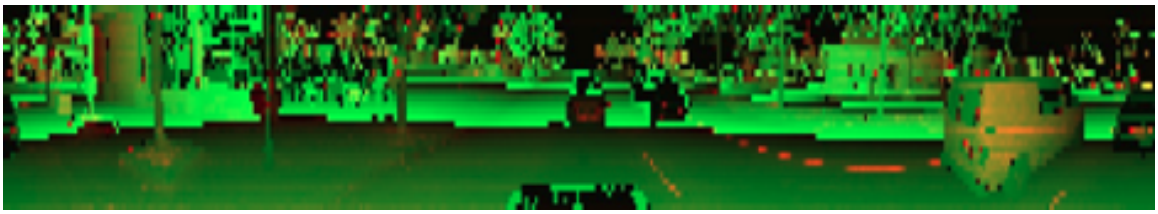

Figure 6: 2D LiDAR frame parsed from raw data (Ponnaganti, Moh, and Moh, 2020).

3.4. Dataset Generation

In order to train and evaluate a neural network with LiDAR data, two datasets were created from a raw packet capture. The first was used as a training set, and the

second for testing and validation. This data stream captured several vehicles on a road traveling in multiple directions, as well as many other surrounding objects. For the purposes of this research, vehicles were selected as the target object for detection. This can of course be modified to the object of your choosing, however, there would need to be enough examples of it in the data that was collected. In the case of this experiment, vehicles were the most abundant. The vehicles in the data were also the easiest to distinguish for labeling. For the purposes of this experiment, the training dataset contained 400 images and the test dataset contained 50.

3.5. Defining Preprocessing

Similar to the methodologies used to develop image processing algorithms, a preprocessing method was explored to understand if it could help improve prediction precision. Several image processing and detection algorithms utilize a preprocessing step that can either simplify, or enhance aspects of the image deterministically in order to improve the performance of the network prediction. Since LiDAR contains accurate distance and spatial information, this data can be leveraged to effectively discard points that are not relevant to the detection. This can be used to help isolate objects and potentially improve detection precision.

In looking at the 2D image in Figure 6, there is a significant amount of data that does not pertain to the vehicle seen to the right. Much of this data can be discarded before the CNN by filtering points based on height, distance, and reflectivity.

Distances greater than 7.5 meters were discarded for this test, as the vehicles outside of that range were difficult to label by hand. This can be seen in Figure 7, where

the vehicle outside of this range was too difficult to recognize until it got closer to the sensor in later frames.

Figure 7: Subsection of a LiDAR frame with a vehicle just out of sight [21].

Through experimentation, the ground values were filtered out based on their cartesian coordinate Z value. This Z value was obtained using the formula provided in the Velodyne Programming guide, also shown in Figure 8 [18]. where in the case of this experiment, they were Z values that were farther than 1.7 meters below the sensor.

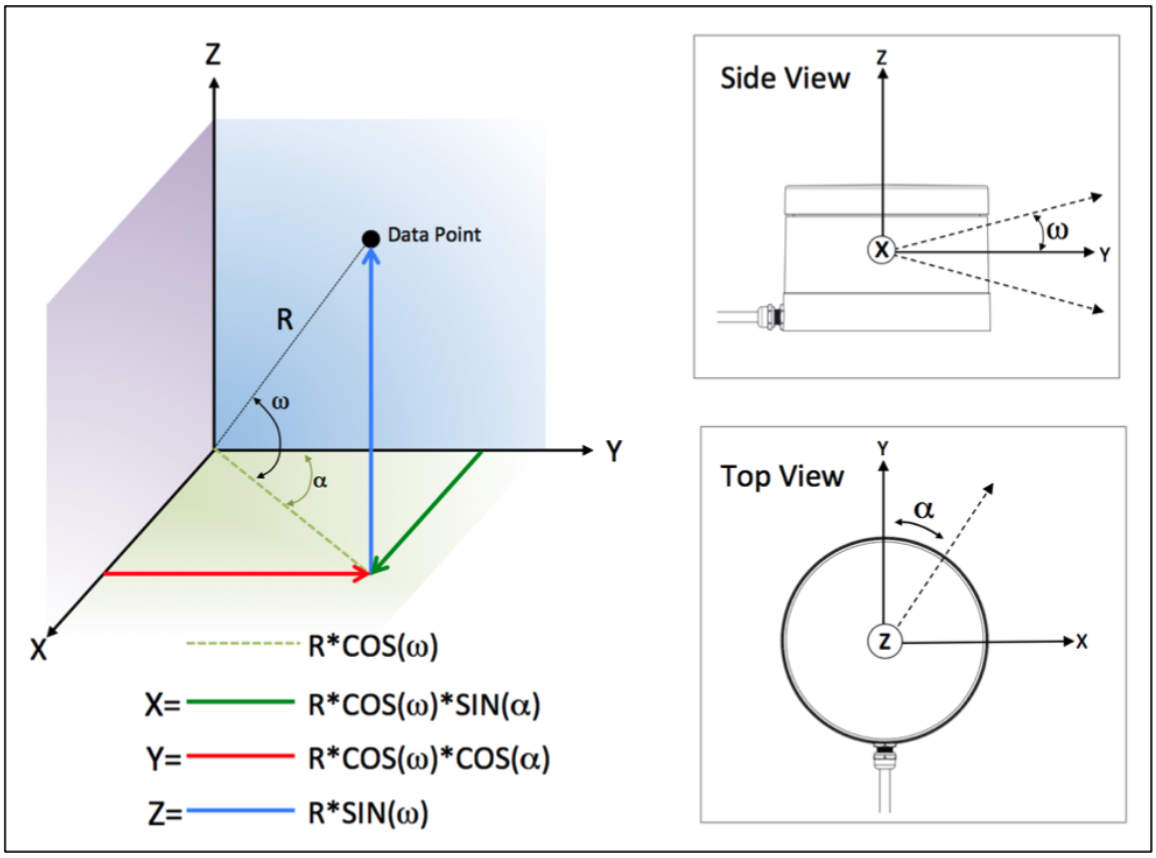

Figure 8: Visual of polar coordinate to cartesian coordinate conversion [18].

While not entirely perfect, the ground filtering reduced the amount of data that was in the image, better isolating the subject. An example of where the ground filtering did not remove all the ground data can be shown in Figure 9, where the curb is higher than the Z filter value and is still visible.

Figure 9: LiDAR frame during preprocessing where part of the curb was not filtered out [21].

Lastly, through trial and error, points were discarded that contained a reflectivity value that was not within the range of 24 and 100%. This was determined by interrogating the reflectivity value, or the red value on pixels on the cars that were captured in the dataset to understand the range they all would fit into.

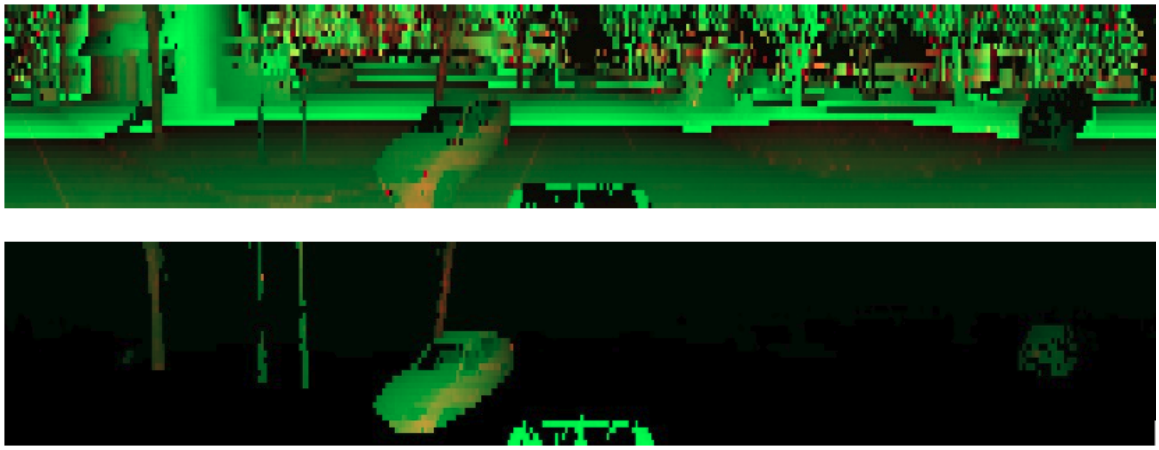

Figure 10: Before and after preprocessing [21].

Preprocessing in this project could also be considered a misnomer in this case, as it is more of a filter that can process points as they are read from the raw packet capture. The term preprocessing, however, was used to keep a consistent naming convention between image processing and LiDAR data processing to reduce confusion.

3.6. CNN Selection

For this project, existing CNNs were evaluated, as this would be an ideal starting point for smart cities that were transitioning from camera-based detection solutions to LiDAR-only. In the case of this experiment, the two networks from the "You Only Look Once," family of CNNs that will be evaluated are YOLOv3-608 and tiny-YOLOv3, two popular CNN architectures used with camera-based images. These two networks were also chosen because they provide a comparison of a high accuracy, lower frame rate

detection and a lowered accuracy, higher frame rate solution. Evaluating such tradeoffs with LiDAR based CNN detection will help to understand how LiDAR performs when compared to the existing studies conducted with camera-based images. This will also later be relevant when understanding the runtime performance on embedded systems and the framerate of LiDAR sensors. Also, YOLOv3-608, when compared with Faster RCNN, another popular and state of the art CNN, performed 70% faster with a minimal drop in precision [10]. This, paired with the online documentation and support made YOLOv3-608 the starting point when selecting a network.

3.7. Training and Evaluation in a Desktop Environment

Training and initial performance evaluation for both YOLOv3-608 and tiny-YOLOv3 was conducted on a desktop computer with an NVIDIA GTX 980 graphics card with 4 Gigabytes of memory. The computer was running Ubuntu 16.04 and is a popular operating system for this development environment. Other relevant specifications can be found in Table 1.

In order to understand how preprocessing might have affected the training results, both CNNs were trained and evaluated using pre-processed and raw LiDAR frames. The difference in the precision results will determine whether the preprocessing step was necessary, or if it can be bypassed in future development.

Component	Model
Operating System	Ubuntu 16.04
CPU	Intel 3820
RAM	16 GB DDR3
GPU	NVIDIA GTX 980 (4GB)

Table 1: Development Computer Specifications [22].

3.8. Embedded System Evaluation

In order to understand the runtime performance of this CNN solution, it was also important to evaluate the performance on a couple of embedded platforms to see how this detection method performs at the edge for real-time processing. The accuracy is not expected to change between devices, however, the runtime and maximum achievable frame rate will. This is due to the varying computational resources among different platforms. For this experiment, the NVIDIA Jetson Nano and Xavier devices were evaluated.

The Jetson product family targets a range of low-power consumption embedded computing platforms that can be used for deep-learning based workloads. These are systems that can be ideal target systems for smart city applications such as intersection monitoring, or other real time computing needs, where real-time inference is required in order to make decisions for pedestrian or vehicle safety. More detailed specifications of the sensors can be seen in Table 2.

For this experiment, both the Jetson Nano and Xavier devices will be used to understand the runtime performance of both YOLOv3-608 and tiny-YOLOv3 to

understand if the produced framerate can process the sensor stream in near real-time. The LiDAR sensor used for this test was producing frames at about 10 frames per second, and therefore, processing frame rates that can exceed this will be considered successful.

	Desktop (GTX980)	**Nano**	**Xavier**
Memory	4GB	4GB (shared)	32GB (shared)
Cores	2048	128	512
GFLOPS (64 double precision)	155.6	7.368	705.0

Table 2: Comparison of the embedded platform specs and the development desktop environment for reference [22].

4. RESULTS

After defining the method and experiment for this project, the results were collected from the training and evaluation phases, as well as the embedded system performance.

4.1. YOLOv3 and Tiny-YOLOv3 Performance

Initially, YOLOv3-608 and tiny-YOLOv3 were trained using the preprocessing method for both training and validation. This yielded promising results where both YOLOv3-608 and tiny-YOLOv3 were able to successfully detect objects. A sample of a vehicle detected can be seen in Figure 11, where a van driving towards the right side of the image is bounded with a prediction.

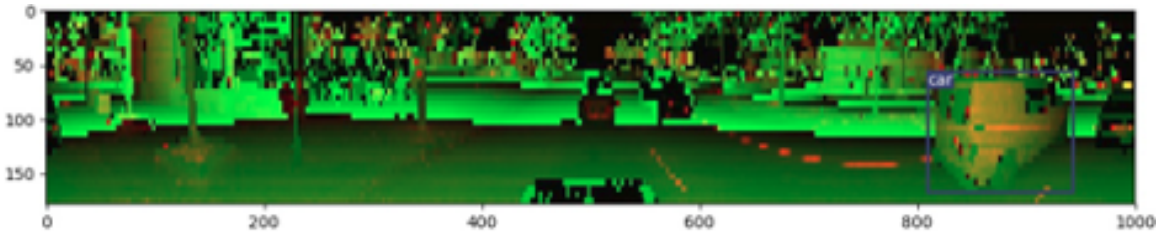

Figure 11: Detected vehicle in LiDAR data [21].

The same training and validation were also performed without the preprocessing step and resulted in a similar outcome. This was rather unexpected as, intuitively, it was assumed that preprocessing would help enhance the performance of the network. This was, however, not the case as the mean Average Precision only improved both tiny-YOLOv3 and YOLOv3-608 by only 0.007 or 0.7%. These results can be seen in Table 3.

	YOLOv3-tiny	YOLOv3-608
Training Time	0.2 hours (16 batches)	2 hours (2 batches)
mAP (0.5 IOU) PREPROCESSED	0.477	0.480
mAP (0.5 IOU) RAW	0.470	0.473

Table 3: Mean Average Precision results for preprocessed and raw frames on two different CNNs [22].

While these results show that preprocessing did not have a significant result in the end precision value, there was some impact on the training performance. The mean Average Precision was plotted over each epoch to understand how it was improving as the network trained, and based on the graphs in Figure 12, the mAP value converged faster in both preprocessed training sessions.

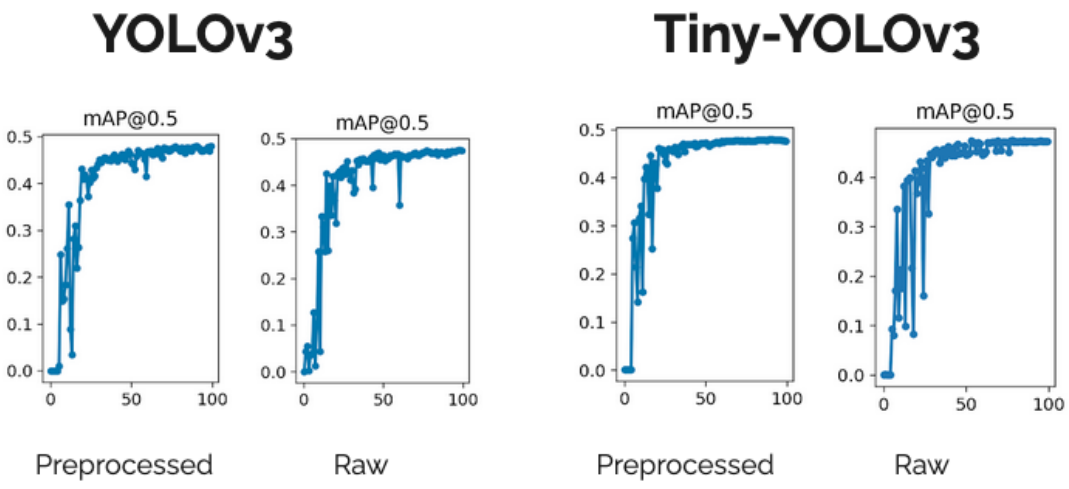

Figure 12: Mean Average Precision values plotted over epochs for Preprocessed and Raw training sessions for YOLOv3-608 and tiny-YOLOv3 [22][5].

Another result worth noting was that the precision drop between tiny-YOLOv3 and YOLOv3-608 was minimal, and only about 0.3% in both cases when moving to the smaller less accurate network. This is significant because when the same test was conducted with a camera image-based dataset, the drop was far more significant, and resulted in a loss of 0.248 or 24.8% [10].

4.2. Embedded System Runtime Performance

After collecting the precision results, the runtime was evaluated on the embedded systems. As expected, the lower-end Jetson Nano performed significantly slower, with 2 frames per second when running YOLOv3-608 and 16 frames per second when running tiny-YOLOv3. When running the same test on the Jetson Xavier, the performance was significantly higher with 16 frames per second when running YOLOv3-608 and 30 frames per second with tiny-YOLOv3. These results, as well as the desktop used for development can be seen in Table 4.

	Desktop w/ GTX980 GPU	**Jetson Nano**	**Jetson Xavier**
YOLOv3 FPS	9fps	2fps	16fps
Tiny FPS	20fps	16fps	30fps

Table 4: Achieved frames-per-second on three different devices [22].

5. CONCLUSION

After reviewing the results, it is clear that object detection with LiDAR data using existing CNNs is possible. This method has also shown promise when compared with cameras as it had a higher resilience to a precision drop when choosing a scaled down network. This was seen in the comparison between YOLOv3-608 and tiny-YOLOv3 when trained on either preprocessed or raw frames. The camera suffered a drop in precision of 24.8%, where the LiDAR only lost about 0.3%. Also, with the use of the Jetson embedded platforms, detection is possible in near-real time with either the full YOLOv3-608 or tiny-YOLOv3 on either embedded platform, again, without losing significant precision.

These results show that the shift to LiDAR-only solutions in smart cities can be done with some new benefits. As cities are switching from camera solutions due to privacy concerns, cities and companies can take advantage of the inherent benefits of LiDAR based object detection without having to re-develop entirely new CNNs. Leveraging existing camera based CNNs can be possible, and through the results shown in this research, loses less precision with smaller CNNs on lower powered edge devices. These results can also be achieved in near-real time to the sensor output of about 10 frames per second, a critical requirement in real time applications where pedestrian or vehicle safety is at stake.

The preprocessing method explored in this paper was also an interesting discovery. The use of preprocessing did not have an effect on the CNN's final precision, however, it affected the number of epochs needed for the neural network to converge at a

stable value. Since the preprocessing used in this research was primarily a filter used at the time of creating the LiDAR frames, there was no significant impact to runtime with or without it for this application. This is different when compared with current camera based inference solutions where a network trained to detect apples was able to achieve a 2.2% increase in precision using YOLOv3, compared to the 0.7% improvement noticed with the LiDAR preprocessing method used in this research [1].

6. FUTURE WORK

Next steps for LiDAR inference in smart cities should include real world testing with multiple different sensors and different object types. As new sensors are developed, understanding how sensors with different underlying technologies compare against each other would make a more comprehensive study. Detecting different types of objects of varying sizes would also help to narrow down the limitations of LiDAR sensors, as well as identify where different sensors perform better than others.

Real world testing can also include increased understanding of what is required for a LiDAR dataset to effectively train a CNN. LiDAR sensors are inherently not affected by visible light the same way cameras are, which could reduce the amount of training data required, as there may be fewer environmental situations to account for. This could significantly reduce development time, as researchers and engineers would be able to spend less time collecting and labeling sample data.

Since LiDAR-based inference is still a relatively new topic, there are still many areas for improvement and exploration. It is clear that further testing and experimentation is required as the sensors and processing capabilities develop over time.

REFERENCES

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[2] A. Kuznetsova, T. Maleva, and V. Soloviev, "Using YOLOv3 Algorithm with Pre- and Post-Processing for Apple Detection in Fruit-Harvesting Robot," *Agronomy*, vol. 10, no. 7, p. 1016, 2020.

[3] B. Li, "3D fully convolutional network for vehicle detection in point cloud," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[4] C. Kiemle, A. Schafler, M. Wirth, A. Fix, and S. Rahm, "Airborne lidar observations of water vapor transport," *2012 IEEE International Geoscience and Remote Sensing Symposium*, 2012.

[5] G. Jocher, et al., "ultralytics/yolov3: 43.1mAP@0.5:0.95 on COCO2014," 2020.

[6] H. Kim, B. Liu, and H. Myung, "Road-feature extraction using point cloud and 3D LiDAR sensor for vehicle localization," *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2017.

[7] I. Z. Ibragimov and I. M. Afanasyev, "Comparison of ROS-based visual SLAM methods in homogeneous indoor environment," *2017 14th Workshop on Positioning, Navigation and Communications (WPNC)*, 2017.

[8] J. Chen, P. Ye, and Z. Sun, "Pedestrian Detection and Tracking Based on 2D Lidar," *2019 6th International Conference on Systems and Informatics (ICSAI)*, 2019.

[9] J. Liu, Q. Sun, Z. Fan, and Y. Jia, "TOF Lidar Development in Autonomous Vehicle," *2018 IEEE 3rd Optoelectronics Global Conference (OGC)*, 2018.

[10] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[11] K. Banerjee, D. Notz, J. Windelen, S. Gavarraju, and M. He, "Online Camera LiDAR Fusion and Object Detection on Hybrid Data for Autonomous Driving," *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[12] M. K. Hsu, K. Byrd, T. N. Lee, C. Hsu, and H. Szu, "Dual IR spectral video inspection of a concealed live animal," *2008 37th IEEE Applied Imagery Pattern Recognition Workshop*, 2008.

[13] M. Velas, M. Spanel, M. Hradis, and A. Herout, "CNN for very fast ground segmentation in velodyne LiDAR data," *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2018.

[14] *Online LIDAR point cloud viewer*, 03-Sep-2018. [Online]. Available: http://lidarview.com/.

[15] *OpenCV*. 2015.

[16] *OpenFrameworks*.

[17] T. Grey, "How lidar powers smart cities and protects privacy," *Ouster*, 01-May-2020.

[18] *USER'S MANUAL AND PROGRAMMING GUIDE*. Velodyne.

[19] "Velodyne Lidar Adds the 'Smart' to Blue City Technology's Smart City Solution," *Business Wire*, 17-Jul-2020. [Online]. [Accessed: 2020].
[20] "VeloView," *ParaView*.
[21] V. Ponnaganti, M. Moh, and T.-S. Moh. (2020) Deep Learning for LiDAR-based Autonomous Vehicles in Smart Cities. Accepted to appear in: Augusto J. (eds) Handbook of Smart Cities. Springer, Cham.
[22] V. Ponnaganti, M. Moh, T.-S. Moh, "Utilizing CNNs for Object Detection with LiDAR Data for Autonomous Driving," (Submitted).
[23] Y. Kunisada, T. Yamashita, and H. Fujiyoshi, "Pedestrian-Detection Method based on 1D-CNN during LiDAR Rotation," *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.
[24] Y. Tatebe, D. Deguchi, Y. Kawanishi, I. Ide, H. Murase, and U. Sakai, "Pedestrian detection from sparse point-cloud using 3DCNN," *2018 International Workshop on Advanced Image Technology (IWAIT)*, 2018.