Fast Intent Prediction of Multi-Cyclists in 3D Point Cloud Data using Deep Neural Networks

Khaled Saleh[a,1], Ahmed Abobakr[b], Mohammed Hossny[c], Darius Nahavandi[c], Julie Iskander[d], Mohammed Attia[e], Saeid Nahavandi[c]

[a]*Data Science Institute, University of Technology Sydney (UTS), NSW, Australia*
[b]*Faculty of Computers and Information, Cairo University, Egypt*
[c]*Institute for Intelligent Systems Research and Innovation, Deakin University, VIC, Australia*
[d]*Walter and Eliza Hall Institute of Medical Research, VIC, Australia*
[e]*Medical Research Institute, Alexandria University, Egypt*

Abstract

Inferring the intended actions of road-sharing users with autonomous ground vehicles in particularly vulnerable ones like cyclists is considered one of the tough tasks facing the wide-spread deployment of autonomous ground vehicles. One of the main reasons for that is the scarcity of the available datasets for that task due to the difficulty in obtaining those datasets in real environments. In this work, we first propose a pipeline that can synthetically produce 3D LiDAR data of cyclists hand-signalling a set of intended actions that are commonly done in real environments. Given the synthetically-produced labelled 3D LiDAR data sequences, we trained a framework that can simultaneously detect, track and give predictions about the intended actions of multi-cyclists in the scene on time. The proposed framework was evaluated using both synthetic and real data from a physical 3D LiDAR sensor. Our proposed framework has scored competitive and robust results in both synthetic and real environments with 88% in F_1 measure with higher frame per second rate (12.9 FPS) than the 3D LiDAR sensor frame rate (10 Hz).

Keywords: cyclist, intent, LiDAR, neural networks, autonomous vehicles

[1]Corresponding author: email address: khaled.aboufarw@uts.edu.au

1. Introduction

Recently, autonomous ground vehicles (AGVs) have made great strides in many applications, ranging from self-driving taxis to last mile delivery vehicles [1]. Since these vehicles will be interacting and/or dealing with humans on a daily-basis, it has become a necessity for them to have the capability of understanding and predicting the humans' behaviours and intentions specially the vulnerable ones such as pedestrians and cyclists. One example of such behaviours that are commonly happening in traffic environments is when pedestrians are walking on a curbside and want to cross the road, usually they make eye-contact and/or head movement just to make sure the drivers in oncoming vehicles acknowledge their intentions. Another example is when cyclists are intending to make a left/right turn in front of a vehicle, they often rely on hand signals to notify the driver behind them of their intentions. Such simple hand and body gestures are easily comprehended by most motorists but AGVs on the other hand are still facing some challenges with them [2, 3, 4]. As a result, in the literature the problem of understanding and predicting vulnerable road users (VRUs) (e.g pedestrians and cyclists) intentions in the context of AGVs has got some momentum over the past few years. Given the fact that the VRUs intention is a latent variable that can not be easily observed, in the literature different VRUs attributes were explored as a proxy to their intention. For example, the gait/postures of pedestrians as well as their past trajectories were utilised for pedestrians intent prediction [5, 6]. Whereas for cyclists, their hand signals were shown to have a strong correlation with their intentions [7]. That being said, the majority of the work however that have been done on the VRUs intent prediction problem was focused mainly on the pedestrians [8, 9, 10, 11]. One of the main reasons for that is due to the relatively larger number of available public datasets of pedestrians when compared to cyclists. One promising solution for this problem of scarce datasets is the utilisation of high-fidelity simulation frameworks [12, 13]. Using such simulators in conjunction with deep representation learning-based approaches, was shown to be effective and can be generalised to real scenarios in many tasks such as: autonomous driving [14], scene parsing [15] and semantic segmentation [16].

Given that, in this work we will adopt a similar approach for the cyclist intent pre-

diction problem. More specifically, we will be relying on the hand-signalling that is frequently carried out by cyclists on the road in order to predict their intended actions. In order to overcome the scarcity of data for cyclists, we propose a novel pipeline that generates synthetic 3D LiDAR scans of cyclists in different traffic environments carrying out different realistic behaviours. Using the generated dataset, a novel framework is proposed to simultaneously detect, track and predict the intentions of cyclists in both the generated simulated scenes and real physical scenes. As part of the proposed framework, one of state-of-the-art deep learning-based models for point cloud data processing will be adopted. The main benefit of using a deep-learning based approach over other traditional geometry-based approaches, is to overcome the challenge of hand-crafting features from the point cloud data which is both a time-consuming task and do not scale well to different scenarios. The rationale behind generating synthetic 3D LiDAR scans rather than synthetic RGB is two folds:

- the robustness of the 3D LiDAR sensors that exists in most of the current AGVs that can detect up to 200 meters with 360 degree field of view independently of light conditions and severe weather conditions unlike visual/IR cameras.
- the similarity between the generated point cloud scans from simulated 3D LiDARs and real 3D LiDAR sensors which is not greatly effected by the domain-shift problem [17] that is quite prevalent between real and simulated RGB cameras.

This work extends and builds up on our preliminary results published in [18]. The contribution of this work in comparison to our early work can be summarised as follows:

- Taking into consideration the case of multi-cyclists in the scene by proposing an integrated fast 3D cyclist tracking model.
- Extensive analysis of the performance of the proposed 3D cyclist tracking model in comparison to other baseline approaches from the literature.
- Evaluating the proposed framework that was trained entirely using synthetic

point cloud data only on real point cloud data collected from realistic traffic environment using a physical 3D LiDAR sensor.

The rest of the paper is organised as follows. An overview of the related work from the literature will be briefly discussed in Section 2. In Section 3, the proposed approach and methodology will be covered. The experimental results and the performance of the proposed approach will be provided in Section 4. Section 5, concludes our paper.

2. Related Work

In the literature, the intent prediction problem of VRUs is commonly formulated as a trajectory prediction problem [9, 19, 20]. Since most of these works were mainly focused on pedestrians, thus the few work that has been done on the cyclist intent prediction problem was following the same paradigm. Most recently, Meijer et al. [21], proposed an approach for the cyclist intent prediction problem based on the kinematics of the bicycle itself. They fitted a bicycle with a number of inertial measurement units (IMU) to calculate its kinematics when a cyclist was riding it during manoeuvring in a controlled traffic environment. The signals they were relying on for their kinematics-based cyclist intent prediction model were as follows: steering angle, roll angle, velocity, wheel speed, peddling frequency and acceleration. With the help of these signals, they feed it to a graphical hidden Markov model to predict the probability of a cyclist will continue cycling in straight path or will take a left/right turn. Given the invasive nature of their approach, their proposed model would be challenged when tested in real traffic scenarios as it would be hard to acquire the aforementioned signals of the bicycle dynamics. Similarly, another trajectory prediction approach for cyclists was introduced in [22]. In their model, they relied on non-invasive vehicle-based sensors such as stereo camera unlike [21]. They observe the past trajectory of a cyclist over a window of time, then using a motion-dynamics-based model, they predict the intended trajectory direction of the cyclist. Two motion dynamics were proposed in their approach based on Kalman filter (KF). The first model was a typical constant velocity linear KF model. Whereas, the other model was a mixture of switching KF

models. They were switching between five different motion models based on a priori regarding the trajectory direction of the cyclist.

In another related work, Benedek et al. [23], introduced an action recognition model for pedestrians' action prediction from a 3D LiDAR sensor for surveillance applications. In their model, they relied on set of extracted features from the point cloud scans coming from the 3D LiDAR sensor to classify two actions; bending and waving. Then, they feed the extracted features to a simple shallow convolutional neural network (ConvNet) model that consists of 4 layers. Over the past few years, deep learning has been making huge strides in a number of perception tasks [24, 25]. More recently, deep-learning based approaches have been also achieving promising results in a number of 3D computer vision tasks in comparison to the pure traditional geometry-based approaches [26, 27, 28]. One of the main backbone architectures for many deep learning models for 3D computer vision tasks is the PointNet architecture [26]. PointNet is one of the earliest architectures that were dealing with raw point cloud data processing directly without any prior information from other sensor modalities or geometrical requirements. It learns representations from point cloud data automatically in an end-to-end fashion on a different levels of abstractions. PointNet has been upgraded to PointNet++ [28], which has achieved quite resilient results in challenging tasks such as 3D semantic scene segmentation. This upgrade was to overcome the inability of the original PointNet architecture to capture the local structure within the point cloud data. As a result, this enabled PointNet++ to learn the structure and patterns of the point cloud data of complex scenes. In the PointNet++ architecture, it relies on the combined (sampling, grouping and pointnet) layers which is referred to as the "set abstraction" (SA) module. In the sampling layer, it selects the subset of points from the input point cloud data which represents centroids of local neighbour regions. Then, the grouping layer clusters the input point cloud data based on the K-nearest neighbour points. Lastly, the pointnet layer encompasses a multi-layer perceptron (MLP) network for embedding every single point from the input 3D scan data, followed by a maximum-pooling layer to extract a universal vector of features from the input 3D scan data. Additionally, a set of consecutive SA modules are hierarchically repeated

inside PointNet++ on a number of levels that can effectively encode the input point cloud data. At each SA level, larger local regions of the point cloud are abstracted to produce lower number of points. The hierarchy of SA levels can be viewed as the feature pyramid networks that are commonly used with 2D computer vision tasks using ConvNets [29]. Finally, interpolation layers are added at the end of the SA modules in order to decode the down-scaled input 3D scan data to its first input shape. For more detailed description of the interpolation layers, it can be found in [28]

Another architecture that has been successfully applied in 3D perception tasks is the VoxelNet architecture which is mainly targeted for the task of 3D object detection. It consists of two simultaneous stages; features extraction for object proposals and 3D bounding box regression which are trained jointly. As the name implies, VoxelNet, firstly segments the input point cloud into voxel chunks and for each chunk, a set of discriminative voxel feature encoding (VFE) layers are trained in an end-to-end style.

3. Proposed Method

In this section we will discuss the details of the proposed method we pursued to tackle the multi-cyclists intended action prediction task. The first subsection will be covering the procedure we followed to overcome the scarcity of data available for the cyclist intent prediction problem. Then, our novel framework for joint detection, tracking and intent prediction of multi-cyclists from 3D LiDAR sensors' point cloud data in urban traffic environment will be introduced.

3.1. Cyclist Hand Signals Data Generation

The number of available datasets for cyclists' intent prediction is almost non-existent. Thus, for overcoming the scarcity of data, we use a data generation pipeline similar to the ones employed by previous works on posture analysis [30] and VRUs detection [31, 32]. This data-generation pipeline was shown to be effective in tackling these tasks specially when coupled with powerful end-to-end approaches based on deep representation learning which provides good generalisation capabilities when tested on real data in physical environments. The first part of the data generation pipeline starts

Figure 1: The motion capture (MoCap) stage of our data-generation pipeline.

with the motion capture (MoCap) of real motion behaviours of cyclists hand signalling their intended actions. We collected the MoCap data using an XSENS MoCap system from four volunteer cyclist subjects on a stationary cyclist inside a GYM (as shown in Figure 1).

We instructed the volunteer subjects to perform a set of three distinctive hand signals while they are cycling, namely left turn (LTRN), right turn (RTRN) and stopping (STOP) while recording their MoCap data (roughly one minute per each signal). Additionally, we sampled from the recorded MoCap data, sequences where the cyclist were not doing any action and we labelled it as an extra signal and we refer to it as the no-action signal (NACT). The four hand signal actions are shown in Figure 2 (starting from left actions are: LTRN, NACT, STOP and RTRN respectively). As a result, we obtained a total of 16 MoCap sequences representing 4 signals for each volunteer subject.

In the second part of the data generation pipeline, we map the resultant MoCap data from the first part to animate simulated cyclists inside simulated traffic environments

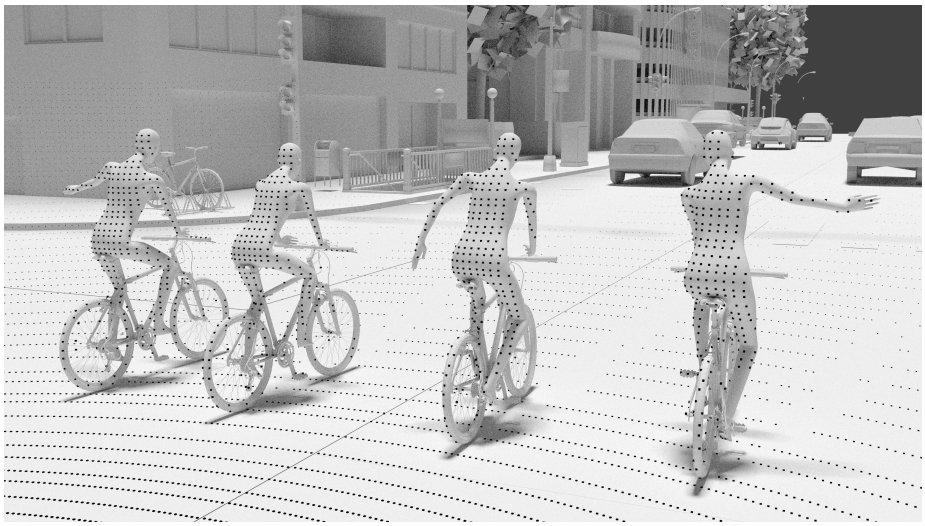


Figure 2: The four hand signals we generated using our pipeline. Starting from left actions are: LTRN, NACT, STOP and RTRN respectively.

Table 1

Distribution (mean$\pm$*std*) of the anthropometric measures of the animated virtual cyclist models used in generating the multi-cyclist synthetic point cloud data.

3D Cyclist Models	Height	Weight	Body Mass Index (BMI)	Body Surface Area (BSA)
Females	158.98$\pm$6.73	50.29$\pm$9.8	19.81$\pm$3.19	1.47$\pm$0.15
Males	173.06$\pm$7.16	70.9$\pm$13.09	23.58$\pm$3.59	1.81$\pm$0.18

designed using the sensor simulator toolbox, Blensor [12]. Blensor is an add-on to the open source 3D computer graphics software, Blender. Inside Blensor a number of simulated 3D LiDAR sensors are modelled such as the Velodyne HDL-64E/32E models. For the virtual animated cyclist models, we made sure that we take into account the different anthropometric measures of cyclists to emulate the real different body shapes of cyclists in real traffic environments. In table 1, we show the diverse set of anthropometric measures we utilised in our data generation pipeline.

In order to generate point cloud data of the cyclists hand signals, we rendered the photo-realistic simulated environment using a simulated 3D LiDAR sensor (Velodyne with 64 laser channels). The cyclists were captured using the simulated Velodyne sen-

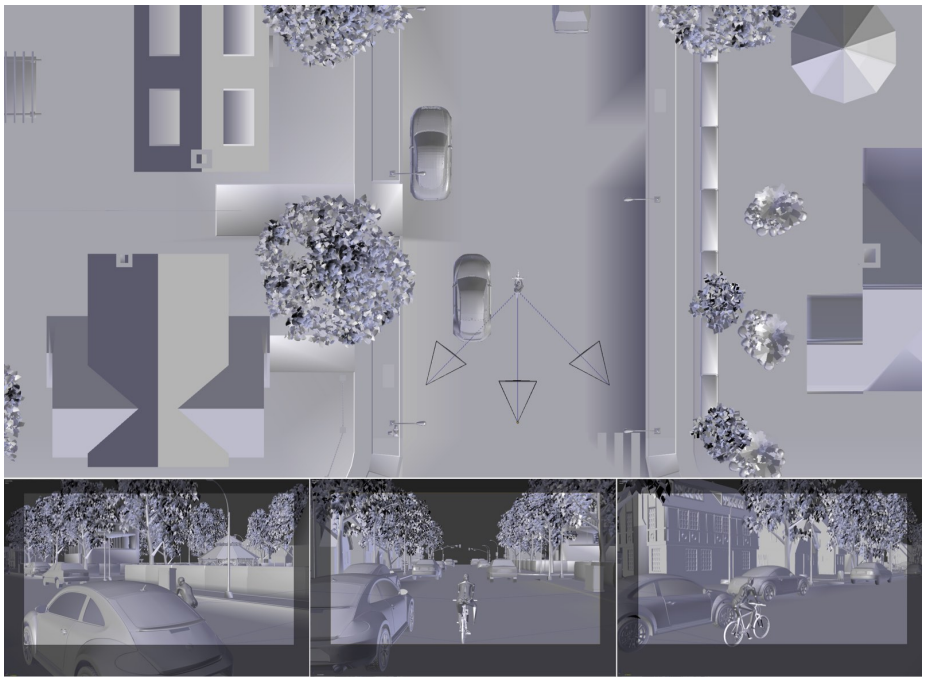


Figure 3: Pipeline for synthetic point cloud data generation. Virtual cyclists are placed in simulated traffic scenes and animated using MoCap data. The scene is then rendered using simulated 3D range sensor (Velodyne HDL-64E)

sor on a variable distance ranging from 5 to 20 meters. Additionally, the scenes covered four different blocks within a simulated urban traffic city with moderate traffic as shown in Figure 3. The complexity of the scenarios in the generated dataset simulated the ones involving cyclist instances in the KITTI dataset [33]. The following is some statistics to reflect the complexity of the simulated four-blocks city. The total number of vehicles in each block were around 100 vehicles, the number of buildings were around 45 buildings and the number of cyclists were at least 8 cyclists per each scene.

3.2. Multi-Cyclist Tracking

Since the hand signals of cyclists commonly take no less than a second, thus the necessity to observe and track the cyclists over this window of time to predict their intentions is inevitable. Motivated by the 'tracking-by-detection' paradigm that has achieved state-of-the-art results for many 2D multi-object tracking (MOT) benchmarks. This

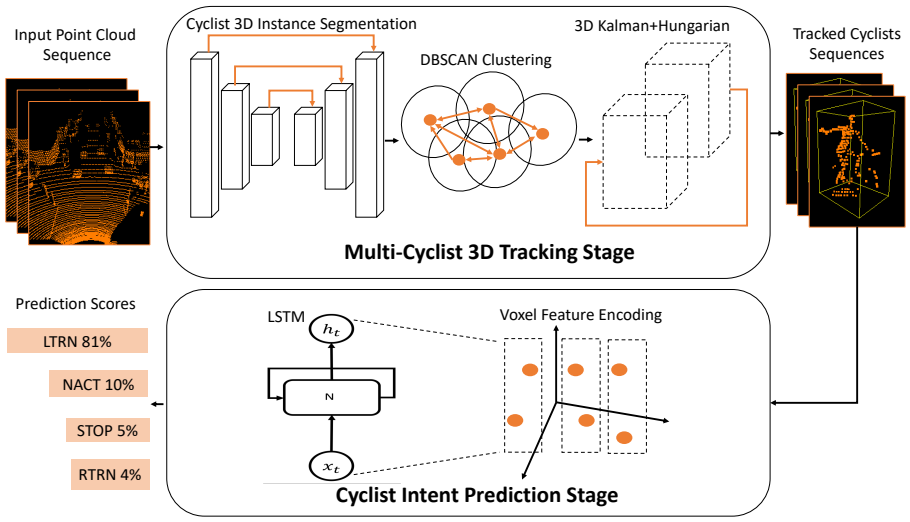


Figure 4: Our framework for the multi-cyclists intended action prediction task. The input to the framework is a short sequence of 3D laser scan data. The framework in returns, track all 3D detected bounding boxes of cyclists in the scene. Then, for each tracked cyclists a classification score probability is provided based on four intended action classes (namely LTRN, NACT, STOP and RTRN).

paradigm has recently been also extended for 3D MOT [34] and it continued to provide competitive results. Given that, we will utilise a modified version of the AB3DMOT baseline [34] which was ranked the second place in the KITTI MOT benchmark leaderboard. The advantage of AB3DMOT baseline in our case is that it provides resilient performance with super fast real-time performance of more than 200 FPS on CPU. Similar to the 'tracking-by-detection' paradigm in 2D MOT, the AB3DMOT baseline also relies on prior 3D bounding boxes of objects of interest in LiDAR point cloud data. Then using a simple state estimation via 3D Kalman filter and data association via a Hungarian algorithm [35], it can achieve resilient real-time results on tough 3D MOT benchmarks like KITTI. This will be the first stage (multi-cyclist 3D tracking stage as shown in Figure 4) of our proposed framework for the multi-cyclists intent prediction problem.

3D Detection using PointNet++ and Clustering

In the original AB3DMOT baseline, they relied on the PointRCNN model [36] for the 3D bounding boxes detections of objects in 3D LiDAR point cloud data. The real time performance of the PointRCNN model was reported to be a GPU-hungry specially during the inference phase with 0.2 second. Because the whole multi-cyclist tracking stage is just a subset of our approach for the multi-cyclists intent prediction problem, thus we can not afford the expensive computational requirements for the PointRCNN model. As a result, we are proposing a different approach for obtaining the 3D bounding boxes of the cyclists. Our approach is based on the PointNet architecture. Given an input point cloud data of cyclist instances, we train an improved version of the PointNet architecture called, PointNet++ [28] (which was described in Section 2) for per-point cyclist instance segmentation. The output of our PointNet++ model is a binary class labels prediction (cyclist or background) per each single point in the input 3D scan data. Subsequently, we extract all the points that corresponds to the cyclist class and we further cluster them into unique groups using the DBSCAN clustering algorithm [37] in order to discard any false-positive predicted cyclist instances. The rationale behind using DBSCAN over other classic clustering algorithms such as k-means, is the fact that DBSCAN does not require the assumed known number of clusters beforehand as it is the case with K-means. In our use-case scenarios, there could be an unknown number of cyclists in the traffic scene, so using K-means in our case would not be a viable solution. Additionally, other classic clustering techniques such as K-means cannot solve the clustering of irregular/arbitrary shapes such as point cloud of cyclists. So, density-based DBSCAN was developed to systematically solve this problem as it was shown in prior LiDAR point cloud data processing works in the literature [38, 39, 40].

The minimum number of points for the DBSCAN was set to 10 points within a radius value (ε) of 0.4. Based on the clustered cyclist points, we fit a 3D oriented bounding box (bbox) for each clustered groups in order to get the 3D bboxes of all cyclists in the input 3D scan data. The fitting procedure for the oriented bbox was based on the technique introduced in [41], which computes a covariance matrix for each clustered point set and then find the eigenvectors of their covariance matrices. Finally, each detected

cyclist 3D bbox is defined using the tuple $(x, y, z, l, w, h, \theta)$ to represent the location co-ordinates for the 3D bbox, length, height and rotation angle of 3D box around y-axis respectively.

Estimation using 3D Kalman Filter

Similar to the AB3DMOT baseline, we rely on a linear constant-velocity Kalman filter that estimates the next-frame state of each detected 3D bbox of cyclists in the 3D world. The state $\mathbf{x}$ of each cyclist object is parametrised using the following:

$$\mathbf{x} = [x, y, z, \theta, l, w, h, \dot{x}, \dot{y}, \dot{z}]^T \tag{1}$$

where $\dot{x}, \dot{y}, \dot{z}$ are the linear velocity of the cyclist in the 3D space. Given all the observed 3D bboxes from the previous frame at time T_{t-1}, the state of each i 3D bbox T^i_{pred} is predicted in the next frame. In order to associate the observed 3D bboxes to the predicted 3D bboxes, a data-association technique based on the Hungarian algorithm [35] is utilised. For each pair from the observed and the predicted 3D bboxes, an 3D intersection over union (IoU) is calculated to check if they are matched or not. As a result, we get unique IDs for each cyclist in the input cloud data over time.

3.3. Cyclist Intent Prediction Model

Given a sequence of tracked cyclists in the scene over a period of time t_{obs} from the multi-cyclist tracking stage, we subsequently feed them to our cyclist intent prediction model (as shown in Figure 4). The spatio-temporal nature of the problem is a crucial property that needs to be considered in the proposed model. The spatio-temporal information is represented by the set of tracked points for each cyclist over a period of time. The spatial information (that is not uniform in terms of number of points across the observed time t_{obs} due to the change in the distance to object) needs to be encoded in a way that can preserve the unique geometrical shape of the different cyclists' hand signals so that it could be easily predicted. Additionally, the temporal dependency between consecutive frames of the performed hand signal need to be exploited so that hand signals such as LTRN and STOP do not get confused with each others. To this end, we are proposing the Voxel-LSTM model for the multi-cyclists intended action

prediction problem. The Voxel-LSTM model consists of two main components; the voxel feature encoding (VFE) layers and the LSTM layers. VFE was firstly introduced in the VoxelNet model for 3D object detection [27]. VFE can automatically differentiate shape-like information from input point cloud data of cyclists. Moreover, the VFE layer acts as an encoder of the shape area in the input voxels of cyclists. It performs a per-point aggregation operation over the extracted features via a fully connected network (FCN). On the other hand, the LSTM layer is one variant of the recurrent neural networks (RNN) which models the temporal dependency in the dynamics of the performed hand signals by the cyclists. More formally, using V which is the input voxel, our proposed model starts to sample an N fixed number of points from it. The reason for that is to eliminate probable problems such as imbalanced density of the voxels as well as minimising the complexity. Afterwards, each voxel is augmented with an offset distance from the centroid of its neighbour cyclist voxel. As a result, voxel V is transformed into $V_f = \{p_i = [x_i,\ y_i,\ z_i,\ x_i - \bar{x},\ y_i - \bar{y},\ z_i - \bar{z}\ ]^T \in \mathbb{R}^6\}_{i=1}^{N}$. Where x, y, z represents the 3D coordinates of i-th point within the voxel, and $\bar{x}, \bar{y}, \bar{z}$ are the local centroid coordinates. Then, for each p_i pointwise features $f_i \in \mathbb{R}^k$ are extracted via FCN. Each FCN consists of three consecutive layers (namely a fully connected layer (FC), a batch normalisation layer (BN) and a rectified linear unit (ReLU) activation layer. Then, using a 1D max-pooling operation, the set of features output from the preceding FCN operation are aggregated locally throughout each f_i. Each f_i are then concatenated with their corresponding local aggregated features. Next, in order to model the temporal dependency between consecutive voxel features of each cyclist a set of stacked LSTM layers are utilised. Lastly, the Voxel-LSTM model is ended with an FC layer with four hidden units that corresponds to the four predicted intended actions.

4. Experimental Results

We will firstly describe the process we adopted for training the two stages of our unified framework. Afterwards, we will analyse and evaluate the performance of our framework against both the synthetically generated point cloud data and real collected

point cloud data.

4.1. Training PointNet++ for the Multi-Cyclist Tracking Stage

In our multi-cyclists tracking stage, it is following the tracking by detection paradigm as described in Section 3.2, where we will be relying mainly on 3D detections from a combination of trained PointNet++ model and the DBSCAN clustering. In order to train our PointNet++ model, we generated 10K labelled 360° point cloud scans of urban traffic scenes with cyclists instances according to the procedure discussed in Section 3.1. As a pre-processing step for the point cloud data before training, we crop the generated point cloud data into fixed-size cropped boxes from the full generated point cloud scan. The size of the crops is +/-30 meters (front/behind the 3D LiDAR) by +/-10 meters (right/left to the 3D LiDAR). The reason for this pre-processing step is because in typical real traffic scenarios, we would only be interested in the actions of the cyclists who are within this range as recommended by the designers of collision warning/avoidance systems [42]. Additionally, we augmented the generated synthetic point cloud scan with random voxel down-sampling, Gaussian noise, rotation and translation. The augmentation is required to help in both introducing some of the artefacts exists in real point cloud scans and increasing the dataset size which in returns improves the generalisation capabilities of the trained model. In our cyclist instance segmentation PointNet++ model, we used 4 layers of SA modules with 4 corresponding interpolation layers to up-sample the down-sampled points after the SA modules. Our model was trained on a Nvidia Titan X GPU for 500 epochs with the Adam algorithm as our optimiser. We used a learning rate of 0.001 and a batch size of 16 samples.

4.2. Training the Cyclist Intent Prediction Stage

This Voxel-LSTM stage predicts the most probable action that a segmented and tracked cyclist instance will perform. It consists of two main modules: a feature learning module and a temporal dynamics module. The feature learning model encompasses 2 VFE layers; VFE-1(6, 32) and VFE-2(32, 128). This model encodes randomly sampled $N = 150$ points from the cyclist voxel in a dense representation of 128 features. The reason for this specific number of points was because we found that 150 points

is the minimum possible number of points for cyclists within the range of (5-20) meters (as it was outlined in Section 3.1). The temporal analysis module consists of two stacked LSTM layers with 100 hidden units each. It takes as input a sequence of $T = 20$ cyclist voxel features and passes the activations to the final FC layer of 4 classes to predict the intended action.

We trained our Voxel-LSTM model on synthetic point clouds generated using the pipeline presented in Section 3.1. This pipeline generates point-wise labelled point clouds. The dataset contains a total of 3416 actions balanced between the four intended actions classes. The duration for each intended action was capped to 25 consecutive synthetic 3D laser scans. The intended actions were captured in four different simulated environments from 16 virtual cyclist models animated using MoCap data captured from four real volunteer subjects. We used the data of three subjects during the training phase and one subject for validation. We further augmented the training data with translation, rotation and scale transformations to mitigate the effect of overfitting. Then, we randomly sampled 10 sequences of length $T = 20$ consecutive scans from the total 25 scans per each generated intended action sequence to construct the training dataset. Another pre-processing step was done on the generated dataset to make sure that each cyclist voxel is no less than 75 points because otherwise it will not capture any meaningful intended action. As a result, we obtained a total training split data of 80K sequences and a validation split of 1600 sequences. The Voxel-LSTM model was trained for 100 epochs using the Adam optimiser. The hyper-parameters were: an initial learning rate of 0.001, batch size of 16 sequences and weight decay of 0.0005. The training was done on a Nvidia Titan X GPU. The loss function was the cross entropy loss as the objective function for our training.

4.3. Performance of Intent Action Prediction Framework

In order to quantify the performance of our proposed framework for the cyclists intent prediction problem, we will be using three of the evaluation metrics that are commonly utilised for classification problems. These metrics are precision, recall and F_1 measures [44] which are calculated as follow:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

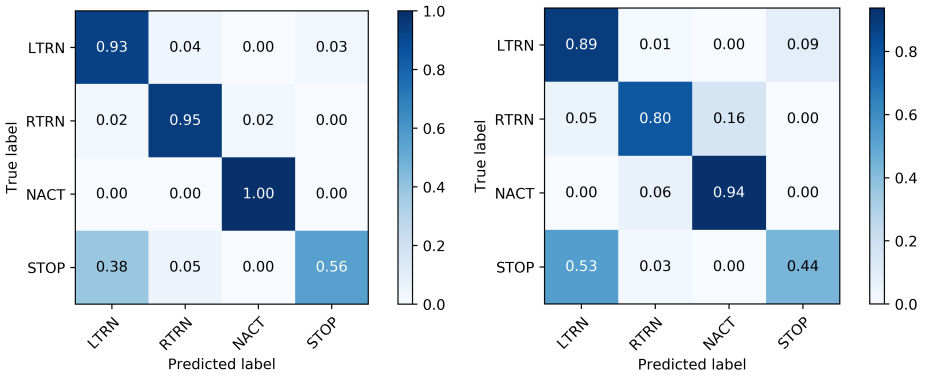
Table 2: Performance evaluation of our proposed framework for multi-cyclists intended action prediction in comparison to two baseline approaches. The Synthetic column represents the testing split of our generated dataset. Whereas the Real column represents the collected point cloud data using a physical Velodyne VLP-32C sensor. Higher is better.

Approach	Synthetic			Real		
	Precision	Recall	F_1-Measure	Precision	Recall	F_1-Measure
FPFH-RF	0.37	0.35	0.32	0.18	0.25	0.15
3D-CNN [43]	0.87	0.84	0.84	0.78	0.74	0.73
Voxel-MLP (ours)	0.94	0.93	0.93	0.80	0.78	0.79
Voxel-LSTM (ours)	**0.99**	**0.99**	**0.99**	**0.89**	**0.88**	**0.88**

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

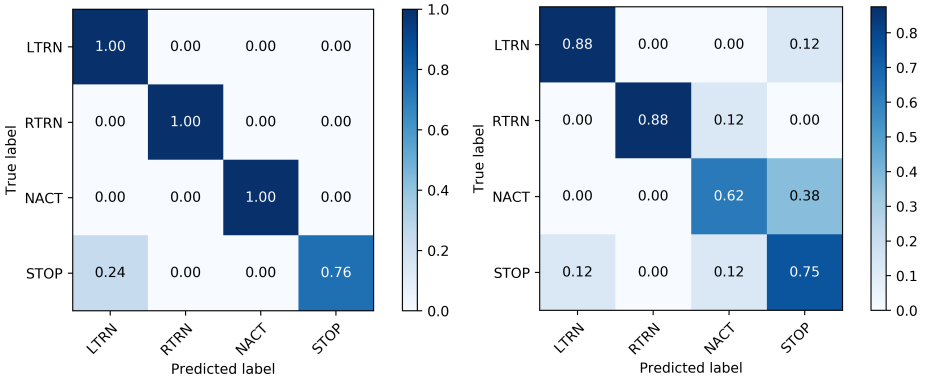
$$F_1 - Measure = 2 * \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

where TP is the true positive counts, FP is the false positive counts and FN is the false negative counts. Since our model was trained entirely on synthetic point cloud data, we will firstly evaluate its performance on the testing split of our synthetic point cloud data generated in Section 3.1. In Table 2, we report the results of the aforementioned three evaluation metrics on the testing split (synthetic part). Moreover, we compare our approach against three different baseline approaches. The first approach is an adapted implementation of the approach proposed in [45, 46]. This approach relies on hand-crafting set of features based on the fast point feature histograms (FPFH) descriptor [47] from the input sequence point cloud data. Given the extracted features, we pass them to a random forest (RF) classifier ensemble [48] and we refer to this approach as "FPFH-RF". The hyper-parameters we used in our experiments for the FPFH are 0.25 for the radius and 50 for the number of the nearest-neighbours points of its KDTree search algorithm. In the RF classifier, the number of trees in the forest was set to 200. The second baseline we are comparing against is another deep-learning based approach that was firstly introduced in [43] for hand gestures recognition from



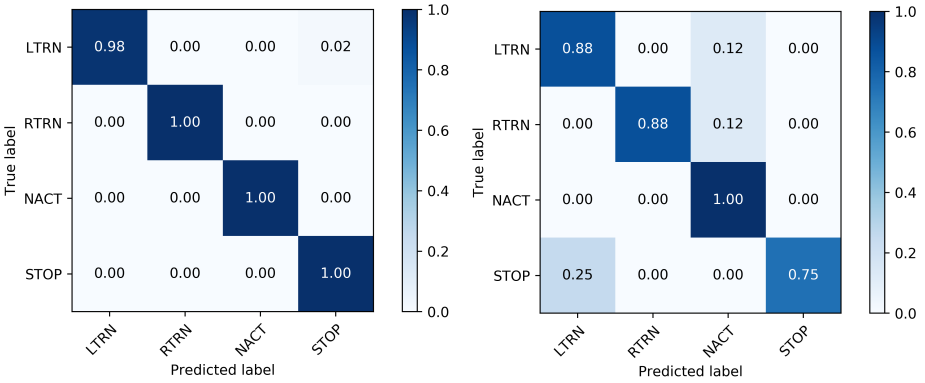
(a) 3D-CNN (Synthetic)

(b) 3D-CNN (Real)



(c) Voxel-MLP (Synthetic)

(d) Voxel-MLP (Real)



(e) Voxel-LSTM (Synthetic)

(f) Voxel-LSTM (Real)

Figure 5: Normalised confusion matrix of proposed Voxel-LSTM model both 3D-CNN and Voxel-MLP models over both the generated synthetic dataset and real point cloud data captured using a physical Velodyne VLP-32C sensor. The first column figures (a, c, e) corresponds to the synthetic data, while the second column figures (b, d, f) corresponds to the real point cloud data.

point cloud data. This approach was relying on 3D convolutional neural network that takes a sequence of point cloud data of hand-gestures as a sequence of 3D occupancy grids of size ($24\times24\times24$). We refer to this approach as "3D-CNN" and we used the same architecture of the model as in [43], which consists of four 3D convolutional layers interleaved with two 3D max pooling layers, two fully connected layers and one output dense layer. The third baseline approach we compared it against our proposed framework is quite similar to our proposed Voxel-LSTM model but we replaced the last two LSTM layers with a multi-layer perceptron model (MLP). We refer to this approach as "Voxel-MLP". In the Voxel-MLP model, we used the same hyper-parameters for the VFE layers as our Voxel-LSTM model while for the MLP we used two hidden layers with 256 neurons for each one. Additionally, we used the ReLU as the activation function for the MLP network. It is worth noting that all the three baseline approaches were fed with the same sequence length of of 20 consecutive point cloud scans of tracked cyclists similar to our Voxel-LSTM model.

As it can be noticed from Table 2, our Voxel-LSTM model achieved the highest scores over the generated point cloud testing split in precision, recall and F_1. The second best model was the Voxel-MLP model and this shows how effective was the choice of LSTM layers which are more capable of capturing the underlying temporal dependency between the tracked cyclists voxels over time unlike the MLP and the 3D-CNN models. From the table, we can also notice how the effectiveness and expressiveness of the automatically extracted features using VFE layers in the Voxel-LSTM and the Voxel-MLP models helped in boosting their performance over both the traditional hand-crafted features such as FPFH and the other automatic ones captured using 3D-CNN. Furthermore, we test and evaluate the generalisation capability of our framework on real point cloud data from a physical Velodyne VLP-32C 3D LiDAR sensor mounted on a vehicle. We collected a total of 100 sequences of point cloud scans from 4 subjects on a bicycle doing the four different hand signals (roughly 1 minute for each signal) described in Section 3.1 while in a low to moderate real urban traffic environment. The sequences were recorded using the 3D LiDAR sensor while the vehicle was both stationary and moving with 50 sequences each. For each sequence, at least two cyclists exist in the scene with intermittent occlusion with other traffic objects (i.e, ve-

hicles, pedestrians). For each hand signal we manually annotated its starting and end

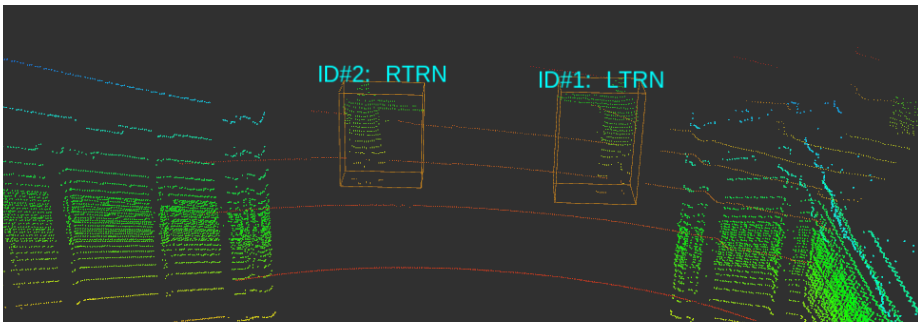


Figure 6: Sample predictions of our proposed multi-cyclist intent prediction framework on real point cloud scan from a physical Velodyne VLP-32C 3D LiDAR.

time in order to quantitatively assess the performance of our proposed framework. As it can be noticed from Table 2 (Real part), our proposed framework has continued to provide a robust prediction on real data (as it can be shown from Figure 6) despite the fact that it was trained using synthetic point cloud data only. In Figure 5, we further analyse the performance of the two proposed models (Voxel-MLP and Voxel-LSTM) against the other deep-learning based model "3D-CNN" using confusion matrices over both the generated synthetic and real point cloud data from the Velodyne VLP-32C 3D LiDAR sensor. As it can be noticed from the confusion matrices, the Voxel-LSTM model continued to show robust results across the four intended actions. The only challenging intended action was the STOP since it is quite similar to the LTRN since they are both involve using the left hand (as it shown in Figure 2). One of the main reason for that, is because of the powerful learned representations and captured using the underlying PointNet, VFE and LSTM layers. The other reason is the data-augmentation procedure we followed which helped in introducing some of the properties exists in real point cloud data which in returns made our trained model able to generalise to unseen real point cloud data. The same also goes for the Voxel-MLP model which was able as well to provide a quite good generalisation capabilities. On the other hand, the FPFH-RF model due to its relying on hand-crafted features which can not be generalised to other unseen real point cloud data, it only achieved an accuracy of 0.25 which is almost a random guessing.

Table 3: Quantitative comparison of the multi-cyclist tracking stage on real collected point cloud data using a Velodyne VLP-32C sensor.

Method	MOTA (%) ↑	MOTP (%) ↑	FPS ↑
AB3DMOT (PointRCNN) [36]	**93.33**	24.89	4.5
AB3DMOT (ours)	89.90	**65.40**	**13.2**

4.4. Effect of Noisy Observations on Multi-Cyclist Tracking

Since our multi-cyclist tracking stage highly relies on the quality of the observations (which are commonly noisy in real data) from the 3D cyclist detection model. Thus, from the collected real data, we manually annotated 32 sequences (each is roughly 1 minute long with frame rate of 10 Hz) with 3D bounding boxes of cyclists in order to assess the impact of the noisy observations from our 3D cyclist detection model. Furthermore, we labelled each cyclist in each sequence with unique ID throughout the sequence. In Table 3, we report a quantitative comparison between our proposed multi-cyclist tracking stage based on AB3DMOT and the baseline AB3DMOT where the only difference between them is the underlying 3D cyclists detection model.

In the baseline AB3DMOT, the 3D detections are based on the PointRCNN model [36], while ours is based on the approach described in Section 3.2. Similar to [34], we utilised the 3D multi-object tracking accuracy (MOTA) and 3D multi-object tracking precision (MOTP) evaluation metrics from the KITTI-3DMOT evaluation tool [34] for quantifying the performance of the multi-cyclist tracking stage. Additionally, we evaluate the run time for our tracking against the baseline AB3DMOT based on the frame per second (FPS) needed for their computations. As it can be noticed from Table 3, our proposed multi-cyclist tracking provided resilient scores in the MOTP metric without compromising the accuracy score of the MOTA metric. Although the baseline AB3DMOT achieved a slightly higher MOTA score than our proposed approach, but we achieved higher scores in both MOTP and FPS which are considered more crucial metrics for the type of the problem we are tackling. The reason for that, is because MOTP is directly correlated to the false positive rate which needs to be kept to the minimum in safety-critical applications such as AGVs where the decisions need to be

Table 4: Evaluation of the run-time requirements of our proposed framework with its two stages (tracking+prediction) for the multi-cyclists intent prediction in terms of ms/FPS. Lower ms is better while higher FPS is better.

Method	Tracking (ms)	Prediction (ms)	Total (ms/FPS)
FPFH-RF	75.5	110	185.5/5.3
3D-CNN [43]	75.5	260	335.5/2.9
Voxel-MLP (ours)	75.5	7	82.5/12.12
Voxel-LSTM (ours)	75.5	**2**	**77.5/12.9**

inferred in a timely manner (i.e. higher FPS).

4.5. Runtime Analysis

Given the multi-task nature of our proposed framework for cyclist intent prediction, we need to evaluate the total run-time requirements which is one of the main factors that determine the feasibility of deploying it to real AGVs. In Table 4, we report the average run-time in milliseconds (ms) and FPS for our proposed framework in comparison to the aforementioned three baseline models. Our proposed Voxel-LSTM has achieved the highest score in terms of FPS with an average score of 12.9 FPS while the Voxel-LSTM comes in the second place while 3D-CNN is the lowest with only 2.9 FPS. It worth noting that the FPFH-RF was only utilising the CPU while the other three models, namely Voxel-LSTM, Voxel-MLP and 3D-CNN were utilising the GPU.

5. Conclusion

In this work, we have introduced a novel framework for fast intent prediction of multi-cyclists in urban traffic environment from point cloud data. Our framework jointly detects, tracks and predicts four distinctive intended actions commonly performed by cyclists in urban traffic scenarios. The framework has achieved exceptional results in terms of precision, recall and F_1 measure and it showed robust results when tested on real point cloud data from Velodyne VLP-32C 3D LiDAR sensor.

References

[1] B. Philanthropies, The aspen institute.(2017). taming the autonomous vehicle: A primer for cities.

[2] P. Fairley, The self-driving car's bicycle problem, IEEE Spectrum.
URL https://spectrum.ieee.org/cars-that-think/transportation/self-driving/the-selfdriving-cars-bicycle-problem

[3] L. Laker, Street wars 2035: can cyclists and driverless cars ever co-exist?, The Guardian.
URL https://www.theguardian.com/cities/2017/jun/14/street-wars-2035-cyclists-driverless-cars-autonomous-vehicles

[4] S. Hanley, Bicycles and autonomous cars are on a collision course, CleanTechnica.
URL https://cleantechnica.com/2017/08/21/bicycles-autonomous-cars-collision-course/

[5] K. Saleh, M. Hossny, S. Nahavandi, Spatio-temporal densenet for real-time intent prediction of pedestrians in urban traffic environments, Neurocomputing 386 (2020) 317–324.

[6] J. F. P. Kooij, N. Schneider, F. Flohr, D. M. Gavrila, Context-based pedestrian path prediction, in: European Conference on Computer Vision, Springer, 2014, pp. 618–633.

[7] I. Walker, Signals are informative but slow down responses when drivers meet bicyclists at road junctions, Accident Analysis & Prevention 37 (6) (2005) 1074–1085.

[8] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, S. Srinivasa, Planning-based prediction for pedestrians, 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009 (2009) 3931–3936doi:10.1109/IROS.2009.5354147.

[9] C. G. Keller, D. M. Gavrila, Will the pedestrian cross? A study on pedestrian path prediction, IEEE Transactions on Intelligent Transportation Systems 15 (2) (2014) 494–506. doi:10.1109/TITS.2013.2280766.

[10] K. Saleh, M. Hossny, S. Nahavandi, Intent prediction of vulnerable road users from motion trajectories using stacked lstm network, in: Intelligent Transportation Systems Conference (ITSC), 2017 IEEE International Conference on, IEEE, 2017.

[11] K. Saleh, M. Hossny, S. Nahavandi, Early intent prediction of vulnerable road users from visual attributes using multi-task learning network, in: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, 2017, pp. 3367–3372.

[12] M. Gschwandtner, R. Kwitt, A. Uhl, W. Pree, Blensor: blender sensor simulation toolbox, in: International Symposium on Visual Computing, Springer, 2011, pp. 199–208.

[13] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun, CARLA: An open urban driving simulator, in: Proceedings of the 1st Annual Conference on Robot Learning, 2017, pp. 1–16.

[14] A. Bewley, J. Rigley, Y. Liu, J. Hawke, R. Shen, V.-D. Lam, A. Kendall, Learning to drive from simulation without real world labels, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 4818–4824.

[15] M. Wrenninge, J. Unger, Synscapes: A photorealistic synthetic dataset for street scene parsing, arXiv preprint arXiv:1810.08705.

[16] Y. Chen, W. Li, X. Chen, L. V. Gool, Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1841–1850.

[17] Y. Luo, L. Zheng, T. Guan, J. Yu, Y. Yang, Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation, in: Pro-

ceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 2507–2516.

[18] K. Saleh, A. Abobakr, D. Nahavandi, J. Iskander, M. Attia, M. Hossny, S. Nahavandi, Cyclist intent prediction using 3d lidar sensors for fully automated vehicles, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 2020–2026.

[19] J. F. Kooij, F. Flohr, E. A. Pool, D. M. Gavrila, Context-based path prediction for targets with switching dynamics, International Journal of Computer Vision 127 (3) (2019) 239–262.

[20] K. Saleh, M. Hossny, S. Nahavandi, Contextual recurrent predictive model for long-term intent prediction of vulnerable road users, IEEE Transactions on Intelligent Transportation Systems.

[21] R. Meijer, S. de Hair, J. Elfring, J. Paardekooper, Predicting the intention of cyclists, in: 6th Annual International Cycling Safety Conference, 21-22 September 2017, Davis, California, 2017, pp. 1–3.

[22] E. A. Pool, J. F. Kooij, D. M. Gavrila, Using road topology to improve cyclist path prediction, in: Intelligent Vehicles Symposium (IV), 2017 IEEE, IEEE, 2017, pp. 289–296.

[23] C. Benedek, B. Gálai, B. Nagy, Z. Jankó, Lidar-based gait analysis and activity recognition in a 4d surveillance system, IEEE Transactions on Circuits and Systems for Video Technology 28 (1) (2018) 101–113.

[24] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[25] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, 2015, pp. 91–99.

[26] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 652–660.

[27] Y. Zhou, O. Tuzel, Voxelnet: End-to-end learning for point cloud based 3d object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4490–4499.

[28] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Advances in Neural Information Processing Systems, 2017, pp. 5099–5108.

[29] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2117–2125.

[30] A. Abobakr, M. Hossny, S. Nahavandi, Body joints regression using deep convolutional neural networks, in: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, 2016, pp. 003281–003287.

[31] K. Saleh, M. Hossny, S. Nahavandi, Kangaroo vehicle collision detection using deep semantic segmentation convolutional neural network, in: 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), IEEE, 2016, pp. 1–7.

[32] K. Saleh, M. Hossny, A. Hossny, S. Nahavandi, Cyclist detection in lidar scans using faster r-cnn and synthetic depth images, in: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2017, pp. 1–6.

[33] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[34] X. Weng, K. Kitani, A Baseline for 3D Multi-Object Tracking, arXiv:1907.03961arXiv:1907.03961.
URL https://arxiv.org/pdf/1907.03961.pdf

[35] H. W. Kuhn, The hungarian method for the assignment problem, Naval research logistics quarterly 2 (1-2) (1955) 83–97.

[36] S. Shi, X. Wang, H. Li, Pointrcnn: 3d object proposal generation and detection from point cloud, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 770–779.

[37] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise., in: Kdd, Vol. 96, 1996, pp. 226–231.

[38] H. Aljumaily, D. F. Laefer, D. Cuadra, Urban point cloud mining based on density clustering and mapreduce, Journal of Computing in Civil Engineering 31 (5) (2017) 04017021.

[39] J. Guo, W. Feng, J. Xue, S. Xiong, T. Hao, R. Li, H. Mao, An efficient voxel-based segmentation algorithm based on hierarchical clustering to extract lidar power equipment data in transformer substations, IEEE Access 8 (2020) 227482–227496.

[40] C. Wang, M. Ji, J. Wang, W. Wen, T. Li, Y. Sun, An improved dbscan method for lidar data segmentation with automatic eps estimation, Sensors 19 (1) (2019) 172.

[41] S. Gottschalk, D. Manocha, M. C. Lin, Collision queries using oriented bounding boxes, Ph.D. thesis, University of North Carolina at Chapel Hill (2000).

[42] R. Anderson, S. Doecke, J. Mackenzie, G. Ponte, D. Paine, M. Paine, Potential benefits of forward collision avoidance technology, Injury 44 (15) (2012) 24.

[43] J. Owoyemi, K. Hashimoto, Spatiotemporal learning of dynamic gestures from 3d point cloud data, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 5929–5934.

[44] S. Godbole, S. Sarawagi, Discriminative methods for multi-labeled classification, in: Pacific-Asia conference on knowledge discovery and data mining, Springer, 2004, pp. 22–30.

[45] R. B. Rusu, J. Bandouch, F. Meier, I. Essa, M. Beetz, Human action recognition using global point feature histograms and action shapes, Advanced Robotics 23 (14) (2009) 1873–1908.

[46] M. Khokhlova, C. Migniot, A. Dipanda, 3d point cloud descriptor for posture recognition., in: VISIGRAPP (5: VISAPP), 2018, pp. 161–168.

[47] R. B. Rusu, Semantic 3d object maps for everyday manipulation in human living environments, Ph.D. thesis, Computer Science department, Technische Universitaet Muenchen, Germany (October 2009).

[48] T. K. Ho, Random decision forests, in: Proceedings of 3rd international conference on document analysis and recognition, Vol. 1, IEEE, 1995, pp. 278–282.