Proceedings of the Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)
IEEE Xplore Part Number:CFP20OSV-ART; ISBN: 978-1-7281-5464-0

Vehicle Trajectory Prediction using GAN

Chinmayi Hegde
Department of CSE
PES University
Bangalore, India

Suman Dash
Department of CSE
PES University
Bangalore, India

Pooja Agarwal
Department of CSE
PES University
Bangalore,India

poojaagarwal@pes.edu

***Abstract*—Autonomy is becoming more alluring in today's world due to its applications in various fields such as defense, transportation and exploration of unknown terrain. Autonomy in transportation will be common in the near future and prediction of trajectory helps in decision making. Trajectory prediction is crucial and significant for efficient navigation. The behavior of vehicles is not typical like most other dynamic objects due to physical factors such as vehicle size, driver behavior, etc. and social factors such as avoidance of obstacles, an inter-vehicle distance of neighboring vehicles, etc. data-driven vehicle-trajectory prediction model using Generative Adversarial Networks (GANs) is proposed which uses the partial history of the vehicles and learns the complex behavior of a vehicle. The proposed scheme feeds the model the sequence of vehicle positions as the coordinates that are obtained from a traffic scene and result in the probabilistic vehicle position based on vehicle velocities, prioritization and the above social and physical factors. The model is evaluated on aerial views of traffic data such as VisDrone data set and the TRAF data set. The proposed model can learn vehicle behaviors such as overcoming, merging, etc. Independent of vehicle type, the proposed model predicts the future path of each vehicle in the scene given the history of the vehicle and the results are measured using the average and final displacement errors.**

Keywords—GANs, Trajectory prediction, Sequence prediction, Autonomous vehicles

I. INTRODUCTION

According to the statistics acquired from the Bengaluru Traffic Police, in 2017 alone, the number of injuries due to accidents on road was 4,256; there were 3,453 non-fatal accidents and 1,002 cases of reckless driving. Being a part of a busy city, one realizes the importance and relevance of autonomy in machines, especially in environments like roads where the situations are uncertain and dynamically changing. For instance, having self-driving cars to control accidents in traffic or even having social robot assistants for the differently-abled. Autonomy brings ease and safety in the day to day activities. Although many vehicles are equipped with the technology of sensors and detectors that even alert the driver if the possibility of an accident is detected, it would only have an if the navigation of the driver changes accordingly. In most parts of metropolitan cities, surveillance video cameras are availed to monitor traffic roads and intersections but that alone is not enough to avoid accidents. In the above examples, one of the important aspects of bringing about autonomy would be to understand the behavior of the surrounding agents, such as vehicles or pedestrians, and to take appropriate measures accordingly to avoid collisions. Trajectory prediction plays a significant role in accomplishing the task by providing a probabilistic future path for a vehicle and it gives us an understanding of pedestrian and vehicle interactions. It is a utilitarian to navigate safely for self-driving vehicles, traffic forecasting or congestion management.

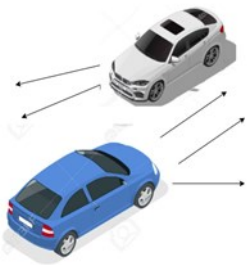

Fig. 1. A conceptual illustration that shows a scene where two cars are approaching each other. The arrows show the multiple plausible paths they can take to avoid colliding into each other. This paper aims to predict all such plausible future trajectories.

II. RELATED WORK

The increasing availability of cameras, computer vision and a wide range of sensors make it easy to track traffic road-agents and alert the system to avoid a collision but these methods are purely based on the proximity of the traffic agents and cannot be efficient enough to avoid a collision entirely. Traffic agents refer to buses, cars, scooters as well as pedestrians and bicycle riders. Trajectory prediction can be seen as a forecasting problem or as a sequence prediction problem. Agrim Gupta et al. [1] use generative adversarial networks to predict human trajectories in crowded scenes, training the model according to the human-space as well as human-human interaction, considering human behavior in crowded scenes, i.e.,

interpersonal, socially acceptable and multimodal properties of human motion. Using the ETH and UCY data sets, they use Long Short-Term Memory (LSTM) algorithm for the Recurrent Neural Network (RNN) for encoder and decoder with a pooling mechanism. They introduce a new pooling mechanism that encodes the slightest indication for all the humans in the scene. However, the model works only for humans which are a minor, although important, part in the traffic scene and due to this, the model does not work of datasets that are heterogeneous in nature.

Amir Sadeghian et al. [2] presents a framework for path prediction that leverages dependencies between agents' behaviors and the navigation environment using a Recurrent Network. Its architecture comprises of feature extractor, visual attention module and the recurrent module, an LSTM network with the Adam optimizer. The model uses the Stanford Drone dataset and Formula One car racing data set. While the former data set is useful for descriptive analysis, the latter dataset highly differs from a traffic dataset in terms of velocities of agents, heterogeneity of the vehicles and density of the traffic as well as the structure of the road.

Yuexin Ma et al. [3] uses LSTM for real-time traffic prediction. It treats each vehicle as an agent which is denoted by A_i^t which is (x_i^t, y_i^t, c_i^t) where *(x, y)* are the coordinates and c is the label of a vehicle i at time t. The model uses the instance layer to capture the movement and category layer to capture the behavior of vehicles. While it considers heterogeneity, dense traffic and works with a realistic data set, it does not consider velocity. An important part to be taken into consideration in collision analysis is the velocity with which the vehicle is being driven.

Rohan Chandra et al. [4] describes the trajectory prediction problem as a sequence prediction and give important observations regarding the nature of the interaction of a traffic agent with other agents, concluding it to be a semi-elliptical region in the field of view. The model considers weighted interactions and assigns weights based on the shape and size of the vehicles. An LSTM-CNN (Convolutional Neural Network) hybrid network to achieve the intention.

Alexandre Alahi et al. [5] describes the trajectory prediction problem as a sequence generation exercise wherein they model crowds in which humans use their common sense and follow certain social norms to not disturb their neighbor's personal space. But predicting human behavior is not a simple task as it needs a detailed understanding of social behaviour. This problem is solved by their proposed model which uses an LSTM model for each person in the scene with social pooling layer in which information and weights are shared among the different layers. This model generalizes the behavior of all the people in the context moving with different speeds and having distinct paths. Then a sequence of steps is followed for future path prediction, they are: position estimation, occupancy map pooling and finally path prediction. The model is tested on ETH and UCY datasets with a reduced error in path prediction.

Xue Hao et al. [6] says that the problem of trajectory prediction of pedestrians is a difficult task in crowds because of the chaotic movements of people and the fact that the arrangement of the surroundings of the agent needs to be considered. Hence, they propose a hierarchical model called Social Scene LSTM which consists of three LSTMs to tackle the limitation mentioned above. The social LSTM models the relationship between any pedestrian and its immediate neighbors at any instant of time on a map. The scene LSTM considers the characteristics of the scene in general such as the obstacles present. Finally, the person LSTM takes the past coordinates as input and predicts the future trajectory of every person present in the scene.

Nachiket Deo et al. [7] says that when an autonomous vehicle drives through some traffic it needs to make decisions based on where its final destination is and what the current position of each vehicle near it and accordingly either change its speed, accelerate or slow down, change its lane etc. For this it needs to predict the path of its surrounding vehicles which is not an easy task as there can be many types of variables involved, the destination of each driver can vary, the interaction between vehicles and also each of them has his way of driving. But some things which can be done to tackle the above is maneuver and to describe a blueprint of lane structure on highways. The model proposed is an LSTM model that has an encoder, decoder and convolution social pooling that takes into account the past trajectory of each vehicle and outputs a multi-modal future trajectory taking into consideration interdependencies among vehicles.

Yuexin Ma et al. [8] talks about a critical problem in autonomous driving cars moving in traffic which is local navigation with more than one agent. The solution to this problem lies in predicting the trajectories of each vehicle in the scene to avoid any collision and also take care of the dynamic constraints such as traffic rules. They propose AutoRVO, an algorithm for movement in dense traffic with more than one agent having different features. AutoRVO uses the CTMAT representation and inverse collision avoidance to calculate the feasible trajectories and velocities of every agent in the scene to ensure collision avoidance.

Amir Sadeghian et al. [9] says that when people move in a crowd, they adjust themselves to avoid any upcoming obstacles in their paths and also interact physically with others in the scene. But this is not an easy task as one has to confirm with the physical limitations of the environment around him, predict the path and social behavior of others around him and find multiple alternative paths to take. To tackle the problems above they propose SoPhie, a GAN based model considers the effect of all the vehicles in the given context and also their interactions with each other to predict their trajectories. The model has a feature extractor which uses a CNN to take features from the scene, an attention module that takes into account physical limitation and social interaction among the vehicles and the GAN module which uses LSTM to predict feasible trajectories.

Na lin et al. [10] provides insight into driving styles and an overview of the identification of driver behavior control for automotive control. Driver behavior is not solely based on the traffic and lane rules but critically varies not only according to the experiences, emotions and driving preferences but also according to his age, gender, etc. To identify the driver characteristic, the driving intention is broken into several small

and simple driving intentions. The paper discusses different technologies that are useful to identify driver behavior characteristics such as classification using fuzzy logic.

III. TERMINOLOGY

A. *Generative Adversarial Network*

Generative Adversarial Networks (GANs) [11], a class of neural networks used for unsupervised learning. It consists of two networks - a generator *G* and a discriminator *D*. The generator *G* create new data instances that resemble the training data by capturing the data distribution and the discriminator *D* learns to distinguish the true data from the data generated by the generator *G* by estimating the probability of the sample data coming from training data or the data generated by the generator *G*. The discriminator is trained to maximize the probability of correct prediction for both training examples and the data from *G* while concurrently training *G* to minimize *log(1 - D(G(z)))*. The *D* and *G* play a mini-max game with value function *V(D,G)*:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}\left[log\left(D(x)\right)\right] + E_{z\sim p_z}\left[log\left(1\text{-}D(G(z))\right)\right] \quad (1)$$

B. *Long Short Term Memory*

Long Short-Term Memory (LSTMs) [12], are an extension to the Recurrent Neural Network (RNN) which can be used for classification and prediction purpose. They overcome the shortcomings of the traditional RNNs some of which are the vanishing gradient problem and long-term dependency. An LSTM is made up of a series of interconnected modules whose basic idea is that of a cell state which is used to remember information. There are three gates-input gates, output gate and forget gate which is used to alter or control the flow of information. The input gate regulates the value entering a cell, the forget gate regulates the time and degree to which the information remains in the cell and the output gate regulates the output activation.

IV. PROPOSED METHOD

A. *Problem Statement*

This paper aims to predict the future trajectories of all the vehicles in a given scene given their past trajectories for a certain time frame. The input trajectory of a vehicle i is given as: $X_i = (x_i^t, y_i^t)$ for a certain time frame where t denotes the time step. Similarly, the predicted trajectory can be denoted as:$Y_i = (x_i^t, y_i^t)$. If the nature of the trajectory of the surrounding objects is known, the vehicle would be able to take the most plausible path in the desired directions to avoid the collision.

B. *Proposed Architecture Diagram*

The proposed architecture is shown below in figure 2. Traffic images from various data sets are fed as input on which preprocessing is done to improve the quality of the images and to extract the required features. The object detection is then performed to obtain the coordinates of each vehicle in the scene.

These coordinates are passed on to the GAN module which consists of the generator and the discriminator which predicts future trajectories taking into account past trajectory history and social rules. The socially acceptable and most plausible trajectory is then output by the model. Each of these blocks is explained below in detail.

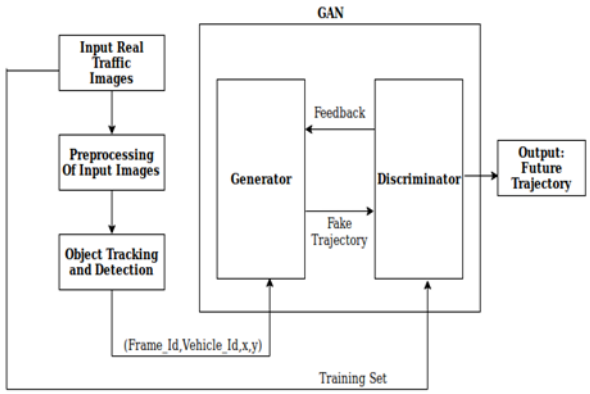

Fig. 2. Proposed architecture for trajectory prediction

C. *Input*

Real images of traffic consisting of various vehicles like cars, trucks, scooters etc. are used as input to the model. The images are divided into training, testing and validation sets. Images from the training set are used to train the discriminator of the GAN. The images are preprocessed to extract the required features and improve the efficiency of the model.

D. *Preprocessing of input images*

Prior to the object detection and tracking, each image sequence or the video frame is pre-processed with two steps: Bilateral filtering and Adaptive Gaussian thresholding. The bilateral filter uses filters in the space domain and the pixel intensity domain. This method retains the edges. The adaptive gaussian thresholding is used for foreground-background segmentation. This helps to retain regions despite different lighting conditions.

E. *Object Tracking and Detection*

The first step in the model is to detect and track vehicles across frames in a given video or sequence of images which is the initial input. Given an input sequence of images or a video of traffic consisting of many vehicles on a road, the YOLOv3 algorithm is applied to perform object detection and tracking.

For object detection and tracking, the pre-trained YOLOv3 model has been used. The detection is run after every few frames, which could be specified and tracking operates throughout all frames. Objects that do not appear in the frame, for a specified number of successive frames, are considered to be absent or disappeared. The number of frames is decided by a parameter.

Tracking takes place in three steps. The first step is the detection of the object. The tracker is initialized for each detected object, i.e., around each bounding box. A unique ID is given to it. In successive frames, the tracker identifies the object in the bounding box in the new frame. The Euclidean

distance is computed between the frames and the correct ID is assigned to the new boxes.

This algorithm draws a bounding box around all the vehicles detected in a frame and then calculates the centroid of each vehicle. The algorithm then calculates the distance between the centroids of the vehicles in subsequent frames and assigns each vehicle a unique id accordingly which is stored as the vehicle. The coordinates are stored in a text file for each frame one by one in the following format: (FrameID, VehicleID, X, Y), where X and Y are the centroid coordinates of a vehicle.

F. GAN model for trajectory prediction

The GAN model is made up of three modules: Generator, Pooling module and a Discriminator. The generator is made up of an encoder-decoder network and is responsible for producing future trajectories, the pooling module takes care of the interaction between vehicles by performing max pooling and the discriminator classifies the incoming trajectories as real or fake based on a score it assigns to them.

Generator: The text file obtained in the previous step is given as the input to this module. This text file contains the history of each vehicle in the form of coordinates at each time step for a certain observed time period. The generator is made up of encoder and decoder network and a pooling module.

Encoder: The location of each vehicle is stored in a perceptron to get a vector. This vector is fed into the encoder which is made up of LSTM cells. The recurrent equations can be summarized as follows for each time step in the next two equations:

$$e_i^t = \phi(x_i^t, y_i^t, w_e) \tag{2}$$

where e_i^t is the fixed vector storing location of each vehicle, $\phi(.)$ is the embedding function with ReLU activation, (x_i^t, y_i^t) are the centroid coordinates of each vehicle at the given time t and w_e is the embedding weight of the encoder cells.

$$h_{ei}^t = LSTM(h_{ei}^{t-1}, e_i^t, w_{encoder}) \tag{3}$$

where h_{ei}^t is the hidden state of the LSTM cells which make up the encoder, e_i^t is the fixed vector storing location of each vehicle obtained in the previous equation and $w_{encoder}$ is the weight of the LSTM cells.

Hence the encoder stores the history of every vehicle. This data is then passed onto the pooling module which models the interaction between all the vehicles in the given scene abiding to the social norms. The intermediate states are pooled to get a pooled tensor P for each vehicle.

Decoder: Now the decoder consisting of LSTM cells produces future trajectories taking into account the past trajectory and the pooled tensor generated by the pooling module, which can be summarized in the following equations: The decoder is initialized as follows as shown in the equation below:

$$c_i^t = \gamma(P_i, h_{ei}^t, w_c) \tag{4}$$

$$h_{di}^t = [c_i^t, z] \tag{5}$$

where $\gamma(.)$ is a multi-layered perceptron with ReLU activation, P_i is the pooled tensor generated by pooling module and w_c is the embedding weight and c_i^t is the initial vector fed to decoder cells and z can be any random initial value.

The equations of the decoder used to make predictions can be summarized in the next three equations:

$$e_i^t = \phi(x_i^{t-1}, y_i^{t-1}, w_{ed}) \tag{6}$$

where ϕ is the embedding function with ReLU activation, (x_i^{t-1}, y_i^{t-1}) are the centroid coordinates of vehicle i at time t-1 and w_{ed} is the embedding weight used in decoder cells.

$$h_{di}^t = LSTM(\gamma(P_i, h_{di}^{t-1}), e_i^t, w_{decoder}) \tag{7}$$

where h_{di}^t are the hidden states of LSTM making up decoder cells, γ is an MLP, P_i is the pooled tensor for each vehicle obtained from the pooling module, and $w_{decoder}$ is the embedding weight of decoder cells.

$$(X_i'^t, Y_i'^t) = (\gamma(h_{di}^t)) \tag{8}$$

where $(X_i'^t, Y_i'^t)$ are the predicted coordinates of vehicle i, γ is an MLP and h_{di}^t is obtained from the previous equation.

Pooling Module: This layer is introduced to model the interaction among the vehicles and the social rules that govern them. It lies as an intermediate between the encoder and decoder. The difference between the coordinates of the desired vehicle and all the other vehicles present in the scene is calculated and the relative positions then obtained are embedded into the hidden state of that vehicle and pooled to get a pooled tensor P_i for each vehicle.

Discriminator: The discriminator is made up of an encoder. An MLP is applied on its last hidden state and a score is calculated. With training, the discriminator will be able to classify more accurately the trajectories as real or fake.

G. Output

The future trajectories of the vehicles in the scene are predicted by the GAN and plotted on graphs. The results are discussed in detail in the next section.

V. EXPERIMENTAL RESULTS

A. Datasets

The datasets that have been used are the VisDrone dataset[13] and TRAF dataset. The VisDrone dataset consists of a sequence of images that shows real-time traffic. These images are taken by drones with mounted high-resolution cameras that capture vehicles in traffic such as cars, buses etc. The total number of frames are: 6471 for training, 548 for validation and 1580 for testing. The TRAF dataset has around 22 videos of dense traffic consisting of buses, scooters, cars etc. It depicts cases of heterogeneous traffic.

B. Implementation

Python 3.6 programming language has been used for the implementation. YOLOv3 algorithm has been run using OpenCV. TensorFlow and PyTorch libraries have been imported for the GAN module. The results are tested on a

system with a good RAM capacity, high processing GPU and inbuilt Nvidia Driver. The batch size for the generator and discriminator is 64 and the number of epochs used is 200. The learning rate is 0.001.

C. *Evaluation Metrics*

To evaluate the model, the following two metrics have been used: average displacement error (ADE) and final displacement error (FDE).

Average Displacement Error (ADE) is a measure of the deviation of the predicted position of the object with respect to the actual position over a time period or number of future positions predicted.

Final Displacement Error (FDE) is the distance between the predicted final position point and true final position point of the vehicle at the end of prediction over a time period or after a fixed number of positions.

Each segment is composed of 8 past positions, which are fed to the generative adversarial network (GAN) as sequential inputs, and 8 or 12 future positions are used to evaluate predictions. This is the standard temporal setup for most trajectory prediction problems such as the Stanford Drone Dataset, ETH-UCY dataset, etc.

D. *Results*

Fig. 3 shows the output of the object detection and tracking module which draws bounding boxes (shown in green) around all the vehicles detected in a given frame and finds their coordinates. The YOLOv3 algorithm is applied to each image frame and each vehicle is assigned a unique id and its centroid is calculated and marked in the figure as shown and also stored in a text file. Given in blue text is the probability of an id belonging to the object based on the tracking of the vehicles.

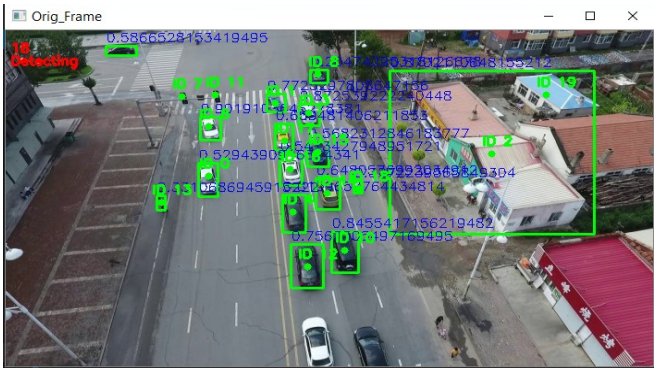

Fig. 3. Object detection is shown in one image frame.

Fig. 4 shows the graph plotted for the trajectories predicted by the GAN for a particular given vehicle. The predicted trajectory (in red) and ground truth trajectory (in blue) are shown below.

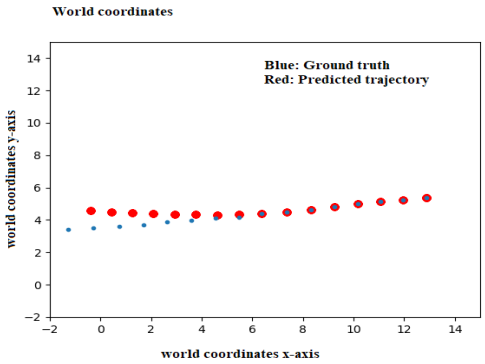

Fig. 4. Predicted trajectory and ground truth trajectory of a particular vehicle.

Fig. 5 shows the trajectory for another randomly selected vehicle from the given input images. The predicted trajectory is quite close to the actual trajectory with minimal deviation.

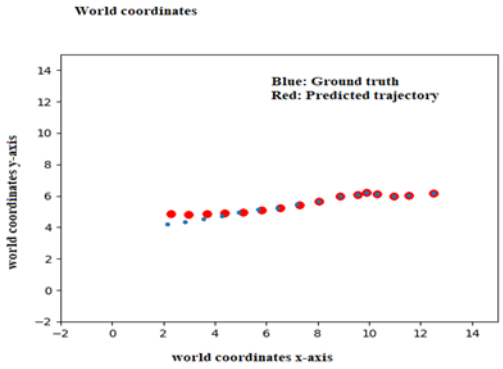

Fig. 5. Predicted and ground truth trajectory of another vehicle.

Table I displays the results obtained by the proposed model. It consists of the predictions by segment with the Pred Length, i. e., the number of future positions predicted by the model (time period), the average displacement error (ADE) and the final displacement error (FDE) calculated for the corresponding number of time steps.

The predictions are obtained for 8-time steps (3.2 seconds) or 12-time steps (4.8 seconds). The ADE and FDE are calculated and reported in meters. The number of time steps (Pred Len) is a parameter that can be changed by the user.

As seen in Table I, the displacement errors i.e. ADE and FDE obtained using GAN are considerably less as compared to the existing trajectory prediction methods such as linear regressor, LSTM, Social LSTM etc. which give comparatively higher errors.

Table I. Displacement errors

S.No.	Pred Length	ADE	FDE
1.	12	0.55	1.37
2.	8	0.49	1.04
3.	12	0.49	1.07

4.	8	0.42	0.62
5.	12	0.47	1.10
6.	8	0.19	0.53
7.	12	0.18	0.50
8.	8	0.10	0.22
9.	12	0.17	0.56

Table II shows a comparison between the proposed objective and the actual outcome obtained after running the model.

Table II. Proposed Objective versus Obtained Outcome

Proposed Objective	Obtained Outcome
There lies an uncertainty in the paths of the vehicles. Therefore, there could be multiple possible plausible trajectories. The objective of this paper was to propose the most plausible trajectory based on the past history of the vehicle taking into account numerous factors.	Based on the social constraints and the history of the vehicle as observed, the model detects vehicles and then predicts the most plausible path for the vehicle as to the output. The most plausible path can be considered the safest in terms of following inherent social norms (as observed in the input data) and obstacle avoidance.

VI. Conclusion

In this paper, a vehicle trajectory prediction model using Generative Adversarial Networks has been proposed. The model considers the interpersonal and multimodal properties and predicts the most plausible path for each vehicle present in the scene with the given history of the vehicle which includes factors such as driver behavior and vehicle size. Using a large amount of data available, the generative adversarial network (GAN) was trained to predict the vehicles' location in the future to avoid collisions and accidents. The predicted trajectories which are socially acceptable are plotted on graphs and metrics such as ADE and FDE show how the proposed method is better than the existing ones. The conducted experiments show that the model provides a reasonably accurate prediction for the vehicle for the given datasets.

VII. Future work

The work has a few limitations. The model does not work real-time as the images have to be processed before feeding it to the GAN. Because it is trained from a drone data set, it can be used to learn and understand the dynamics of a complex traffic system but the inability to collect this type of data real-time and process it is a tedious task.

References

[1] Gupta, Agrim, et al. "Social gan: Socially acceptable trajectories with generative adversarial networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[2] Sadeghian, Amir, et al. "Car-net: Clairvoyant attentive recurrent network." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[3] Ma, Yuexin, et al. "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. 2019.K. Elissa, “Title of paper if known,” unpublished.

[4] Chandra, Rohan, et al. "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[5] Alahi, Alexandre, et al. "Social LSTM: Human trajectory prediction in crowded spaces." Proceedings of the the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),pp. 961-971.

[6] Xue, Hao, et al. "SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction." In Applications of Computer Vision (WACV), 2018 IEEE Winter Conference on, pages 1186–1194. IEEE, 2018.

[7] Deo, Nachiket, et al. "Convolutional Social Pooling for Vehicle Trajectory Prediction." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2018, pp. 1468-1476.

[8] Ma, Yuexin, et al. "AutoRVO: Local Navigation with Dynamic Constraints in Dense Heterogeneous Traffic." In Computer Science in Cars Symposium (CSCS). ACM, 2018.

[9] Sadeghian, Amir, et al. "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints." The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2255-2264.

[10] Lin, Na, et al. "An overview on study of identification of driver behavior characteristics for automotive control." Mathematical Problems in Engineering 2014 (2014).

[11] Goodfellow, Ian, et al. "Generative Adversarial Nets." In Advances in neural information processing systems, 2014.

[12] Greff, Klaus, et al. "LSTM: A Search Space Odyssey." IEEE Transactions on Neural Networks and Learning Systems (2017).

[13] Zhu, Pengfei, et al. "Vision Meets Drones: Past, Present and Future." arXiv preprint arXiv:2001.06303 (2020).