Chapter 2

Theoretical Foundation and Literature Survey

This chapter presents the survey of the state-of-the-art techniques for Object Detection, Multi-object Tracking, Trajectory Prediction and Motion Planning. Section 2.1 presents a theoretical foundation for the perception and planning module of the AVs. In section 2.2 the literature of the various submodules has been discussed that represents the workflow of the thesis. The identified research gaps are mentioned in section 2.3. Section 2.4 discusses the datasets used for the perception and planning module of the AVs. Section 2.5 presents benchmarks and evaluation metrics for the sub-modules of the perception and planning of the AVs. The chapter concludes in section 2.6.

2.1 Introduction

Artificial intelligence, robotics, machine learning and signal processing belong to the versatile field of computer vision. The aim of computer vision is a program to an automatic system to understand, analyze and process images and videos. The essential tasks of computer vision are object detection, object tracking, pose estimation, activity recognition and segmentation. The sensors work as the eyes of the driver for the self-driving car and computer vision helps the car to understand the surroundings and plan to make the decision. A detailed literature review of perception and planning techniques has been done in this chapter. This review has been done in two different sections- the first is dedicated to perception (object detection, multi-object tracking and trajectory prediction) and the second is for Planning (motion planning). This chapter also summarizes the research gaps reported in the literature. Except for the literature review, the theoretical background of the research has been discussed in this chapter. The literature survey includes publicly available datasets used to validate the proposed models and performance measures used for the evaluation purpose.

2.2 Literature Review

This section has reviewed some state-of-the-art techniques for Object Detection, Multi-Object Tracking, Trajectory Prediction and Motion Planning.

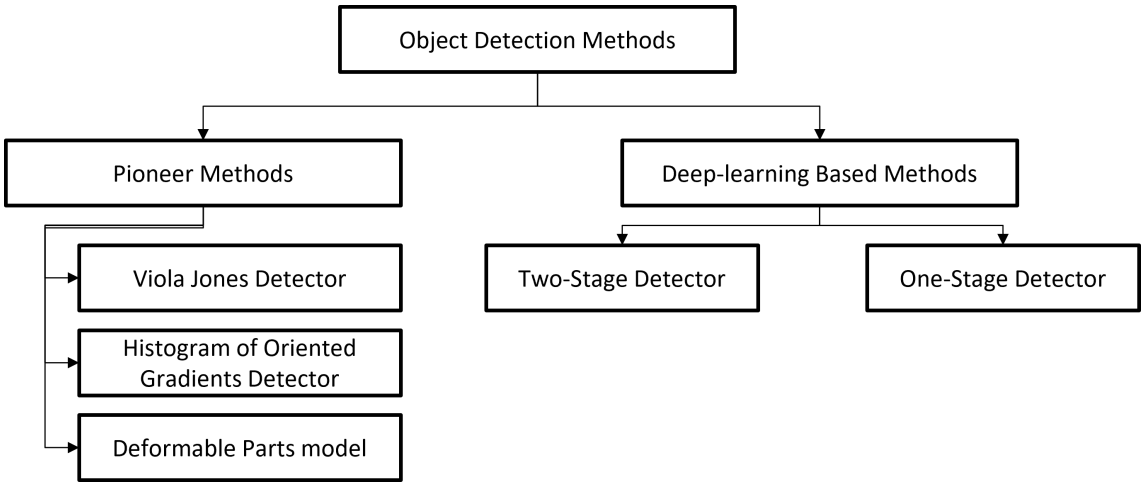

FIGURE 2.1: Taxonomy of the Object Detection methods

2.2.1 Literature Review of Object Detection

Over the past decades, much work has been done on object detection. In literature, different kind of methods exists for activity recognition. This section presents a detailed survey of various object detection frameworks based on pioneer and deep-learning methods. The taxonomy of the object detection is shown in Fig-2.1

2.2.1.1 Pioneer Methods

The pioneer methods are used conventional feature extraction methods and classifiers. This type of method can be categorized in 3 parts.

Viola Jones

This method combined multiple methods Haar-like features, Adaboost, integral images and cascading classifier. The first step of this method is to search for haar features by sliding the window through the complete input image and calculating it using an integral image. Then it uses a trained Adaboost to find the classifier on each haar feature and at last, cascade it.

Histogram of Oriented Gradient Detector

In this method, the HOG descriptor is used to extract the features for object detection. This method creates the feature table using edge gradient and orientation features extracted by the HOG. The image is divided into grids and a feature table is then used to histogram each cell in the grid. The HOG features are generated for the region of interest into a linear SVM classified for detection.

Deformable Part Model

It used individual parts of the object for detection and achieved higher accuracy than HOG. The parts of the detection are individually detected during inference time and a probable arrangement is marked as detection.

2.2.1.2 Deep-Learning Methods

The deep-learning-based object detector has been categorized into two parts, a Two-stage detector and a Single-stage detector and the architecture of both methods is shown in Fig- 2.2

Two-Stage Detector:

In a two-stage object detector, the former is used to get the region using a sliding window from the image and then apply classification of the region. A model with an individual module to produce the region proposals is called a two-stage detector. These approaches generate a random number of object region proposals of an image after the first stage and in the second stage, the objects are classified and localized. The two-stage method generally takes longer to produce the proposals,

has a complex model design and lacks of global features.

Region-Based Convolution Neural Network: The first version of the R-CNN family is the Region-based Convolutional Neural Network (R-CNN) [15], presenting improved detection performance using the CNN. The region proposal module is a class agnostic with CNNs that convert the detection into localization and classification issues. The mean-subtracted image using as an input that passes through the module of region proposal, which generates the 2000 number of object candidates. The region proposal module selects different image sections for finding an object with a higher probability using Selective Search [16]. The candidates are propagated and warped along with the CNN network that extracts a feature vector of size 4096-dimension for every proposal. Girshick et al. adopted AlexNet [17] using as the backbone network for the detection. The support vector machine is applied to the feature vectors to obtain the object confidence score. The score regions are passed through Non-maximum suppression (NMS), based on class and its IoU. The class is identified once the model predicts Bbox using a regressor of trained Bbox that predicts the four parameters i.e., center point coordinates along with height and weight of its Bbox.

Spatial Pooling Pyramid Network: He et al. presented the Spatial Pyramid Pooling (SPP) layer [18] that is used to process the different aspect ratios and random size images. The model observed that a fully connected part of CNN required a fixed size of input images. SPP-Net [19] added the pooling layer and replaced the convolution layer of the CNN before the module of the region proposal,

reducing the computations and making the network independent from the size or aspect ratio. The selective search generated the candidate windows. The image input is passed along the convolution layers of a ZF-5 network [20] to extract the feature map. The feature map and candidate window are mapped together and converted subsequently into a fixed length of spatial bins of the pyramidal pooling layer. The obtained feature vector passed through the FCN and used SVM to get the object score and predict class. The SPP-Net has also used the post-processing layer like R-CNN [15] to refine the localization using bounding box regression. This method also used similar multistage training, but the FCN is done the fine-tuning.
Similar to R-CNN [15], the post-processing layer of SPP Net is used to improve the location of Bbox regression. The model also uses a similar process for training, except for fine-tuning the FCN. SPP-Net has considerably comparable accuracy but is faster than the R-CNN. The deformation of an object is not effective due to the warping of input so that the model can process any shape or aspect ratio images. However, the SPP-Net and R-CNN have the analogous architecture to each other, and it also shares disadvantages of R-CNN, like computationally expensive training time and multistage training.

Fast RCNN: The primary issue is that the RCNN/SPP-Net needs many systems to train the model separately. Fast R-CNN [21] creates a single end-to-end trainable system to resolve this issue. The image and its proposal passed through the network as input. The feature map is extracted from the mapping of image and object proposals fed into the convolution layers. Girshick et al. used SPP-Net [19]

to have one spatial bin instead of the pyramidal structure of pooling layers known as the RoI pooling layer. The RoI pooling layer is combined with two FCN, then branches out a bounding box regression and N+1 class softmax layer that uses a fully connected layer as well. A multi-task loss function is used to train the model. The Fast RCNN model performed better by introducing the loss function from L2 to smooth L1 loss function.

Faster RCNN: Although the Fast R-CNN has speed like a real-time system (object detector), the region proposal generation was still slower (2 sec/image compared to 0.2 sec/image). Ren et al. proposed a fully convoluted network [22] also known as a region proposal network (RPN) in [23] that considers a random number of the input image and a candidate window set as output. The objectness score is associated with each window that determines the object likelihood. Like the predecessors, some methods [24] [25] [26] uses image pyramids for scale variant of objects; RPN represents anchor boxes. The multiple boundary boxes with different aspect ratios are used and applied regression for object localization.

The feature map is extracted from the input image when forwarded to the CNN. The obtained feature map is passed through the RPN that generates the boundary boxes and classification. The proposals are selected and mapped with the feature map obtained from the preceding CNN layer in the RoI pooling layer and fed into the fully connected layer that passes through the bounding box regressor and classifier. The faster R-CNN is like grouped Region Proposal and Fast R-CNN network.

Feature Pyramid Network: The image pyramid is used to get the feature pyramid or (image pyramid features) at different levels and is commonly used to increase small object detection. The FPN increases the detection accuracy (Average Precision) and the inference time increases substantially. Lin et al. introduced a top-down architecture known as a feature pyramid network (FPN) [27] with lateral connections to extract high-level features at various scales. There are two pathways of FPN, the former one is used ConvNet computing feature hierarchy at several scales for bottom-up pathways and the later one is used coarse feature maps obtained by upsampling from higher levels with high resolution features for top-down pathways. These two pathways are associated with lateral connection by a 1×1 convolution operation to improve the useful information of the feature. The region proposal network (RPN) of Faster R-CNN based ResNet-101 [28] is used by FPN. FPN could generate high-level features at different scales, which decreased the detection error rate.

Region-based Fully Convolutional Network: Dai et al. introduced a Region-based Fully Convolutional Network (R-FCN) [29] that follows all shared computations through the network while the resource-intensive techniques are used by the previous 2 stage detection methods. This method appreciates the use of fully connected layers instead of convolutional layers. However, the localization task is non-effective due to the convolutional network; deeper layers are translation invariant. The authors introduced score maps for position-sensitive to overcome it. These maps are pooled with exact location identification and encoded the object's relative

spatial information.

The later averaged score is used for object class prediction. The combination of 4 convolutional networks is used by R-FCN. The feature map is extracted by passing the image input through ResNet-101 [28]. The intermediate output is used to identify the RoI proposal by passing it through RPN while the convolutional layer further processes the final output and input to the regressor and classifier. The RoI proposal and position-sensitive map are grouped in classification for predictions, while the regression output is Bbox. R-FCN is trained as Faster-RCNN in a four-step manner with box regression loss and cross-entropy loss. The R-FCN also used the ten OHEM (online hard example mining) [30] in training. Dai et al. introduced a technique to solve the invariant translation problem with CNN. RFCN uses the combination of FCN and Faster-RCNN for more accurate and fast detection results. However, the RFCN is 2.5 to 20 times faster but not more improved accuracy.

Mask RCNN: Mask RCNN [31] is an extension of faster R-CNN with an extra branch parallel to another for pixel-level instance segmentation. The fully connected network branch also applied to RoIs to generate the segment by classifying each pixel with a small computational cost. It adopts the basic faster R-CNN architecture for region proposals. However, extend a mask head parallel to the bounding box regressor and classification head. The significant difference between the RoIpool layer with RoI aligns to avoid pixel-level. The Mask RCNN avoids the misalignment of a pixel due to spatial quantization and uses the RoIAlign instead of RoIPool. The backbone network is ResNeXt-101 [32] is used along with FPN to get better speed

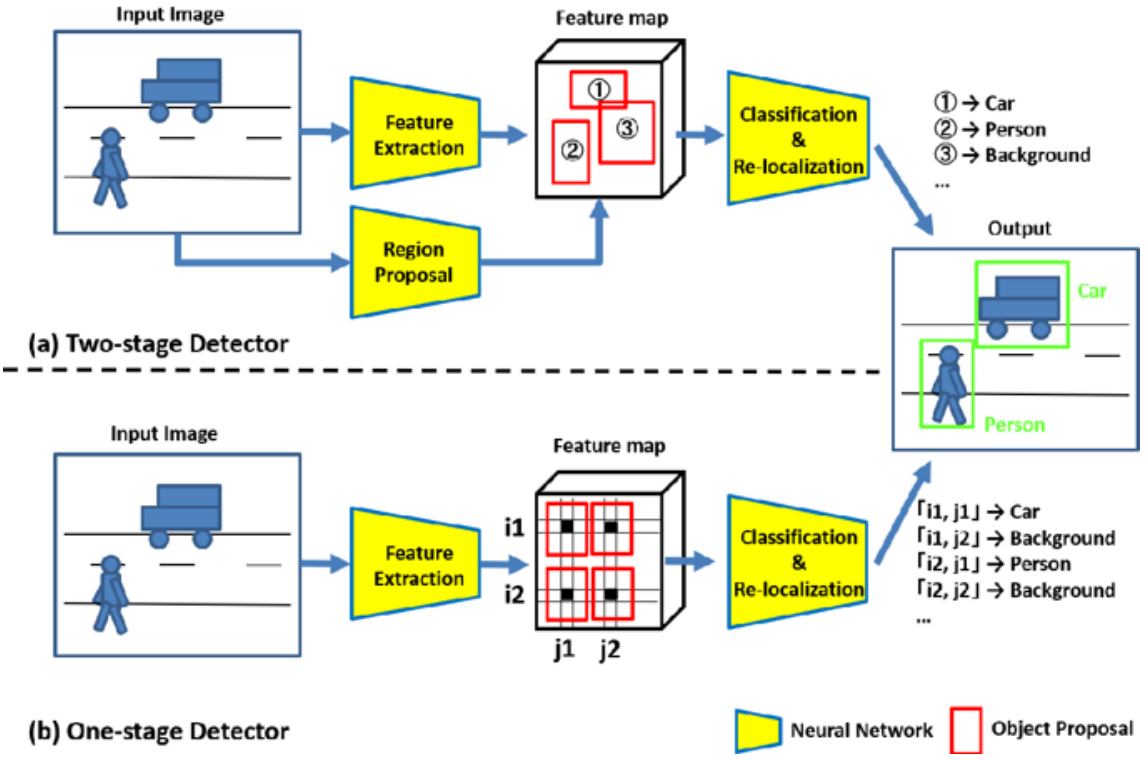

FIGURE 2.2: Architecture of (a) Two-Stage Object Detection and (b) Single-stage Object Detection

and accuracy. The Mask RCNN uses five anchor boxes with three aspect ratios as used by FPN and updated Faster RCNN with mask loss to define the loss function. The overall Mask-RCNN is trained as Faster RCNN. The small overhead in computations and adding an extra parallel branch for instance segmentation, make the model very effective, flexible, simple and well generalized in various applications like human pose estimation, keypoint detection etc. However, the Mask-RCNN is still below the performance of real-time (>30 fps).

Single-Stage Detector Single-stage detectors localize and classify semantic objects in an end-to-end manner using dense samples. Single-stage detection methods are used for predefined boxes/critical points of different aspect ratios and scales

for object localization if edge-stage detectors are simple to design for real-time performance.

You Only Look Once Object detection is considered as a classification problem in a two-stage method; the first module generates the candidates, which can be classified as either background or object. However, YOLO considers object detection as regression which can directly predict the pixel image as objects and its Bbox. YOLO divides the image input into the grid (S $\times$ S), where the center of the objects in the cell is responsible for detecting the objects. A cell predicts the many boundary boxes and five elements of each prediction array: x, y (bounding box center), w, h (bounding box dimensions), and object confidence score. The image classification model of YOLO was inspired by the GoogLeNet [33] and uses the cascaded layers of smaller convolution networks [34]. Yolo achieves more accuracy to pre-trained the model on ImageNet data [24] and adding arbitrarily initialized the fully connected layers and convolution to modify this. The cell predicts only one class as it converges better during training time, but it is increased at the inference time. The combined loss or multitask loss for all predicted objects is used for model optimization. Non-maxima Suppression (NMS) removes the multiple detections for a specific class. YOLO surpassed the one-stage model by a large margin in speed and accuracy. However, YOLO had some significant shortcomings, such as clustered or small objects localization accuracy, and the major drawback is the limited number of objects. These problems are resolved by the other versions of YOLO [35] [36] [37] [38].

Single-Shot Multibox Detector: The first one-stage detector is the Single Shot MultiBox Detector (SSD) [39] that beats the two-stage detector accuracy like Faster RCNN [23], with the speed of a real-time system. The SSD uses VGG-16 [25] as a backbone network associating some auxiliary structures for performance improvement. The auxiliary convolution layer was added as the last layer of the model, with decreases the size progressively. SSD is useful to detect small objects earlier in the network if the features of the image are not crude. The deeper layer of the model was responsible for the aspect ratio and offset of the default boxes [36]. SSD method uses Jaccard overlap to match every ground truth box with default boxes during training, similar to Multibox [40]. They also used heavy data augmentation and hard negative mining. Similar to DPM [41], the SSD used confidence loss and location weighted sum to train the model. The NMS is used to get the final output. The SSD was significantly more accurate than YOLO and faster than RCNN. It had hard to difficult to detect small objects. These issues are resolved by some backbone networks like ResNet and small fixes.

YOLOv2 & YOLO 9000: The improved version of YOLO [38] is YOLOv2 [37], which provides a tradeoff between accuracy and speed, while the YOLO9000 method predict 9000 class of object in real-time. YOLOv2 is used DarkNet-19 as a backbone network instead of GoogLeNet [33]. It incorporated various methods such as Batch Normalization [19] (improve the convergence), detection system (increase detection classes), joint training of classification and removing fully connected layers for speed increment and using learned anchor boxes to improve recall and have better

priors. The WordNet [42] is used for hierarchical structure by combining the detection and classification. The higher conditional probability for hyponym is predicted by the WordTree, if the hyponym is not correctly classified, thereby increasing the system's overall performance. The YOLOv2 can obtain better flexibility with speed and accuracy, and the new architecture uses a few parameters. As suggested in the paper title, it was "better, faster and stronger"

RetineNet: Lin et al. proposed the reason of single-stage detector lag is "extreme foreground-background class imbalance [27]" given the accuracy difference between two-stage and single-stage detectors. They introduced the reshaped cross-entropy loss, known as Focal Loss, used to remedy the imbalance. The Focal loss parameter reduces the loss contribution. The RetinaNet [27] is a simple and single-stage detector to demonstrate the efficacy of the model, which predicts the objects from input image dense sampling, aspect ratio and scale. The backbone network of model is ResNet [28] enlarged by FPN [27] for 2 similar subnets: Bbox regressor and classification. The FPN detected objects at a different scale, passing each layer to the subnets. The object score predicts by the classification subnet while the offset of each anchor to the ground truth regresses by the box regression. The small FCN is for subnets and parameters sharing across the individual network. Unlike most of the past works, the model employs a class agnostic Bbox regressor and is observed to be the same effect. The model can train simply, easy implementation and faster converge. The model achieved better accuracy and speed than the two-stage methods. The new loss function is introduced by the RetinaNet that pushed the

advanced ways of object detector [37].

YOLOv3: The incremental improvements of previous versions of YOLO [37] [38] is YOLOv3. Redmon et al. used a Darknet-53 as a backbone for the feature extraction. YOLOv3 incorporated many techniques such as multi-scale training, data augmentation and batch normalization, among others. The softmax is used as a classifier instead of the logistical classifier. Although the YOLOv3 is faster than the YOLOv2 [37] while it lacked the groundbreaking change of its predecessor. Although the accuracy of YOLOv3 is lesser than the one-year-old detector [27].

CenterNet: Zhou et al. [43] proposed a novel approach in which the object is considered a point instead of a bounding box representation. CenterNet utilizes the center of the bounding box that predicts the object as a point. The image input generates the heatmap using the FCN, whose peaks generate the detected center point of objects. It uses an ImageNet pre-trained stacked Hourglass-101 [44] for the feature extraction and has three heads: Heatmap Head (for object center point), Dimension Head (for object size estimation) and Offset (to correct object point offset). The loss function is the combination of multitasking loss (all three heads) during training, while the offset head output is used to represent the object point and generate the bounding box. There is no need for NMS for post-processing because the prediction result is a point instead of a bounding box.

CenterNet makes a year of progress in the object detection field, which is a fresh perspective. It takes less inference time than the previous version of YOLO and provides more accurate detection. The CenterNet has high detection results

for multiple applications like keypoint estimation, 3D object detection, instance segmentation, orientation detection, pose estimation and others. However, CenterNet required different architecture as a backbone network that is well performed for other detectors while poor for it and vice versa.

EfficientDet: EfficientDet [45] generates the scalable detector with higher efficiency and accuracy. It proposes multi-scale features, model scaling and BiFPN. Bidirectional feature pyramid network (BiFPN) for learnable weights with input cross-connection at different scales. NAS-FPN [46] improved complex network and heavy training by adding lateral connection and removing one node input. EfficientNet removes the less efficient nodes and increases feature fusion at a high level. Unlike the existing method of detection that scale up with stacking FPN or deeper backbone, a bigger network, EfficientDet proposes a compounding coefficient that can be used to "jointly scale up all dimensions of backbone network, BiFPN network, class/box network and resolution" [45]. EfficientNet [47] is used as a backbone network in EfficientDet which has multiple BiFPN layers stacked in series for feature extraction. Each output of the final BiFPN layer is passed for box prediction and class. The model is trained with an SGD optimizer and synchronized the model with Swish activation [48] and batch normalization instead of standard ReLU, which has better performance, is differentiable and more efficient. It achieves better accuracy and efficiency than existing detectors while being computationally cheaper and small. This method is well generalized for other applications, easy to scale and is for single-stage object detection.

YOLOv4: YOLOv4 [36] has many extraordinary ideas to develop an easy and fast model to train the detector that could perform in a real-time production system. The "bag of freebies" i.e., methods that increase the training time without affecting the inference time. YOLOv4 has used data augmentation methods, class label smoothing, CIoU-loss [49], Cross mini-Batch Normalization (CmBN), Self-adversarial training, Cosine annealing scheduler [50] and other tricks to improve training. Methods that only affect the inference time, called "Bag of Specials" are also added to the network, including Cross-stage partial connections (CSP) [51], SPP-Block [52], Mish activation [53], Path aggregated network (PAN) [54], Multi-input weighted residual connection (MiWRC), etc. It utilizes the genetic algorithm to search hyper parameters. YOLOv4 has a pre-trained ImageNet and CSPNetDarknet-53 as a backbone, YOLOv3, PAN and SPP block as detection head. Multiple GPUs are needed to train most existing methods, while A single GPU is sufficient to train the YOLOv4. YOLOv4 is two times fast as EfficientDet with comparable performance.

SwinTransformer: Transformer [55] is adopted the self-attention mechanism and used primarily for the Natural Language Processing (NLP) applications and domain hence its inception. The variant of transformer network are GPT (Generative Pre-trained Transformer) [56], BERT (Bidirectional-Encoder- Representation from Transformers) [57], T5 (Text-To-Text-Transfer-Transformer) [58] etc. used in different state-of-the-art papers. The Transformers use attention to establish dependencies among the sequence elements and use longer features than other sequential

architectures. The success of transformer networks in NLP has grown interest in computer vision applications. The CNN is very popular in computer vision with advancement but has shortcomings such as fixed post-training weights [59], lack of importance of global features etc. Swin Transformer [55] is a transformer backbone network for computer vision applications. It divides the image input in non-overlapping multiple patches and creates embedding to convert it. Numerous Swin Transformers is used 4 stage patches to maintain the hierarchical representation by reducing the number of patches with each successive stage. The swin transformer block used an alternating shift patch of successive blocks to compose local multi-headed self-attention (MSA) modules. The window shifted enables the cross-window connection with linear computation complexity with the size of an image in local self-attention. [60] introduced the work of shifted window that increases detection accuracy with small overhead. Swin Transformer surpassed the state-of-the-art techniques on MS COCO but used higher parameters comparatively convolutional models.

2.2.1.3 Summary

The deep-learning-based object detection methods have become more significant and widespread compared to pioneer methods. The pioneer methods need to follow a long-step process, while the two-stage method of deep-learning approaches is also needed to generate region proposals and then extract the feature from each

region. However, the single-stage method is more prominent than these two in respect of both time and accuracy. A comprehensive analysis of a few methods is given in Table 2.1.

TABLE 2.1: Comprehensive details of few existing methods for object detection using Deep-learning.

References	**Datasets Used**	**Features Type**	**Methods**	**Challenges handled**	**Limitations**
[61]	KITTI [1], Caltech	Multi-scale Features	Multi-scale CNN with Receptive field	Reduce the memory and computational cost, for small object detection	Can not detect in real-time
[62]	KITTI [1]	Deep CNN	Dense Convolution Neural Network	Scale Senstive, for a large variance of scale	Very slow for real-time system
[63]	UA-DETRAC, BDD	Enhanced Features	Assembled comprehensive enhance features with SSD	Small Object detection, scale invariant	Computationally expensive
[64]	Pascal VOC 2007,12 ,MS COCO	Transfer connection block (TCB), converting the features from the ARM to the ODM for detection	Filter out negative anchors and used refined anchors to the detection and regression	High accuracy with efficiency	More computational task
[65]	Pascal VOC 2007, MS-COCO	Robust features	SSD+ Receptive field block	Computational cost	Suffer from class imbalance problem
[66]	KITTI [1], BDD [2]	Multi-scale features	Using Gaussian parameters and redesigning the loss function	Location uncertainty	Using the high-resolution image as an input for high accuracy
[67]	MS-COCO	Multi-scale features	Generate Focal loss by reshaping the standard cross-entropy loss	Class imbalance	Using anchor increases the computational cost
[68]	MS-COCO	Single scale features.	Avoid all hyper-parameters related to anchor boxes and very sensitive to the final detection performance using Post-processing and NMS.	Anchor free to provide fast result	Not more efficient for the Class imbalance problem
[69]	MS-COCO, BDD	Enhanced comprehensive features	SSD + comprehensive features	Computational cost	Class imbalance.
[70]	KITTI, BDD	Multi-scale features	Using Extended focal loss by reformulating the standard softmax cross-entropy loss	The class imbalance between foreground and background objects	Slow needs to be faster for the real-time system.

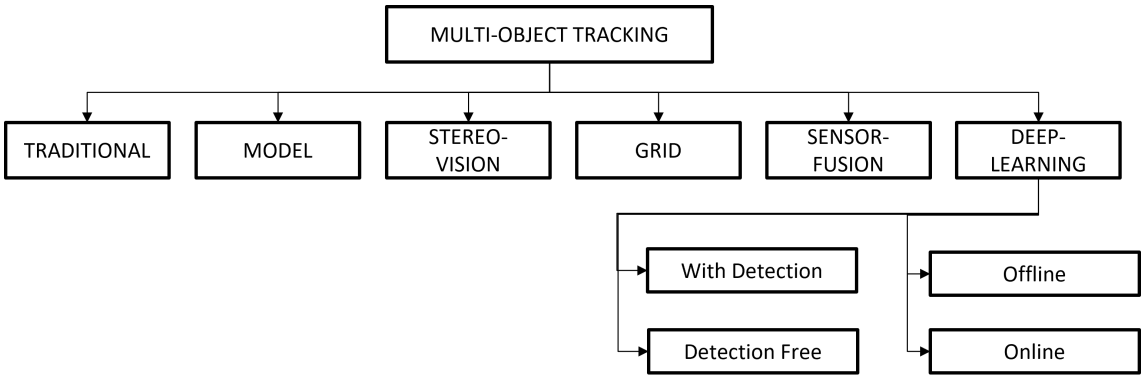

FIGURE 2.3: Taxonomy of Multi-Object Tracking methods

2.2.2 Literature Review of Multi-Object Tracking

The task of the Moving Objects Tracker subsystem (also known as Detector and Tracker of Moving Obstacles – DATMO) is to detect and track moving road agents (vehicles and pedestrians) around the surrounding area AVs. The object tracking subsystem is responsible for deciding on the vehicles' further actions to avoid collisions and abide by the traffic rules. The position of moving objects is estimated from the data captured by sensors such as RADAR and LIDAR or Stereo and Monocular cameras. Monocular camera images are very useful in providing rich appearance features to explore improving the moving object hypothesis. The sensor measurement uncertainty resolved with Bayes filters (example, Particle and Kalman filter) is employed for state prediction. There are various techniques for the MOT in the literature. These methods can be categorized into traditional, model-based, grid map, stereo vision-based, sensor fusion-based and deep-learning-based techniques. The most current developments and pertinent literature from the previous ten years are covered here. Fig -2.3 depicts the Multi-Object Tracking taxonomy.

2.2.2.1 Traditional Multi-Object Tracking

The three steps are followed in the traditional-based MOT; data segmentation, association and filtering [71]. The sensor data can be segmented using pattern recognition and clustering in the data segmentation phase. In the data association phase, associate the segmented data with the targets (moving road-agents) using data association methods. The geometric mean of the data is used to estimate the position of the target in the filtering phase. The estimated position is updated by particle or Kalman filter. Amaral et al. [72] introduced a traditional method and segmented the 3D LIDAR point into data clusters using Euclidean distance. This cluster is used to detect and track moving vehicles using 3D LIDAR sensors. The observed clusters (obstacles) of the current scan clusters are associated with the observed cluster of the previous scan clusters using the nearest neighbor algorithm. The vehicles are defined as moving objects if the velocity of the vehicle is greater than a given threshold. Zhang et al. [73] using 3D boundary boxes for each object and different boxes for different clusters to determine whether objects are distinguishing. An optimization technique is used to solve the data association. The employed of multiple hypothesis tracking is used to mitigate the association error. Hwang et al. [74] used monocular camera images to filter out 3D LIDAR points that do not belong to moving objects (vehicles, cyclists and pedestrians). The feature extraction from 3D Lidar points and images are used to match the segments to perform the object tracking after filtering. Model-Based MOT The three terms for model-based

MOT where the physical models for sensors, geometric models for objects and non-parametric filters (particle filters) are used to infer the model-based MOT directly [71]. There is no separate need for object segmentation and association because geometric models of objects associate data with targets. Petrovskaya et al. [75] introduce the model-based technique for detecting and tracking moving road-agents adopted by the AVs, Junior (the second place in the 2007 DARPA Urban Challenge by the car of Stanford University). Dynamic road-agent uses detected hypotheses between two consecutive frames over LIDAR data. The new sensor data are included the updated state of the vehicle target and the combination of vehicle geometry and pose. The hybrid formulation combines the Rao-Blackwellized particle filter (RBPF) and Kalman filter. The revised work of Petrovskaya and Thrun by He et al. [76] introduces the combination of RBPF and the geometry fitting and motion estimate throughout the complete tracking process by the Scaling Series Particle Filter (SSPF). The tracked variable represented by the geometry predicts the current state using its past state.

2.2.2.2 Stereo-Vision Based MOT

The stereo-vision-based method is known for the depth and color information obtained by the images stereo pairs for tracking and dynamic road agents in the surrounding. Ziegler et al. [20] explain the modified architecture of the Mercendes-Benz S-Class S500, Bertha, that automatically drove on the route (Historic Bertha-Benz-Memorial-Route). For MOT, Semi-Global Matching (SGM) is used to reconstruct

the dense disparity images from pair of stereo images. Thin sets predict all road-agents of the 3D surroundings of AVs and vertically oriented rectangles are known as stixels and superpixels. The Kalman filter is used to track the stixels. Finally, the stixels are classified into shape, motion and spatial constraints of dynamic road-agents and static backgrounds. Chen et al. [77] calculate the disparity map from the pair of stereo images using a semi-global matching algorithm. The disparity map assisted in simple linear iterative clustering to generate the image segmentation boundaries. The simple linear iterative clustering is classified into occlusion, hinge and coplanar. The RANSAC algorithm uses ego-motion estimation to obtain the moving points (RANdom Sample Consensus). Finally, dynamic road-agents are extracted by superpixels merging according to their movements and boundary types.

2.2.2.3 Grid-Based MOT

Azim et al. [78] used an octree-based 3D local occupancy grid map that divides the environment into free, unknown voxels and occupied. A local grid map is constructed to detect the moving road-agents based on the inconsistency between occupied and observed free spaces in the local grid map. Moving road-agents are divided into known categories (buses, cars, bikes and pedestrians) using geometric features extracted from each layer. Ge et al. [79] used a 2.5D occupancy grid map to model the dynamic road-agents and static background. The average height of 3D points is stored in a grid cell during the cell space domain for 2D projection. The motion prediction is detected from discrepancies between the background model and

the current grid.

2.2.2.4 Sensor-Fusion Based MOT

The sensor fusion-based model is used to fuse the different sensor data (like LIDAR, camera and RADAR) to explore the characteristics of each sensor individually to improve the surrounding perception. In DARPA challenge 2007, Darms et al. [80] introduced a sensor fusion model for detecting and tracking the dynamic road-agents for the AVs. The subsystem of MOT is divided into two parts. The sensor layer extracts the features from the sensors that can be used to represent the dynamic road-agents either using the boundary boxes or center points. The sensor layer is also responsible for feature association with the currently predicted hypothesis from the fusion layer. The feature generates new proposals if it cannot be associated with an existing hypothesis. Na et al. [81] combine the tracks of dynamic road-agents produced from different sensors, such as 2D-3D LIDARs and Radars. Data from 2D LIDAR is projected on a 2D plane and dynamic road-agents are tracked using Joint Probabilistic Data Association Filter (JPDAF). The region growing algorithm is partitioning the image into moving road-agents by projecting the 3D LIDAR data onto the image.

At last, the interactive closest points (ICP) matching or data association are used to estimate and update the track poses. Xu et al. [82] proposed a context-aware tracking for dynamic road-agents that maintain the distance used by the AVs of the (CMU) Carnegie Mellon University. A Region of Interest is produced in the road

network for the given behavioral context. The target candidates existing in ROI are found and projected into road coordinates. To obtain the distance maintaining targets, all candidate targets are associated from different sensors (RADAR, Camera and LIDAR). Xue et al. [70] fuse the camera and LIDAR data to detect the pedestrian more accurately. The false detections are reduced with previous knowledge of pedestrian height. They use the pinhole camera to estimate the pedestrian height, combining the LIDAR and Camera data.

2.2.2.5 Deep-Learning Based MOT

The deep neural network are used to detect the position and dynamic road-agents geometries, tracking their future position by using current camera data in deep-learning approaches. Huval et al. [83] proposed an overfeat convolutional neural network (CNN) to detect the dynamic vehicles. Mutz et al. [84] presented the model for tracking the dynamic road-agents for various applications known as "follow the leader," which is very relevant for autonomous vehicle convoys. The top of the GOTURN (Generic-Object-Tracking-Using-Regression-Networks) is responsible for the tracking method. Held et al. [85] introduced a pre-trained network to track generic objects without further fine-tuning and training. Initially, image input and manually determined boundary boxes of the vehicles are passed through GOTURN. The center of bounding boxes is assumed as the object of interest. Subsequently, GOTURN estimates the geometry (width and height) and location of the Bbox as an output for the new image.

Detection based Multi-Object Tracking

In detection-based tracking, the two consecutive video frames are passed through a pre-trained object detector that generates the boundary boxes for multiple objects and this detected object is used as an input for the object tracking. This method is very popular because it terminates the disappearing object and detects the new object which enters the scene.

Detection-free Multi-Object Tracking

This method needs the initialization of a fixed number of objects manually and then localizes the object in other subsequent frames. When any new object enters the frame, then this method can not deal with this and is not able to localize them.

Online Object Tracking

This method is applied when the prediction is needed immediately. This method can not use the future frame to improve the results.

Offline Object Tracking

Offline tracking method is applied in frame sequences of recorded video.

2.2.2.6 Summary

Multiple multi-object tracking methodologies are examined in the section above based on architecture. The deep-learning-based approaches have different categories and the methods for that categories are mentioned in Table- 2.2. The observations during literature are as follows: the detection-based methods have more prominent

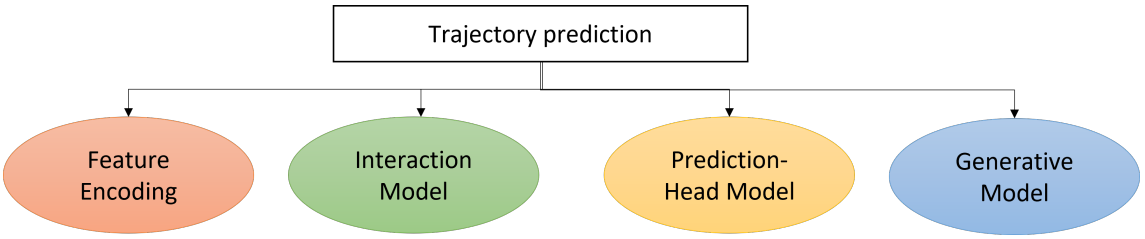

FIGURE 2.4: Taxonomy of the Trajectory Prediction Models

results than detection-free methods. Although offline methods provide more significant results than offline methods, the online method is more beneficial for real-time systems.

2.2.3 Literature Review of Trajectory Predition

To achieve better prediction performance, researchers propose novel models with various architectures, such as Multi-Layer Perception (MLP), Convolution Neural Network (CNN), Recurrent Neural Network (RNN), and Graph Neural Network (GNN). Several typical design choices are compared with feature encoding, interaction modeling, and prediction head. Some approaches that adopt the generative model are also reviewed.

2.2.3.1 Feature Encoding based TP

Considering the trajectory as sequential data, a lot number of papers [99] [100] [101] [102] [103] [104] [105] introduce feature encoder using RNN like the LSTM and GRU [106]. Some combine approach with 1D-conv or MLP with RNN extract the features of the scene context input [107] or the trajectory input [108] [107] [109].

TABLE 2.2: Comparison details of few existing methods for Multi-Object Tracking using Deep-Learning Approaches

S.No.	Year	Author	Model Used	Approaches			
				With Detection	Detection Free	online	offline
1	2019	Hefeng Wu et al. [86]	Multi-branch Neural Network	Y	N	N	Y
2	2016	Ricardo Sanchez-Matilla [87]	Density particle filter for probabilty hypothesis.	Y	N	Y	Y
3	2017	Min Yang [88]	Data association with multi commodity, local target-specific method	Y	N	Y	N
4	2017	Amir Sadeghian [89]	Recurrent Neural Network for data association	Y	N	Y	N
5	2015	Yu Xiang [90]	markov decision process+similarity function	Y	N	Y	N
6	2019	Jiarui Xu [91]	an unified model to calculate similarity with spatial-temporal domain	Y	N	Y	N
7	2019	Peng Chu [92]	Instance-aware tracking for integrating single object tracking in multi-object tracking	Y	N	Y	N
8	2019	Philipp Bergmann [93]	Appearance Model +Re-Id	Y	N	Y	N
9	2017	Seung-Hwan Bae [94]	Object model learning (Appearance Learning) and confidence based association	Y	N	Y	N
10	2017	Long Chen [95]	Kalman Filter+Re-ID+Heirarchical Data Association	Y	N	Y	N
11	2018	Kuan Fang [96]	Recurrent Autoregressive Network	Y	N	Y	N
12	2020	Bo Pang [97]	TubeTK with using bounding tube for spatial temporal localization of objects	Y	N	Y	N
13	2021	Yifu Zhang [98]	CenterNet based object detection +Re-ID	Y	N	Y	N

Some researcher have more attention in transformer network for feature extraction [102], [110][86] after the great success in natural language processing. Some approaches [111] [103] [109] [112] [113] [114] [115] directly borrow convolutional feature encoder from object segmentation [116] tasks and object detection [28] [117]. VectorNet [118] adopts the idea of PointNet [117] and extracts the instant-level features for the vectorized input. TNT [117] directly adopts the backbone of VectorNet for feature extraction. The author uses the discrete input as the point cloud and mimics the point cloud encoding in their work [119]. Lane RCNN [120] and Lane GCN [121] propose graph convolutional neural network modules at the encoding stage and aggregate the features across the graph constructed based on road connectivity or temporal sequence.

2.2.3.2 Interaction Modeling Based TP

The interaction among the road-agents such as vehicles and pedestrians is extremely crucial yet complicated. The interaction between agent-to-scene and agent-to-agent modelled at a time, researchers enhance the social pooling method as introduced in [117] [118] by adding the scene context feature maps [104] [108] [114]. Many researchers [101] [102] [103] [105] [107] [109] motivated with success of the attention mechanism design the interaction modeling module. Furthermore, Liu et al. [110] and Ngiam et al. [122] build the entire interaction module with multi-head attention. The GNN layer and Graph modeling are also frequently involved because the message passing of GNN can fuse the features of different agents (and road elements).

According to the theory, the interaction between road-agents and the behavior of a feature in fusion is the same. There exists an exception that TPCN [119] does not explicitly consider modeling the interaction but still achieves a good performance.

2.2.3.3 Prediction Head Based TP

Few researchers define the hidden markov model based prediction and produce the trajectory prediction with RNN [123] [101] [102] [103] [104] [105]. Some other researchers consider the prediction like regression process and features decoded with MLP in [118] [100] [107]. The concept of anchor proposal influence significantly [105] [23] in the computer vision field, most of the researchers include a classification branch in prediction head to predict a confidence score or probability for each proposed trajectory anchor [102] [103] [109] [120] [121] [124]. Motivated by the prediction uncertainty of the deep-learning model against the unseen cases, CoverNet [115] and PRIME [125] further deploy a model-based planner to propose trajectory anchors that satisfy the constraints imposed by the kinematics of the vehicle and the scene context. The uncertainty in the deep-learning-based prediction model against the unseen cases motivated the researchers to propose models like CoverNet and PRIME further implement a model-based planner to introduce trajectory anchors satisfies the imposed constraints of the scene context and the kinematics of the vehicle.

2.2.3.4 Generative Model-Based TP

SMART [114] and DESIRE [104] consider the trajectory prediction as selection process and conditional sampling. Firstly, the recognition module projects the observation of trajectory and ground truth prediction to the latent space. The z is a latent variable assumed to satisfy a distribution prior parameterized by the encoding of the recognition module. The latent variable z and given conditional input is used to recover the future trajectory from the conditional decoder. However, one cannot obtain the likelihood of each trajectory sampled from the generative model. DESIRE designed the ranking and refinement module to evaluate the trajectories generated from the CVAE module.

2.2.3.5 Summary

To achieve better prediction performance, the Researcher proposed various novel methods using CNN, RNN and GNN etc. The trajectory prediction, as indicated above is classified according to design choices. The interaction model is suitable for the autonomous vehicle application because the interaction between the road-agents can provide more semantic information about the surroundings.

TABLE 2.3: Comprehensive details of few existing methods for Trajectory Prediction using Deep-learning.

Author	Year	Datasets Used	Methodology	Advantages	Limitations
L Paparusso et al. [126]	2021	Nuscenes	Using LSTM, fuses road geometry information with past dynamics of driver-vehicle system	*Itisbasedonaflexibleproblemformulation,suchthattheusercanchooseagenericsetc*	It do not define a minimal set of drivers and tracks so it has less the generalisation capabilities of the neural network on any new driver/track configuration.

Author	Year	Datasets Used	Methodology	Advantages	Limitations
Park et al. [127]	2020	Nuscenes	distribution of multiple modes, social behaviour of road agents and conditioned on the scene context. Capture agent-to-agent and agent-to-scene interactions for each agent interest	This model learns agent-to-agent interactions and agent-to-scene interactions using attention mechanisms, resulting in better prediction in terms of precision, diversity, and admissibility.	Since this model learns agent-to-agent interactions and agent-to-scene interactions it has better precision and better prediction but it is less time efficient.
Cheng et al. [128]	2020	ETH, UCY, Standford Drone	Two LSTM are used for temporal feature extraction with spatial context. The Conditional Variational Auto-Encoder (CVAE) module is used to encode the spatial-temporal features	It is effective for predicting accurate trajectories for homogeneous agents in various real-world traffic scenes, even without modeling interactions explicitly and it's ranking method correctly estimates the multiple predictions and recommends a reliable candidate for the single-path trajectory prediction task	This model do not have the method for learning the impact from environment/static context, e.g., space layout and scene deployment, to further enhance the performance of trajectory prediction.
Styles et al. [129]	2020	ETH, UCY	Multi-camera trajectory forecasting (MCTF), where the future trajectory of an object is predicted in a network of cameras. It not only considers forecasting trajectories in a single camera view but also this work is the first to consider the challenging scenario of forecasting across multiple non-overlapping camera views	It implements 3 purely learned approaches using the normalized bounding box coordinates as inputs : Fully connected network, Long short-term memory, and Gated recurrent unit. Each camera uses a separate classification network.	It focus on this next-camera prediction problem only. It do not focus on full trajectory information which is available in WNMF. It is unable to do further improvement in using recurrent models over a fully-connected network .
Liu et al. [130]	2020	ETH,UCY	Introduced a social contrastive loss that encourages the encoded motion representation to preserve sufficient information for distinguishing a positive future event from a set of negative ones	This method suggests that incorporating negative data augmentations by means of contrastive learning can be a promising alternative to conventional interactive data collections for building robust neural motion models	In this method the random negative sampling is not able to provide any significant performance gain.
Ivanovic et al. [131]	2020	Standford Drone	Mixtures of Affine Time-varying Systems (MATS) used as an output representation for trajectory forecasting that is more amenable to downstream planning and control use.	It is beneficial as it leads to an easier integration with downstream planning and control algorithms as well as being more immediately interpretable since all values are in the original (not latent) state space	In this model feasibility is also not guaranteed since it considers a finite horizon, it also does not currently guarantee safe behavior and leads to uncertain predictions of other agents

Author	Year	Datasets Used	Methodology	Advantages	Limitations
H. Park et al. [132]	2020	Standford Drone	It is a model that addresses the lack of diversity and admissibility for trajectory forecasting through the understanding of the multimodal environmental context. It is a new annotation-free approach to estimate the true trajectory distribution based on the drivable-area map.	It tackles the task of diverse trajectory forecasting with a special emphasis on admissibility in dynamic scenes and proposes a new task metric that specifically assess models on the basis of these attributes.	It is unable to capture complex interactions in the high dimensional input space and propose methods to explicitly model these interactions via sophisticated attention mechanisms.
Liang et al. [133]	2020	Nuscenes	Using multi-view 3D simulation data for training. This method utilizes the hard camera view in training and enabling the model to learn the robust representation that can generalizes to unseen camera views.	It achieves comparable or even better performance than other state-of-the-art models that are trained on in-domain real videos.	In fewer number of data and less diverse views, the performance drops suggesting that it should use more views
Weng et al. [134]	2020	Standford Drone	Graph neural network is used to capture the interactions between the roag-agents. The GNN is able to model complex hierarchical interactions, improve the discriminative feature learning for MOT association, and provide socially-aware context for trajectory forecasting.	The accuracy metrics are increased and reach the highest performance with two layers of GNN, showing the effectiveness of GNN feature interaction. This method has a higher sample efficiency and can cover different modes of the future trajectory distribution.	It perform feature extraction to get impressive performance for each object independently, but ignores interaction between the objects.
Graber et al. [135]	2020	Argoverse	A "Dynamic Neural Relational Inference" (dNRI), model that recover the agent interactions at every time step.it predicts static relation between agents and provides an interpretable representation of the dynamic agents.	Since static relations improperly model the data. In response to this, it develop a Dynamic Neural Relational Inference (dNRI), which incorporates insights from sequential latent variable models to predict separate relation graphs for every time-step.	It do not adapt additional methods used by recent sequential latent variable models, such as auxiliary loss functions, to further improve the performance.
Monti et al. [136]	2020	Standford Drone	A new recurrent generative model for both single agents future goals and interactions between different agents. The double attention based graph neural network extracts mutual information among different agents.	This model keeps in consideration the coordination between the different subjects. It uses an attentive module to associate importance weights to different interactive nodes	In case of complex trajectories, the predictions do not always precisely resemble the expected output. it also fail to generalise properly and struggle in complex scenarios, limiting the results in terms of prediction accuracy.

Author Year	Datasets Used	Methodology	Advantages	Limitations
Kothari 2020 et al. [137]	Argoverse	Two knowlwdge based data driven methods to capture the socail interactions have been proposed effectively.	This proposed designs show significant gains in reducing model prediction collisions. This models outperform competitive baselines on TrajNet++ synthetic dataset by benchmarking against several popular interaction module designs in the field.	This model impose not only strong priors but also have limited capacity when modelling complex interactions. The filter does not model social interactions so it is associated with high error of EKF. A significant yet missing component in this field is an objective and informative evaluation of these interaction-based methods.
Mangalam 2020 et al. [138]	ETH,UCY, Argoverse, Nuscenes	Predicted Endpoint Conditioned Network (PECNet) for flexible human trajectory prediction. Aditionally a simple "truncation trick" for improving diversity and multimodal trajectory prediction performance is presentedIt generates socially acceptable trajectories via a learned reward function.	These socially-aware approaches do not take into account the pedestrians' ultimate goals, which play a key role in shaping their movement in the scene. They ignore the presence of other pedestrians in the scene which is key for predicting shorter term motions which are missed by just considering the environment.	In case of complex trajectories, the predictions do not always precisely resemble the expected output.It is unable to capture complex interactions in the high dimensional input space.
Giuliari 2020 et al. [139]	ETH,UCY	LSTM,Transformer and large BERT	The proposed Transfomers have shown better long-term prediction behavior, the capability to predict sensible multiple future trajectories and the unique feature of coping with missing input observations, as it may happen when dealing with real sensor data.	In complex scenario such as TrajNet, dropping input frames impact the prediction performance : the more dropped frames, the larger the performance decrease.

2.2.4 Literature Review of Motion Planning

motion planning is kinetics features, human poses, velocities and dynamic obstacles nearly when AVs move towards the destination. Sometimes, motion planning is

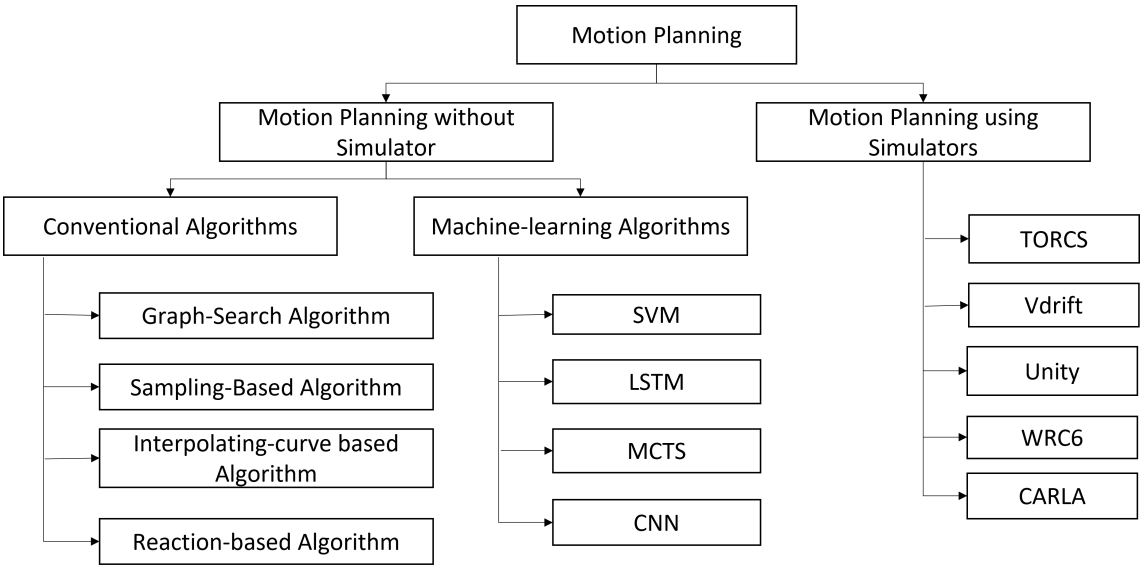

FIGURE 2.5: Texonomy of the motion planning techniques

considered a short-term sub-optimal and optimal reactive strategy to make a reactive or instant response. Motion planning is achieved by linear and rotary control in hardware (e.g., Servo, Motor) from the perspective of control and robotic engineering. On the other end, Motion Planning should achieve long-term optimal planning goals as path planning robots interact with the environment. The Motion Planning algorithms are divided into three parts. One is a Traditional algorithm, the second is Machine-Learning based algorithms and the third one is Motion Planning using simulators, as shown in Fig-2.5 according to their principles and the era they are invited.

2.2.4.1 Traditional Algorithms

The traditional algorithm is categorized into four categories.

Graph-Search Algorithm: The graph-search algorithm for motion planning

is divided into Breath-First Search Algorithm, Depth-First Search Algorithm and Best-first Algorithm. The DFS algorithm builds a search tree as fast and deep as possible from start to destination until an optimized path is found. The BFS is a search tree like the DFS that extends the tree as quickly and broadly as possible until an appropriate path is found. The best-first search algorithm added a numerical criterion (cost and value) at each edge and node in a search tree. The value calculated guides the search process to decide which branch should be extended and whether the search tree should be expanded. The build search tree process is repeated until an appropriate path is found. The most popular algorithms are the Dijkstra algorithm [140] and A* algorithm [141] which are used to compose the Graph-Search Algorithm.

Sampling-Based Algorithm: Sampling-Based algorithm is random samples a fixed workspace to generate sub-optimal paths. The RRT and PRM (probabilistic roadmap method) are common algorithms used in motion planning. The RRT algorithm is more popular and widely used for commercial and industrial purposes. This method constructs a tree that attempts to explore the workspace rapidly and uniformly via a random search [142]. The RRT algorithm can consider non-holonomic constraints, such as the maximum turning radius and momentum of the vehicle [143]. The PRM algorithm [144] is normally used in a static scenario. it is divided into two phases: Learning Phase and Query Phase. In the learning phase, a collision-free probabilistic roadmap is constructed and stored as a graph. in the query phase, a path that connects original and targeted nodes is searched from the probabilistic

roadmap.

Interpolating Curve Algorithm: This process constructs or inserts a set of mathematical rules to draw trajectories. The interpolating curve algorithm is based on techniques e.g., Computer Aided Geometric Design (CAGD), to draw a smooth path. Mathematical rules are used for path smoothing and curve generation. Typical path smoothing and curve generation rules include line and circle [145], clothoid curves [146], polynomial curves [147], Bezier curves [148] and spline curves [149].

Reaction-Based Algorithm: Unlike graph-search algorithms that cost a long time to plan high-level or global-level paths, reaction-based algorithms are about making reactions or doing local path planning quickly and intuitively, as the description of algorithms in reactive architecture [150]. Here three reaction-based algorithms that are widely used in engineering and manufacturing are presented: potential field method (PFM), velocity obstacle method (VOM), and DWA.

2.2.4.2 Machine-Learning and Deep-Learning Based Algorithm

Here, basic principles of four classical but pervasive ML algorithms for motion planning are presented. These algorithms include three supervised (SVM, LSTM and CNN) and one Reinforcement Learning (Monte-Carlo Tree Search) MCTS.

Support Vector Machine is a well-known supervised learning algorithm for classification. The basic principle of SVM is drawing an optimal separating hyper lane between inputted data by training a maximum margin classifier [151]. the inputted data is in the form of the vector obtained by performing trained classifier

SVM is used in 2-class classification that can no suit the real-world task, but its variant multi-class SVM [152] works.

Long-Short Term Memory [153] [154] is a variant of RNN. LSTM can remember inputted data (vectors) in its cells. Because of the limited capacity of cells in storage, a part of data will be dropped when cells are updated with past and new data and then a part of data will be remembered and transferred to the next time step. These functions in cells are achieved by a neural network. In robotic motion planning, the robot's features and labels in each time step are fed into neural networks in cells for training. Therefore, the decision for motion planning is made by performing a trained network.

Monte Carlo Tree Search is a classical RL algorithm and it is the combination of Monte-Carlo method [155] and search tree [156]. MCTS is widely used in games (Go and Chess) for motion prediction. MCTS mechanism comprises four processes: selection, expansion, simulation and backpropagation. In AVs motion planning, the node of MCTS represents a possible state of the robot and stores the state value of the robot in each step.

The first selection is to choose some possible tree nodes based on known state values. The second tree expands to an unknown state by tree policy (e.g., random search). The third expansion simulation is made on a newly expanded node by default policy. Until the terminal state of the AVs and reward R is obtained. Finally, backpropagation is made from the newly expanded node to the root node and state

values in these nodes are repeated until the convergence of state values in the tree. The AVs can therefore plan their motion according to state values in the tree. MCTS fits discrete-action tasks e.g., AlphaGo() and time-sequential tasks like autonomous driving.

Convolutional Neural Network 1998 has become a research focus of ML after LeNet-5 [157] was introduced and successfully applied into hand written digits recognition. CNN is one of the essential types of Neural networks because it is good for extracting high-level features from high-dimensional, high-resolution images by convolutional layers. CNN makes the AVs avoid obstacles and plans motions of AVs according to human experience by models trained in forwarding propagation and backpropagation processes, especially the backpropagation. In the back propagation, a model with a weight matrix/ vector θ is updated to record features of obstacles. Note that $= (W_i, b_i)_i^L$ where w and b represent weight and bias and i represents the serial no. of w-b pairs. L represents the length of weight. The training images of obstacles are used as inputs for CNN. Outputs are probability distributions obtained by the softmax function. Loss value $Loss_{CE}$ is cross-entropy (CE) and that is obtained by

$$Loss_{CE} = -\sum_i p_i . \log q_i \tag{2.1}$$

Where p represents the output probability distribution (observed real value), q is defined the expected probability distribution $(p, q\epsilon(0, 1))$ and i represents the serial number of each batch of images in training. The weight is updated in optimizer by minimizing the loss value using gradient descent approach. Therefore, new weight

w_i^{new} is obtained by

$$w_i^{new} = w_i - \eta . \frac{\partial loss}{\partial w_i} \tag{2.2}$$

where w represent the weight, ∂ represent the Learning rate and i is used to represent the serial number of each batch of images in training. Improved variants of CNN are also widely used in Motion Planning. e.g. Residual Network [158] [159].

2.2.4.3 Motion Planning using simulator

Although the model without a simulator produces notable results in real-time applications, its implementation has numerous obstacles and difficulties on public roads. Even though end-to-end models have made significant strides in practical applications, many obstacles still prevent their full autonomy on the road. Many of these difficulties are caused by AVs being rarely trained or tested in all conditions (including corner cases). However, personally developing these scenarios and gathering data in the actual world may be a time-consuming and frequently unrealistic procedure.

As a highly portable multi-platform car racing simulation, TORCS [160] has a multi-platform high portable car racing simulation which can be used like ordinary car racing game, but it is also used for the research platform. The researchers first attempt vehicles control on TORCS in an end-to-end manner [161] using Reinforcement Learning. For the vision based driver's perspective, the competition [162] is later processed using CNN and RNN. Later, many end-to-end Reinforcement

learning algorithms have been used to control the vehicle's motion on TORCS simulator, such as deep deterministic policy gradient (DDPG) [163] [164] [165], safeDAgger [166], and A3C [167] etc. The DDPG algorithm is the first to handle the complex scenarios in [168] and action spaces in the continuous domain. This is very useful to produce continuity in action space that can adapt to the complex real-world driving scenarios.

Both CARLA [169] and Unity [170] is used very efficiently to create realistic simulation environments with more sensors and physical complexity. The major difference between CARLA and Unity simulators are that CARLA being an explicit driving simulation while the basic Unity is a more generic engine and does not describe a particular realization. CARLA has been developed to support training, validation and development for AVs that can also generates an interface allowing to control the vehicles and interact with a dynamic environment in town 1 with Hard Rainy. Since a vehicle trained end-to-end to imitate an expert cannot be controlled at test time (i.e., cannot take a specific turn at an upcoming intersection), a condition imitation learning approach was proposed in [171] to enable the learned driving policy to behave as a chauffeur that handles sensory motor coordination but also continues to respond to navigational commands. To alleviate the inefficiency of exploring large continuous action spaces that often prohibits the use of classical RL in challenging real driving tasks, [172] proposed a controllable imitative RL based on DDPG to explore over a reasonably constrained action space guided by encoded experiences that imitate human demonstrations.

The experiments on CARLA simulator demonstrate the superior performance withe respect to route completion and effective generalization capability in unseen surroundings. Similarly, the learning strategies of agent for acting in unstable and complex environments and to achieve better robustness of agents,an advantage actor-critic algorithm was implemented in [173] with multi-step returns. High-fidelity realistic driving environments. The CARLA simulator has a pre-made and high fidelity to provide a benchmark for comparison of performance.

2.2.4.4 Summary

Although the end-to-end model has significant results in real-time applications, and these methods still have many challenges. These challenges arise from the fact that they rarely occur. The simulator can gain all the experiences for training and testing at negligible costs.

TABLE 2.4: Comparison details of few existing methods for Motion Planning using Deep-Learning Approaches

S.No.	Year	Author	Algorithm Used	Advantages	Limitations
1	2018	Y Pan et al. [174]	CNN	Learn driving at low cost camera with high speed	trained only for elliptical race tracks with no other vehicles, requires iteratively building the dataset with the reference policy
2	2022	J Li et al. [175]	DQN	used to solve the low-dimension problem in dynamic enviornment and can improve the high dimension processing speed	simple and only for static enviornment
3	2018	Bansal et al. [176]	CNN with RNN	can easily transfer from simulator to real world, for trajectory deviation its robustness	perform aggressively with other road-agents in new scenario, waypoints make turns infeasible
4	2021	Zhou et al. [177]	A2C	higher success rate with faster motion planning, it converges stable and faster tahn optimal value of reinforcement learning	few goals are still missing, need to concentrate

S.No.	Year	Author	Algorithm Used	Advantages	Limitations
5	2022	Y Li. et al. [178]	DDPG	"A motion planning approach based on DDPG algorithm and artificial potential field is proposed in this paper to achieve the position and orientation alignment of the space manipulator end-effector during on-orbit operation."	DDPG can become unstable and heavily dependent on searching the correct hyperparameters for the current task. DDPG algorithm risk overestimating the Q values in the critic (value) network
6	2021	Wang et al. [179]	IVMP	The IVMP first employs a semantic map. The predicted semantic maps not only provide useful interpretable information, but also allow our motion planning module to handle objects with low probability, thus improving the safety of autonomous driving.	less vulnerability for unseen testing and different structures like turns, hills.
7	2018	M. Jartitz et al. [180]	A3C	The newly proposed reward and learning strategies lead together to faster convergence and more robust driving using only RGB image from a forward facing camera. An Asynchronous Actor Critic (A3C) framework is used to learn the car control in a physically and graphically realistic rally game, with the agents evolving simultaneously on tracks with a variety of road structures (turns, hills), graphics (seasons, location) and physics (road adherence).	it wouldn't be efficient with a GPU since the updates are not batched, and GPUs are more efficient with batched data
8	2016	Zhang et al. [181]	SAFE DAGGER	it shows that it indeed requires less queries to a reference policy. A considerable acceleration of convergence that is hypothesised to be caused by the impact of automated curriculum learning.	need of better realization of road trajectories for collision free motion planning.
9	2021	Hoque et al. [182]	LAZY DAGGER	LazyDAgger improves the performance and robustness of the learned policy during both learning and execution while limiting burden on the supervisor. Simulation experiments suggest that LazyDAgger can reduce context switches by an average of 60 percent over SafeDAgger on 3 continuous control tasks while maintaining state-of-the-art policy performance.	method is not capable for manipulation task such as cable untangling and suturing and can improve the robustness of the model.
10	2019	Jaafra et al. [173]	A2C	the deep actor-critic algorithm demonstrated higher performance and faster learning capabilities than a standard deep RL. Furthermore, the results showed a certain vulnerability of the approach when facing unseen testing conditions.	Working on to tackle the issue of non-stationary environments impact on RL methods robustness as a multi-task learning problem.

S.No.	Year	Author	Algorithm Used	Advantages	Limitations
11	2021	Chen et al. [183]	MONTE CARLO LOCAL-IZATION	The experimental results show that our method can reliably and accurately localize a mobile system in different environments and operate online at The LiDAR sensor frame rate to track The vehicle pose. Also LiDAR-based global localization is needed for autonomous driving, especially in GPS-denied environments or situations where GPS cannot provide accurate localization results.	high computational cost, unable to measure the distance through heavy rain, fog and snow

2.3 Research Gaps

2.3.1 Object detection

Object detection is used in various applications like security, surveillances, automated vehicle system and forensics. The real time object detection methods should maintain the accuracy with speed. Some of the challenges faced in the object detection are as follow

- **Detection at different scale** The common problem faced by the object detector is to detect the object at single scale or may not be detected on other bigger/smaller scale. Thus, the idea should be generate a generalize feature extraction method which can be utilize for any scale.

- **Training for different scale** To train the model at every sized input image is for the generic object detector. Most of the classifier and regressor are fully

connected layers so resizing it at run time is not possible. The training of network at one resolution is not very effective than the other.

- **Speed/Accuracy** To maintain the trade-off between accuracy and speed is the biggest challenge for the object detection that are faced by the industries. The major concern is to provide the heavy detection with cheap embedded device which can easily manage the both (speed and accuracy).

- **Class Imbalance** The total number of a class of data (positive) is far less than the total number of another class of data (negative). To overcome from the class imbalance issue, the model combined the undersampling and over-sampling on dataset to produce the equal ratio of negative (Background) and positive (foreground obstacles) sample.

2.3.2 Multi-Object Tracking

- **Occlusion** The object in question is partially or completely occluded as shown

- **Identity switches** ID Switching occurs when two similar objects overlap or blend, causing the identity switching hence, keeping track of the object id is difficult.

- **Motion Blur** Object is blurred due to the motion of the object or camera. Hence, visually the object doesn't look the same anymore.

- **View-Point Variation** Different viewpoint of an object may look very different visually and without the context, it become very difficult to identify the object using only visual detection.
- **Scale Change** Huge changes in object scale may cause a failure in detection.
- **Background Clutters** Background near object has similar color or texture as the object. Hence, it may become harder to separate the object from the background.
- **Illumination Variation** Illumination near the target object is significantly changed. Hence, it may become harder to visually identify it.
- **Low Resolution** When the number of pixels inside the ground truth bounding box is very less, it may be too hard to detect the objects visually.

2.3.3 Trajectory Prediction

- It is complex to predict such trajectories over long time horizons.
- There is uncertainty about latent variables such as drivers' motivations and goals.
- Drivers can take multiple possible decisions under the same traffic situation (multi-modality nature of the decision-making processes).

2.3.4 Motion Planning

- **Obstacle and Robot Speed** Decreasing the robot max speed or increasing the speed of obstacle is the main reason to increase the planning difficulty, in dynamic surroundings. The guarantees for safety can be possible only if the maximum speed of robot is higher than the speed of obstacles.

- **Obstacle Motion Uncertainty** The AVs has the another challenge that the prediction of obstacle motion may not be exactly. The noisy sensors of robot is the cause of uncertainty in surrounding sensing, or the obstacles stochastic behavior. e.g., human driver or pedestrian, is called as uncertainty in environment predictability.

- **Number of Obstacles** There are multiple moving obstacles operates with the robots or AVs. For the collision-free planning solution space have been reduced by the high obstacle density. The performance of planning algorithm is poor in crowded scenes.

2.4 Benchmark Datasets and Simulator used for training and evaluation

Several datasets with different classes and modality have been considered during experimental setup. A comprehensive list of datasets used in the work is mentioned below. These datasets are used in this thesis to evaluate the proposed model.

2.4.1 KITTI [1]

The most popular dataset is used for autonomous driving and mobile robotics is KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) dataset [1]. The traffic scenarios for an hour is recorded with different sensor modalities such as grayscale stereo cameras, a 3D laser scanner and a high-resolution RGB that contains by the KITTI dataset. This is very popular but not consisting the semantic segmentation ground truth. The car, pedestrian and cycle are the three classes of KITTI dataset which consists 7481 training image samples and 7518 testing image sample with 1242 $\times$ 375 resolution. The KITTI dataset divides the training sample in half for the validation and training while it does not have the GT for the Test sample.

2.4.2 Berkley Driving Dataset (BDD) [2]

The lane marking annotation detection, instance segmentation, detection of dynamic obstacles and driveable area segmentation is properly annotated in BDD dataset [2]. The road obstacles detection consists 10 classes bus, motor, rider, traffic sign, traffic light, train, truck, person and car in 100000 images for 2D Bboxes annotations. The dataset is divided in 2:1:7 ratio for testing, validation and training respectively. The geographic, weather diversity and environmental possesses by the BDD dataset which is useful for model training that are less likely to be surprised by the new conditions.

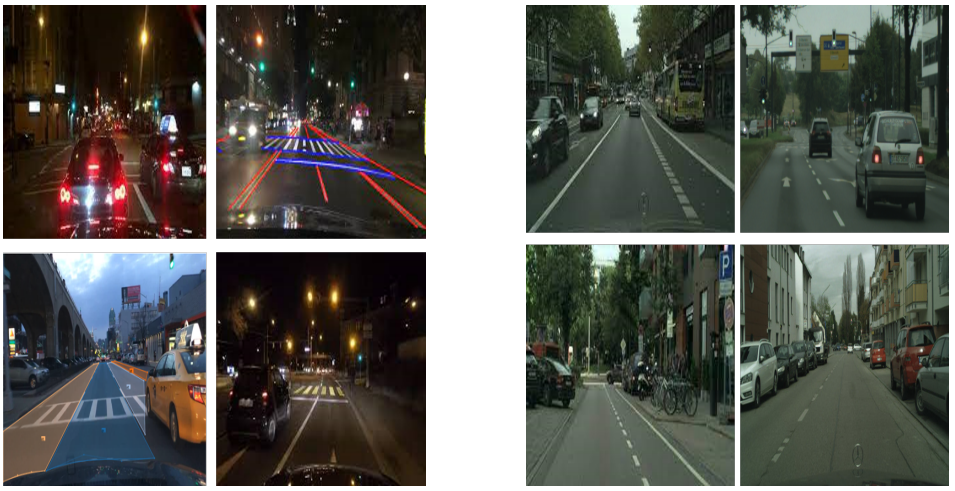

FIGURE 2.6: Visualization of datasets in different conditions (a)The night vision of BDD dataset (b) cloudy weather of KITTI dataset

TABLE 2.5: Details of Training, Testing and validation set for KITTI, BDD, Waymo, MOT variants, Argoverse, Apolloscape, Lyft Datasets

Datasets	**Training**	**Validation**	**Testing**
KITTI	86000	7000	1000
BDD	70000	10000	20000
Waymo	790k	200k	148k
MOT16	110407	-	182326
MOT17	336891	-	564228
MOT20	1336920	-	765465
Argoverse	232566	25626	66365
Apolloscale	18905	2338	4964
Lyft	98294	10831	28049

2.4.3 Waymo [3]

Waymo is a open dataset [184] that consists 5 camera which captures the immage from 5 different angles: front, left front, right front, side left and right side. The 3990 videos that contains the 790k images (3990 videos) for training, 200k validation images (1010 videos) and 148k testing images (750 videos). Three annotated class are available for evaluation.

2.4.4 Multi Object tracking (MOT) [4]

The MOT challenge datasets [4] are created for the multiple object tracking tasks. There are various variants for this dataset released in each year, like MOT15, MOT16, MOT17 and MOT20.

- **MOT 15** This is a variant of MOT dataset. It consists 11 different outdoor and indoor public place scenes with lots of pedestrians as the objects, where camer angle, image condition and camera motion vary. The ACF-based detection is used to generate the detection.

- **MOT 16** This dataset contains the existing data with few new data that has 14 challenging real-world videos of both dynamic and static scenes, 7 for testing, 7 for training. The 110407 Bboxs are using for training set and 182326 Bbox are using for testing. The strict standard is using to annotated the video sequences and get the high accurate ground-truths for semantic evaluation.

- **MOT 17** MOT17 challenge consists 7 different outdoors and indoor public place scenes with pedestrian as the obstacles. Each scene of the video is splits into 2 part where one for testing and one for training. The three detectors are used for the Bbox detection i.e. DPM, Faster-RCNN and SDP. The offline and online challenge accepted and later allowed for predict the tracks from future videos.

- **MOT 20** MOT 20 consists 8 challenging sequences of video (4 test, 4 train) of crowded place in uncontrainted environments like town squares, sport stadium

and train stations.

2.4.5 ARGOVERSE

A tracking benchmark with 30k scenarios captured in Miami and Pittsburgh for Argoverse dataset [185]. The 10 Hz sampled of sequence frames for each scenario. 'Agent' is an interesting obstacle containing by each sequence and the task is to predict the 3 seconds future location of agents. There is no geographical overlap in splitting. These sequences are divided into 205942, 39472 and 78143 for training, validation and testing respectively.

2.4.6 APOLLOSCAPE

The 140,000 video frames are contain by the ApolloScape dataset [186] which consist the 73 street scenes in various location of china under different weather conditions. The 2D scene is represented by pixel-wise semantic annotation while 3D is represented by the point-wise semantic annotation for 28 classes. The dataset consists 2D lane marking annotations.

2.4.7 LYFT

The LYFT [187] is a motion prediction dataset for self-driving car that containing 1000 hours data. The 20 AVs are fleet in a fixed route of Palo Alto, California for 4-month period to collect the data. The 170000 scenes captured where each scene contain 25 seconds long and output perception of AVs, which encodes the precise

locations and motions of nearby cyclists, pedestrians and vehicles over time. The number of scenes in this dataset is 170,000 and each scene of dataset is 25sec long that captures the output perception of AVs. It encodes the nearby dynamic obstacles motion and precise positions over time.

2.4.8 CARLA Dataset

Carla (Car Learning to Act) [188] is an open source simulator developed to support AVs research, in particular the development, training and validation of autonomous urban driving system. Carla enables a first and third person camera view from the perspective of the vehicle. Carla is able to simulate advanced and realistic weather condition, urban environments and non-player characters. The simulator is built with python as a layer on top of Unreal Engine 4 (UE4).
CARLA has a two "towns" urban traffic environments where in vehicles may be rendered and tested. These two towns are currently, the only environments available. It is also uncertain whether CARLA enables a path as an input. It is possible for CARLA to test an AVs, a so called "agent" along with an "experimentation suite". The agent in this case is a decision maker, while the experimentation suite is a set of experiments consisting of tuples of start and end points. As stated however, the aim is not to use a decision maker within Carla. Instead the aim is to generate a path using the algorithm outside of CARLA and use only for visualization of the model along the path and to compare against the benchmark.

2.5 Evaluation Metrices

2.5.1 Precision

It defines the percentage accuracy of prediction. It measures how much percentage of correct prediction among the complete predictions.

$$Precision = \frac{TP}{TP + FP} \tag{2.3}$$

Where, TP is True Positive which represents the number of correct predictions for positive class and FP is False Positive which represents the incorrect predictions for positive class.

2.5.2 Multi-Object Tracking Accuracy

It stands for Multi-Object Tracking Accuracy. A model that achieves a high MOTA value shows improved performance and better results.

2.5.3 ID F1 Score

It stands for ID F1 Score. It denotes the number of detections that are correctly identified among the average number of detections computed and the ground truth. The higher the IDF1 score, the better the model is.

2.5.4 Mostly Tracked Targets

It stands for Mostly Tracked Targets. The higher value of MT results in better tracking performance. It is the ratio of GT(ground truth) trajectories that a track hypothesis covers for at least 80% of their lifespan.

2.5.5 Mostly Lost Targets

It stands for Mostly Lost Targets. It is the complement to MT. So, the lower the ML is, the better the model will perform. It is the ratio of GT(ground truth) trajectories that a track hypothesis covers for at most 20% of their lifespan.

2.5.6 IDs Identity Switches

It stands for Identity switches. It is the number of times, an object is assigned a new ID in its track. The less number of times identity switches occur, the better it is for the model.

2.5.7 FRAG

The total number of times a trajectory is fragmented or interrupted in between the detection.

2.5.8 Final Displacement Error

It measures the RMSE (Route Mean Square Error) between the final location prediction and the corresponding GT location of the TP i.e., the Euclidean mean

distance for the final predicted locations and the corresponding GT locations. The FDE of the ith dynamic obstacle is represented as:

$$FDE_i = \sqrt{\left(x_{i(pred)}^{t_f} - x_{i(GT)}^{t_f}\right)^2 + \left(x_{i(pred)}^{t_f} - x_{i(GT)}^{t_f}\right)^2} \tag{2.4}$$

where t_f is the last time-step or last frame in the predicted sequence, $x_{i(pred)}^{t_f}$ and$y_{i(pred)}^{t_f}$ are the predicted x and y-coordinates of the i^{th} object during the last time-step t_f , and, $x_{i(GT)}^{t_f}$ and $y_{i(GT)}^{t_f}$ are the x and y-coordinates corresponding to the GT position of the object at time-step t_f. The overall FDE is the mean of the FDEs of all the concerned objects for which trajectories are predicted. The FDE metric shows the long-term prediction ability of the model.

2.5.9 ADE Average Displacement Error

It measures the root means square error (RMSE) between all predicted positions and GT locations during particular prediction window, i.e., the Euclidean mean distance for all predicted locations and the GT locations during particular prediction time. The ADE of i^{th} dynamic obstacle is represented as:

$$ADE_i = \frac{1}{t_f}\sum_{t=1}^{t_f}\sqrt{\left(x_{i(pred)}^{t} - x_{i(GT)}^{t}\right)^2 + \left(x_{i(pred)}^{t} - x_{i(GT)}^{t}\right)^2} \tag{2.5}$$

where, t_f is the number of time-steps or, frames in predicted sequence, $x_i^t(pred)$ and $y_i^t(pred)$ are the predicted x and y-coordinates of the i^{th} object at time t, and, $x_i^t(GT)$ and $y_i^t(GT)$ are the x and y-coordinates corresponding to the GT position of

the object at time t. The overall ADE is the mean of the ADEs of all the concerned objects for which trajectories are predicted.

2.5.10 Infraction Management

Infractions or, violations while driving are considered using this metric. The infractions which are taken into consideration are collisions with pedestrians, vehicles, or static elements, violation of a red light, or a stop sign, and route deviations. IM is given by:

$$IM = \prod_{j} (P_j)^{No.of\ \mathrm{inf}\ ractionsj} \tag{2.6}$$

Where, p_j is the infraction penalty coefficient for every instance of infraction j done by the ego vehicle in a particular route.

2.5.11 Driving Score

DS is the weighted average of the RC with the infraction multiplier, and is given by:

$$DS = \frac{1}{N} \sum_{i=1}^{N} [(R_i)\,(IM_i)] \tag{2.7}$$

where, N is the total number of routes, R_i is the percentage of the i^{th} route completed by the ego vehicle, and IM_i is the infraction multiplier for the i^{th} route.

2.5.12 Route Completion

RC is the average percentage of the total route distance completed by the agent, or ego vehicle across different routes. RC is given by:

$$RC = \frac{1}{N} \sum_{i=1}^{N} R_i \tag{2.8}$$

where, N is the total number of routes, and R_i is the percentage of the i^{th} route completed by the ego vehicle.

2.6 Conclusion

This chapter studied the state-of-the-art approaches in the disciplines of Object Detection, Multi-Object Tracking, Trajectory Prediction and motion planning. It also examined the challenges and issues of these domains. This chapter also discussed the benchmark databases utilized to perform the performance evaluation of the proposed methods. Finally, the evaluation metrics were also explained, which are used to do the Performance evaluation.