Received April 24, 2021, accepted May 21, 2021, date of publication May 26, 2021, date of current version June 4, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3083890

Steering Angle Prediction Techniques for Autonomous Ground Vehicles: A Review

HAJIRA SALEEM[1], FAISAL RIAZ[1], LEONARDO MOSTARDA[2], MUAZ A. NIAZI[3], (Senior Member, IEEE), AMMAR RAFIQ[4], AND SAQIB SAEED[5]

[1]Control, Automotive and Robotics Laboratory, National Center of Robotics and Automation (NCRA), HEC, Department of Computer Science and Information Technology, Mirpur University of Science and Technology (MUST), Mirpur 10250, AJK, Pakistan
[2]Department of Computer Science, University of Camerino, 62032 Camerino, Italy
[3]Department of Computer Science, School of Information Technology, NUTECH, Islamabad 44000, Pakistan
[4]NFC Institute of Engineering and Fertilizer Research, Faisalabad 38090, Pakistan
[5]Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, Dammam 31441, Saudi Arabia

Corresponding author: Hajira Saleem (hajira396@gmail.com)

ABSTRACT Unintentional lane departure accidents are one of the biggest reasons for the causalities that occur due to human errors. By incorporating lane-keeping features in vehicles, many accidents can be avoided. The lane-keeping system operates by auto-steering the vehicle in order to keep it within the desired lane, despite of changes in road conditions and other interferences. Accurate steering angle prediction is crucial to keep the vehicle within the road boundaries, which is a challenging task. The main difficulty in this regard is to identify the drivable road area on heterogeneous road types varying in color, texture, illumination conditions, and lane marking types. This strenuous problem can be addressed by two approaches, namely, 'computer-vision-based approach' and 'imitation-learning-based approach'. To the best of our knowledge, at present, there is no such detailed review study covering both the approaches and their related optimization techniques. This comprehensive review attempts to provide a clear picture of both approaches of steering angle prediction in the form of step by step procedures. The taxonomy of steering angle prediction has been presented in the paper for a better comprehension of the problem. We have also discussed open research problems at the end of the paper to help the researchers of this area to discover new research horizons.

INDEX TERMS Computer vision, machine learning, neural network, lane detection, steering angle.

I. INTRODUCTION

Road accidents cause numerous causalities every day. Khatib *et al.* [1] concluded that the distraction of drivers is one of the key reasons for accidents. Unintentional lane departure accidents point to a major class of accidents caused due to driver distraction. According to Mammeri *et al.* [2], lane departure crashes counted 51% of total accidents reported in the United States in the year 2011. There is a continuous quest to improve vehicles and driving conditions in order to eliminate the chances of lane departure accidents. In this regard, driver assistance technologies such as lane-keeping systems are being incorporated in vehicles.

Lane-keeping is a lateral control system aiming to automatically perform the steering of an autonomous vehicle (AV) in order to keep it within the road boundaries despite of changes in road conditions and other interferences [3]. This is effectuated by deriving the desired steering wheel angle using the lateral dynamics. This predicted steering wheel angle is the main impetus to the lateral controller to perform lane keeping maneuver in AVs.

For the successful steering angle prediction, robust road region understanding is first and the foremost step. Road region understanding is a challenging task, as there are a variety of road conditions and their geometries. For example, roads can be structured or unstructured, paved or unpaved, occluded or unoccupied, and marked or unmarked. Even lane markings have various variations e.g., continuous/disjoint, white/yellow, curved/straight, and single/double markings. Moreover, varying illumination conditions, weather conditions, and artifacts on the road make the road region understanding a complex and difficult task. The very first step of road region understanding is to perceive the vehicle's surrounding environment by using one or a combination of sensors like camera, LIDAR, SONAR, and GPS, etc. Then the obtained information is processed through various techniques

The associate editor coordinating the review of this manuscript and approving it for publication was Long Xu.

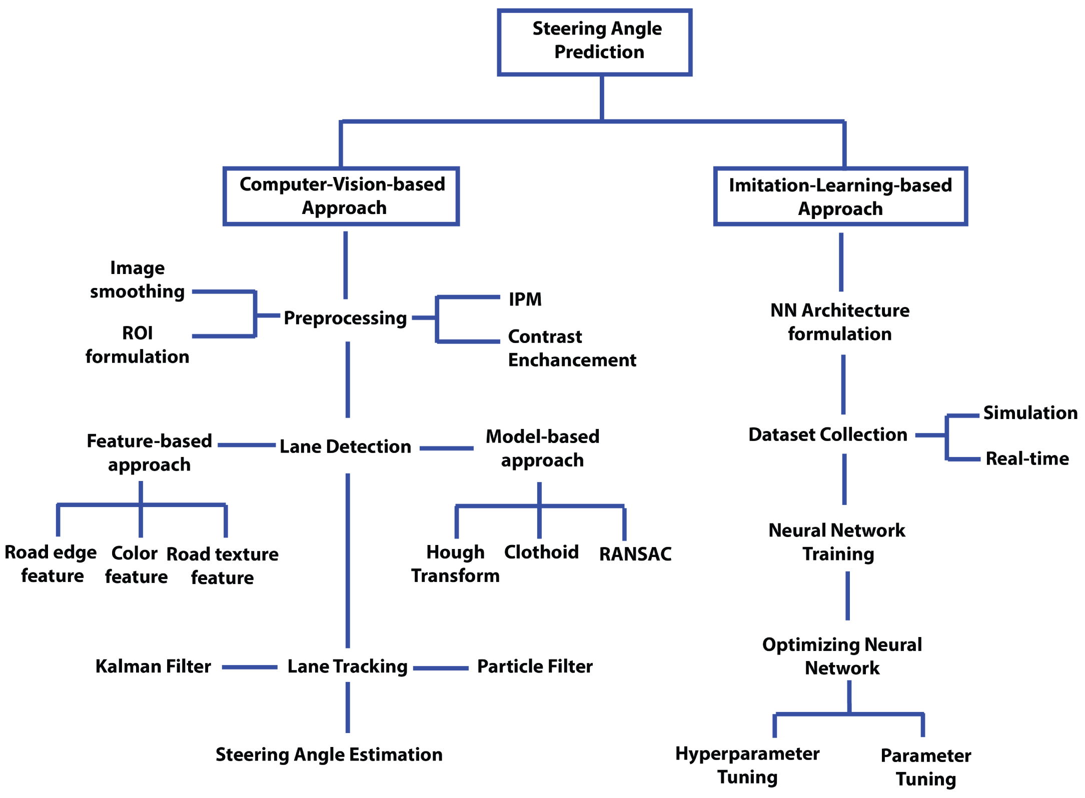

FIGURE 1. Taxonomy of steering angle prediction.

in order to make it usable for the subsequent process of steering angle prediction. This paper covers literature only on vision-based steering angle prediction techniques.

The steering angle prediction systems can broadly be categorized into two approaches. The first approach involves extracting road boundary coordinates and then applying mathematical or statistical models to predict the steering angle. The second approach is based on imitation learning strategy. The latter technique involves utilizing artificial neural network (ANN) models, and the lane boundary extraction process is not done explicitly by the practitioner. Rather, the machine learns through demonstration of expert and utilize this knowledge for predicting steering angle without human intervention. Imitation-learning-based steering angle prediction involves using only images or sequence of images as input to ANN models to generate steering angle as output. The predicted steering angle in both approaches is utilized along with other lateral parameters for keeping AV within the road boundaries. Figure 1 shows the taxonomy of vision-based steering angle prediction.

A. EXISTING SIMILAR RECENT STUDIES

Recently, Gidado *et al.* [4] presented a survey paper which covers the following topics: deep learning architectures (deep reinforcement learning and convolutional neural network); application of deep learning architectures for steering angle prediction, longitudinal & lateral control; history and other details of frameworks being used for designing and training ANN architectures; analysis of year-wise and frequency wise publication trend of deep learning applications in steering control. Oussama and Mohamed [5] presented a short literature review on computer vision and deep learning approach for steering angle prediction. Yet, to the best of our knowledge, there is no available literature review of the techniques of steering angle estimation part used in computer-vision-based approach. Our review study is unique in its detailed coverage of step-by-step processes involved in computer-vision-based as well as the imitation-learning-based approaches for the steering angle prediction. It presents the solutions various researchers adopted to handle different challenges in the steering angle prediction (e.g., different illumination and weather conditions etc). This in-depth review will help novice researchers get insight into the problem of steering angle prediction and various methods to implement it.

Organization of the paper: Rest of the paper is structured as follows: Section 2 discusses computer-vision-based approach accompanied by image processing techniques for steering angle prediction, which is accomplished in steps, including image frames preprocessing, road region identification, road boundary tracking and steering angle estimation based on processed visual information. Section 3 covers imitation-learning-based approach for steering angle prediction. This Section is further divided into subsections based on the steps involved to solve the problem, including ANN model architecture formulation, dataset collection, training the ANN model and optimizing the model. Section 4 highlights open research questions. Lastly, section 5 presents concluding remarks.

II. STEERING ANGLE PREDICTION USING COMPUTER VISION APPROACH

This method of steering angle prediction involves explicitly extracting road boundary coordinates or lane markings based on which the steering angle is computed. Firstly, image frames are captured through a live feed from the camera mounted on the car. Then these image frames are preprocessed in order to increase the likelihood of accurately extracting relevant road features. The next stage is lane boundary extraction using the preprocessed images. Some researchers performed lane tracking after this step, by considering consecutive frames for successful road detection despite the missing or erroneous road information. Tracking is achieved using particle filter, Kalman filter, and Bayes filter. Description for each of these steps and the related literature is elucidated in the next subsections.

A. PRE-PROCESSING

Calibration of camera, illumination changes, poor visibility due to bad weather conditions, shadows, and light reflections on the road may cause faulty lane detection and tracking [6]. Therefore, image pre-processing is done for the sake of enhancing and modifying input frames as a means to increase the likelihood of delivering useful information to subsequent stages. Conventionally, the preprocessing stage includes one or a combination of the following methods: downsampling the image, image segmentation, image smoothing, extraction of Region of Interest, and the application of Inverse Perspective Mapping (IPM).

As the computation time is the most important criterion in real-time applications. Therefore, the only relevant portion of the image frame is taken for further processing, hence rejecting portions of images that do not contribute to lane detection and tracking (e.g., pixels of the image containing sky). The portion containing the important information taken for further processing is referred to as 'Region of Interest' (ROI). Any coordinate that lies beyond the ROI is disregarded. ROI should be taken such that it covers left and right lanes and the point where both lanes appear to intersect at each other in the image; this intersection point is mostly used to find the steering angle. ROI can either be determined with prior road area detections or it can roughly be chosen as the lower portion of the image [7]. Carefully choosing ROI is significant in improving computation efficiency and lane detection accuracy.

Another measure to reduce the computational load is to downsample the images using grayscale conversion [8]–[12], inter-area interpolation [13], or using the image binarization approach [14]. Among these, grayscale is mostly used for downsampling. Figure 2 (b) shows a grayscale image of a road. Apart from grayscale conversion, a lot of researchers found that conversion of RGB to other color formats may present better lane detection results instead of processing raw RGB camera captured images. Sun *et al.* [15] argued that applying a loose threshold method on the HIS color model of the image yields better detection compared to RGB. Yet, reducing the size and changing the format of the images may result in the loss of useful information, especially in complex situations.

1) IMAGE SMOOTHING, SHARPENING, AND SHADOW REMOVAL

Image smoothing is done to blur the noisy details diminishing the impact of pixels which are not part of the lane markings. The most widely used smoothing filters for road lane extraction are Gaussian filter [16], [17] and Median filter [18] or both [19], [20]. It can be seen in Figure 2 (c, d), that both median and Gaussian filters blur out noise and unnecessary details, yet they can sometimes eradicate information crucial for lane detection, such as in Figure 2(c) lane markings information is lost. Therefore, a careful selection of the size of the filter is necessary which again are road conditions and application-specific.

Objects (such as buildings, bridges, and other vehicles) may cast shadows on the road and alters the impact of road texture by producing artifacts onto the road surface. Smoothing and eliminating these shadows in the image is important for error-free road boundary detection. Assidiq *et al.* [21] used a method for shadow removal from the frames. The method works by first deriving 1-d illumination invariant, free of the shadow image. Then this invariant image was used to locate the edges of shadows. These edges were then set to zero in an edge representation of the original image, and lastly, the obtained edge representation was reintegrated to the original image by a method parallel to lightness recovery. However, a problem with their approach is that the intensity values are unlikely to remain consistent over the different construction materials used to build the road. This can cause a disparity in results for extracting shadow from the illumination of a single color.

According to Kucukmanisa *et al.* [22], B color channel of RGB can easily detect and separate out white lane markings using an MSER-based approach [23] despite of any kind of shadows on the road. However, enhancing the B channel of the image gives better detection results for yellow lanes as well irrespective of shadows.

Parajuli *et al.* [24] applied a vertical gradient on the image to remove the effect of shadows, as according to the authors, shadows cast on the road are usually horizontal. The high pass filter has also been used to eliminate shadows in omnidirectional images [25]. Since gradient operators are high pass filters, which are sensitive to noise, hence can also be used for lane detection, but they have difficulty in detecting degraded lane markings [26].

Processing the image frames without applying contrast enhancement after smoothing gives rise to fading and phantom edges [27]. Hence, after applying a single or a combination of smoothing filters, image enhancement has

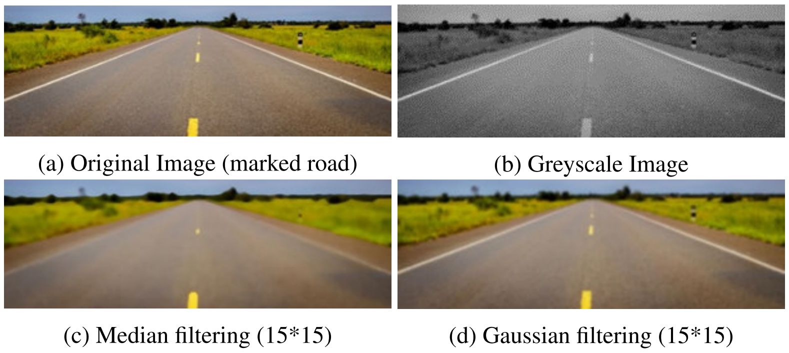
(a) Original Image (marked road)
(b) Greyscale Image
(c) Median filtering (15*15)
(d) Gaussian filtering (15*15)

FIGURE 2. Preprocessing a marked road.

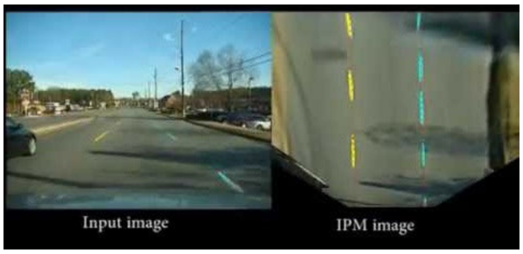

FIGURE 3. IPM [42].

been done in some researches in order to retain contour details [28]–[30].

2) IPM

The perspective mapping process of 3D road scene to the 2D image plane brings forth many problems in the lane boundary extraction process. If the lane boundary extraction process is done directly on the original image, many non-target edges exterior to the lane boundary will also be extracted, e.g., pedestrians, trees and traffic signs, etc. These are basically sources of interference for lane detection, whose effect can be minimized using IPM. IPM is top view or the bird's eye view of the road scene, having nearly vertical lines on a darker background whose detection is more convenient removing outliers on the roadsides. Originally IPM formula was derived by Bertozz *et al.* [31] and was applied to the GOLD AV successfully. After that, this method is being extensively used in the preprocessing stage of the lane and obstacle detection [32]–[41]. In Figure 3, IPM of roads can be examined.

B. ROAD BOUNDARY DETECTION & TRACKING

Road boundary detection aims to identify a drivable region on the road by preparing the system for the accurate identification of the lane marking (for marked roads) or road boundary (for unmarked roads). The key for vision-based road area detection is the ability to classify image pixels as belonging or not belonging to the road surface. Almost every lane detection algorithm follows three essential steps: lane feature extraction, outlier removal, lane boundary representation. In some methods, lane tracking has been performed as well, which is aimed to track lane positions after performing road boundary detection. For this purpose, Kalman filter and Particle filter are widely used methods that refine the detection results and predict lane coordinate positions more efficiently. While a few researchers used Bayes filter and custom-built tracker. It can be observed in Table 1, that how different researchers achieved the task of lane detection and tracking by combining different approaches.

Road boundary detection is a lot easier for structured and properly marked roads as compared to unstructured and unmarked roads. Vision-based road detection techniques can broadly be divided into three categories, namely: feature-based techniques, model-based techniques, and machine-learning-based techniques.

1) FEATURE-BASED TECHNIQUES FOR DRIVABLE ROAD AREA DETECTION

Many features such as the geometric shapes, edges, color, gradient, and road texture can be used to identify the driveable road region. A brief overview of the techniques utilizing these features is given below:

a: TECHNIQUES USING LANE EDGE OR GRADIENT INFORMATION

As the road lanes have brighter intensity values as compared to the rest of the road, hence a gradient of dark-bright-dark exists at lane lines. This gradient feature has been utilized by gradient-based edge detectors to extract left, right, and center lanes. In this regard, the Canny edge detector is the most widely used gradient-based edge detector in the literature [43]–[51]. It works by first smoothing the image using a Gaussian filter, followed by calculating the gradient direction and its amplitude on which non-maximal suppression is performed, lastly using a double threshold algorithm edges are detected and connected. Hence the resultant image after processing from the canny edge detector is a binarized image with highlighted edges.

Another rather less commonly used gradient-based edge detector for lane detection is Sobel operator [52]–[54], Prewitt operator [55] and other custom edge detectors [11]. Yet, the Canny edge detector is found to outperform other gradient-based lane detection methods [56]. As the Canny edge detector also performs image smoothing hence it is

TABLE 1. Review of lane detection and tracking techniques.

Ref	Year	Preprocessing and extraction of edges	Lane marking extraction	Tracking	Evaluation	Critic
[11]	2018	Grayscale conversion, Custom edge detection technique(EDline)	Hierarchical agglomerative clustering and slope filter	Kalman filter	99%, 96% and 93% accuracy during day with clear, rainy and snowy conditions respectively; 98% and 93% during night with clear and rainy conditions respectively.	Testing is done only on straight roads (curve scenarios are not handled)
[57]	2017	Image Binarization	Hough Transform	Kalman filter	97% accuracy in daytime and 95% accuracy in night-time. Operates on 15fps	Their method doesn't operate well on dotted lanes
[58]	2013	IPM, Gaussian filtering, Denoising template filtering	Hough transform, RANSAC B-Spline Fitting algorithm	Kalman filter	Robust in straight as well as curved roads with 92% accuracy and 72ms maximum processing time for one frame	Not tested in low illumination conditions or night light
[59]	2018	IPM, Grayscale conversion, Canny edge detector	Hough transform, Linear interpolation, Cubic spline and cubic spline with shape preservation	—	87∼98% of accuracy, Among the three interpolation functions, best result was obtained with cubic spline with shape preservation	Testing was not done in Realtime
[63]	2019	Grayscale conversion, image binarization, dividing ROI in two sections	Hough transform, lane curve model	Custom built tracker	92∼93% accuracy in various challenging scenarios like various curves, night light, foggy and rainy weather conditions,	Testing was not done for cases with intrusion on lane markings
[62]	2014	Canny edge Detector	Hough transform	Kalman filter	91.0% of visible lane boundaries were detected successfully and 90.9% of the detected results were correct	Poor performance in heavy traffic and uneven illumination, Suitable for straight roads but not for curved
[68]	2017	Gaussian smoothing, binarization, contrast enhancement, RGB to HSV conversion	RANSAC	—	86.21% accuracy with 4.3% standard deviation in real time	Not tested in low illumination conditions

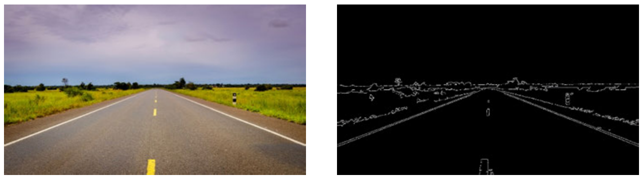
FIGURE 4. Canny edge detection.

robust to slight noise. As can be seen in Figure 4, the image processed with the canny edge detector still have edges around the road which are not part of the lane marking. So, to remove these outliers and locate actual lane markings, Hough transform [57]–[63] and other methods are applied afterward.

The most popular methods for lane detection using the gradient information extraction technique involve firstly to convert the image into grayscale, followed by applying the canny edge detector and lastly applying Hough transform. Conversion into grayscale substantially reduces computation time, but it may lose the gradient of lane lines leaving them undetected. Yoo *et al.* [64] proposed a solution to this problem by enhancing the gradient through contrast enhancement of white and yellow lanes of the road with respect to the road. As the RGB intensity values of the same objects can vary under different illumination, therefore enhancing intensity values of a color with the same ratio is impractical. Therefore, they used two random adaptable vectors for yellow and white lanes, which produce a grayscale image having maximum road lanes gradient.

Other methods to preserve lane gradient invariant of shadow and other attenuation effects include processing images in a specific channel of RGB or conversion of images to other formats such as HSI [65], YCbCr [66] YUV [7], [67] and HSV [68] etc.

Another approach to preserving the lane markings gradient minimizing attenuation effect is proposed by Liu *et al.* [69]. They first performed IPM on images. Then wavelet decomposition was performed, followed by wavelet reconstruction in order to enhance vertical edges and to weaken the horizontal edge information (because in IPM lanes are transformed into vertical lines). Then the Sobel operator and the Canny edge detector are applied to these processed images. These operators are aimed to produce a binarized image highlighting the road lanes and converting all the pixels other than the road lanes to black. Finally, this binarized image is processed using Hough transform to convert lane points into coordinate points. Another variation of edge detection approach is "middle to side strategy". This approach begins with the search for non-zero pixel or negative/positive gradient on both sides of the road and moving upwards connecting points in successive upward locations. This method is more prone to misdetection because the first pair of points might be the artifacts [70].

b: TECHNIQUES USING ROAD TEXTURE FEATURE

A popular technique of road detection using texture information is intensity thresholding. As the whole road region mostly has darker intensity values as compared to the surrounding. Hence, using connected component and threshold intensity values, the road region is segmented out from pixels outside the road. Seo and Rajkumar [71] performed intensity thresholding to identify the boundary of drivable regions. Firstly, they applied IPM on acquired image frames, followed by intensity thresholding for road region detection. Lastly, the Bayes filter and unscented Kalman filter were used to track detected boundaries. A limitation to intensity thresholding technique is that the threshold has to be adjusted according to the illumination condition and the detection result varies by the threshold value [72].

c: TECHNIQUES USING COLOR FEATURE

All the methods for drivable road region estimation involve using color feature. Typical image frames are in RGB format, which might be converted to other color formats in order to reduce computation complexity or to aid better detection. Illumination conditions keep varying constantly in different times of the day, hence affecting other features used for road region detection. For example, the gradient of lane markings may not be preserved in very low illumination conditions, hence leading to faulty lane detection. In short, the color feature is essential in all methods of drivable road area estimation. Hence the color feature is taken into consideration for devising road detection methods employing other road features.

2) MODEL-BASED APPROACHES FOR ROAD AREA DETECTION

In this approach, road boundary points are matched with templates or lane models such as a parabola, hyperbola, spline, and linear line; the best-fitted shape determines the road boundaries [73]–[78]. Hough transform and Random Sampling Consensus (RANSAC) are the most popular model-based approaches used for fitting lines and other geometric shapes on the road boundaries for detecting lanes.

Through a detailed analysis of the literature regarding lane detection, it is found that the majority of techniques utilize Hough transform at some stage for lane detection. Hough transforms works by finding the slope at each edge point and proposing a single line that has the majority vote at edge points. Hence a straight line can be drawn at the lane position on the road using Hough transform, or in other words lane boundaries can be represented in coordinates by straight Hough lines. This process also has the advantage of connecting dotted or disconnected lane edges. Yet, a major drawback is that it is not efficient in the curve lane detection. Another problem is that artifacts on the road, such as cracks, navigational text, and arrows, etc. often have features similar to straight lines. Hence Hough transform may mis-classify these artifacts as the best candidate for lanes.

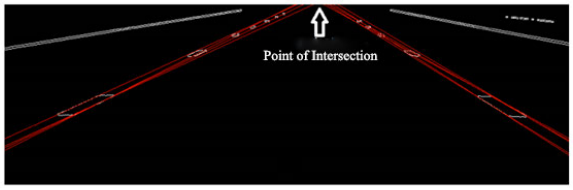

FIGURE 5. Point of intersection of left and right boundary lines.

Another widely used method in this regard is the RANSAC method. RANSAC first proposed by [79], performs iterative estimation of parameters of a mathematical model from observed data points. Robust estimation of model parameters and separating out the inliers from the outliers with high accuracy is the main advantage of this method.

RANSAC is found to be more efficient in curved as well as straight lane detection. Many researchers combined model-based and feature-based approaches, for attaining improved results. For example, applying Hough transform on canny edge detected image. In Table 2, a short review of feature-based and model-based approaches have been presented.

C. STEERING ANGLE ESTIMATION

After performing road boundary extraction process, required steering angle is computed. Different techniques have been proposed by researchers for this purpose. The extracted boundary points and the heading direction of the vehicle are used for this purpose. In a technique proposed by Dev *et al.* [6], required steering angle is determined as the angle between Point of Intersection (POI) and the heading direction of the vehicle. POI is the (x,y) coordinate, where the extracted left and right boundary lines meet, as shown in the Figure 5.

If the POI and the center point of road is aligned, the required angle for AV to steer is zero. In the other case, a center line is determined connecting the POI and the center of image, on the basis of which the required steering angle is computed. Let s_r and s_l be the slopes of right and left extracted line. Let b_r and b_l be the slopes of angle bisectors determined by s_r and s_l. The slope of the center line ($slope_c$)

TABLE 2. Feature and model-based lane detection techniques.

Ref	Year	Contribution	Approach used	Type of roads and scenarios used for testing	Color channel
[71]	2014	Efficiently performed detection and tracking of unmarked road boundaries by applying connected component labeling of an intensity threshold image followed by unscented Kalman filter	Feature-based approach	Unmarked paved public roads, images with snow by the sides of roads	RGB
[73]	2020	Achieved Shadow and illumination robust lane detection by firstly applying top-hat transformation to filter out non-lane objects and enhance the contrast, then applying threshold segmentation algorithm followed by Hough transform algorithm with polar angle and distance constraint.	Feature-based + Model-based approach	Multi-lane urban roads with various complex scenarios like shadows, road marking interference, damaged road surface, road painted with other shapes in addition to lane markings, and scenarios with different illumination conditions	Grayscale
[74]	2020	Robust lane detection by 'dynamic region of interest technique' achieved through using vehicle speed data and lane line equation	Feature-based + Model-based approach	Highways, rural roads, wet roads, curved and straight roads, day and night scenarios	Grayscale
[75]	2020	Computationally efficient lane detection in the scenarios with interrupted lane marking, through the use of model of road structure and extended kalman filter.	Feature-based + Model-based approach	Various complex scenarios, such as, strong light, lane-changing, zebra crossing, dashed markings, curved road, and presence of shadows etc	Grayscale
[76]	2016	Improved curved lane detection by combining Hough transform (for section of road near the vehicle) and parabolic model fitting (for section of road far from the road)	Model fitting approach	Roads with dark lane boundary markings	Grayscale
[77]	2010	Improved lane detection by linking edges using Ant colony optimization, lines were extracted by the Hough transform after applying canny edge detector	Feature-based + Model-based approach	Straight roads	Grayscale
[78]	2020	Efficiently detected curved as well as straight lane markings by fitting third order spline over regions with highest density of nonzero pixels of the image obtained after edge and color thresholding	Feature-based + Model-based approach	Straight as well as curved roads	Grayscale

can be computed using following equations:

$$slope_{c1} = (s_l * b_r) - \frac{s_r * b_l}{b_r - b_l}$$

$$slope_{c2} = (s_l * b_r) + \frac{s_r * b_l}{b_r + b_l}$$

If abs($slope(c_1)$) > abs($slope(c_2)$), $slope(c_1)$ is chosen as slope of center line; the intercept of middle line is computed as follows:

$$inter_c = \frac{(c_l * b_r) - (c_r * b_l)}{b_r - b_l}$$

Otherwise $slope_{c2}$ is selected and the slope of centerline is computed as follows:

$$inter_c = \frac{(c_l * b_r) + (c_r * b_l)}{b_r + b_l}$$

The pair ($slope_c$, $inter_c$) characterizes a line in cartesian plane, which is the navigating path for the vehicle. A reference point from this line (x_2,y_2) and the POI (x_1,y_1) are utilized to determine the required steering angle, as can be seen in Figure 6.

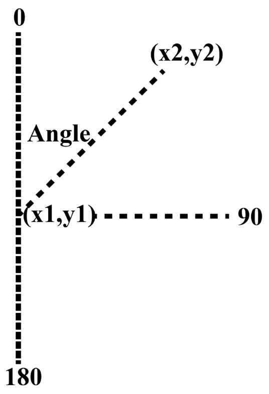

FIGURE 6. Steering angle computation method proposed by Dev *et al.* [6].

The equation used for the computation of required steering angle 'θ' is as follows:

$$\theta = \arctan \frac{(y_2 - y_1)}{(x_2 - x_1)}$$

TABLE 3. Conditions for different road scenarios.

Scenario	Steer Straight	Steer Right	Steer Left
ϕ	-30<ϕ<30	b2>b1 ϕ<-30	b2<b1 ϕ>30
θ	μ1-μ2=0	μ2>μ1 θ=negative	μ2<μ1 θ=positive

The main limitation to this technique is that it does not handle the cases when only one boundary line or no boundary line is detected. Yet in a technique proposed by Umamaheswari *et al.* [80], a virtual boundary line is computed if only one line is detected, using the following equations:

For left virtual boundary line-

$$\psi = -(1/3) * (\eta + 45) + 10,$$

For right virtual boundary line-

$$\psi = -(1/3) * (\eta - 45) - 10.$$

where ψ is the virtual edge inclination and η is the inclination of detected opposite edge. In their proposed technique, steering angle is computed by drawing a line from the middle of the road to the starting position of both extracted boundaries. Afterwards, the following equations are computed:

$$\mu 1 = \arctan(b1/a1),$$
$$\mu 2 = \arctan(b2/a2),$$
$$\theta = \mu 1 - \mu 2,$$
$$\phi = b1 - b2,$$

a1 = a2 = image width/2 = pixel distance b1 = distance of (image height,0) value from edge.

b2 = distance of (image height, image width) value from edge.

θ = steering angle required to align the vehicle in the center of the road. ϕ is the perpendicular steering distance. The value of ϕ varies between 150 to 100 based on the scenario. A negative value of ϕ indicates rightward drag of the vehicle and θ denotes the steering angle to align it along the mid-line of lane. Similarly, a positive ϕ value indicates leftward drag.

The scenarios handled by their proposed algorithm and their implication are listed in Table 3 and depicted in the Figure 7.

In another approach proposed by Sujatha *et al.* [81], steering angle prediction is done based on three cases: (1) neither of the left or right boundaries are detected (2) both the boundaries are detected and (3) one of the boundaries is detected. Description of the steering angle prediction in all these cases is as follows:

- Case1
  When none of the boundary is extracted, the vehicle is allowed to move straight along its path till it finds one or both lane boundaries.
- Case2
  In this case when both boundaries are extracted, the slope and position of both extracted lines is determined. The line in the right half of image and having a positive slope is the right boundary; whereas the line in the left half of the image and having a negative slope is determined as left boundary. From the slope of both boundaries, the common POI is calculated. The required steering angle will be the deviation of POI of the boundaries from the orientation of the vehicle (centerline of the image). The whole procedure from finding the slope to the steering angle computation, is performed using the following equations.

$$Slope_{right} = (Y_4 - Y_3)/(X_4 - X_3),$$
$$Slope_{left} = (Y_2 - Y_1)/(X_2 - X_1),$$
$$C_2 = Y_{right} - (Slope_{right} * X_{right}),$$
$$C_1 = Y_{left} - (Slope_{left} * X_{left}),$$
$$InterX = (C_2 - C_1)/(Slope_{left} - Slope_{right}),$$
$$InterY = ((Slope_{left} * InterY) + C_1),$$
$$S = ((InterX - (w/2)) - (InterY - h)),$$
$$Steering\ angle = 90 - (\tan^{-1}(1/S) * (180/\pi),$$

where X1; X2; X3; X4 and Y1; Y2; Y3; Y4 are the coordinate points, whereas C_1; C_2 are the constants of the extracted lines. Inter X and Inter Y are the coordinates of POI of both extracted lines. S is the slope between POI of the boundaries and the vehicle heading direction, whereas h and w are the height and width of the image in pixels. In Figure 8, marked points 1-4 represent the coordinate points; marking 5 represents the POI of both the boundaries; and marking 6 represents the required steering angle. White dotted lines represent the centerline of the image.

- Case3
  When only one of the boundaries is visible within the field of view of the camera, two approaches that can be adapted are: (a) maintaining the vehicle in the center of the road; (b) maintaining the vehicle towards the visible boundary.
  - Case3a
    In this case the vehicle is maintained at the center of the road. If only one boundary is extracted, the other

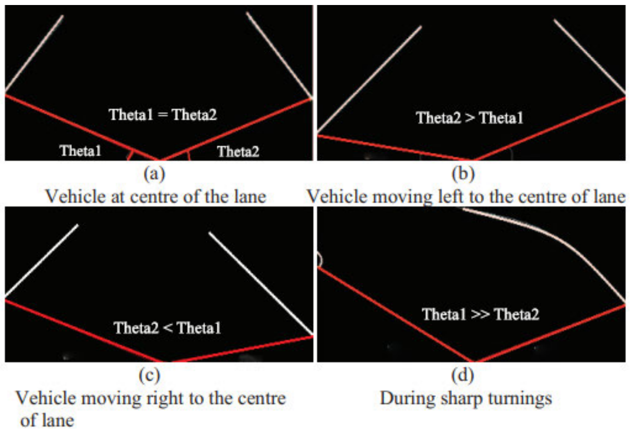
Theta1 = Theta2
Theta1
Theta2
(a)
Vehicle at centre of the lane
Theta2 > Theta1
(b)
Vehicle moving left to the centre of lane
Theta2 < Theta1
(c)
Vehicle moving right to the centre of lane
Theta1 >> Theta2
(d)
During sharp turnings

FIGURE 7. Cases of angular aberration from center of the road [80].

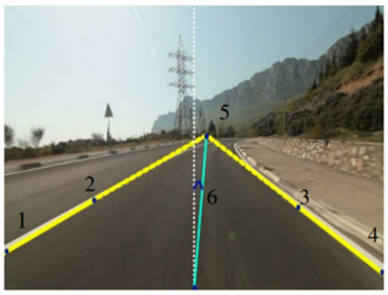

FIGURE 8. Depiction of points used for steering angle computation [81].

boundary is usually considered the last column of pixels on the other side of the image. For example, if right boundary is not visible and only left boundary is visible, then. The last pixel column on the right side of the image is considered the right boundary. After calculating the slope of both, the steering angle computation is done using the same procedure as mentioned for case 2.

– Case3b
In this case the vehicle is maintained at a certain distance from the visible boundary. If the width of road is too large, it is difficult to maintain the vehicle in the middle of the road since this way it will move in a zigzag manner, especially on turns. Hence, the vehicle can be maintained at a certain distance from the road boundary instead of maintaining at middle of the road. To keep the vehicle at a certain distance from the boundary a base offset needs to be computed, which is done as follows. Let a distance to be maintained from the required boundary be 2m. An imaginary straight-line boundary is assumed of which the two sides are $[x_1, y_1, z_1] = [x_1, 2,$ height to which the camera is mounted] and $[x_2, y_2, z_2] = [x_1 +$ displacement, 2, height to which the camera is mounted]. The points are converted into camera coordinates and base offset angle is computed using following equations:

$$\begin{aligned}
\theta 1 &= (180/\pi) * (\tan^{-1}(y_1/x_1)),\\
\phi &= 90 - \frac{180}{\pi} * \tan^{-1}(\frac{\sqrt{x_1^2 + y_1^2}}{z_1}),\\
I_x &= rows/horizontal\, fov,\\
I_y &= cols/vertical\, fov,\\
x_{im1} &= rows - (c_x + (\theta 1 * I_x)),\\
y_{im1} &= cols - (c_y + (\phi 1 * I_y)),\\
\theta 2 &= (180/\pi) * (\tan^{-1}(y_2/x_2))\\
\phi 2 &= 90 - \frac{180}{\pi} * \tan^{-1}(\frac{\sqrt{x_2^2 + y_2^2}}{z_2}),\\
x_{im2} &= [rows - c_x + (\theta 2 * I_x)],\\
y_{im2} &= cols - (c_y + \phi 2 * I_y),\\
\phi_d &= \tan^{-1}(\frac{(y_{im2} - y_{im1})}{(x_{im2} - x_{im1})}) * (\frac{180}{3.14}),
\end{aligned}$$

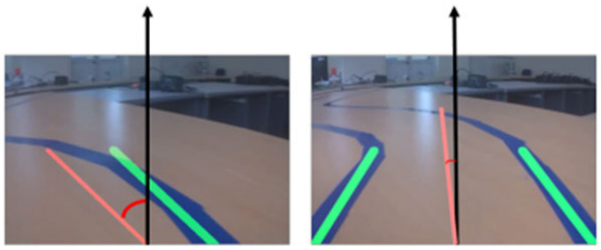

FIGURE 9. Steering angle computation proposed by Tu *et al.* [82].

where $x_1; y_1; z_1; x_2; y_2; z_2$ are the assumed edges of the boundary, $x_{im1}; y_{im1}; x_{im2}; y_{im2}$ are the corresponding image coordinates, c_y is the center y-coordinate of the image, c_x is the center x-coordinate of the image, cols are the pixels along height of the image, rows are the pixels along width of the image, l_y is the vertical pixel per degree information, l_x is the horizontal pixel per degree information, and θ_d is the base offset angle in degrees. This base offset angle is maintained until vehicle has to maneuver with one boundary. The angle between the extracted boundary and the frame of reference is computed. For every successive frame, the required steering angle is the difference between the base offset and the angle of frame of reference with the extracted boundary line.

Another recent technique of steering angle prediction proposed by Tu *et al.* [82] also handles the cases when only one boundary or none of boundary lines are detected. In their proposed approach, a vertical line in the center of the image is drawn and then following cases are considered:

- If a single left or right line is detected, the required steering angle is the angle between the vertical line and the detected lane.
- If both lines at the left and the right are detected: Linear average of the two lanes is computed and the required steering angle is the angle between the vertical centerline and the linear average.
- If no lines are detected, preceding angle is used.

In Figure 9, vertical centerline is drawn in black and the required steering angle is the angle between extracted lane and linear average for single and both extracted lanes respectively.

Another approach to steering angle estimation proposed by Abdelrahman *et al.* [83] is by using decision tree. The current situation of vehicle on the road is interpreted using decision tree and respective heading line behavior is selected. In this technique, after performing lane marking extraction process, road is divided into segments. Two decision trees are then used, one for upper and other for lower segments, see Figure 10 and 11.

In the decision tree. The first decision factor is whether the boundary lines are detected or not and in which parts lines are detected. If both left and right lines are detected, then the distance between the two lines is measured in order to check

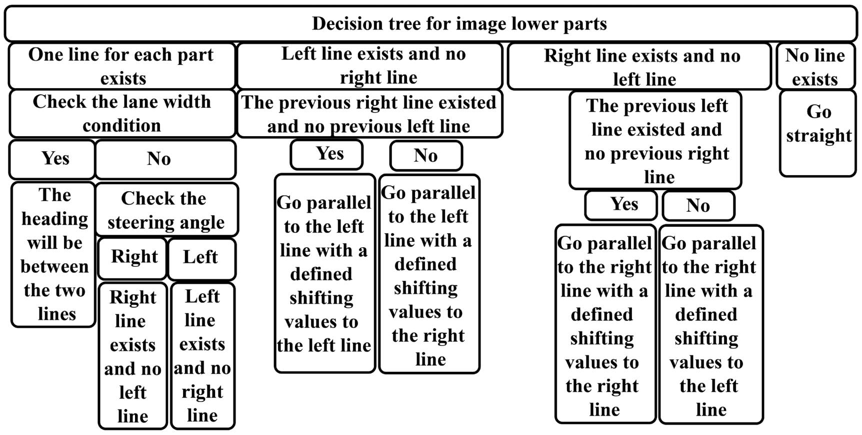
Decision tree for image lower parts
One line for each part exists
Left line exists and no right line
Right line exists and no left line
No line exists
Check the lane width condition
The previous right line existed and no previous left line
The previous left line existed and no previous right line
Go straight
Yes
No
Yes
No
Yes
No
The heading will be between the two lines
Check the steering angle
Right
Left
Right line exists and no left line
Left line exists and no right line
Go parallel to the left line with a defined shifting values to the left line
Go parallel to the left line with a defined shifting values to the right line
Go parallel to the right line with a defined shifting values to the right line
Go parallel to the right line with a defined shifting values to the left line

FIGURE 10. Decision tree for the lower parts of the image.

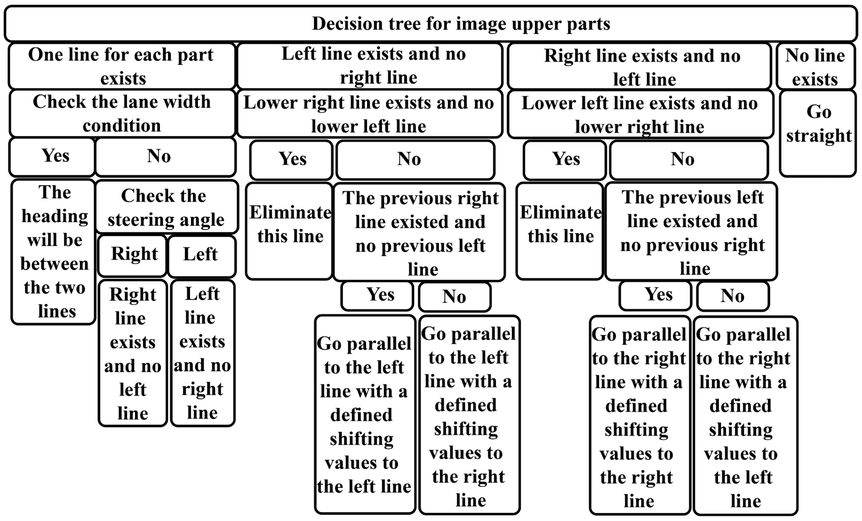
Decision tree for image upper parts
One line for each part exists
Left line exists and no right line
Right line exists and no left line
No line exists
Check the lane width condition
Lower right line exists and no lower left line
Lower left line exists and no lower right line
Go straight
Yes
No
Yes
No
Yes
No
The heading will be between the two lines
Check the steering angle
Right
Left
Right line exists and no left line
Left line exists and no right line
Eliminate this line
The previous right line existed and no previous left line
Eliminate this line
The previous left line existed and no previous right line
Yes
No
Yes
No
Go parallel to the left line with a defined shifting values to the left line
Go parallel to the left line with a defined shifting values to the right line
Go parallel to the right line with a defined shifting values to the right line
Go parallel to the right line with a defined shifting values to the left line

FIGURE 11. Decision tree for the upper parts of the image.

lane width. This is done to solve a problem of false detection which comes from objects around the track or uneven lighting. If the condition is false, the current heading direction of the vehicle is checked. If heading direction determines that the vehicle is turned to the right, the left detected line is not considered in the steering angle estimation and vice versa. If only one line is detected in the image, it is checked that whether the vehicle is still within the lane boundary or not. This verification is vital because the vehicle might be about to leave the lane boundary, or the vehicle is approaching a curvature. If the detected lane line appears in the right part at the present step and in the previous step there was a single line appearing in the left part, then the decision will be to move the vehicle parallel to the detected line but in the right side. But if the condition is false, the vehicle heading will be parallel to the line but in the left side. If there is no line, this means that the vehicle is still within the lane and the decision is to proceed straight.

The decision tree for the upper and lower parts are the same, other a condition to check if the detected line appears in the upper right part and not the left lower part or vice versa. This condition is checked since this line may be an extension

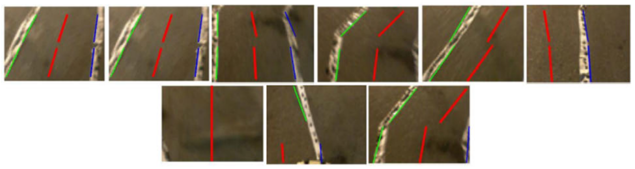

FIGURE 12. Examples of the algorithm behavior after passing through the decision trees(As a reference the green is the left detected line, the blue line is the right detected line and red line is the calculated vehicle heading).

of the line detected in the opposite lower part (curved lane line). If this condition is true, the line in upper part is not considered in steering angle calculation step. But if the condition is false, the decision tree will follow the same conditions that are already checked of the lower parts. Realtime testing results of their approach are depicted in Figure 12.

When both the left and the right lines exist, the desired steering angle is based on the average of the two lines' slopes. But if only one line exists, the desired steering angle is based on the slope of line detected. In case when no line exists, the vehicle should proceed with zero steering angle.

III. STEERING ANGLE PREDICTION USING IMITATION-LEARNING-BASED APPROACH

Recent years have witnessed emergence and prevalence of ANNs which has evoked a storm in intelligent transportation systems. Among these, leveraging ANN models for vision and control mechanisms required for lane keeping are notable breakthroughs. An amazing development in this regard is steering angle prediction through imitation-learning-based approach. Research in imitation-learning-based steering angle prediction started with second competition involving the Udacity self-driving car. The winner of the competition developed nine-layered CNN model [84], which was determined to be successful in autonomously driving the vehicle in the Udacity simulator. Since then, continuous research has been conducted on the design of optimal neural network architectures for steering angle prediction and the discovery of hyperparameters that deliver optimal training results.

Employing the imitation learning technique, the machine learns the steering angles to be actuated on different scenarios through human driving demonstration. For this purpose, images/sequence of images and steering angles induced during driving the vehicle are collected simultaneously, which are used as training data the ANN model. That is, the steering angle is predicted by the ANN model using raw image pixels as input.

Solving any problem using ANN involves following steps: ANN model architecture formulation, dataset collection, training ANN model & ANN model testing, and lastly optimizing ANN and retraining it to improve the results.

1) NN ARCHITECTURE FORMULATION

For the purpose of steering angle prediction using imitation learning approach, various architectures of ANNs and their efficiency have been explored. Most widely used ANNs for this purpose are Convolutional Neural Network (CNN), Long-short-term-memory network (LSTM) and their variants. Table 4 presents some recent studies conducted for steering angle prediction using imitation-learning-based approach.

a: CNN

It has been convincingly shown over the last few years, that CNNs can produce a rich representation of the input image by embedding it to a fixed-length vector, such that this representation can be used for a variety of vision tasks [85]. CNNs are basically proficient in analyzing visual imagery.

The basic structure of a CNN has an input and an output layer, as well as multiple hidden layers. Hidden layers of a CNN consist of series of convolution layer, pooling layer, normalization layer and fully connected layers. Convolutional layers basically convolve over the data (usually images) with a multiplication or other dot product. This layer is regarded as convolution only by convention. Mathematically, a sliding window or filter performs dot product with the given image. This operation has great importance for the indices in the matrix, in that it affects how weights are determined at a specific index point. The output of this layer is then directed to the next layer called pooling layer, which is intended to reduce the size of data in order to reduce computational complexity [86], [87].

Due to remarkable performance in visual imagery understanding, CNN and its variants are being used for the task of steering angle prediction. Various CNN architectures, differing in number of layers and neurons in each layer, have been explored for the purpose of steering angle prediction [4], [84], [88]–[93].

b: LONG SHORT-TERM MEMORY NETWORK

Unlike standard feedforward neural networks, LSTMs have feedback connections. LSTM not only can process single data points (e.g., images), rather they can proficiently process entire sequences of data (such as speech or video) [94]. Due to the outstanding performance of LSTM in sequence learning and understanding, it has been used to AVs for efficient lane keeping by analyzing the dependencies between consecutive image frames. LSTMs are used to interpret temporal dependencies and is mostly placed after CNN which detects features. This CNN-LSTM assorted model has been employed by several studies [95]–[97].

2) DATASET COLLECTION

To utilize ANN for any purpose, training it with suitable data is the first step. To train ANNs for the purpose of steering an AV, different parameters (like steering angles and speed etc.) are obtained through human demonstration during driving. The required dataset in this regard can be produced using a simulation software (like CARSIM, CARLA and TORCS) [98]–[100], or by a human driven vehicle whose actions and decisions are recorded using onboard cameras and angle sensor [101]. Moreover, many state-of-the-art datasets (such as, DIPLECS, Udacity and Comma.ai) are available publicly which can also be utilized to train ANN models for steering angle prediction. Purpose of training ANNs is to make the system familiar with various scenarios and appropriate decisions to be taken under different circumstances. Hence a dataset for steering angle prediction should provide comprehensive coverage of various types of roads and illumination conditions. Collecting the dataset covering all kinds of illumination conditions and road structures in real world is an expensive process in terms of resources and time. A solution to this problem is 'data augmentation', which is basically a technique to increase the amount of training data by applying transformations on available data.

Tian *et al.* [102] proved through experiments that applying realistic transformations on images and using the original as well as transformed images for training, elevated the performance of ANN. These image transformations include intensity, color, dimension, spatial and other realistic transformations. The majority of other studies also performed the image transformations for more exposure of scenarios to the

TABLE 4. Imitation learning approach for steering angle prediction.

Ref	Year	ANN used	Number of layers of ANN	Dataset used	ANN model Testing	Critic	Evaluation metrics
[84]	2016	CNN	9 layers	Manually collected data	Testing on simulated as well as real-time environment	Lane changing scenarios were not handled	Model was able to drive accurately 98% of time for 10 miles of driving in real-time
[88]	2018	Inception V3 (variant of CNN)	48 layers	25 hour driving in GTAV game	Testing was done by driving car in simulated environment of GTAV game	Pretrained inception V3 weights were not able to recognize the patterns in input frame from GTAV game	35% classification accuracy
[89]	2017	CNN-LSTM	1 input layer, 2 hidden layers, 1 output layer (total 4 layers)	Manually collected dataset using prototype car	Testing was done on a prototype car using Raspberry Pi module	Testing done on a toy car rather than a real car	95 % classification accuracy
[90]	2017	CNN	3 convolutional + 4 Relu + 3 FC (total 10 layers)	152K frames from "comma.ai" dataset	Testing was done on videos from "comma.ai" dataset	Night illumination scenarios were not handled, model was not tested to drive car in simulation nor in real world	MSE of 2.42 and standard deviation error of 3.26
[91]	2017	CNN	1 output, 1 output, 3 convolution, 1 FC layers	Dataset collected through driving in CARSIM simulator	Testing was done by driving car in CARSIM simulator	Does not have reasonably well accuracy on unseen data	ME of 0.0338 and 0.0346 standard deviation using NAG optimizer
[92]	2019	CNN-LSTM	Input layer, CNN (having 6 convolution, 2 FC, 1 input layer), 2 FC and a LSTM layer at the end	dataset collected through driving in CARLA simulator	Testing was done by driving car in CARLA simulator	A single end-to-end network for lane steering control, speed adjustment, traffic light and speed limit signs identification is computationally expensive	0.014 validation loss & 0.022 testing loss on CNN-LSTM
[93]	2018	CNN	1 input layer, 1 output layer, 2 hidden layers (total 4 layers)	10868 manually collected image frames	Testing was done on a prototype car using Raspberry Pi module	Lane changing scenarios are not handled	Model drove the prototype vehicle

ANN e.g., adding shadow to random parts of images and flipping images horizontally etc.

3) ANN TRAINING AND TESTING

Training of an ANN model is of critical importance. The training process of any ANN model is basically aimed to find a set of optimal network parameters. These optimal parameters are saved in some form (depending on software being used) as a trained model. Efficiency and accuracy of this trained model are tested on test dataset and validation dataset through evaluation metrics. Mostly used evaluation metrics for measuring the performance of any ANN model on test dataset is the mean error (ME), mean square error (MSE), root mean square error (RMSE) and standard deviation error. Whereas, some researchers used percentage accuracy for reporting the results. Supervised training is the dominant trend followed to train ANN models for steering angle prediction purpose. Yet a few researchers used unsupervised technique for this purpose. Yang *et al.* [103] used an unsupervised learning algorithm to train an improved auto-encoder at the first stage of training. Then in fine tuning stage, which is the second stage of training, the model was trained using supervised learning.

ANN training frameworks which are mostly used for training steering angle predictive models include: Keras with tensorflow as backend, Caffe [104], and Pytorch. Research studies in which these frameworks have been used are listed in the Table 5.

TABLE 5. Frameworks for training ANN models for purpose of steering angle prediction.

Frameworks	References
Caffe	[91], [105]–[112]
Pytorch	[113]–[117]
Keras with tensorflow as backend	[88], [118]–[124]

For testing the performance of the ANN models, three approaches have been used in the literature: (i) Using offline images and videos as test set (ii) Using an autonomously driving toy car (iii) Testing on driving simulator (iv) On-road testing

In the first method of testing, researchers reported average prediction results on test dataset of images or videos which have already been recorded. For example, Chen and Huang [90] used CNN model having 3 convolutional layers, 4 relu layers, and 3 fully connected layers. Five video clips having 152K frames were taken from comma.ai dataset for training and 2 video clips were taken for testing from comma.ai dataset. The mean absolute error found was 2.42 and the standard deviation of the error was 3.26.

In the second method of testing, a toy car is used. For example, Jain [93] used steering angle predictive CNN to autonomously steer an Arduino driven car. Implementation was done using Raspberry Pi and a camera module mounted on the top of the prototype car. CNN architecture used consisted of 128 input nodes, two hidden layers, each having 32 neurons and lastly an output layer having four neurons for each of four outputs (i.e., left, right, stop and forward). 10868 images frames extracted from video were used for training process. Though their proposed model was able to efficiently drive a prototype. Yet they did not perform Real-time road testing.

In the third type of testing method, driving simulators are used for testing performance of the ANN model. Rausch *et al.* [91] developed CNN-based autonomous vehicle steering model and simulated it using car simulator (CARSIM). It contained 3 convolutional layers, 1 fully connected layer, 1 input & 1 output layer and a batch size of 128. They used a single camera as a sensor for analysis of environment. For training of their model, they gathered frames labeled by steering angles data obtained through human driver demonstration using joystick wheel in CARSIM simulation. They used neural network framework CAFFE. For updating weights and bias they used three solvers Nesterov's accelerated gradient (NAG) solver getting mean error of 0.0338, SGD solver getting error of 0.0395 and Adam solver whose error was 0.0465. Among these Nesterov's accelerated gradient (NAG) solver was found to perform best on their model. Haavaldsen *et al.* [92] simulated AV driving on Carla software [100] to train and test CNN as well as CNN-LSTM models. CNN used was inspired by DAVE-2 [84] developed by NVIDIA. They used 10 million simulation steps for training to auto steer the vehicle in Carla environment. Chaudhari *et al.* [88] used Inception V3 to enable learning steering angles, throttle, acceleration and deceleration decisions based on a given image frame. The dataset was collected by 25 hours driving car in GTAV game (used as simulator). Testing was also done using different tracks of GTAV game and 35% accuracy was obtained.

The fourth type of testing involves utilizing the steering angle predicted by the ANN model to autonomously steer a real car in realtime. Bojarski *et al.* [84] trained their model using 9 layers of convolutional neural network (CNN) using 10 frames per second (FPS) of video. They tested their model by 3 hours of 100 miles driving in a simulated environment, and 10 miles of real road driving. The model was able to drive accurately in real time for 98% of the time. The problem with their approach is that it did not tackle scenarios with obstacles, vehicle or any object in front. Moreover, no lane changing scenarios was taken into consideration for training.

4) OPTIMIZING ANN FOR OPTIMIZED RESULTS

Any neural network has parameters and hyperparameters, each configuration of which gives different results. Therefore, neural network designed for any problem needs tuning, to get high accuracy and reduced error. Hence parameter as well as hyperparameter tuning of a neural network is of chief importance.

a: PARAMETER TUNING OF NEURAL NETWORK

For the parameter tuning of ANNs, gradient methods are most widely used, among which, stochastic gradient descent using the backpropagation algorithm is the most popular one [125]. Backpropagation has some drawbacks. Firstly, it has ''scaling problem'' i.e., it works well on simple and less complex problems, yet its performance degenerates rapidly as the complexity of problem increases. Secondly, it has a high tendency of getting trapped in local minima especially in multimodal problems. There is a way to escape these local minima with a high enough momentum, yet it lacks knowledge of whether the succeeding one will be better or worse. When the global minima are hidden among local minima, it can keep bouncing between local minima without having overall improvement [126]. The third drawback of backpropagation is that it has slow convergence. Another chief weakness of the gradient methods is that the derivative information is essential such that the error function to be minimized has to be differentiable and continuous. Moreover, backpropagation has a high dependency on the initial parameters. To solve these problems, nature-inspired algorithms are being used for parameter training of neural network used for huge, complex, multimodal and nondifferentiable domain [126]–[128]. Various studies have demonstrated that nature inspired optimization algorithms outperform significantly than backpropagation in training the neural networks [129]–[133]. Another popularly used optimizer used for steering angle prediction is Adam optimizer and was found to outperform standard SGD optimizing method [97], [122], [134]–[137]. Rausch *et al.* [91] also employed Nesterov's accelerated gradient (NAG) and found that it outperforms Adam and SGD optimizer.

b: HYPERPARAMETER TUNING OF NEURAL NETWORK

Hyperparameter tuning is another critical aspect of training a neural network. The efficiency of a neural network highly depends on its hyperparameter tuning. Hyperparameters of a neural network include learning rate, number of epochs, batch size, activation function, dropout for regularization, number of hidden layers and units. A convolutional neural network has convolution, Relu, pooling and fully connected layers. The organization of these layers and the size of filters in each one of them are chosen during hyperparameter tuning of hidden layers and units of CNN. Each combination of all the hyperparameter values has different impact on learning network parameters of neural network.

Hyperparameter optimization is an attempt to identify set of optimal hyperparameters which minimizes the generalization error for the given problem. This becomes very challenging when the dimensionality of the hyperparameter space increases. Especially, deep neural networks have many different hyperparameters that can be adjusted to any given input data set, resulting in a high-dimensional search space.

For this process of optimizing hyperparameters, brute force technique has been widely used in the past, i.e., manually adjusting each hyperparameter by practitioner and finding at which combination of these values the model gives the best result. Especially in case of CNN with a potentially high number of filters on each layer hyperparameter setting can take a long time. So, using brute force for this purpose is not an efficient way. Recently, EA (GA and differential algorithms, etc.) and swarm-based algorithms (e.g., Particle swarm optimization algorithm) are emerging as optimization techniques for this purpose. These techniques are inspired by biological phenomenon of evolution and collaborative social behavior of animals. Mostly Genetic algorithm [138]–[141], Particle swarm optimization [142]–[147] and their variants are used for this purpose.

Nature inspired algorithms proved efficiency for parameter and hyperparameter search problems separately. Yet several researchers also used these algorithms to simultaneously train ANN network and optimize its architecture [148]–[150]. An another area of research in this field is to combine nature inspired algorithms by traditional gradient search techniques for speeding up convergence and having higher accuracy [151], [152].

IV. OPEN RESEARCH QUESTIONS

Despite several advancements in the steering angle prediction of AVs, there are still many challenges that require further study. In this section, we outline some of the challenges and open research questions revealed in the literature survey which are as follows:

A. CONVOLUTIONAL NEURAL NETWORK OPTIMIZATION

Bat algorithm has been used for optimization of hyperparameters by a simple feedforward neural network; and showed better results for hyperparameter tuning of feedforward ANN as compared to GA and PSO [153]. Yet, to the best of our knowledge Bat algorithm has not been used for the optimization of CNN hyperparameters. Hyperparameter optimization of a neural network is a non-convex, non-linear and complex global optimization problem. To solve these kind of problems, Bat algorithm and its variants showed efficient results as compared to various other popular nature inspired metaheuristic algorithms [153]–[155]. Wang and Guo [154] proposed a variant of Bat algorithm designed by incorporating pitch adjustment operation of Harmony Search. To verify effectiveness of this improved Bat algorithm, they applied fourteen standard benchmark functions and found that it has superior performance in global optimization problems as compared to Ant colony optimization, Differential evolution algorithm, Genetic algorithm, and particle swarm optimization. By considering these groundworks, it is expected that a variant of Bat algorithm would outperform in hyperparameter optimization of CNN for the purpose of predicting steering angle for an autonomous vehicle. Therefore, exploring the efficiency of bat algorithm for optimizing CNN structure can be an interesting area of research. In order to achieve this, each hyperparameter of CNN can be represented by a dimension of bat. Hence a set of hyperparameters can be encoded as a position of a bat. In this way, number of bats represent the number of hyperparameter's set and the fitness of bat represents the evaluation metric of the CNN model. Each hyperparameter set is improved by updating the position of bat using equations of bat algorithm. The process of updating the position of bats in the population is to be repeated upto maximum number of specified generations.

B. AFFECTIVE DRIVABLE ROAD AREA DETECTION

Riaz & Niazi [156] concluded through their research study that more robust collision avoidance can be achieved in AVs through combining human emotions with a cognitive agent. Likewise, a potential research horizon can be to incorporate emotion feature in road boundary detection. As in real world driving, various kinds of risky road scenarios are encountered including bridges and roads on the hills, etc. In developing countries most of the hilly roads are unstructured, unmarked and not properly built. Moreover, scenarios where the system is uncertain about the drivable road area due to heavy traffic occluding the lane markings or due to overexposure, should be discerned as high risk scenarios. For these kinds of scenarios, an element of fear can be incorporated into AVs for safe autonomous lane keeping. To achieve this, fuzzy rules can be applied based on the intensity of fear (e.g., very high fear, high fear, medium fear, low fear, very low fear). The intensity of fear corresponds to the level of risk. In high risk scenarios, speed of the vehicle should be reduced and an emergency handling module should be activated. The emergency handling module can be designed in such a way that firstly it analyses the problem and then act upon solving the problem accordingly. For example, in case of unstructured road and occluded lane markings, the steering angle can be based on position of the front vehicle in horizontal plane in the frame, instead of determining the angle based on road. Quick analysis of the problem can be done through a function determining the average pixels of the image.

C. TESTING SCENARIOS

Literature regarding real world imitation-learning-based steering angle prediction deprives a comprehensive study covering various testing scenarios varying in road types, weather and illumination conditions. The researchers in the domain of ANN-based steering control of AVs heavily rely on artificial datasets for the experimentation. However, the use of artificial datasets has challenges in conducting experiment with the aim of deploying the results in a real-world environment [4]. There is a need for a comprehensive research study covering real world road scenarios with different weather conditions (such as snow, rain, sunny and cloudy), illumination conditions and road types (including unpaved and broken roads).

D. WEIGHTS INITIALIZATION IN ANN-BASED STEERING ANGLE PREDICTION

The performance and convergence of majority ANN highly depends on its weights initialization [157], [158]. Modern deep learning frameworks such as Keras, Caffe and Torch etc provide a facility of high level synthesis of ANN models and controlling its training parameters. Researchers are using automatic weight initialization using these deep learning frameworks for the purpose of steering angle prediction. Pretrained weights of various outperforming ANN models are available the classification task. Yet, employing these pretrained weights for regression task of steering angle prediction is not feasible. Hence there is a need for a detailed research study on weight initialization for ANN architectures designed for the purpose of steering angle prediction.

E. PRACTICAL COMPARISON OF IMITATION-LEARNING-BASED APPROACH AND COMPUTER-VISION-BASED APPROACH

To be best of our knowledge, a practical comparison of the two approaches of steering angle prediction has yet to be done. Hence a study has to be conducted presenting the comparison of processing time, accuracy and other evaluation factors. For this purpose, recent baseline researches of both approaches can be implemented by practically and then results can be evaluated and compared.

F. RESEARCH STUDY EVALUATING DIFFERENT TECHNIQUES OF DRIVABLE ROAD AREA DETECTION

Various techniques have been proposed by researchers for drivable road area detection. As the testing scenarios undertaken in each research study varies, supremacy of an approach over the other cannot be assured. Hence, there is a need for a research study which evaluates different techniques using a single dataset. The selected dataset should cover various scenarios i.e., different weather conditions, traffic conditions, curves, colors of lane markings, and types of lane markings etc.

V. CONCLUSION

Advanced driver assistance technologies such as lane keeping are being incorporated into the vehicles in order to reduce chances of accidents. Lane keeping systems are determined to counter unintentional road departures, for which an accurate steering angle prediction is crucial. In this paper, the problem of steering angle prediction and various techniques for solving it has been discussed. Vision-based steering angle prediction involves analyzing road area using camera and steering the vehicle autonomously within the road boundaries. Major challenge in this regard is to make the system robust in various scenarios, such as illumination changes, curved & straight roads, urban roads & highways, traffic conditions in surrounding of ego vehicle, and different artifacts on the roads (e.g., shadows and cracked roads etc). In order to meet this challenge, various solutions proposed by researchers have been highlighted in this paper.

The first approach to steering angle prediction is through computer vision escorted by image processing techniques. The first step to this approach is capturing the vehicle environment through the camera. The obtained image frames are then preprocessed to enhance the required features or portions of the image. These preprocessed image frames are then used for road detectors and/or tracking algorithms. By analyzing the drivable road region, steering angle and other parameters are derived by AV for maneuvering the vehicle within the lane boundaries of the road. This is done by the vehicle controller continuously sending commands to the actuators according to the road dynamics and vehicle state.

The second approach to steering angle prediction is through ANNs. This involves estimation of steering angle by inputting frames or sequences of frames without performing extra road region extraction processes. The obtained steering angle along with other parameters is leveraged by the controller for keeping ego vehicle on the road. Any neural network model requires a dataset to be trained on for achieving a particular task. For the neural network designed for steering angle prediction, the dataset is obtained through human demonstration through driving a car. After designing any neural network model, optimization is crucial for obtaining the required results. Traditionally, this is done through hit and trial by the practitioner, but this is time consuming and inefficient method. A more efficient way is to use nature-inspired optimizartion algorithms for this purpose, which are being in the research focus these days.

In this paper, various techniques adopted by various researchers under both the approaches for steering angle prediction are discussed. At the end of the paper, open research problem regarding steering angle prediction has been highlighted. For future work, we are planning to conduct a review analysis of the latest researches on the various perception devices for autonomous vehicles.

REFERENCES

[1] A. El Khatib, C. Ou, and F. Karray, "Driver inattention detection in the context of next-generation autonomous vehicles design: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4483–4496, Nov. 2020.

[2] A. Mammeri, G. Lu, and A. Boukerche, "Design of lane keeping assist system for autonomous vehicles," in *Proc. 7th Int. Conf. New Technol., Mobility Secur. (NTMS)*, Jul. 2015, pp. 1–5.

[3] A. Khodayari, A. Ghaffari, S. Ameli, and J. Flahatgar, "A historical review on lateral and longitudinal control of autonomous vehicle motions," in *Proc. Int. Conf. Mech. Electr. Technol.*, Sep. 2010, pp. 421–429.

[4] U. M. Gidado, H. Chiroma, N. Aljojo, S. Abubakar, S. I. Popoola, and M. A. Al-Garadi, "A survey on deep learning for steering angle prediction in autonomous vehicles," *IEEE Access*, vol. 8, pp. 163797–163817, 2020.

[5] A. Oussama and T. Mohamed, "A literature review of steering angle prediction algorithms for self-driving cars," in *Proc. Int. Conf. Adv. Intell. Syst. Sustain. Develop.* Marrakech, Morocco: Springer, 2019, pp. 30–38.

[6] V. S. Dev, V. V. S. Variyar, and K. P. Soman, "Steering angle estimation for autonomous vehicle," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 871–876.

[7] Y. Xing, C. Lv, L. Chen, H. Wang, H. Wang, D. Cao, E. Velenis, and F.-Y. Wang, "Advances in vision-based lane detection: Algorithms, integration, assessment, and perspectives on ACP-based parallel vision," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 3, pp. 645–661, May 2018.

[8] Z. M. Sani, H. A. Ghani, R. Besar, and W. Loi, "Daytime road marker recognition using grayscale histogram and pixel values," *Internetworking Indonesia J.*, vol. 8, no. 1, pp. 11–16, 2016.

[9] A. S. Rathore, "Lane detection for autonomous vehicles using OpenCV library," *Int. Res. J. Eng. Technol.*, vol. 6, no. 1, pp. 1326–1332, 2019.

[10] S. P. Narote, P. N. Bhujbal, A. S. Narote, and D. M. Dhane, "A review of recent advances in lane detection and departure warning system," *Pattern Recognit.*, vol. 73, pp. 216–234, Jan. 2018.

[11] C. Lee and J.-H. Moon, ''Robust lane detection and tracking for real-time applications,'' *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 4043–4048, Dec. 2018.

[12] Q. Wen, Z. Yang, Y. Song, and P. Jia, ''Road boundary detection in complex urban environment based on low-resolution vision,'' in *Proc. 11th Joint Int. Conf. Inf. Sci.* Beijing, China: Atlantis Press, 2008, pp. 208–214.

[13] A. Chahal, ''*In situ* detection of road lanes using raspberry Pi,'' Dept. Comput. Sci., Utah State Univ., Logan, UT, USA, 2018.

[14] R. Klette, Y. Xu, Z. Xiao, R. Song, and H. Chen, ''Lane detection algorithm based on geometric moment sampling,'' *Scientia Sinica Informationis*, vol. 47, no. 4, pp. 455–467, Apr. 2017.

[15] T.-Y. Sun, S.-J. Tsai, and V. Chan, ''HSI color model based lane-marking detection,'' in *Proc. IEEE Intell. Transp. Syst. Conf.*, Sep. 2006, pp. 1168–1172.

[16] M. Li, Y. Li, and M. Jiang, ''Lane detection based on connection of various feature extraction methods,'' *Adv. Multimedia*, vol. 2018, pp. 1–13, Aug. 2018.

[17] Y. Kortli, M. Marzougui, B. Bouallegue, J. S. C. Bose, P. Rodrigues, and M. Atri, ''A novel illumination-invariant lane detection system,'' in *Proc. 2nd Int. Conf. Anti-Cyber Crimes (ICACC)*, Mar. 2017, pp. 166–171.

[18] Y. Feng, W. Rong-Ben, and Z. Rong-Hui, ''Research on road recognition algorithm based on structure environment for ITS,'' in *Proc. ISECS Int. Colloq. Comput., Commun., Control, Manage.*, vol. 1, 2008, pp. 84–87.

[19] Q.-B. Truong and B.-R. Lee, ''New lane detection algorithm for autonomous vehicles using computer vision,'' in *Proc. Int. Conf. Control, Autom. Syst.*, Oct. 2008, pp. 1208–1213.

[20] Q. B. Truong, B. R. Lee, N. G. Heo, Y. J. Yum, and J. G. Kim, ''Lane boundaries detection algorithm using vector lane concept,'' in *Proc. 10th Int. Conf. Control, Autom., Robot. Vis.*, Dec. 2008, pp. 2319–2325.

[21] A. A. Assidiq, O. O. Khalifa, M. R. Islam, and S. Khan, ''Real time lane detection for autonomous vehicles,'' in *Proc. Int. Conf. Comput. Commun. Eng.*, May 2008, pp. 82–88.

[22] A. Kuçukmanisa, G. Tarim, and O. Urhan, ''Real-time illumination and shadow invariant lane detection on mobile platform,'' *J. Real-Time Image Process.*, vol. 16, no. 5, pp. 1–14, 2017.

[23] J. Matas, O. Chum, M. Urban, and T. Pajdla, ''Robust wide-baseline stereo from maximally stable extremal regions,'' *Image Vis. Comput.*, vol. 22, no. 10, pp. 761–767, Sep. 2004.

[24] A. Parajuli, M. Celenk, and H. B. Riley, ''Robust lane detection in shadows and low illumination conditions using local gradient features,'' *Open J. Appl. Sci.*, vol. 3, no. 1, p. 68, 2013.

[25] K. Ishikawa, K. Kobayashi, and K. Watanabe, ''A lane detection method for intelligent ground vehicle competition,'' in *Proc. SICE Annu. Conf.*, vol. 1, 2003, pp. 1086–1089.

[26] Y. M. L. J. W. Hong and Z. Bo, ''Vision based real time vehicle guidance on THMR V Part I: Unstructured road detection,'' Tech. Rep., 2001.

[27] G. Deng and L. W. Cahill, ''An adaptive Gaussian filter for noise reduction and edge detection,'' in *Proc. IEEE Conf. Rec. Nucl. Sci. Symp. Med. Imag. Conf.*, Oct. 1993, pp. 1615–1619.

[28] J. Ton, A. K. Jain, W. R. Enslin, and W. D. Hudson, ''Automatic road identification and labeling in Landsat 4 TM images,'' *Photogrammetria*, vol. 43, no. 5, pp. 257–276, Jun. 1989.

[29] J. Wang, Z. Ji, and Y.-T. Su, ''Unstructured road detection using hybrid features,'' in *Proc. Int. Conf. Mach. Learn. Cybern.*, vol. 1, Jul. 2009, pp. 482–486.

[30] T. Liu, Z. Chen, Y. Yang, Z. Wu, and H. Li, ''Lane detection in low-light conditions using an efficient data enhancement: Light conditions style transfer,'' in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1394–1399.

[31] M. Bertozz, A. Broggi, and A. Fascioli, ''Stereo inverse perspective mapping: Theory and applications,'' *Image Vis. Comput.*, vol. 16, no. 8, pp. 585–590, Jun. 1998.

[32] G. Y. Jiang, T. Y. Choi, S. K. Hong, J. W. Bae, and B. S. Song, ''Lane and obstacle detection based on fast inverse perspective mapping algorithm,'' in *Proc. IEEE Int. Conf. Syst. (SMC), Man Cybern., Cybern. Evolving Syst., Humans, Org., Complex Interact.*, vol. 4, Oct. 2000, pp. 2969–2974.

[33] J. Wang, T. Mei, B. Kong, and H. Wei, ''An approach of lane detection based on inverse perspective mapping,'' in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2014, pp. 35–38.

[34] Z. Ying and G. Li, ''Robust lane marking detection using boundary-based inverse perspective mapping,'' in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 1921–1925.

[35] A. M. Muad, A. Hussain, S. A. Samad, M. M. Mustaffa, and B. Y. Majlis, ''Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system,'' in *Proc. IEEE Region Conf. (TENCON)*, Nov. 2004, pp. 207–210.

[36] J. Shin, E. Lee, K. Kwon, and S. Lee, ''Lane detection algorithm based on top-view image using random sample consensus algorithm and curve road model,'' in *Proc. 6th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2014, pp. 1–2.

[37] Y. Huang, Y. Li, X. Hu, and W. Ci, ''Lane detection based on inverse perspective transformation and Kalman filter,'' *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 2, pp. 643–661, 2018.

[38] D. Chen, Z. Tian, and X. Zhang, ''Lane detection algorithm based on inverse perspective mapping,'' in *Proc. Int. Conf. Man-Mach.-Environ. Syst. Eng.* Zhengzhou, China: Springer, 2019, pp. 247–255.

[39] F. Rakotondrajao and K. Jangsamsi, ''Road boundary detection for straight lane lines using automatic inverse perspective mapping,'' in *Proc. Int. Symp. Intell. Signal Process. Commun. Syst. (ISPACS)*, Dec. 2019, pp. 1–2.

[40] Z. Yu, X. Ren, Y. Huang, W. Tian, and J. Zhao, ''Detecting lane and road markings at a distance with perspective transformer layers,'' 2020, *arXiv:2003.08550*. [Online]. Available: http://arxiv.org/abs/2003.08550

[41] W. Li, F. Qu, Y. Wang, L. Wang, and Y. Chen, ''A robust lane detection method based on hyperbolic model,'' *Soft Comput.*, vol. 23, no. 19, pp. 9161–9174, Oct. 2019.

[42] R. F. Berriel, E. de Aguiar, V. V. D. S. Filho, and T. Oliveira-Santos, ''A particle filter-based lane marker tracking approach using a cubic spline model,'' in *Proc. 28th SIBGRAPI Conf. Graph., Patterns Images*, Aug. 2015, pp. 149–156.

[43] W. Yanqing, C. Deyun, S. Chaoxia, and W. Peidong, ''Vision-based road detection by Monte Carlo method,'' *Inf. Technol. J.*, vol. 9, no. 3, pp. 481–487, Mar. 2010.

[44] M. Hu, W. Yang, M. Ren, and J. Yang, ''A vision based road detection algorithm,'' in *Proc. IEEE Conf. Robot., Autom. Mechatronics*,vol. 2, Dec. 2004, pp. 846–850.

[45] Y. Wang, D. Chen, and C. Shi, ''Vision-based road detection by adaptive region segmentation and edge constraint,'' in *Proc. 2nd Int. Symp. Intell. Inf. Technol. Appl.*, vol. 1, Dec. 2008, pp. 342–346.

[46] N. S. Parameswaran, E. R. Achan, V. Subhashree, and R. Manjusha, ''Road detection by boundary extraction technique and Hough transform,'' in *Proc. Int. Conf. ISMAC Comput. Vis. Bio-Eng.* Palladam, India: Springer, 2018, pp. 1805–1814.

[47] W. Xiang, Z. Juan, and F. Zhijun, ''Unstructured road detection based on contour selection,'' in *Proc. 4th Int. Conf. Smart Sustain. City (ICSSC)*, Shanghai, China, 2017.

[48] R. Jahan, P. Suman, and D. K. Singh, ''Lane detection using canny edge detection and Hough transform on raspberry Pi,'' *Int. J. Adv. Res. Comput. Sci.*, vol. 9, no. 2, p. 85, 2018.

[49] W. Song, L. Liu, X. Zhou, and C. Wang, ''Road detection algorithm of integrating region and edge information,'' in *Proc. Int. Conf. Artif. Intell. Robot. Int. Conf. Autom., Control Robot. Eng. (ICAIR-CACRE)*, 2016, pp. 1–6.

[50] X. Yan and Y. Li, ''A method of lane edge detection based on canny algorithm,'' in *Proc. Chin. Autom. Congr. (CAC)*, Oct. 2017, pp. 2120–2124.

[51] Y. Li, L. Chen, H. Huang, X. Li, W. Xu, L. Zheng, and J. Huang, ''Nighttime lane markings recognition based on canny detection and Hough transform,'' in *Proc. IEEE Int. Conf. Real-time Comput. Robot. (RCAR)*, Jun. 2016, pp. 411–415.

[52] Q. Li, N. Zheng, and H. Cheng, ''Springrobot: A prototype autonomous vehicle and its algorithms for lane detection,'' *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 300–308, Dec. 2004.

[53] D. Gao, W. Li, J. Duan, and B. Zheng, ''A practical method of road detection for intelligent vehicle,'' in *Proc. IEEE Int. Conf. Autom. Logistics*, Aug. 2009, pp. 980–985.

[54] C.-L. Fan and Y.-Y. Ren, ''Study on the edge detection algorithms of road image,'' in *Proc. 3rd Int. Symp. Inf. Process.*, Oct. 2010, pp. 217–220.

[55] M. Samuel, M. Mohamad, S. M. Saad, and M. Hussein, ''Development of edge-based lane detection algorithm using image processing,'' *Int. J. Informat. Visualizat.*, vol. 2, no. 1, pp. 19–22, 2018.

[56] J. He, H. Rong, J. Gong, and W. Huang, ''A lane detection method for lane departure warning system,'' in *Proc. Int. Conf. Optoelectronics Image Process.*, vol. 1, Nov. 2010, pp. 28–31.

[57] D.-K. Lee, J.-S. Shin, J.-H. Jung, S.-J. Park, S.-J. Oh, and I.-S. Lee, ''Real-time lane detection and tracking system using simple filter and Kalman filter,'' in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2017, pp. 275–277.

[58] J. Deng and Y. Han, "A real-time system of lane detection and tracking based on optimized RANSAC B-spline fitting," in *Proc. Res. Adapt. Convergent Syst.*, Montreal, QC, Canada, 2013, pp. 157–164.

[59] D. C. Andrade, F. Bueno, F. R. Franco, R. A. Silva, J. Henrique, Z. Neme, E. Margraf, W. T. Omoto, F. A. Farinelli, A. M. Tusset, S. Okida, M. M. D. Santos, A. Ventura, S. Carvalho, and R. dos Santos Amaral, "A novel strategy for road lane detection and tracking based on a Vehicle's forward monocular camera," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1497–1507, Apr. 2019.

[60] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2008, pp. 7–12.

[61] A. Borkar, M. Hayes, and M. T. Smith, "Robust lane detection and tracking with ransac and Kalman filter," in *Proc. 16th IEEE Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 3261–3264.

[62] Y. Li, A. Iqbal, and N. R. Gans, "Multiple lane boundary detection using a combination of low-level image features," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2014, pp. 1682–1687.

[63] H. Wang, Y. Wang, X. Zhao, G. Wang, H. Huang, and J. Zhang, "Lane detection of curving road for structural highway with straight-curve model on vision," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5321–5330, Jun. 2019.

[64] H. Yoo, U. Yang, and K. Sohn, "Gradient-enhancing conversion for illumination-robust lane detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1083–1094, Sep. 2013.

[65] T.-T. Tran, C.-S. Bae, Y.-N. Kim, H.-M. Cho, and S.-B. Cho, "An adaptive method for lane marking detection based on HSI color model," in *Proc. Int. Conf. Intell. Comput.* Berlin, Germany: Springer, 2010, pp. 304–311.

[66] J. Son, H. Yoo, S. Kim, and K. Sohn, "Real-time illumination invariant lane detection for lane departure warning system," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 1816–1824, Mar. 2015.

[67] N. Apostoloff and A. Zelinsky, "Robust vision based lane tracking using multiple cues and particle filtering," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2003, pp. 558–563.

[68] K. B. Kim and D. H. Song, "Real time road lane detection with RANSAC and HSV Color transformation," *J. Inf. Commun. Converg. Eng.*, vol. 15, no. 3, pp. 187–192, 2017.

[69] W. Liu, J. Gong, and B. Ma, "Research on lane detection method with shadow interference," in *Proc. Chin. Control Conf. (CCC)*, Jul. 2019, pp. 7704–7709.

[70] A. Borkar, M. Hayes, M. T. Smith, and S. Pankanti, "A layered approach to robust lane detection at night," in *Proc. IEEE Workshop Comput. Intell. Vehicles Veh. Syst.*, Mar. 2009, pp. 51–57.

[71] Y.-W. Seo and R. R. Rajkumar, "Detection and tracking of boundary of unmarked roads," in *Proc. 17th Int. Conf. Inf. Fusion (FUSION)*, 2014, pp. 1–6.

[72] A. J. Humaidi, M. A. Fadhel, and A. R. Ajel, "Lane detection system for day vision using altera DE2," *Telkomnika*, vol. 17, no. 1, pp. 349–361, 2019.

[73] G. Fan, B. Li, and Q. Han, "Robust lane detection and tracking based on machine vision," *ZTE Commun.*, vol. 18, no. 4, pp. 69–77, 2020.

[74] J. Hu, S. Xiong, Y. Sun, J. Zha, and C. Fu, "Research on lane detection based on global search of dynamic region of interest (DROI)," *Appl. Sci.*, vol. 10, no. 7, p. 2543, Apr. 2020.

[75] J. Xiao, W. Xiong, Y. Yao, L. Li, and R. Klette, "Lane detection algorithm based on road structure and extended Kalman filter," *Int. J. Digit. Crime Forensics*, vol. 12, no. 2, pp. 1–20, Apr. 2020.

[76] B. Dorj and D. J. Lee, "A precise lane detection algorithm based on top view image transformation and least-square approaches," *J. Sensors*, vol. 2016, pp. 1–13, Jan. 2016.

[77] P. M. Daigavane and P. R. Bajaj, "Road lane detection with improved canny edges using ant colony optimization," in *Proc. 3rd Int. Conf. Emerg. Trends Eng. Technol.*, Nov. 2010, pp. 76–80.

[78] R. Muthalagu, A. Bolimera, and V. Kalaichelvi, "Lane detection technique based on perspective transformation and histogram analysis for self-driving cars," *Comput. Electr. Eng.*, vol. 85, Jul. 2020, Art. no. 106653.

[79] M. A. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[80] V. Umamaheswari, S. Amarjyoti, T. Bakshi, and A. Singh, "Steering angle estimation for autonomous vehicle navigation using Hough and Euclidean transform," in *Proc. IEEE Int. Conf. Signal Process., Informat., Commun. Energy Syst. (SPICES)*, Feb. 2015, pp. 1–5.

[81] J. Sujatha, "Computer vision based novel steering angle calculation for autonomous vehicles," in *Proc. 2nd IEEE Int. Conf. Robotic Comput. (IRC)*, Jan. 2018, pp. 143–146.

[82] K. D. N. Tu, H. D. Nguyen, and T. H. Tran, "Vision based steering angle estimation for autonomous vehicles," in *Proc. Int. Conf. Adv. Technol. Commun. (ATC)*, Oct. 2020, pp. 187–192.

[83] M. AbdElrahman, A. Bayoumy, G. Elbayoumi, and M. Bayoumi, "Vision-based road tracking of wheeled mobile robot," in *Proc. Int. Conf. Aerosp. Sci. Aviation Technol.*, vol. 15, May 2013, pp. 1–14.

[84] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Müller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*. [Online]. Available: http://arxiv.org/abs/1604.07316

[85] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*. [Online]. Available: http://arxiv.org/abs/1312.6229

[86] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proc. Int. Conf. Eng. Technol. (ICET)*, Aug. 2017, pp. 1–6.

[87] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," 2013, *arXiv:1312.5402*. [Online]. Available: http://arxiv.org/abs/1312.5402

[88] R. Chaudhari, S. Dubey, J. Kathale, and R. Rao, "Autonomous driving car using convolutional neural networks," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 936–940.

[89] H. M. Eraqi, M. N. Moustafa, and J. Honer, "End-to-end deep learning for steering autonomous vehicles considering temporal dependencies," 2017, *arXiv:1710.03804*. [Online]. Available: http://arxiv.org/abs/1710.03804

[90] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1856–1860.

[91] V. Rausch, A. Hansen, E. Solowjow, C. Liu, E. Kreuzer, and J. K. Hedrick, "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 4914–4919.

[92] H. Haavaldsen, M. Aasboe, and F. Lindseth, "Autonomous vehicle control: End-to-end learning in simulated urban environments," in *Proc. Symp. Norwegian AI Soc.* Trondheim, Norway: Springer, 2019, pp. 40–51.

[93] A. K. Jain, "Working model of self-driving car using convolutional neural network, Raspberry Pi and Arduino," in *Proc. 2nd Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Mar. 2018, pp. 1630–1635.

[94] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," 2014, *arXiv:1402.1128*. [Online]. Available: http://arxiv.org/abs/1402.1128

[95] R. Valiente, M. Zaman, S. Ozer, and Y. P. Fallah, "Controlling steering angle for cooperative self-driving vehicles utilizing CNN and LSTM-based deep networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2423–2428.

[96] L. Chi and Y. Mu, "Deep steering: Learning end-to-end driving model from spatial and temporal visual cues," 2017, *arXiv:1708.03798*. [Online]. Available: http://arxiv.org/abs/1708.03798

[97] S. Du, H. Guo, and A. Simpson, "Self-driving car steering angle prediction based on image recognition," 2019, *arXiv:1912.05440*. [Online]. Available: http://arxiv.org/abs/1912.05440

[98] X. Ji, X. He, C. Lv, Y. Liu, and J. Wu, "Adaptive-neural-network-based robust lateral motion control for autonomous vehicle at driving limits," *Control Eng. Pract.*, vol. 76, pp. 41–53, Jul. 2018.

[99] Q. Zhu, Z. Huang, D. Liu, and B. Dai, "An adaptive path tracking method for autonomous land vehicle based on neural dynamic programming," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Aug. 2016, pp. 1429–1434.

[100] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," 2017, *arXiv:1711.03938*. [Online]. Available: http://arxiv.org/abs/1711.03938

[101] S. Chen, S. Zhang, J. Shang, B. Chen, and N. Zheng, "Brain-inspired cognitive model with attention for self-driving cars," *IEEE Trans. Cognit. Develop. Syst.*, vol. 11, no. 1, pp. 13–25, Mar. 2019.

[102] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated testing of deep-neural-network-driven autonomous cars," in *Proc. 40th Int. Conf. Softw. Eng.*, May 2018, pp. 303–314.

[103] Y. Yang, Z. Wu, Q. Xu, and F. Yan, "Deep learning technique-based steering of autonomous car," *Int. J. Comput. Intell. Appl.*, vol. 17, no. 2, Jun. 2018, Art. no. 1850006.

[104] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, Nov. 2014, pp. 675–678.

[105] S. Yang, W. Wang, C. Liu, W. Deng, and J. K. Hedrick, "Feature analysis and selection for training an end-to-end autonomous vehicle controller using deep learning approach," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1033–1038.

[106] T.-M. Hsu, C.-H. Wang, and Y.-R. Chen, "End-to-end deep learning for autonomous longitudinal and lateral control based on vehicle dynamics," in *Proc. Int. Conf. Artif. Intell. Virtual Reality (AIVR)*, 2018, pp. 111–114.

[107] X. Hu, B. Tang, L. Chen, S. Song, and X. Tong, "Learning a deep cascaded neural network for multiple motion commands prediction in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, early access, Jul. 3, 2020, doi: 10.1109/TITS.2020.3004984.

[108] S. Yang, W. Wang, C. Liu, and W. Deng, "Scene understanding in deep learning-based end-to-end controllers for autonomous vehicles," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 1, pp. 53–63, Jan. 2019.

[109] J. Jhung, I. Bae, J. Moon, T. Kim, J. Kim, and S. Kim, "End-to-End steering controller with CNN-based closed-loop feedback for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 617–622.

[110] S. Song, X. Hu, J. Yu, L. Bai, and L. Chen, "Learning a deep motion planning model for autonomous driving," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1137–1142.

[111] C. J. Holder and T. P. Breckon, "Learning to drive: Using visual odometry to bootstrap deep learning for off-road path prediction," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 2104–2110.

[112] Y. Zhao and Y. Chen, "End-to-end autonomous driving based on the convolution neural network model," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Nov. 2019, pp. 419–423.

[113] L. Du, Z. Zhao, F. Su, L. Wang, and C. An, "Jointly predicting future sequence and steering angles for dynamic driving scenes," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 4070–4074.

[114] F. Johnson and K. Dana, "Feudal steering: Hierarchical learning for steering angle prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 1002–1003.

[115] Q. Wang, L. Chen, B. Tian, W. Tian, L. Li, and D. Cao, "End-to-end autonomous driving: An angle branched network approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 11599–11610, Dec. 2019.

[116] S. Hornauer, K. Zipser, and S. Yu, "Imitation learning of path-planned driving using disparity-depth images," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, Munich, Germany, 2018, pp. 542–548.

[117] L. Chen, Q. Wang, X. Lu, D. Cao, and F.-Y. Wang, "Learning driving models from parallel end-to-end driving data set," *Proc. IEEE*, vol. 108, no. 2, pp. 262–273, Feb. 2020.

[118] V. John, A. Boyali, H. Tehrani, K. Ishimaru, M. Konishi, Z. Liu, and S. Mita, "Estimation of steering angle and collision avoidance for automated driving using deep mixture of experts," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 4, pp. 571–584, Dec. 2018.

[119] Ø. K. Grimnes, "End-to-end steering angle prediction and object detection using convolutional neural networks," M.S. thesis, Dept. Comput. Sci., NTNU, Trondheim, Norway, 2017.

[120] M. V. Smolyakov, A. I. Frolov, V. N. Volkov, and I. V. Stelmashchuk, "Self-driving car steering angle prediction based on deep neural network an example of CarND udacity simulator," in *Proc. IEEE 12th Int. Conf. Appl. Inf. Commun. Technol. (AICT)*, Oct. 2018, pp. 1–5.

[121] N. Merrill and A. Eskandarian, "End-to-end multi-task machine learning of vehicle dynamics for steering angle prediction for autonomous driving," in *Proc. 21st Int. Conf. Adv. Vehicle Technol., 16th Int. Conf. Design Educ.*, Aug. 2019.

[122] T.-D. Do, M.-T. Duong, Q.-V. Dang, and M.-H. Le, "Real-time self-driving car navigation using deep neural network," in *Proc. 4th Int. Conf. Green Technol. Sustain. Develop. (GTSD)*, Nov. 2018, pp. 7–12.

[123] I. G. A. Setta, O. M. Shehata, and M. A. Awad, "Multivariate prediction of correct lane for autonomous electric vehicle using deep learning models," in *Proc. 8th Int. Conf. Control, Mechatronics Autom. (ICCMA)*, Nov. 2020, pp. 127–130.

[124] T. Tiwari, N. Shastry, and A. Nandi, "Deep learning based lateral control system," in *Proc. IEEE Int. Symp. Sustain. Energy, Signal Process. Cyber Secur. (iSSSC)*, Dec. 2020, pp. 1–5.

[125] S. Vempaty and Y. He, "A review of car-trailer lateral stability control approaches," SAE Tech. Paper 0148-7191, 2017.

[126] D. J. Montana and L. Davis, "Training feedforward neural networks using genetic algorithms," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 89, 1989, pp. 762–767.

[127] H. H. Örkcü and H. Bal, "Comparing performances of backpropagation and genetic algorithms in the data classification," *Expert Syst. Appl.*, vol. 38, no. 4, pp. 3703–3709, Apr. 2011.

[128] H. A. Abbass, "Speeding up backpropagation using multiobjective evolutionary algorithms," *Neural Comput.*, vol. 15, no. 11, pp. 2705–2726, Nov. 2003.

[129] J. N. D. Gupta and R. S. Sexton, "Comparing backpropagation with a genetic algorithm for neural network training," *Omega*, vol. 27, no. 6, pp. 679–684, Dec. 1999.

[130] C.-Y. Huang, L.-H. Chen, Y.-L. Chen, and F. M. Chang, "Evaluating the process of a genetic algorithm to improve the back-propagation network: A Monte Carlo study," *Expert Syst. Appl.*, vol. 36, no. 2, pp. 1459–1465, Mar. 2009.

[131] R. S. Sexton and R. E. Dorsey, "Reliable classification using neural networks: A genetic algorithm and backpropagation comparison," *Decis. Support Syst.*, vol. 30, no. 1, pp. 11–22, Dec. 2000.

[132] R. S. Sexton, R. E. Dorsey, and J. D. Johnson, "Toward global optimization of neural networks: A comparison of the genetic algorithm and backpropagation," *Decis. Support Syst.*, vol. 22, no. 2, pp. 171–185, Feb. 1998.

[133] R. S. Sexton and J. N. D. Gupta, "Comparative evaluation of genetic algorithm and backpropagation for training neural networks," *Inf. Sci.*, vol. 129, nos. 1–4, pp. 45–59, Nov. 2000.

[134] W. Yuan, M. Yang, H. Li, C. Wang, and B. Wang, "End-to-end learning for high-precision lane keeping via multi-state model," *CAAI Trans. Intell. Technol.*, vol. 3, no. 4, pp. 185–190, Dec. 2018.

[135] M.-T. Duong, T.-D. Do, and M.-H. Le, "Navigating self-driving vehicles using convolutional neural network," in *Proc. 4th Int. Conf. Green Technol. Sustain. Develop. (GTSD)*, Nov. 2018, pp. 607–610.

[136] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2942–2950.

[137] W. Farag and Z. Saleh, "Behavior cloning for autonomous driving using convolutional neural networks," in *Proc. Int. Conf. Innov. Intell. Informat., Comput., Technol. (ICT)*, Nov. 2018, pp. 1–7.

[138] J. Wu, J. Long, and M. Liu, "Evolving RBF neural networks for rainfall prediction using hybrid particle swarm optimization and genetic algorithm," *Neurocomputing*, vol. 148, pp. 136–142, Jan. 2015.

[139] Z. Arabasadi, R. Alizadehsani, M. Roshanzamir, H. Moosaei, and A. A. Yarifard, "Computer aided decision making for heart disease detection using hybrid neural network-genetic algorithm," *Comput. Methods Programs Biomed.*, vol. 141, pp. 19–26, Apr. 2017.

[140] S. Wang, Y. Zhang, Z. Dong, S. Du, G. Ji, J. Yan, J. Yang, Q. Wang, C. Feng, and P. Phillips, "Feed-forward neural network optimized by hybridization of PSO and ABC for abnormal brain detection," *Int. J. Imag. Syst. Technol.*, vol. 25, no. 2, pp. 153–164, Jun. 2015.

[141] A. Bhandare and D. Kaur, "Designing convolutional neural network architecture using genetic algorithms," in *Proc. Int. Conf. Artif. Intell. (ICAI), Steering Committee World Congr. Comput. Sci.*, 2018, pp. 150–156.

[142] T.-Y. Kim and S.-B. Cho, "Particle swarm optimization-based CNN-LSTM networks for forecasting energy consumption," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2019, pp. 1510–1516.

[143] Y. Wang, H. Zhang, and G. Zhang, "CPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks," *Swarm Evol. Comput.*, vol. 49, pp. 114–123, Sep. 2019.

[144] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, "Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition," *IET Intell. Transp. Syst.*, vol. 12, no. 8, pp. 939–946, Oct. 2018.

[145] T. Yamasaki, T. Honma, and K. Aizawa, "Efficient optimization of convolutional neural networks using particle swarm optimization," in *Proc. IEEE 3rd Int. Conf. Multimedia Big Data (BigMM)*, Apr. 2017, pp. 70–73.

[146] T. Sinha, A. Haidar, and B. Verma, "Particle swarm optimization based approach for finding optimal values of convolutional neural network parameters," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2018, pp. 1–6.

[147] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor, ''Particle swarm optimization for hyper-parameter selection in deep neural networks,'' in *Proc. Genetic Evol. Comput. Conf.*, Jul. 2017, pp. 481–488.
[148] D. Whitley, T. Starkweather, and C. Bogart, ''Genetic algorithms and neural networks: Optimizing connections and connectivity,'' *Parallel Comput.*, vol. 14, no. 3, pp. 347–361, Aug. 1990.
[149] S. A. Harp and T. Samad, ''Optimizing neural networks with genetic algorithms,'' in *Proc. 54th Amer. Power Conf.*, Chicago, IL, USA, vol. 2, 1992, pp. 41–44.
[150] W.-Y. Ling, M.-P. Jia, F.-Y. Xu, J.-Z. Hu, and B.-L. Zhong, ''Optimizing strategy on rough set neural network fault diagnosis system,'' *Proc. CSEE*, vol. 23, no. 5, pp. 98–102, 2003.
[151] I. Aljarah, H. Faris, and S. Mirjalili, ''Optimizing connection weights in neural networks using the whale optimization algorithm,'' *Soft Comput.*, vol. 22, no. 1, pp. 1–15, Jan. 2018.
[152] S. Ding, C. Su, and J. Yu, ''An optimizing BP neural network algorithm based on genetic algorithm,'' *Artif. Intell. Rev.*, vol. 36, no. 2, pp. 153–162, Aug. 2011.
[153] N. S. Jaddi, S. Abdullah, and A. R. Hamdan, ''Multi-population cooperative bat algorithm-based optimization of artificial neural network model,'' *Inf. Sci.*, vol. 294, pp. 628–644, Feb. 2015.
[154] G. Wang and L. Guo, ''A novel hybrid bat algorithm with harmony search for global numerical optimization,'' *J. Appl. Math.*, vol. 2013, pp. 1–21, Dec. 2013.
[155] R. Sedaghati and F. Namdari, ''An intelligent approach based on metaheuristic algorithm for non-convex economic dispatch,'' *J. Operation Autom. Power Eng.*, vol. 3, no. 1, pp. 47–55, 2015.
[156] F. Riaz and M. A. Niazi, ''Enhanced emotion enabled cognitive agent-based rear-end collision avoidance controller for autonomous vehicles,'' *Simulation*, vol. 94, no. 11, pp. 957–977, Nov. 2018.
[157] S. Krishna Kumar, ''On weight initialization in deep neural networks,'' 2017, *arXiv:1704.08863*. [Online]. Available: http://arxiv.org/abs/1704.08863
[158] S. Timotheou, ''A novel weight initialization method for the random neural network,'' *Neurocomputing*, vol. 73, nos. 1–3, pp. 160–168, Dec. 2009.

HAJIRA SALEEM received the bachelor's degree in computer science from Fatima Jinnah Women University, Pakistan. She is currently pursuing the M.S. degree in computer science with the Mirpur University of Science and Technology (MUST), AJK, Pakistan. She is also employed with the Control, Automotive and Robotics Laboratory, National Center of Robotics and Automation, Department of Computer Sciences, MUST. Her research interests include machine learning, affective computing, and intelligent transportation.

FAISAL RIAZ received the Ph.D. degree in cyber physical systems (CPS) from Iqra University, Pakistan, in 2018. He is currently serving as an Associate Professor with the Department of Computer Sciences, Mirpur University of Science and Technology. He also heads the Control, Automotive and Robotics Laboratory, National Center of Robotics and Automation, Department of Computer Sciences, Mirpur University of Science and Technology (MUST), AJK, Pakistan. Under the banner of the CAR-Lab, he has innovated the world's first ever cognitive and emotional autonomous vehicle. He has more than 34 research publications in various national/international research journals and conferences. He is the author of three books. His research interests include vehicular cyber–physical systems, cognitive radio, affective computing, and agent-based modeling. He received the Hall of the Fame Award in the recognition of his smart innovations, in 2018. He is also a reviewer of many well reputed journals. He has served as a technical program committee member in many international IEEE conferences.

LEONARDO MOSTARDA was a Research Associate with the Computing Department, Imperial College London, in 2007. In 2010, he was a Senior Lecturer with the Networking Department, Middlesex University, U.K. He is currently an Associate Professor with the Department of Computer Science, Camerino University, Italy. His research interests include the IoT, middleware, and security.

MUAZ A. NIAZI (Senior Member, IEEE) received the bachelor's degree in electrical engineering, and the M.S. degree in computer sciences from Boston University, MA, USA, and the Ph.D. degree in computer sciences from the University of Stirling, Scotland, U.K. He held a postdoctoral position with the COSIPRA Laboratory, University of Stirling. He is currently the Chief Scientific Officer (Professor) with COMSATS Islamabad. He also serves as the Founding Head of the Complex Systems Modeling, Simulation, and Engineering (COSMOSE) Research Group, COMSATS. He also served as the Director Research/ORIC with Bahria University, where he played an active role in laying the foundations of research practice in all university campuses. He regularly organizes and participates in various capacities in conferences, workshops, special sessions, and journal special issues around the globe. He has published in many prestigious journals and conference proceedings besides several books. His research interests include modeling, simulation, and engineering of complex adaptive systems (CAS) using various techniques, such as agent-based and complex-network-based approaches other than distributed pervasive/mobile application development. He is also an Active Member of the IEEE Consumer Electronics Society, the IEEE Computational Intelligence Society, and the IEEE Robotics and Automation Society. He has been listed in the Who's Who around the World and Who's Who in Science and Engineering. He is also the Founding Editor-in-Chief of *Complex Adaptive Systems Modeling* (SpringerOpen/BioMed Central) an Open Access journal and *International Journal of Privacy and Health Information Management* (IGI Global). He also serves as an Associate Editor for *Transactions on Emerging Telecommunication Technologies* (Wiley).

AMMAR RAFIQ received the master's degree in information technology from the National University of Science and Technology (NUST), Islamabad, Pakistan, in 2007, and the Ph.D. degree in computer science from the University of Engineering and Technology Lahore, Pakistan, in 2021. He is currently working as an Assistant Professor with the Department of Computer Science, NFC Institute of Engineering and Fertilizer Research, Faisalabad, Pakistan. His research interests include optical communication and networks, wireless sensor networks, and wireless communication.

SAQIB SAEED received the B.Sc. degree (Hons.) in computer science from International Islamic University Islamabad, Pakistan, in 2001, the M.Sc. degree in software technology from the Stuttgart Technology University of Applied Sciences, Germany, in 2003, and the Ph.D. degree in information systems from the University of Siegen, Germany, in 2012. He is currently a Certified Software Quality Engineer from the American Society of Quality. He is also an Associate Professor with the Department of Computer Information Systems, Imam Abdulrahman Bin Faisal University, Dammam, Saudi Arabia. His research interests include human-centered computing, data visualization and analytics, software engineering, information systems management, and digital business transformation. He is also an Associate Editor of IEEE Access and *International Journal of Public Administration in the Digital Age*, besides being member of the advisory boards of several international journals.