A detailed comparison study about Object Detecting and Lane Detecting Algorithms

Rakshit Jha
Department of Computer Engineering
Pimpri Chinchwad College of Engineering
Pune, India
rakshit.jha18@pccoepune.org

Shruti Sonune
Department of Computer Engineering
Pimpri Chinchwad College of Engineering
Pune, India
shruti.sonune18@pccoepune.org

Mohammad Taha Shahid
Department of Computer Engineering
Pimpri Chinchwad College of Engineering
Pune, India
mohammad.shahid18@pccoepune.org

Santwana Gudhade
Department of Computer Engineering
Pimpri Chinchwad College of Engineering
Pune, India
santwana.gudhade@pccoepune.org

***Abstract*—A self-driving car (driverless, autonomous, robotic car) is a vehicle that can sense its surroundings and navigate without the need for human intervention .Self-driving cars can detect environments using a variety of techniques like radar, lidar, GPS and computer vision. Lane detection is finding the white/yellow color markings on a road to ensure that the vehicles are within lane constraints.It is one of the key features of self driving car. This work provides a survey on lane detection approaches,based on Performance analysis of existing lane detection like CNN based, Haugh Transform, Gaussian filter and canny edge detection. and author's proposed approaches on different datasets such as curved roads,big datasets, rainy days, yellow -white strips ,day and night lights. We also present a detailed direct comparison of You Only Look Once [YOLO] Algorithm with Object detection using color masking and provide an insight on YOLO algorithms' predecessors. YOLO is a simple and straightforward algorithm that has a plethora of categories to detect objects live in real-time using a camera, by an input video provided to it and, also in an image given to it as an input. YOLO v3 is a very fast algorithm and was an incremental leap in the domain of object detection. The most noticeable feature in YOLO v3 being its ability to make detections at 3 completely different scales. YOLO v3 is one of the more prominent classifiers, which is incrementally faster.**

***Index Terms*—Trappings effects, thermal effects, low frequency S-parameters, CAD non-linear model, RF pulsed operation.**

I. Introduction

One of the fundamental challenges that a computer face that humans have never faced is object comprehension. When we look at the world, we are able to identify objects such as books and phones etc., . Computers, on the other hand have hard time doing the same thing. This because, fundamentally, computers only recognize images and videos as a collection of pixels rather than objects as a whole. This problem is crucial as it is one of the first obstacles to overcome to make self-driving cars a reality. The first thing it has to do is identify lane lines on the road so that it knows where to drive. Identifying lanes on the road is a typical job performed by all human drivers in order to keep their automobiles within lane constraints when driving. To Minimize chances of collisions with other cars and to make sure traffic is smooth lane detection is needed to keep the vehicle in its lane constraints as it is a critical task for an autonomous vehicle to perform. YOLO is a unified pipeline of Convolutional Neural Networks developed in 2015 by Joseph Redmon and Ali Farhadi from the University of Washington. Prior to YOLO v1 object detection CNN's such as R-CNN's used Region Proposed Networks (RPNs) to predict and outline a bounding box on the input image. R-CNNs were very difficult to optimize and very slow. YOLO is a mono-stage CNN that can be trained end-to-end, easy to optimize and works in real-time.

II. Algorithmic Survey

A. Object Detection using Color Masking

1) Color Masking: Color masking is a simple technique which provides the programmer a fine control of updating pixel values on screen [14]. By restricting the color channels, each channel can be used to store a completely different image. Color masking can also be used for object detection by setting the algorithm to detect a particular color channel and use it to localize the object in the input grid.

2) Modules/ Libraries used: The library used in the program is cv2. Open-CV (cv2) of python bindings is designed to solve problems based solely on computer vision. To install and import Open-CV into the program we use: Installation:
sudo apt-get install python3-opencv #Ubuntu
pip install opencv-python #Windows
Building from source: https://docs.opencv.org/master/d5/de5/tutorial_py_setup_in_windows.html
Importing: import cv2

3) Algorithm Architecture:

- We mainly will be detecting Blue and Red colors [14],
- Define lower and upper bounds of the Mask that will be detected later
- Prepare text for the Label that will be displayed when object is detected
- Define a loop, set its value to True
- Capture frame using open cv library

- Convert frames to HSV format
- Implement the mask with found colours to HSV Image
- Find and Assign Contours by selecting the appropriate version of Open-CV
- Find the largest Contour
- Draw a bounding box on the current BGR Frame
- Put the text with label on the Frame
- Show BGR Frame with detected Object
- Name the opened window
- Break the loop if 'q' is pressed
- Close all windows

4) Advantages and Disadvantages: If we are provided with an object that is homogeneous in color, and provided that the conditions are just right, the object can be localized and detected with a good accuracy. The algorithm is very fast but comes with a few restrictions. A disadvantage of this algorithm is that it struggles under various light conditions. If the background is the same color as the object, there is a high chance that the background will be detected as an object. The algorithm also works best if the object ad the background are stationary i.e. there is no relative motion of the camera with respect to the object.

5) Verdict: The algorithm of Object Detection with Color Masking is a good algorithm but is bound by a lot of factors in order for it to detect objects. It is very snappy but there is a greater chance of a miss than that of a hit due to its inconsistency in differentiating the object and the background.

B. YOLO v3 Object Detection

YOLO v3 was a drastic improvement in the field of object detection. Taking inputs at 320x320 it runs at 22ms at 28.2 mAP, which is 3 times faster than SSD but equally accurate. [11]

1) YOLO v3: YOLO v3 has an interesting way of predicting bounding boxes around a detected object. It uses dimension clusters as anchor boxes. It predicts 4 corner points for each box tx,ty,tw,th. While training the model, the error function used is the sum of squared error loss and the objectness score is predicted using logistic regression. For multiclass predictions, independent logistic classifiers are used instead of a softmax layer since softmax was unnecessary, and for class predictions a binary cross-entropy loss was used. [11] [9]

Figure 3.adapted from the, this time displaying speed/accuracy tradeoff on the mAP at .5 IOU metric. You can tell YOLOv3 is good because it has a very high value and is placed to the far left. (1.2)Modules used The Libraries used in YOLO v3 are numpy, open-cv and, time. To install and import these libraries we can include the following code: Installing libraries

```
pip install numpy #Windows
pip install python-opencv #Windows
sudo apt install numpy #Ubuntu
sudo apt install python-opencv #Ubuntu
```

Importing the required libraries

```
import numpy as np
```

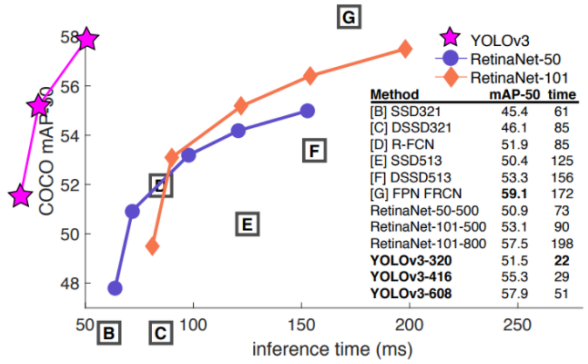

Fig. 1. A graph representing inference times of YOLO v3 as compared to other algorithms

```
import cv2
import time
```

2) Algorithm Architecture:

- Read the video stream from camera and prepare variables for spatial dimension of every frame
- Load the COCO labels
- Load the trained YOLO object Detection file
- Get names as list of all layers from the network
- Get the output layers from YOLO v3
- Set minimum probability and threshold for bounding boxes
- Generate different colors for every different object detected
- Define a loop to accept frames from camera
- Implement a forward pass using blob
- Prepare a list for bounding boxes that are detected
- If confidence value is larger than minimum probability only then object is detected
- Prepare labels and confidence for bounding boxes
- Show results in real-time
- Close all windows

3) Advantages and Disadvantages: A few downsides to the algorithm can be represented by a MAP score between 0.5 to 0.95 IOU, which can be increased by the algorithms' successors.The average precision for medium and large objects can be improved as medium is 5 percent behind and large is 10 percent behind the best of algorithms. A few upsides to the algorithm is that there is a newer architecture that boasts of residual skip connections, and upsampling, and makes predictions at three scales, which are precisely given by downsampling the dimensions of the input image by 32, 16 and 8 respectively. It is very capable of detecting smaller objects when compared to its predecessors or other algorithms. YOLO v3 predicts more bounding boxes than its predecessors for an input image of an equivalent size. Overall, it is a fantastic algorithm to use.

4) Verdict: It is a very fast algorithm and has an on par accuracy with the best in class 2-stage detectors which makes it a very powerful object detection algorithm. YOLO v3 can be

applied in numerous domains some of which contain sensitive environments which require high accuracy and a low latency model i.e. autonomous driving, security, etc. scenarios. It can also be used in product monitoring where a little dip in accuracy can be tolerated for a higher speed, but since this algorithm is very fast and very accurate it makes up a fantastic algorithm for most use cases. YOLO v3 is one of the best object detection algorithms that has been developed so far.

III. YOLO v/s other Algorithms

Prior to YOLO, several algorithms held the mantle for very good object detection algorithms. Some of them are listed below.

A. OverFeat

OverFeat: Integrated Recognition, Localization and Detection [1] using CNNs was also counted amongst the top algorithms. It classified, localized and detected objects. OverFeat proposed a novel DLL approach which approached localization by predicting the boundaries of the object. OverFeat won the localization task of the ImageNet Large Scale Visual Recognition Challenge 2013 (ILSVRC2013). OverFeat uses a single ConvNet to perform object detection.

B. Region Convolutional Neural Networks

R-CNNs improved the mean Average Precision (mAP) by more than 30% when it was initially released. It combined two methods:

- Applying high quality CNNs to bottom-up region proposals for localization and segmentation
- When low training data is present, supervised pre-training following domain-specific fone tuning yields significantly better results.

C. Very Deep Convolutional Networks for Large scale Image Recognition

Very deep CNs have one primary focus. They have high depth architecture with very small 3x3 convolution filters. Deep layers had a significant improvement on prior art configurations by pushing the depths to 16-19 weight layers[2]. This method has a high depth configuration and generalizes well to other datasets, achieving state-of-the-art results.

D. Deep Residual Learning for Image Recognition

Conventional DNNs are tough to train and hence ResNets [30] are designed in a residual framework that are significantly deeper than the ones used earlier. Instead of learning unreferenced functions, the layer inputs learn with rewritten residual functions. Due to a substantial increase in the depth of the model, it performs with a higher accuracy at greater depths and is easier to optimize.

E. Deep Neural Networks for Object Detection

DNNs have an outstanding performance on image classification tasks [31]. Using DNNs for object detection enhances the ability of DNNs since the objects in the image can not only be classified but also be localized within a frame by using bounding box masks. DNNs do not require a hand designed model but instead builds itself up by finding patterns in the data provided to it. This level of simplicity and flexibility has easier application to a wide variety of various class categories and it also outputs a better detection performance across a wider range of objects-rigid and deformable ones.

IV. YOLO and its Version History

A. YOLO v1

The first version of YOLO accepted the input image in a NxN grid [13], where N can be any number but YOLO v1 preferred N=7. If the object were to be present in the preferred grid, the algorithm was programmed to detect it. The grid is assigned based on the center of the object axis. This was an infant stage in object detection and could detect at maximum NxN number of objects present in the grid. Due to this, if a single frame consisted of more than a single object, only one object would be detected at a time. Among the NxN grid, all of the grid cells were detected simultaneously which made YOLO a very fast algorithm in comparison. For each of the bounding boxes inside the NxN grid, the model outputs a confidence score about the prediction of the object. YOLO v1 was trained to detect 20 different classes of objects. The single ConvNetwork is inspired by GoogleNet model for image classification.

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

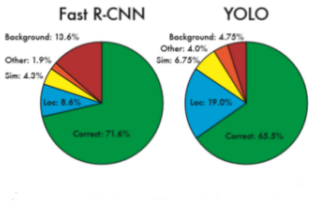

Fig. 2. Real-Time Systems on PASCAL VOC 2007 .We can notice that YOLO struggles to localize objects correctly.

B. FAST YOLO

The name is self defining and lives upto its name. The model uses a 9 layer Convolutional Network instead of a 24 layer. This reduction in the count of layer speeds up the model significantly but also creates an alternative side-effect of a lower mAP. YOLO VGG-16 uses VGG-16 as its main component instead of the original YOLO network. It does provide a higher accuracy but take a toll by being slower than real-time

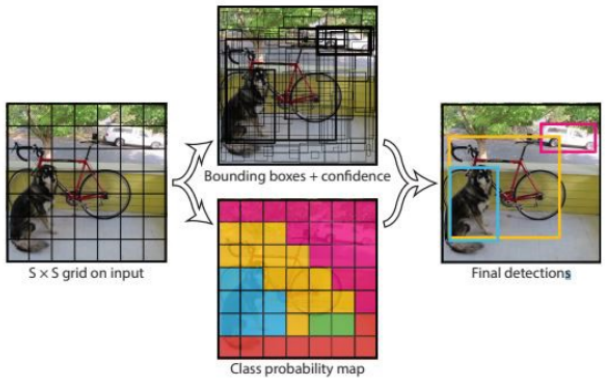

Fig. 3. YOLO version 1 conceptual design (Source:You Only Look Once: Unified, Real-Time Object Detection by Joseph Redmon et al.)

C. YOLO v2

YOLO v1 consisted a major drawback of localization errors and having a low recall. Hence the second version was built on the basis of improving the localization and recall metrics [12]. To achieve a higher performing model a few new implementations were experimented with. The methods being (1)Batch-Normalization- This had a more than 2% improvement in mAP (2)High Resolution Classifier. The resolution was increased to 448x448 from the initial value 224x224 for detection. (3)Convolutional with Anchor Boxes-This idea added a major functionality update to the model, it could now detect multiple objects in the same grid. YOLO v2 was comparatively faster than a variety of detection systems.

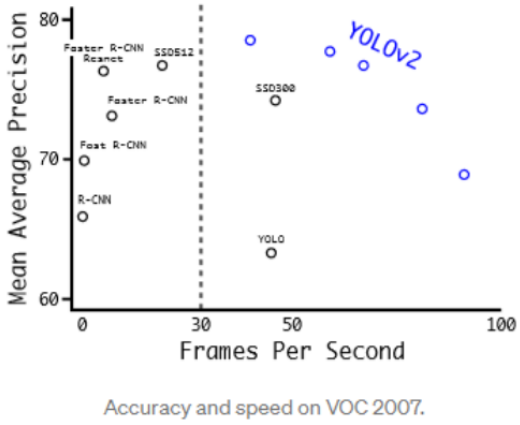

Fig. 4. PASCAL VOC2012 test detection results YOLOv2 performs on par with state-of-the-art detectors like Faster R-CNN with ResNet and SSD512 and is 2-10×faster

D. YOLO9000

This model was solely built for the purpose of identifying a higher number of classes than the original YOLO model. YOLO9000 could detect and classify 9000 different categories of classes[6].

E. YOLO v3

YOLO v3 was bigger in size than the previous models but provided accurate results as outpout. YOLO v3 was very similar to YOLO9000 in terms of predicting the boundary boxes with coordinates (tx, ty, tw, th) around the detected object [9] [11]. By using Logistic Regression each bounding box predicted a confidence score of what the detected object was. Since the Softmax layer could only assign one class per object, this was a potential drawback and was eliminated from this model. This model uses independent Logistic Classifiers for any class. YOLO v1 and v2 struggled detecting tiny objects in the frame and hence v3 used short cut connections to get better results. On a medium and larger size object, YOLO v3 performed worse than the previous models. YOLO v3 showed numerous and significant benefits over other detection systems.

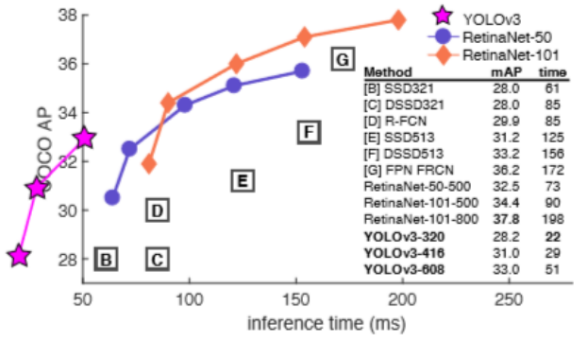

Fig. 5. YOLO version 3 conceptual design (Source:You Only Look Once: Unified, Real-Time Object Detection by Joseph Redmon et al.)

F. YOLO v4

YOLO v4 runs up to twice as fast as the EfficientDet with a comparable performance output. This version improved the YOLO v3's AP and Frame Rate by 10% and 12%. YOLO v4 used multiple data enhancement technologies such as geometric distortion, illumination distortion, etc. [8] [10], creatively used Image Occlusion Random Erase-Cutout-Hide and Seek-Grid Mask -MixUp technologies as well, and multiple image augmented models were also trained by using augmentation types such as MixUp, CutMix, Crop, Rotate, Mosaic, Blur, etc. The author also used Self-Adversarial Training for better outputs.

G. YOLO v5

YOLO v5 has not currently been published. The pipeline hence can only be understood from a code perspective [5]. YOLO v5 is potentially to be a huge performance upgrade over YOLO v4 due to a huge increase in the Mosaic data. Mosaic data has the potential to solve the most troublesome "small object problem" during model training. When comparing YOLO v4 to YOLO v5 we observe a significant improvement in object detection.

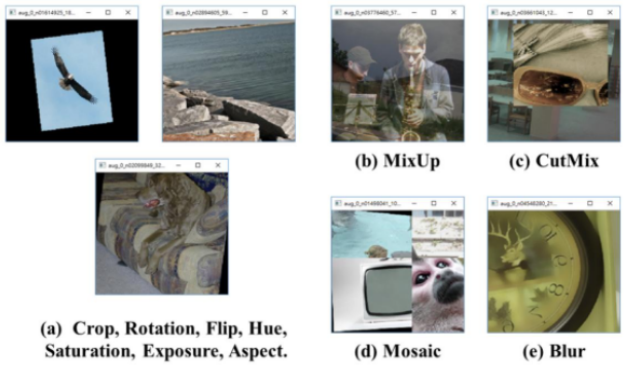

Fig. 6. Augmented Images for training

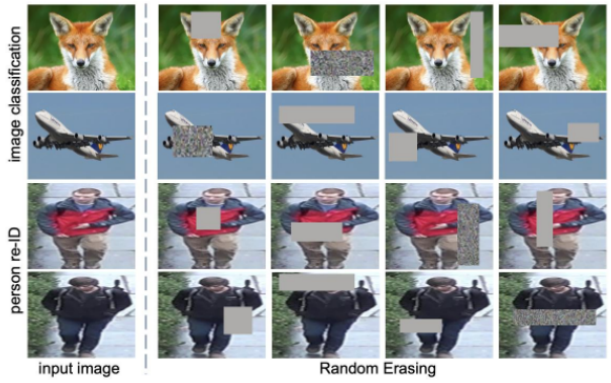

Fig. 7. Augmented Images for training

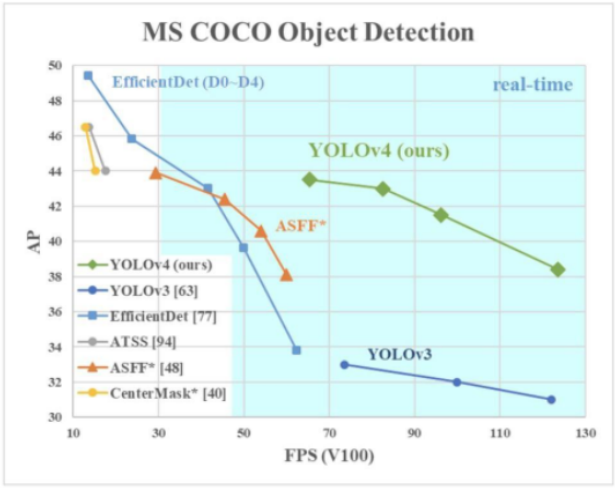

Fig. 8. MS COCO Object Detection Graph

H. PP-YOLO

PP-YOLO is the acronym for PaddlePaddle YOLO [9]. This model creates an active balance between the efficiency of the model(72.9FPS) and its effectiveness(45.2 mAP) ,

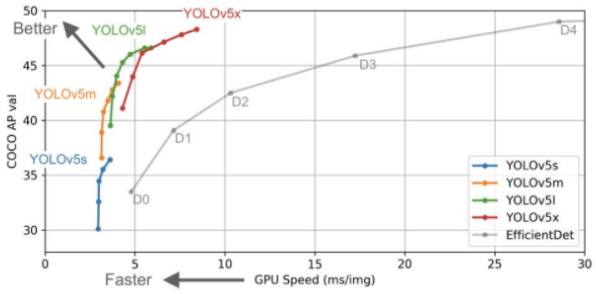

Fig. 9. YOLO v5 vs its other versions

passing the state-of-the-art YOLO v4 model and EfficientDet model. Source code is at https://github.com/PaddlePaddle/PaddleDetection. It works on the principle of a backbone network, a detection net (which is a Feature Pyramid Network) followed by a detection head for classification and localization of objects.

V. A Survey in Lane Detection Approaches

In primitive computer vision techniques, It is often challenging to detect the lane positions based to road conditions, shadows, distortion on day or night lights, heights of the cameras. Also there are different lane markers like solid, double, single, broken markers with number or possible colors. To overcome this problem Alexander et al., [15] presented a convolutional neural network approach from "Deep Lanes: End-To-End Lane Position Estimation Using Deep Neural Networks" which estimated the position of lane marker for input image with respect to the baseline of the image, where the image was taken from down facing camera on the vehicle. The author's approach is not pre-processing, post-processing and handcrafted features dependent. The system resulted the estimation of the lane position was less than five pixel error in real time on embeded automotive platform in 99 percent of the cases.

Assidiq et al., [21] proposed a vision-based lane detection approach capable of real time operation. The system acquires the front view using a camera mounted on the car, then applies a few algorithms to detect the lanes using a pair of hyperbolas attached to the lane's sides. The lanes are then extracted using the Hough transform. The proposed lane detection system by the author are applied on both painted and unpainted road also as curved and straight road in several weather which resulted the scheme to be fast enough for real time requirements but the purposed system could not detect sharp curves in the foreground of the image and the accurate detection of the lanes under heavy rain also the captured frames aren't that stabile because of the vehicle movement.

The standard methodology divides the work into many elements, such as lane detection, path planning, and control logic, which are generally explored independently. Errors might also accrue from one processing stage to the next, resulting in an erroneous final output. To overcome this problem Gurghian

et al., [19] proposed a system from " End-to-end learning for lane keeping of self-driving cars," TO take the end-to-end learning approach to the raw image as input and outputs the control signal automatically using convolutional neural networks (CNNs). The results of the tests suggest that the model can create quite accurate vehicle steering. The author used comma.ao dataset, Despite the fact that they appear to be quite realistic because to the cutting-edge game engine, the information and variances they present are still insufficient to match facts from the real world.The simulator and data augmentation were not included in the research because the dataset did not contain the original sized frames or camera calibration parameters.

The car should be able to distinguish between different items encountered on the road and identify components of interest such as other vehicles, pedestrians, traffic lights, and so on.To build such ability to navigate properly in between the road lanes while complying with the traffic rules. Eadhunath et al., [29] proposed a system from "Self-Driving Car using Convolutional Neural Network and Road Lane Detector" where the author used a yolo algorithm on a single neural network on the full image which divided the image into regions and predicted the bounding boxes and probabilities for each region. The author created functions to approximate the lane structures by feeding them into the regression approach to discover the best fit of the road lane detection by getting right and left points.The final product detected traffic and maneuvered its way on the roads by keeping track of the road lanes on its either sides.The author stated that the single network evaluation is 1000x faster than RCNN and 100x faster than Fast R-CNN. The fault tolerance of the car and its ability to adapt in various environmental conditions is low. Multiple cameras weren't used for different angles to make the car overtake or by-pass other vehicles.

"Robust and Real Time Detection of Curvy Lanes(Curves) with Desired slopes for Driving Assistance and Autonomous Vehicles" [26] by Amartansh Dubey et al., for detecting curves. They have proposed a new model to detect the lanes at diverse environmental conditions by using Gaussian filter and Hough transform. "A machine learning approach for detecting and tracking road boundary lanes" [27] by Mammeri et al.,"Robust lane detection from continuous driving scenarios using deep neural networks" is a new method for detecting road lanes that combines MSER and Hough and achieves good results. "Robust Lane Detection from Continuous Driving Scenes Using Deep Neral Networks" [25] by Zou et al., to detect the road boundary lanes, the author proposed a cnn and rnn-based technique to detect the road boundary lanes. "Vision and odometry based autonomous vehicle lane changing" [28] Peter et al., presented an automated lane changing approach for vehicles by employing environment detection with LIDAR.

Yang Xing et al., [20] provided the literature reviews on the lane detection algorithms, the integration methods, and evaluation methods. And a summary of earlier lane detection systems from "advances in vision based lane detection algorithms integration evaluations and perspectives on acp based parallel vision" were presented. As a notable amount of existing studies don't provide enough information on the mixing methodologies of lane detection, sensors other systems. The author analyzed the mixing methodologies and categorized the ways of integration into senor level, system level and algorithm level. The author introduced a completely unique lane detection system design framework which supported the ACP (Artificial society, Computational experiments, and Parallel execution) parallel theory for more efficient training and evaluation of lane detection models. Artificial society, Computational experiments, and Parallel execution are the three major components of the parallel systems. The ACP-based lane detection parallel system aims to create virtual parallel scenarios to train the models and benefit the corresponding real-world system.

Table 1
Performance analysis of existing lane detection approaches.

Methodology	Accuracy
Guassianfilter+Hough transform [4]	89%
MSER+Hough [5]	92%
HSV-ROI+Hough [6]	95%
CNN Based [7]	96%
Improved LD+Hough [10]	96%
LaneNet [11]	97%

Fig. 10. Table shows the comparison table of related work of analysis of existing lane detection approaches. From paper "A machine learning approach for detecting and tracking road boundary lanes" by Satish Kumar Satti et al., [24]

The suggested methods use a convolutional neural network to extract edge features followed by a normalisation step to achieve the best results. The authors used the most widely used methods for detecting road lanes, including Hough Transform,Canny edge detection using Hough, HSV-ROI and Convolutional Neural Network (CNN, or ConvNet) based.

A. Lidar Vs other sensors

Vehicles must be able to detect their environment with the help of sensors like camera,lidar and radar before being able to drive autonomously.Lidar sensors are used by self-driving car

Table 2 Summary of data collection.			
Image type images	Total	Training (80%)	Testing (20%)
Day time images	1400	1120	280
Low light images	1200	960	240
Night time images	1400	1120	280

Fig. 11. Table shows the comparison table of different image type data collection. From paper "A machine learning approach for detecting and tracking road boundary lanes" by Satish Kumar Satti et al., [24]

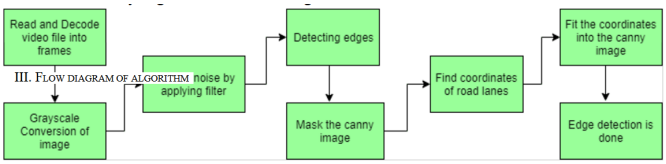

Fig. 12. Flow Diagram of Algorithm for Lane Detection

businesses such as Alphabet's Waymo and General Motors' Cruise. Apple has chosen to outsource portions of its self-driving technology, such as lidar sensors, to third parties. Tesla has filed to use a replacement kind of "millimeter-wave radar sensor" for forward-facing radar which is hidden at the front of the car. LiDAR is essentially blind in bad weather. For this reason, autonomous cars using LiDAR still need to use radar to drive in such inclemency conditions, further adding to autonomous vehicle costs. As cameras can already infer distance and see quite well in good weather.

VI. Conclusion

In this literature survey we gave a summary for lane detection using different approaches Using different algorithms and datasets. This survey will be very useful for those who are building a lane detection algorithm for self-driving cars. In the survey we came to conclude that Most commonly used methodologies to implement the lane detection systems are CNN based, Haugh Transform, Gaussian filter and canny edge detection. and a very less amount of research has been done on unmarked roads. The following are some of the most commonly utilised lane detecting steps:

- Noise reduction improves the quality of video or photos produced by a camera.
- Edge detection algorithms help to detect edges to reduce the complexity in the lane detection
- Hough transform converts the image into Hough space to detect straight lines, circle and ellipses. Hough transform works with simple background but it fails to detect lanes in different lighting conditions.

There are lot of methodologies for lane detection. But, basic steps involved altogether the methodologies are similar. This paper will be helpful for increasing the accuracy of existing or upcoming models related to lane detection. YOLO makes up for an excellent algorithm for object detection as it is highly optimized to detect objects under any and all circumstances whether it be stationary or moving. The agility of the YOLO algorithm makes it suitable for all scenarios. In the above survey we compared YOLO with all state-of-the-art algorithms and YOLO came out to be the best and the most optimized algorithm. YOLO was also compared to its predecessors and each and every new version of YOLO had an incremental improvement turning into the state-of-the-art algorithm it is today.

References

1 Pierre, Sermanet; David, Eigen; Xiang, Zhang; Michael, Mathieu; Rob, Fergus; Yann, LeCun.; (2014)."OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks". arXiv preprint arXiv:1312.6229v4.

2 Karen, Simonyan; Andrew, Zisserman.; (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". arXiv preprint arXiv:1409.1556v6.

3 Ross, Girshick; Jeff, Donahue; Trevor, Darrell; Jitendra, Malik.; (2014)."Rich feature hierarchies for accurate object detection and semantic segmentation". arXiv preprint arXiv:1311.2524v5.

4 Christian, Szegedy; Wei, Liu; Yangqing, Jia; Pierre, Sermanet; Scott, Reed; Dragomir, Anguelov; Dumitru, Erhan; Vincent, vanhoucke; Andrew, Rabinovich.; (2014)."Going Deeper with Convolutions". arXiv preprint arXiv:1409.4842v1.

5 Joseph, Redmon; Santosh, Divvala; Ross, Girshick; Ali, Farhadi.; (2016)."You Only Look Once:Unified, Real-Time Object Detection". arXiv preprint arXiv:1506.02640v5.

6 Joseph, Redmon; Ali, Farhadi.; (2016)."YOLO9000: Better, Faster, Stronger". arXiv preprint arXiv:1612.08242v1.

7 Joseph, Redmon; Ali, Farhadi.; (2018)."YOLOv3: An Incremental Improvement". arXiv preprint arXiv:1804.02767v1.

8 Alexey, Bochkovskiy; Chien-Yao, Wang, Hong-Yuan, Liao.; (2020)."YOLOv4: Optimal Speed and Accuracy of Object Detection". arXiv preprint arXiv:2004.10934v1.

9 Xiang, Long; Kaipeng, Deng; Guanzhong, Wang; Yang, Zhang; Qingqing, Dang; Yuan, Gao; Hui, Shen; Jianguo, Ren; Shumin, Han; Errui, Ding, Shilei, wen.; (2014)."PP-YOLO: An Effective and Efficient Implementation of Object Detector". arXiv preprint arXiv:2007.12099v3.

10 Joseph, Redmon; Santosh, Divvala; Ross, Girshick; Ali, Farhadi.; (2015)."You Only Look Once:Unified, Real-Time Object Detection". arXiv preprint arXiv:1506.02640v4.

11 Joseph, Redmon; Santosh, Divvala; Ross, Girshick; Ali, Farhadi.; (2015)."You Only Look Once:Unified, Real-Time Object Detection". arXiv preprint arXiv:1506.02640v3.

12 Joseph, Redmon; Santosh, Divvala; Ross, Girshick; Ali, Farhadi.; (2015)."You Only Look Once:Unified, Real-Time Object Detection". arXiv preprint arXiv:1506.02640v2.

13 Joseph, Redmon; Santosh, Divvala; Ross, Girshick; Ali, Farhadi.; (2015)."You Only Look Once:Unified, Real-Time Object Detection". arXiv preprint arXiv:1506.02640v1.

14 Khawar, Jamil.; Maneuvering Color Mask into Object Detection Aug. 2020. Accessed on: Aug. 28, 2020. [Online]. Available: https://medium.com/globant/maneuvering-color-mask-into-object-detection-fce61bf891d1

15 Christian, Szegedy; Alexander, Toshev; Dumitru, Erhan.; (2013)."Deep Neural Networks for Object Detection". NIPS'13.

16 "Install OpenCV-Python in Windows," May. 27, 2021. Accessed on: May. 27, 2021. [Online]. Available: $https://docs.opencv.org/master/d5/de5/tutorial_py_setup_in_windows.html$

17 Tsung-Yi, Lin; Priya, Goyal; Ross, Girshick; Kaiming, He, Piotr, Dollár.; (2018)."Focal Loss for Dense Object Detection". arXiv preprint arXiv:1708.02002v2.

18 Joseph, Redmon; Ali, Farhadi.; (2018).”YOLOv3: An Incremental Improvement”. arXiv preprint arXiv:1804.02767v1.
19 Gurghian; T. Koduri; S. V. Bailur; K. J. Carey; V. N. Murali.; (2016)”DeepLanes: End-To-End Lane Position Estimation Using Deep Neural Networks,” 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, USA, 2016, pp. 38-45, doi: 10.1109/CVPRW.2016.12.
20 Yang, Xing; Chen, Lv; Long, Chen; Huaji, Wang; Hong, Wang; Dongpu, Cao; Efstathios, Velenis;Fei-Yue, Wang.; (2018)”Advances in Vision-Based Lane Detection: Algorithms, Integration, Assessment, and Perspectives on ACP-Based Parallel Vision,” IEEE/CAA J. Autom. Sinica, vol. 5, no. 3, pp. 645-661, Mar. 2018. doi: 10.1109/JAS.2018.75
21 A. A. Assidiq; O. O. Khalifa; M. R. Islam;S. Khan,; (2008) ”Real time lane detection for autonomous vehicles,” 2008 International Conference on Computer and Communication Engineering, Kuala Lumpur, 2008, pp. 82-88, doi: 10.1109/ICCCE.2008.4580573.
22 Z. Sun.; (2020) ”Vision Based Lane Detection for Self-Driving Car,” 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA), Dalian, China, 2020, pp. 635-638, doi: 10.1109/AEECA49918.2020.9213624.
23 Z. Chen; X. Huang.; (2017) ”End-to-end learning for lane keeping of self-driving cars,” 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, 2017, pp. 1856-1860, doi: 10.1109/IVS.2017.7995975.
24 Satish, Satti; K. Suganya, Devi; Prasenjit, Dhar; P. Srinivasan,; (2021)”A machine learning approach for detecting and tracking road boundary lanes,” ICT Express,Volume 7, Issue 1,2021,Pages 99-103,ISSN 2405-9595.
25 Qin, Zou, et al., (2019)”Robust lane detection from continuous driving scenes using deep neural networks,” IEEE Trans. Veh. Technol. (2019).
26 Amartansh, Dubey; K.M. Bhurchandi.; (2015)”Robust and real time detection of curvy lanes (curves) with desired slopes for driving assistance and autonomous vehicles,” 2015, arXiv preprint arXiv:1501.03124.
27 Abdelhamid, Mammeri; Azzedine, Boukerche; Guangqian, Lu.; (2014)”Lane detection and tracking system based on the MSER algorithm, hough transform and kalman filter,” in: Proceedings of the 17th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2014. 6.
28 Gábor,Péter; Bálint, Kiss; Viktor, Tihanyi.; (2019)”Vision and odometry based autonomous vehicle lane changing,” ICT Express 5 (4) (2019) 219–226.
29 Eadhunath, V; Amir, Suhail; Jayant, Waghmare; Rishab, Mishra;Prof. K. U. Jadhav., (2019)”Self-Driving Car using Convolutional Neural Network and Road Lane Detector” International Journal of Engineering Research & Technology (IJERT) Vol. 8 Issue 05, May-2019.
30 Kaiming, He; Xiangyu, Zhang; Shaoqing, Ren; Jian, Sun.; (2015).”Deep Residual Learning for Image Recognition.” arXiv:1512.03385
31 Christian, Szegedy; Alexander, Toshev; Dumitru, Erhan; Deep Neural Networks for Object Detection. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/41457.pdf