A New Approach To Estimate Depth Of Cars Using A Monocular Image

1st Seyed Mohamad Ali Tousi, 2nd Javad Khorramdel 3rd Faraz Lotfi 4th Amir Hossein Nikoofard 5th Ali Najafi Ardekani and 6th Hamid D. Taghirad

1st Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran, s.m.ali.tousi@email.kntu.ac.ir
2nd Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran, rteoi96@email.kntu.ac.ir
3rd Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran, F.Lotfi@email.kntu.ac.ir
4th Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran, a.nikoofard@kntu.ac.ir
5th Department of Mechanical Engineering, K. N. Toosi University of Technology, Tehran, Iran, najafi@kntu.ac.ir
6th Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran, Taghirad@kntu.ac.ir

***Abstract*—Predicting scene depth from RGB images is a challenging task. Since the cameras are the most available, least restrictive and cheapest source of information for autonomous vehicles; in this work, a monocular image has been used as the only source of data to estimate the depth of the car within the frontal view. In addition to the detection of cars in the frontal image; a convolutional neural network (CNN) has been trained to detect and localize the lights corresponding to each car. This approach is less sensitive to errors due to the disposition of bounding boxes. An enhancement on the COCO dataset has also been provided by adding the car lights labels. Simulation results show that the proposed approach outperforms those who only use the height and width of bounding boxes to estimate the depth.**

***Keywords*—depth, monocular, autonomous vehicle, convolutional neural networks, CNN, object detection**

I. Introduction

Depth estimation and perception have widespread applications in many fields, especially in autonomous vehicles. Radars, LIDARs and cameras are sensors that are commonly used in order to estimate depth. Radars and LIDARs are not used in many applications because of their high cost. Moreover using LIDARs stipulates hardware with high processing power. Radars will fail in crowded environments and the presence of buildings. The above-mentioned restrictions made cameras to be the focus of research in the past decade. Inspired by depth perception in human vision, stereo cameras have been employed for the task of depth estimation. However, the computational costs and availability of equipment are substantial restrictions in those approaches. Estimating the depth from a single view also has been tried, and many techniques have been developed for that throughout the years. One of the most successful techniques is Structure-From-Motion (SFM) [1]; it uses camera motions (i.e., a sequence of frames) to estimate camera poses through different temporal intervals and then estimate the depth from pairs of progressive views. However, these methods are still falling short in practice because object movements are not handled and in highly dynamic scenes, they tend to fail as they cannot explain object motion [2]. Under restrictions of such environmental assumptions and the absence of suitable equipment, using a monocular image to estimate depth became an ill-posed problem. The undeniable importance of monocular image depth estimation and its widespread applications in many computer vision-based activities made researchers do lots of experiments and researches to develop a fundamental approach to estimate depth from a single RGB image. Odometry based approaches were one of the first solutions [3]. They use cameras intrinsic matrix and geometrical transformations. However, these methods require object dimensions in real-world, which is not desirable to estimate such constraints by hand in practice [2]. Some other approaches assumed super-pixels as a planar and perceived depth through-plane coefficients via Markov Random Fields (MRFs) [4]. Methods based on super-pixels are also considered in [5] and [6]. However, the pre-processing for this method is involved with image segmentation task that is not computationally much efficient. Recently, Convolutional Neural Networks (CNNs) have been used to estimate depth accurately, and obviously, they have been employed to estimate depth from a monocular RGB image. In some approaches, CNNs detect objects and draw bounding boxes around them, then the contact point of detection bounding box with the ground leads to depth. In these cases, one of the essential assumptions is that the objects are in the same plane with the observer. These planarity assumptions are violated in some situations like the presence of speed bumps, road slope, and also the error due to inaccurate position of bounding boxes. It will result in a significant error in estimation. In some other cases, CNNs have to learn an implicit relation between color pixels and depth([7]–[9]). However, these methods encompass a higher complexity than other approaches without CNNs because of their higher degree of freedom and more parameters in a deep CNN.

In this work, an approach has been proposed to estimate depth from a monocular RGB image using cars and their lights in the image to provide suitable data for an autonomous vehicle. Many different approaches have been tried to detect an object through an image. One of the most successful methods is "You Only Look Once" (YOLO) [10]. The YOLO architecture has been used in this approach to be trained az a model for detecting car lights. For that purpose, an

978-1-7281-7301-6/20/$31.00 ©2020 IEEE

enhancement on COCO data-set [11] has been provided to train the model to detect cars with their lights. Then the distance between two central points of light bounding boxes has been mapped into radial depth using a nonlinear function.

This paper is organized as follows. Section 2 describes the necessary backgrounds and some technical discussions about depth estimation, object detection, and virtual environments which are used for depth verification. Section 3 introduces the new model for estimating the depth of cars based on object detection using deep convolutional neural networks (CNNs). Section 4 combines the description of the training process with some discussion about the results. Finally, section 5 presents our conclusion.

II. Required Backgrounds and Materials

A. Dimensions and orientations of the car

It is widely known that distant objects in the image look smaller in dimension in comparison with when they are closer to the observer. Since the height of the bounding box is almost equal to the height of the car in the image, it is desirable to estimate depth from the height of the detection bounding boxes. However, this approach is not robust to variation in car heights. Furthermore, inaccurate bounding box estimation results in an enormous error in the depth prediction process. On the other hand, there are much fewer variations in car widths, and this tends to be a proper parameter for using in the depth prediction process. However, the problem with this method is that when the car isn't just in front of us and has an orientation with respect to the direct line, the width of the detection bounding box is more fabulous than the width of the car in the image. As F. Domini and C. Caudek said in [12], orientation components of the object in the image (azimuth, elevation, and roll) contain important information about the depth of the object. Minor orientation changes would not be distinguishable in vast distances, but it appears in closer objects so that it can be a metric of the depth. Orientation by itself cannot be perceivable from the detection box and requires pixel-level resolutions (i.e., segmentation mask) or more feature points. To attack these limitations, we propose using car lights as key feature points. The distance between rear lights (or front lights) is almost equal to the width of the car in the image. In order to reveal orientation, we can also use the center of the bounding box of the car and the center of rear lights (or front lights). Although in some cases car lights might get occluded by another car, it is the distance to the nearest car, which is more critical than partially occluded cars.

In addition to determining the orientation and width of the car, lights can also be used as feature points in SFM based models [13]. Besides, detecting lights can replace some approaches which employed highly computational deep CNNs to obtain feature points [14]. Some approaches [15], propose using a combination of trackers and object detection networks. Tracking lights, in addition to cars, can reduce the impacts of occlusions.

B. A new dataset for car lights

To best of our knowledge, there is no dataset which includes car lights for training CNN to detect them. So we decided to do a modification on Microsoft COCO [11], which is a dataset to advance the state-of-the-art in object recognition and gathers images of complex everyday scenes containing common objects in their natural context. The original dataset contains photos of 91 objects types and 328k images. In order to use this dataset for training our model to detect car lights, we selected images containing car, truck, and bus objects and added two more labels named rear-light and front-light to them. The enhanced dataset is containing 8530 images of labeled cars and lights for training and 4500 images for validation.

C. Object Detection

Object detection task has been the topic to numerous researches in recent years. Since the Haar-features which had been used in older object detection approaches had some restrictions like high sensitivity to orientations and lightning conditions, convolutional neural networks (CNNs) have been widely employed in recent researches. On the other hand, the high computational cost of the CNNs has reduced the speed of object detection process and in many cases prevents real time object detection. So it is very important to have a trade-off between the pace and accuracy of the process. Many CNN networks have somehow obtained this trade-off and present high speed accurate detectors such as R-CNN [16], Fast R-CNN [17], Faster R-CNN [18], Single Shot Detector (SSD) [19] and You Only Look Once (YOLO) [10]. Fig. 1 shows a comparison between these networks based on their precision and frames per second.

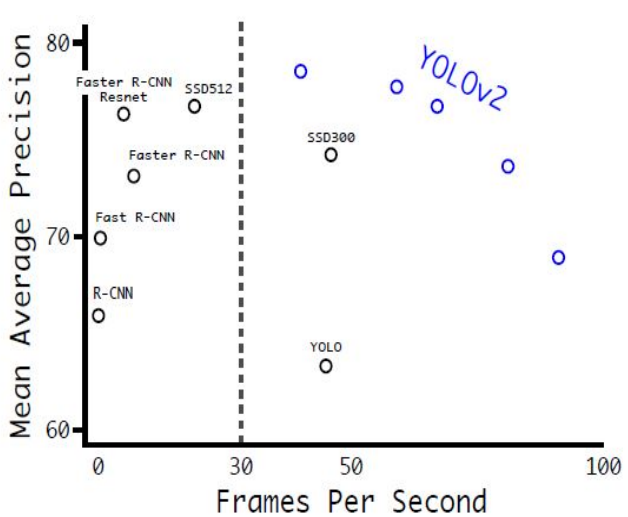


Fig. 1: Comparison between common object detection networks [20].

Regarding Fig. 1, YOLOv2 has considerable accuracy and speed simultaneously. Guangrui Liu in [21] also says that the YOLO networks have the best performance in the case of autonomous driving vehicles.

You Only Look Once (YOLO) was an extremely fast approach to object detection, localization and classification introduced by Joseph Redmon and his colleagues [10]. In this method, the image will be given to a single deep CNN as input and the network will predict the objects and their classes and locations within the image. The architecture of the YOLO network contains 24 convolutional layers followed by 2 fully-connected layers. In this work, we use YOLOv3 [22] architecture, shown in Fig. 2, which is a modified and much faster version of the YOLO network. YOLOv3 has 53 convolutional layers and one fully-connected layer and predicts bounding boxes corresponding to each object using dimension clusters as anchor boxes similar to the first version.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Fig. 2: The architecture of YOLOv3 [22].

D. CARLA simulator

Supervised learning requires verified targets. Since there are many restrictions with gathering real world verified depth data (with some equipment like LIDAR and Radar), simulators have been developed to fill this gap and nowadays their resolution and quality are comparable with real world data. We used the CARLA simulator [23] which provides a virtual environment with valuable data and sensors for autonomous vehicle researches. CARLA (Car Learning to Act) is an open-source simulator which has been introduced for autonomous driving research. This simulator is a server-client system; the client has an API which is implemeneted in Python and the server simulates and renders the scene. The environment which is simulated by CARLA is composed of static objects such as buildings, traffic signs and also dynamic objects like pedestrians and vehicles. Weather conditions, illumination, and numbers of cars and pedestrains can be controlled by the client. One of the sensors provided by the CARLA is RGB Camera which its type, number, and position can be specified by the API. Camera parameters include 3D location, 3D orientation with respect to the car's coordinate system, field of view and depth of field. Images taken by the camera can be recorded for future possible usages. Finally, the exact location of each dynamic object is also accessible. [23]

III. Our Purposed Model

In this work, in order to estimate the depth of cars in the image, a combination of a CNN based detection network with a Multi-layer perceptron (MLP) is proposed. Firstly, the frame captured by a camera will be given to the detection network as input. Then the detection network will output the cars and their light locations and dimensions within the image. In the next stage, using the information come from the first step these three parameters will be calculated and then used as inputs for the MLP network in order to estimate the depth of corresponding car (all in pixel domain):

- In1: Euclidean distance between car rear lights (or front lights)
- In2: Euclidean distance between the center of the car detection bounding box and the centroid of the line that connects two lights (this is implying the orientation of the car)
- In3: height of the car detection bounding box

Fig .3 shows a block diagram of the proposed model.

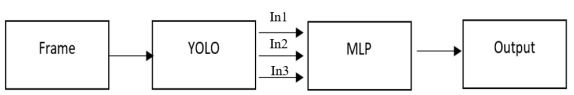


Fig. 3: Block diagram of the proposed model. YOLO detection box will output In1, In2 and In3 which are the distance between two lights (rear or front lights), the distance between the center of the car detection box and the center of a line which connects two lights and height of the car detection box, respectively.

The YOLOv3 network has been employed as our detection network. In addition to that, a MLP (multi-layer-perceptron) has been used as a nonlinear function to estimate the radial distance. This MLP has two hidden layers, each contains 5 neurons.In order to provide ground-truth for our network, exact radial distance to each car is extracted from CARLA [23]. The model parameters have been optimized by the Levenberg-Marquardt optimization algorithm with respect to the ground-truth came from CARLA. Fig. 4 shows the architecture of the MLP which is used for estimating the depth of the cars.

IV. Simulation Results

In the first stage, an object detector must be able to detect and localize car lights. The YOLO network by default cannot do this task, since it wan not trained on a dataset which includes the car lights labels.So the YOLOv3 [22] has been trained on our dataset to detect and localize cars and their lights. In order to increase the accuracy, some classes which are less probable to emerge in front of a vehicle camera like pencils, tie and etc. have been effaced from the dataset. Moreover, because of the variance in the scales of sought-after objects like cars and their lights, multiple anchor boxes

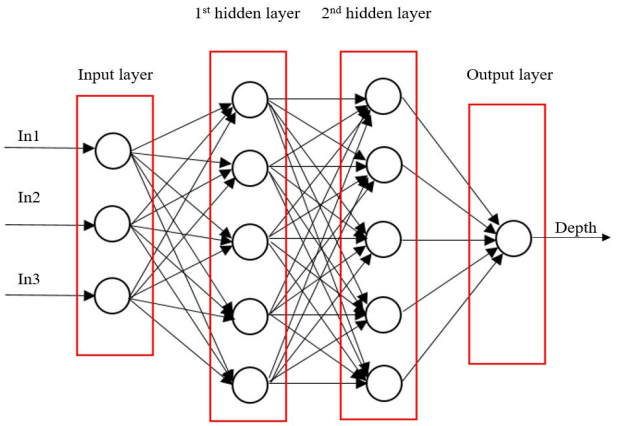


Fig. 4: The architecture of the MLP model.

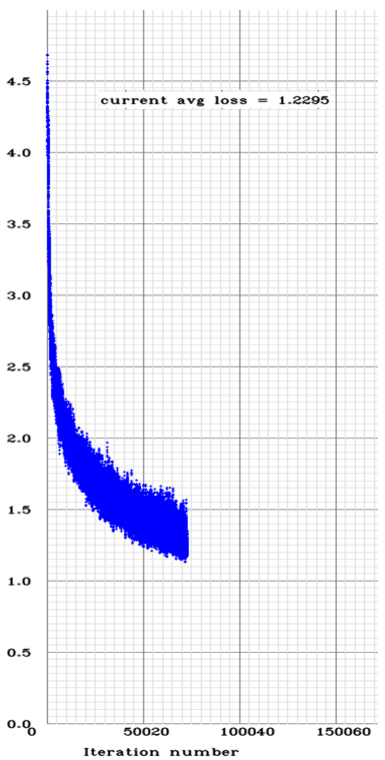
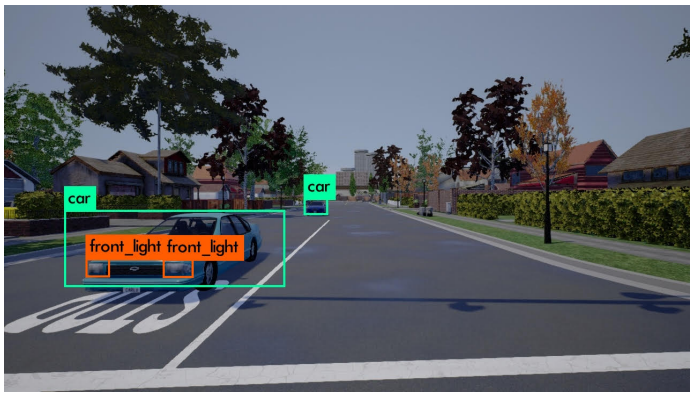
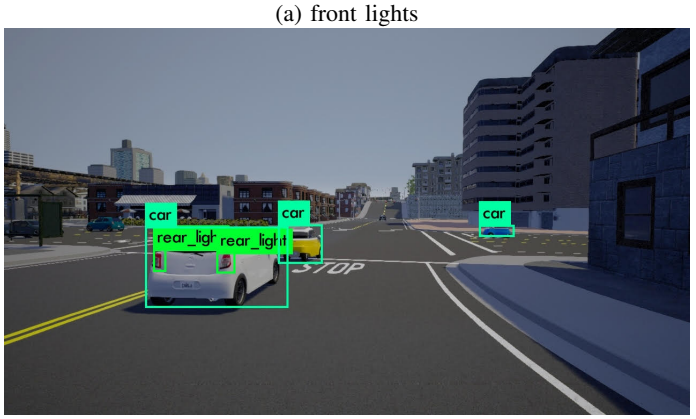


Fig. 5: Average loss of YOLO network during the training process.

with different sizes have been used in the process of training. The results of training YOLOv3 on our dataset are shown in Fig. 5, Fig. 6a and Fig. 6b. Fig .5 indicates the average loss of the YOLOv3 network during the training process. As the plot shows, after roughly 70,000 iterations the average loss is almost near 1.22. Since the frames include highly dynamic scenes, this loss for detecting small objects like car lights and greater ones like the car itself simultaneously seems reasonable. Regarding Fig .6a and Fig .6b, YOLOv3 has been employed in order to detect cars and their lights (rear or front) on some scenes captured from CARLA. In this work, it is the nearest car that has the major importance and as it can be seen, the YOLOv3 has detected and localized car lights corresponding to the nearest car successfully. The success of the network in detecting sought after objects in such synthetic images speaks of both the power of the YOLOv3 network and the quality of the CARLA simulator. Moreover, it indicates that the modified dataset has enough number and quality of data for reaching the goal of detecting car lights.



(a) front lights



(b) rear lights

Fig. 6: result of testing the network on CARLA images to detect cars and their lights.

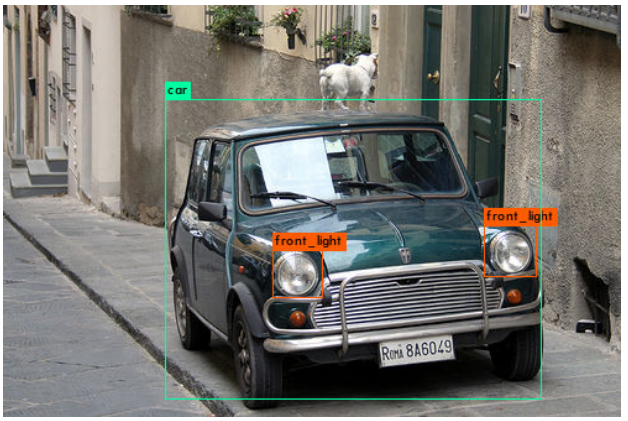
The detection network has also been tested on real world images. Fig. 7a and Fig. 7b show the results of car and their lights detection on real world images. Based on the results, the networks successfully detected the cars and their lights.

After the YOLOv3 training process is done, the detection network is able to provide necessary inputs. This network outputs objects bounding boxes by representing its coordinates in the image, width and height in pixel domain.

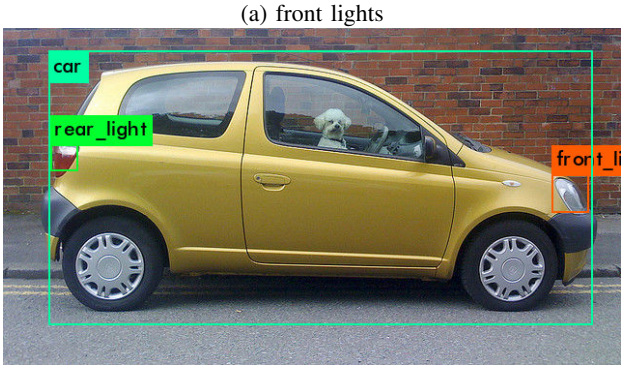
In order to obtain the required data for training the MLP network multiple scenes of driving different cars have been captured from CARLA [23] and the radial distance corresponding to each car has been collected as the ground-truth for the MLP training process. It is very important to feed the model with data which is rich in various car dimensions to avoid errors caused by variance in car heights.

In the next stage, after obtaining the data from running the YOLOv3 network on captured scenes, the MLP network has been trained.

In order to analyze the accuracy of the MLP network, the train and test error histograms have been plotted in Fig .9 and Fig .10. These plots show that both train and test errors



(a) front lights



(b) rear and front lights

Fig. 7: The result of testing the trained YOLOv3 network on a real world image [11].

have a zero-meaned normal distribution. This means the most frequent errors in the network were the least ones. In other words, the frequency of significant errors is negligible.

Fig . 8 shows the train and test MSE (mean-squared-error) of the model during the training process. As the plot shows, the test and train MSE converged to a number lower than 10^{-3} and this means the network performs well in estimating the depth both in frames that it learned before and the new frames.

Fig. 11 indicates depth prediction results. The test data contain 213 samples, each is the data extracted from a frame captured from CARLA. Fig. 11 indicates that with the selected inputs this is desirable to estimate radial distance corresponding to the nearest car accurately. The maximum error in the range of 23 meters is 1.6 meters for training data and 1.4 meters for test data. In the case of regular autonomous vehicles which their speed is slower than usual speeds (i.e. 1 to 5 meters per second) it seems that the control system of the car can easily handle these amounts of error in that specific range.

V. CONCLUSION

In this paper, we proposed a new method to estimate the depth of cars in a monocular image. This approach is more robust than those which use the contact point of car detection bounding box with the ground and also those approaches, which use only the height of the car in the image. In this paper it is proposed to use a combination of a deep CNN architecture named YOLO to detect and localize cars and their

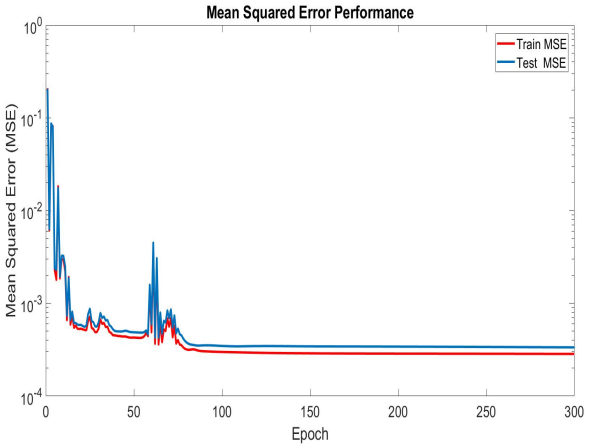


Fig. 8: Train and test MSE during the training process.

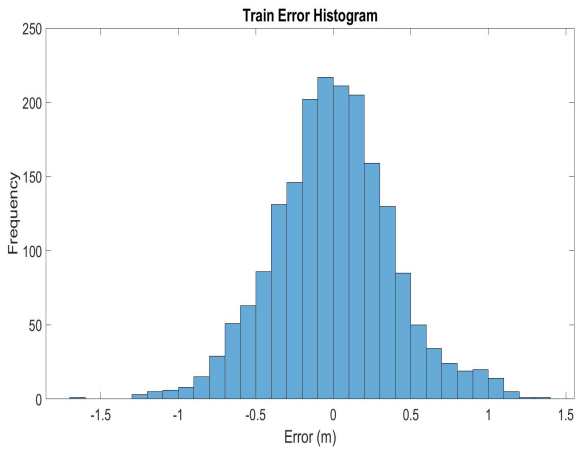


Fig. 9: The error histograms of train data evaluations after the training process.

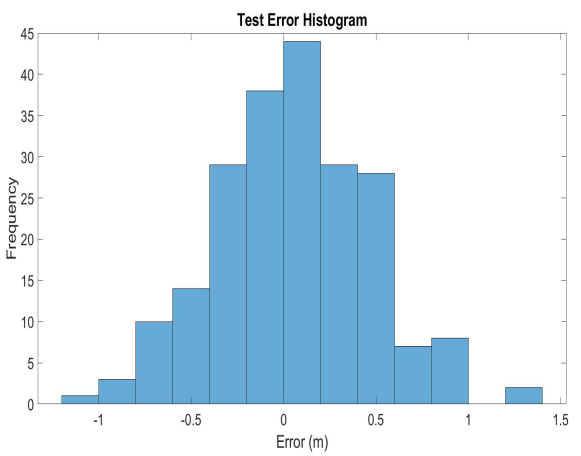


Fig. 10: The error histograms of test data evaluations after the training process.

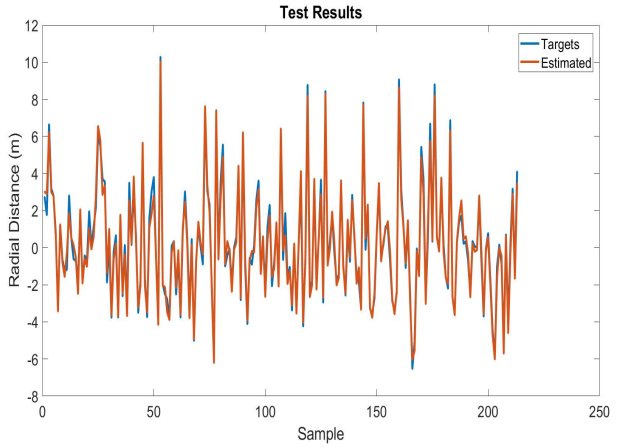


Fig. 11: Results of evaluating the model with test data.

lights with a nonlinear model to estimate depth. The results indicate the effectiveness of this approach and show that it can be a suitable method to estimate depth for autonomous vehicles.

References

[1] Richard Szeliski. *Structure from motion*, pages 303–334. Springer London, London, 2011.

[2] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):8001–8008, Jul. 2019.

[3] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE Robot. Automat. Mag.*, 18:80–92, 12 2011.

[4] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(5):824–840, May 2009.

[5] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[6] Miaomiao Liu, Mathieu Salzmann, and Xuming He. Discrete-continuous depth estimation from a single image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[7] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2366–2374. Curran Associates, Inc., 2014.

[9] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[10] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2015.

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[12] Fulvio Domini and Corrado Caudek. 3-d structure perceived from dynamic information: a new theory. *Trends in Cognitive Sciences*, 7:444–449, 2003.

[13] N Micheletti, Jim Chandler, and Stuart Lane. Structure from motion (sfm) photogrammetry. pages 1–12, 01 2013.

[14] N. Dinesh Reddy, Minh Vo, and Srinivasa G. Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[15] F. Lotfi, V. Ajallooeian, and H. D. Taghirad. Robust object tracking based on recurrent neural networks. In *2018 6th RSI International Conference on Robotics and Mechatronics (IcRoM)*, pages 507–511, Oct 2018.

[16] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2013.

[17] Ross B. Girshick. Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[18] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015.

[19] Weiwei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[20] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2016.

[21] Guangrui Liu. Real-time object detection for autonomous driving based on deep learning. 2017.

[22] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. 04 2018.

[23] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.