2023 International Conference on Computer Graphics and Image Processing (CGIP) | 979-8-3503-3318-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/CGIP58526.2023.00012

Using Monocular Depth Estimation for Distance Estimation in a Moving Vehicle

Lanz Benedict N. De Guzman
Manufacturing Engineering and Management Department
De La Salle University
Manila City, Philippines
lanz_deguzman@dlsu.edu.ph

Aaron Raymond See*
Department of Electrical Engineering
Southern Taiwan University of Science and Technology
Tainan City, Taiwan
aaronsee@stust.edu.tw / 0000-0003-4927-8778

***Abstract*—Accompanying the increase in demand for autonomous systems and robotic solutions is the increase in the relevance of various depth estimation technologies. Monocular Depth Estimation (MDE) is used to predict distances by generating depth maps using only a single RGB camera. However, without out-of-the-box calibration or ground truth reference for generated depth values from MDE models its use case in practical applications is limited. This research introduces a method of actualizing generated depth map values for different applications. The proposed system involves the utilization of machine vision using YOLO for object detection, followed by the computation of the lens optic algorithms to calculate the distance. Results demonstrated a real-time environment detection and depth estimation solution with more than 90% accuracy for measuring object depth in static environments. Furthermore, the system was also successfully tested in a moving vehicle to provide an estimated distance of surrounding vehicles. In the future, further tests will be done to improve the accuracy and calculation speed for use in car safety.**

***Keywords*—monocular depth estimation, machine vision, obstacle detection**

I. Introduction

Significant advancements in the field of robotics engineering and autonomous systems development led to the necessity of precise depth measurement solutions as challenges in obstacle detection, obstacle avoidance, and autonomous navigation are becoming increasingly relevant [1-3]. Conventionally, in measuring depth, distance sensors, 3D scanners, LIDARs, and stereo cameras are used [3-7]; however, consequent to their popularity is their respective expense. Moreover, with the popularity of mobile robotic applications, there is a significant preference for low-form factor and low-cost solutions. This dilemma led to the interest in developing monocular solutions for depth estimation [3,7,8]. Although there are various monocular depth estimation (MDE) methods and solutions [9,10], the conventional depth estimation solution for real-time applications involves a Deep Learning-based approach to measuring distance by using trained models to generate disparity maps from a single camera feed [7,8,11-12]. Nonetheless, a challenge in using monocular depth solutions is their reliability in various precision-based applications [3,6,8]. Results of MDE models are often represented as normalized values rather than actualized measurement values, which limit interpretations, and applications without an accompaniment of intricate control systems such as a fuzzy control system or PID controller [8].

A. Monocular Depth Estimation Models–MiDaS

In pursuit of a diverse and effective MDE model for multiple applications, Ranftl et al. approached training a model by mixing datasets to cover diverse environments without being burdened by certain biases from a single data collection approach. Essentially, five diverse training datasets, including data from stereo cameras, laser scanners, structured light sensors, and 3D films, were used together within a loss function to develop a new state-of-the-art MDE model [12]. Throughout the research study, MDE models from [12] are used.

B. YOLO Object Detection

In computer vision, YOLO, or "You Only Look Once," is among the popular solutions for real-time object detection known for its favorable speed, accuracy, and flexibility [13,14]. This convolutional neural network-based algorithm is used to detect objects within a video feed from a camera in real time. Throughout the application, aside from detecting objects, YOLO's bounding box outputs were extensively used to reference values from the generated depth map by computing for the bounding box centroids and determining detected object pixel height to accompany the actualization of values through the lens optic equation.

C. Actualizing Depth Map Values to Physical Measurement Values

This research introduces a method of actualizing generated depth map values for both static and dynamic scenario applications. The method on top of the MDE model used involves the use of the lens optic formula and YOLO object detection in calculating actual distance information. Accordingly, this research includes an application of the established method in road obstacle detection. The said application can estimate the distance of road elements and decide whether it is a potential danger or not.

II. Methodology

The novelty of this approach is the ingenuous use of the bounding box output of the YOLO object detection, precisely the bounding box height, the computable detection centroid, and the lens optic formula, for continuous computation of the target object's distance. Since the generated depth map of MiDaS is presented in normalized values, the output of this equation, via lens optic equation, and its location via centroid computation, is then used as a ground reference value for the remainder of the depth map that is adequately related through inverse proportion.

The overall process flow is as follows: the program initializes with a camera input that is then converted from BGR to RGB. From this converted feed, the MDE Model is initiated. Simultaneously, from the converted feed, a specific region, ROI, is cropped; within this cropped feed, a YOLO object detection is initiated with models specifically trained for locating the target object. Upon locating the target object, the target object's centroid and bounding box height information is stored. The centroid is primarily used as a reference point for the depth map calculations, while the bounding box height value serves as the pixel height value for the lens optic equation. Consecutively the target object's actualized distance and location within the cropped feed are computed and stored. Only after achieving the previous step will then, if any, surrounding objects be detected. During this step, the non-cropped feed is used, and another YOLO object detection model is initiated to locate surrounding objects. For each object detected, a corresponding depth map value is retrieved based on the object's centroid. From the stored target object's actualized distance and location, a corresponding depth map value from MiDaS is retrieved. The MiDaS value equivalent of the target object's distance and its calculated actualized distance are used as a reference in calculating the actualized distances of the surrounding objects. Relative distances are computed through inverse ratio and proportion. Correspondingly, all detection and calculated values are displayed within the video feed.

A. *YOLO Object Detection Centroid Calculation*

In OpenCV and YOLO, bounding boxes are determinants whenever an object is recognized. Four main variables are used in bounding boxes: x, y, w, and h. Variables x and y serve as the origin and starting coordinate where the box will be drawn. Variables w and h serve as the bounding boxes' width and height [15]. In referencing the center of an object given its bounding boxes, a mid-point formula was used:

$$Centroid\ (x_c, y_c) = \left(\frac{\text{x + (x+w)}}{2}, \frac{\text{y + (y+h)}}{2}\right) \qquad (1)$$

B. *Actualizing Depth Map Values to Physical Measurement Values*

In order to actualize depth map values into measurement quantities such as the SI units, physical relationships were inferred through the lens optic equation together with ground truths from the camera's information and a known object's pixel height [16]. Given this formulaic constraint, the actualizing approach is limited to scenarios where a target object with known physical height measurement is present within a video feed.

$$Distance = \frac{\text{Real Object Height(mm)} * \text{Focal Length(mm)} * \text{SensorHeight(Px)}}{\text{Object Height(Px)} * \text{SensorHeight(mm)}} \qquad (2)$$

C. *Inverse Proportion Equation*

Upon testing, the relationship of the generated depth map was observed to be inversely proportional. Whereby within the normalized scale of 0 - 1, closer values to 1 represent that an object is far and vice versa and is represented by,

$$\text{MDE Actualized Dist} = \frac{\text{Refernce Distance (Reference MiDaS Value)}}{\text{Object MiDaS Value}} \qquad (3)$$

where Reference Distance is the Target Object's calculated distance from Eq. 2. Reference MiDaS Value is the Target Object's centroid equivalent MiDaS value. Surrounding Object MiDaS Value is the Surrounding Object's centroid equivalent MiDaS value.

D. *ROI Centroid Equivalence to Original Feed Calculation*

Since the approach utilized, ROI or cropped region detection, centroid values from the ROI feed are localized and does not refer to the same location within the original feed. This was addressed by adding an offset value to the retrieved points from the ROI feed to compensate when computing with values from the original feed. The offset value was based on ROI used values- (y1:y2,x1:x2), where values of y1 and x1 are added to the retrieved coordinates from the specific object detection.

$$Centroid\ Equivalence\ (x_{CE}, y_{CE}) = (Xc + X1,\ Yc + Y1) \qquad (4)$$

E. *Data Sets and Models*

On the entirety of the research study, the MDE model that was specifically used was the MiDaS v2.1 Small. The small model was favored based on its low computational requirement, suitable for real-time applications whilst being able to output an adequate depth map. Correspondingly, throughout the study, various YOLO versions, data sets, and models were used, such as the YOLOv3 MS COCO data set [17], YOLOv4 Road Obstacle [18], and YOLOv4 License Plate Detection Data Set [19].

F. *Camera and Experimentation Set Up*

Throughout the experimentation and testing period, Logitech's C310, a single USB camera, was used. Correspondingly the following information about the camera was used: a) Camera pixel height: 960 pixels, b) Camera sensor height: 9.6 mm, c) Focal Length: 14 mm. Note that the sensor height is multiplied by two due to the pixel resolution being upscaled by the software from 640 (w) x 480 (h) to 1280 (w) x 960 (h). In scaling the resolution, the sensor height can be 4.8mm, but the sensor height in pixel and object height must be divided accordingly, in this case by 2. Furthermore, the camera was placed close to the car's windshield during road obstacle distance estimation and detection. An Apple M1 MacBook Air was used as the primary computing device throughout the testing period.

III. Results and Discussion

The proposed system was tested in various situations, including controlled and uncontrolled environments. Controlled environments were primarily dedicated to evaluating the approach's accuracy, while the uncontrolled environment was for performance testing. Controlled environment testing was primarily conducted within a bedroom and dining room, while uncontrolled environment testing was conducted outdoors during city and expressway driving. In terms of road application, different models were used. A YOLOv4 model that can detect 5 classes – cars, trucks, pedestrians, bicyclists, and the light- was used to determine the environment [18]. Within this application, the target object used was license plate numbers, being that all cars have license plates that are standardized and have a constant

National Science and Technology Council, Taiwan under Grant NSTC 111-2221-E-218-006,

height. In detecting the license plate, a specifically trained YOLO model [19].

A. *Object Detection Using MS COCO and Depth Estimation Using MiDaS v2.1 Small with Present Surrounding Object*

Fig. 1 shows the result of detecting the target object, the cup, and detecting the surrounding object, the bottle using [17] dataset. Within these tests, the proposed method successfully computed the distance of the cup and related it with the generated depth map to calculate the distance of the bottle.

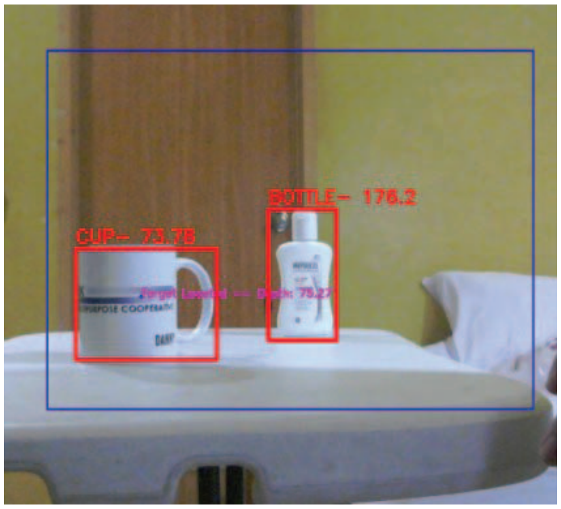

Fig. 1. Detecting the target object (cup) and surrounding object (bottle).

Fig. 2 shows the detected object and the distance using the same models. Table I shows the actual distance and the calculated distance with less than 5% difference reaching an accuracy of around 98.68% and 96.85% in detecting that target object and surrounding object, respectively.

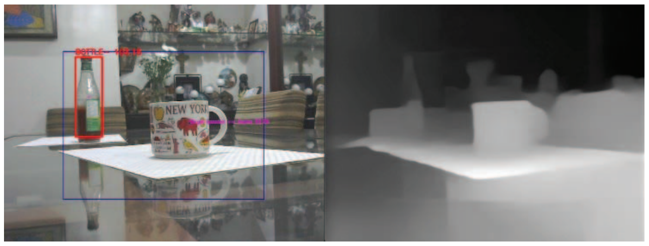

Fig. 2. Static objects with object detection and distance calculation and the depth map of the image.

TABLE 1. DISTANCE ESTIMATION AND DIFFERENCE

	True Value	Computed Value	% Difference
Cup	57 cm	57.75 cm	1.32%
Bottle	100 cm	103.15 cm	3.15%

B. *Road Obstacle Detection and Distance Estimation.*

Fig. 3 shows the successful implementation of using [13] and [14] models together with the proposed method to estimate the distance of the car. The system after detecting the license plate correspondingly computes for the distance of the car.

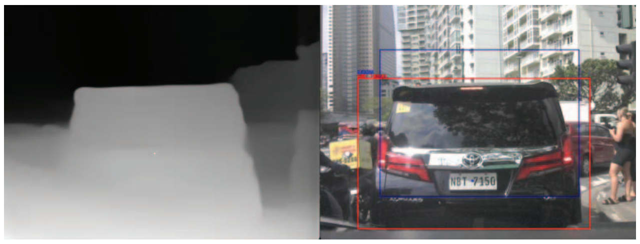

Fig. 3. Detecting the car and locating the plate number in a static environment

C. *MDE Testing in a Moving Vechicle*

Fig. 4. demonstrates the implementation results specific to road obstacle detection using models [18] and [19]. Throughout the test, the surrounding objects are the road elements, while the target object is the license plate. Within this test, bounding boxes are updated based on their level of potential danger. Whenever a detected object is less than 3000 mm or 3 m, around the length of 2 cars, they are regarded as a potential danger and are represented by changing their bounding boxes to the color red. Whenever an object is considered safe, its bounding box colors change to green. This scenario showcases the program's performance in city driving conditions.

Because of the dynamic nature of these scenarios, variability is observed when measuring distances. The use of detecting license plates as target objects had pros and cons. These are only applicable when a car is in front; thus, detection and depth retrieval is rarely observed when turning left or right. Accordingly, the test was highly limited by the machine's processing capability. Frame processing is far from usable and must be improved for further studies and real-time applications. Unable to measure the computed and predicted distance estimation accuracy, the test shows promise and proof of the concept of using MDE in obstacle detection and is a potential tool for autonomous navigation.

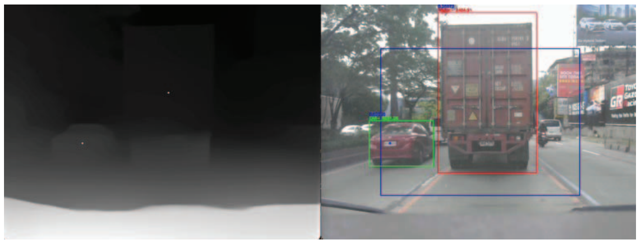

Fig. 4. Road element detection and distance estimation in city driving conditions

D. *Testing During in a Raining Setup*

The system was also tested while driving on a rainy day and it is inferred that the testing setup is not applicable in rainy situations. Being placed close to the windshield, the refraction caused by the water being at its surface essentially obstructs the camera's feed as shown in Fig. 5.

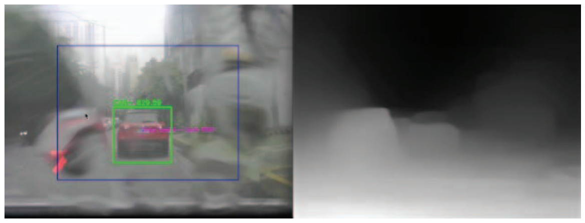
Fig. 5. MDE testing while driving and raining.

IV. Conclusions

The proposed method was able to successfully actualize depth information through the use of the MDE models, various computer vision techniques, and inferring physical relationships. In static scenarios, the proposed method renders at least 90% accuracy. Although most tests utilized cups as target objects, similar performance is achieved using different objects such as the bottle and the license plate. Essentially, the method is flexible for different applications, given a specific YOLO detection model for the application and ground truth information on the target object. With regards to road obstacle detection, the study, together with the integrated crude obstacle safety determination, was able to provide a proof of concept of its potential use in challenges regarding autonomous navigation, specifically obstacle detection and avoidance. Although the approach would not omit the need for fuzzy-based controllers, the established method strengthens MDE's application with appropriate distance information for different precision-based applications.

Throughout the research study, the main recommendation is to perform further experiments to validate the proposed method in dynamic scenarios. This can be performed by running the program alongside a different distance-measuring device, such as a LIDAR or stereo camera, and comparing results accordingly. The limitation of the proposed method is observed when the target object is not perpendicular to the camera and is viewed at a different degree. In detecting the target object in such scenarios, more than 2D bounding boxes would be required. A recommended approach is integrating 3D object detection with 3D bounding boxes so that an adequate pixel height can be inferred in computing distances regardless of viewing the object at different angles. Particularly, to improve detection accuracy and appropriateness, the researchers recommend training models using customized datasets to adequately detect country-specific vehicles such as jeepneys, tricycles, and e-motors. Lastly, the researcher recommends considering a different path to actualizing the measurements using a maker object within the car's hood. This was the initial direction of the paper however was not pursued due to possible road restrictions.

Acknowledgment

This work was supported in part by the National Science and Technology Council (NSTC), Taiwan, under Grant NSTC 111-2221-E-218-006.

References

[1] A. Masoumian, H. A. Rashwan, J. Cristiano, M. S. Asif, and D. Puig, "Monocular depth estimation using Deep Learning: A Review," Sensors, vol. 22, no. 14, p. 5353, 2022.

[2] I. K. Somawirata, K. A. Widodo, S. Achmadi, and F. Utaminingrum, "Road and obstacle detection for Autonomous Electrical Vehicle Robot," *Journal of Image and Graphics*, vol. 8, no. 4, pp. 126–130, 2020.

[3] M. Mancini, G. Costante, P. Valigi, and T. A. Ciarfuglia, "Fast robust monocular depth estimation for obstacle detection with fully convolutional networks," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

[4] Y. Li, A. W. Yu, T. Meng, B. Caine, J. Ngiam, D. Peng, J. Shen, Y. Lu, D. Zhou, Q. V. Le, A. Yuille, and M. Tan, "DeepFusion: LIDAR-camera deep fusion for multi-modal 3D object detection," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[5] J.-J. Cheng, C.-C. Chang, and H.-Y. Lin, "Enhanced depth features of human bodies for image retargeting," *Journal of Image and Graphics*, vol. 3, no. 1, 2015.

[6] N. Smolyanskiy, A. Kamenev, and S. Birchfield, "On the importance of stereo for accurate depth estimation: An efficient semi-supervised Deep Neural Network Approach," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[7] X. Dong, M. A. Garratt, S. G. Anavatti, and H. A. Abbass, "Towards real-time monocular depth estimation for robotics: A survey," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 10, pp. 16940–16961, 2022.

[8] Z. Zhang, M. Xiong, and H. Xiong, "Monocular depth estimation for UAV obstacle avoidance," in 2019 4th International Conference on Cloud Computing and Internet of Things (CCIOT), 2019.

[9] N. Mukai, Y. Matsuura, M. Oishi, and M. Oshima, "Chromatic aberration based depth estimation in a fluid field," *Journal of Image and Graphics*, vol. 6, no. 1, pp. 59–63, 2018.

[10] T.-Y. Kuo, Y.-C. Lo, and Y.-Y. Lai, "Depth estimation from a monocular outdoor image," 2011 IEEE International Conference on Consumer Electronics (ICCE), 2011.

[11] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," arXiv.org, 06-Mar-2020. [Online]. Available: https://arxiv.org/abs/1907.10326v5. [Accessed: 21-Oct-2022].

[12] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for Zero-shot cross-dataset transfer," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 3, pp. 1623–1637, 2022.

[13] U. Handalage and L. Kuganandamurthy, "Real-time object detection using YOLO: A review." Academia.edu, 07-May-2021. [Online].

[14] F. Spiess, L. Reinhart, N. Strobel, D. Kaiser, S. Kounev, and T. Kaupp, "People detection with depth silhouettes and convolutional neural networks on a mobile robot," *Journal of Image and Graphics*, vol. 9, no. 4, 2021.

[15] A. K. Panda, "Ml model to detect the biggest Object in an image — part 1," Hackernoon.com, 12-Feb-2019. [Online]. Available: https://medium.com/hackernoon/single-object-detection-e65a537a1c31. [Accessed: 20-Oct-2022].

[16] "Www.scantips.com," Calculator to compute the Distance or Size of an Object in a photo Image. [Online]. Available: https://www.scantips.com/lights/subjectdistance.html. [Accessed: 21-Oct-2022].

[17] J. Redmon, Yolo: Real-time object detection. [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed: 21-Oct-2022].

[18] Cyuan, Self-driving-with-YOLO: A YOLOv4 model that can detect 5 classes on the road and a comparison with YOLOv3.

[19] G. Kataria, Yolov4-Pytesseract-License-plate-detection-and-reading.