symmetry

MDPI

Article

Vehicle Distance Estimation from a Monocular Camera for Advanced Driver Assistance Systems

Seungyoo Lee †, Kyujin Han †, Seonyeong Park and Xiaopeng Yang *

School of Global Entrepreneurship and Information Communication Technology, Handong Global University, Pohang 37554, Republic of Korea
* Correspondence: yxp233@handong.edu
† These authors contributed equally to this work.

Abstract: The purpose of this study is to propose a framework for accurate and efficient vehicle distance estimation from a monocular camera. The proposed framework consists of a transformer-based object detector, a transformer-based depth estimator, and a distance predictor. The object detector detects various objects that are mostly symmetrical from an image captured by the monocular camera and provides the type of each object and the coordinate information of a bounding box around each object. The depth estimator generates a depth map for the image. Then, the bounding boxes are overlapped with the depth map to extract the depth features of each object, such as the mean depth, minimum depth, and maximum depth of each object. The present study then trained three models—eXtreme Gradient Boosting, Random Forest, and Long Short-Term Memory—to predict the actual distance between the object and the camera based on the type of the object, the bounding box of the object (including its coordinates and size), and the extracted depth features. The present study proposes including the trimmed mean depth of an object to predict the actual distance by excluding the background pixels around an object but within the bounding box of the object. The evaluation results show that the proposed framework outperformed existing studies.

Keywords: vehicle distance estimation; object detection; depth estimation; advanced driver assistance systems

Citation: Lee, S.; Han, K.; Park, S.; Yang, X. Vehicle Distance Estimation from a Monocular Camera for Advanced Driver Assistance Systems. *Symmetry* **2022**, *14*, 2657. https://doi.org/10.3390/sym14122657

Academic Editor: Dumitru Baleanu

Received: 2 November 2022
Accepted: 13 December 2022
Published: 15 December 2022

1. Introduction

Measuring the distances of a driver's vehicle to its surroundings is essential in advanced driver assistance systems (ADAS) for road safety. Existing distance measurement methods can be classified into three groups: active sensor-based, passive vision-based, and fusion-based methods. Active sensor-based approaches use sensors such as radar and light detection and ranging (LiDAR) for distance measurement. Radars are able to detect objects up to 150 m away [1], but they are limited by low resolutions [2]. LiDAR provides higher resolutions [3] but is costly [4–7]. The main advantages of active sensors are that they are efficient in distance measurement [2] and applicable in different visibility conditions [5].

Passive vision-based approaches use vision sensors such as cameras for distance estimation. Existing vision-based methods can be classified into two groups: stereo camera-based methods and monocular camera-based methods. Stereo camera-based methods consider multiple-view geometry and provide depth for each pixel by matching stereo image pairs [2,5]. However, stereo camera-based methods are limited by the complexity of stereo calibration, errors in matching stereo image pairs, and efficiency in actual road scenarios [2,5,6,8]. Monocular camera-based methods use a single camera for distance estimation and therefore are inexpensive [5,6], and they have become a trend in distance estimation.

The monocular camera-based approaches can be further classified into geometric approaches and deep learning-based approaches. Geometric approaches use geometric

properties in a two-dimensional (2D) image and camera parameters for distance estimation. Kim and Cho [9] used the relative position information between the camera and front vehicle, camera setting parameters, and the width of the front vehicle to estimate the inter-vehicle distance. Liu et al. [10] applied inverse perspective mapping transformation to convert an image to a bird's eye view and restore the road plane information to estimate the inter-vehicle distance. Such methods are limited by their heavy dependence on image brightness and the accuracy in measuring the camera parameters and the target size.

Recently, deep learning-based approaches have become popular in distance estimation from a monocular camera. Such methods commonly train various neural networks for distance estimation [6,11–20]. Guizilini et al. [13] proposed using three-dimensional (3D) packing and unpacking blocks in their self-supervised network to preserve spatial information for depth estimation. Zhang et al. [15] proposed a network with regions with convolutional neural network (R-CNN)-based structures for distance estimation and explored several regression methods to improve distance estimation results. Fu et al. [17] proposed a deep ordinal regression network and adopted a multi-scale network structure for depth estimation. Xu et al. [18] proposed fusing the side outputs of multi-scale CNNs with continuous conditional random fields (CRFs) for depth estimation through supervised learning. Liang et al. [6] proposed a self-supervised, scale-aware network to estimate distance. However, their method requires calibrating the camera and integrating the calibrated parameters into their network.

Before distance estimation, object detection needs to be performed to identify different objects that are mostly symmetrical from an image. Object detection methods can be classified into conventional [21–23] and deep learning-based methods [24–35]. Conventional object detection methods usually manually extract features from the selected region of interest and then classify the extracted features. However, the conventional methods are computationally costly and insufficient in accuracy [6]. Deep learning-based methods train various CNN-based or transformer-based models in a supervised learning or self-supervised learning manner for object detection. The results of deep learning-based methods are promising.

This study was intended to propose and evaluate a framework for better accuracy and efficiency in vehicle distance estimation. The proposed framework consists of an object detector and a depth estimator based on a transformer. After depth estimation, different models were applied to predict vehicle distance from depth information to find the best-performing model.

2. Materials and Methods

As shown in Figure 1, the proposed framework in this study consists of an object detector, a depth estimator, and a distance predictor. The object detector detects an object in an image and provides the type of the object and a bounding box around the object with the coordinate information of the bounding box. The depth estimator generates a depth map for the image. Then, the bounding box is overlapped with the depth map to extract the depth features of the object, such as the mean depth, minimum depth, and maximum depth. The distance predictor predicts the actual distance between the object and the camera that captures the image based on the type of the object, the bounding box, including its coordinates and size, and the extracted depth features.

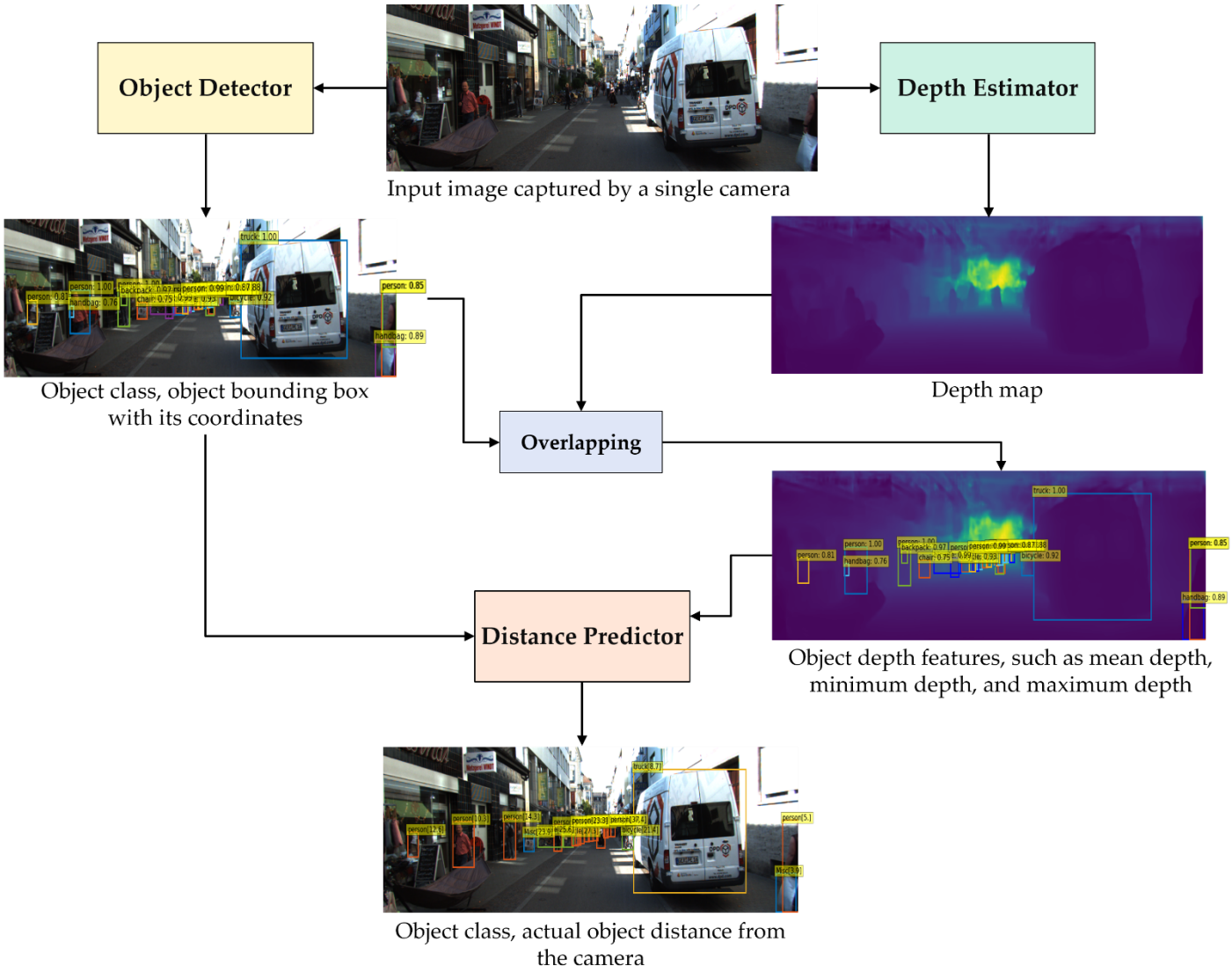

Figure 1. The proposed framework for vehicle distance estimation.

2.1. Object Detector

The present study used a pretrained transformer-based deep learning model named DEtection TRansformer (DETR) [32] for object detection due to its high effectiveness. As shown in Figure 2, the model consists of a CNN backbone, ResNet-101 [36], for extracting the features of an input image, an encoder-decoder transformer, and a feedforward network (FFN) for the final detection. The extracted features of the input image are flattened and supplemented with positional encoding before passing them to the transformer encoder. Then, a small, fixed number of learned positional embeddings, called object queries, is passed to the transformer decoder. Lastly, each output embedding of the decoder is fed to the FFN to predict either an object class with a bounding box or a class without any object. Figure 3 illustrates an example of the objects detected using DETR.

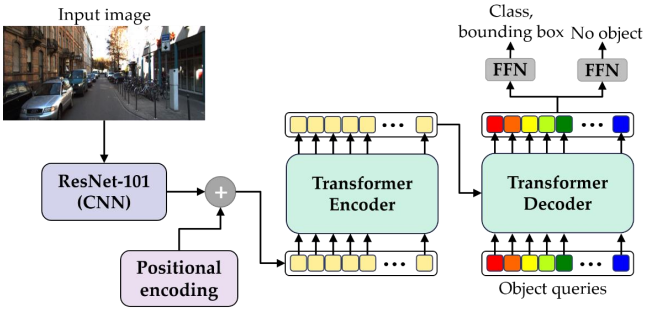

Figure 2. The object detector named DEtection TRansformer [32] for object detection used in this study (FFN: feedforward network).

Figure 3. An example of the detected objects with classes and bounding boxes using the object detector in this study.

2.2. Depth Estimator

A pretrained transformer-based deep learning model named the global-local path network [37] was used for depth map estimation due to its high accuracy and robustness. As shown in Figure 4, the global-local path network consists of a transformer encoder that learns global dependencies to extract features in different scales and a decoder that generates the target depth map from the extracted features by establishing local paths through a skip connection and a selective feature fusion module. Figure 5b shows an example of the estimated depth map using the global-local path network. Then, as shown in Figure 5c, the detected bounding boxes are overlapped with the estimated depth map to extract the depth features of each object, such as the mean, median, maximum, and minimum depths of the pixels in the bounding box of the object. If there is any overlapping area between two bounding boxes, then the overlapping area is excluded before extracting the depth features.

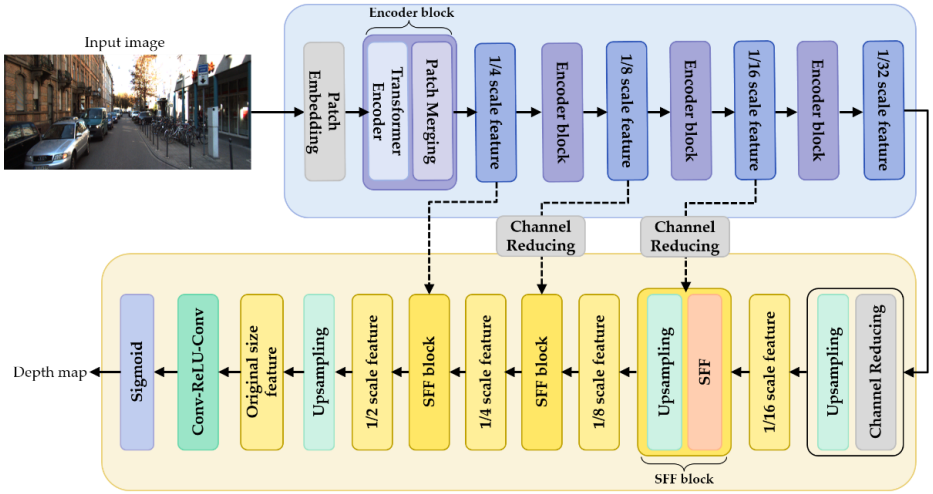

Figure 4. The depth estimator named the global-local path network [37] for depth map estimation in this study (SSF: selective feature fusion; Conv: convolution; ReLU: rectified linear unit).

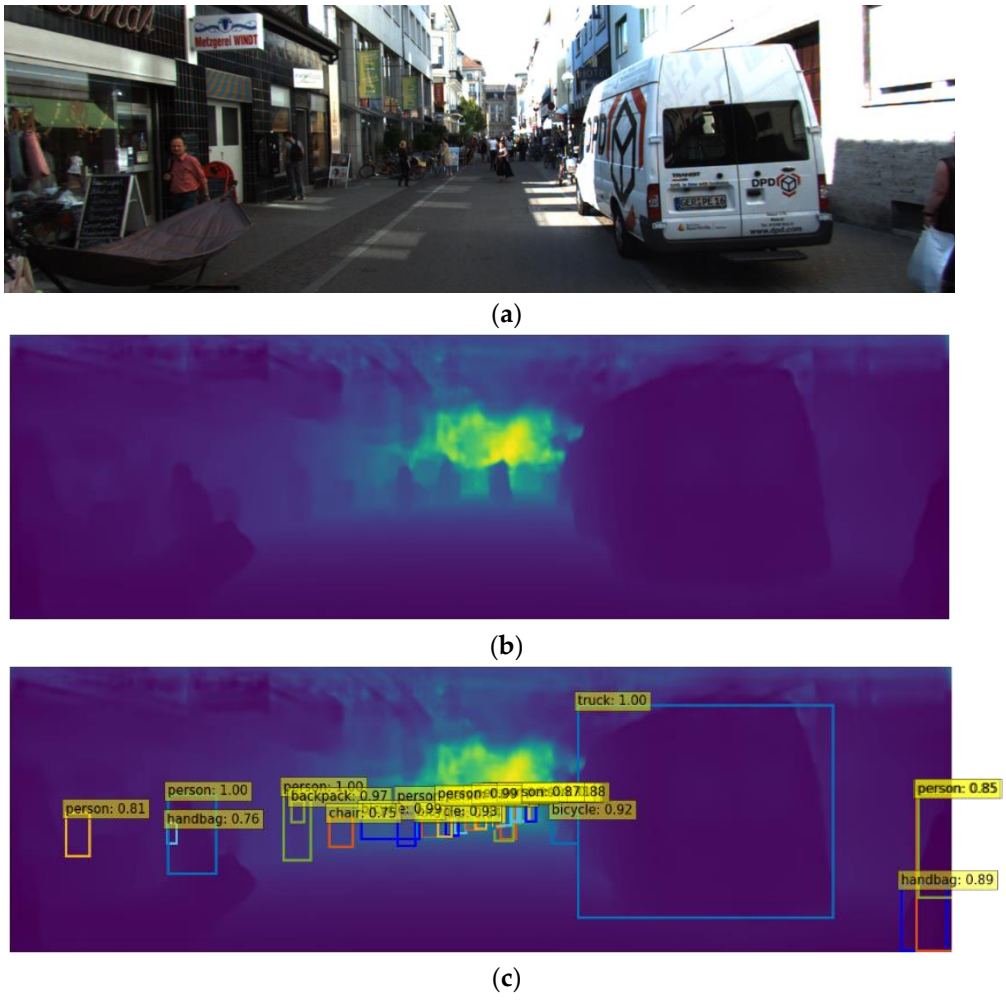

Figure 5. Depth map estimation using the depth estimator in this study: (**a**) the original image, (**b**) the estimated depth map, and (**c**) the overlaid depth map with the identified bounding boxes using the object detector in this study.

2.3. Distance Predictor

Three machine learning models—eXtreme Gradient Boosting (XGBoost) [38], Random Forest (RF) [39], and Long Short-Term Memory (LSTM) [40]—were trained for predicting the absolute distance of an object to the camera based on the information of its bounding box and depth features, and then their performances were compared.

2.3.1. XGBoost

XGBoost is a scalable implementation of the Gradient Boosting framework for supervised learning. Through parallel creation of trees and regularization to avoid overfitting, XGBoost achieves high efficiency and accuracy. XGBoost can be used for regression and classification. With many hyperparameters, XGBoost is highly flexible and therefore can be customized to solve a specific problem.

2.3.2. RF

RF combines many random tree predictors by using ensemble learning to provide solutions to complex problems. RF is also a supervised learning method and can be used for classification or regression. Based on the predictions of the decision trees, RF provides an output by taking the most votes for classification tasks or by taking the average for regression tasks. RF can avoid overfitting and reduce variance through bagging during training, and it requires fewer hyperparameters and little parameter tuning.

2.3.3. LSTM

LSTM is a variation of the recurrent neural network (RNN) that avoids the vanishing gradient problem in RNNs for learning long-term dependencies. As shown in Figure 6, LSTM has a hidden state represented by h_{t-1} and h_t for the previous and current timestamps, respectively. In addition, LSTM has a cell state represented by C_{t-1} and C_t for the previous and current timestamps, respectively. The cell state is known as the long-term memory. The hidden state is known as the short-term memory. The LSTM cell consists of a forget gate, an input gate, and an output gate. The forget gate determines whether one should keep or forget the information from the previous timestamp in the cell state. The input gate tries to learn new information from the input by deciding whether the input flows to the cell state. The output gate determines whether the cell state is passed to the output and the hidden state for the next timestamp. As shown in Figure 7, the structure of the proposed LSTM model consists of three LSTM layers, three FFN layers, and a linear layer.

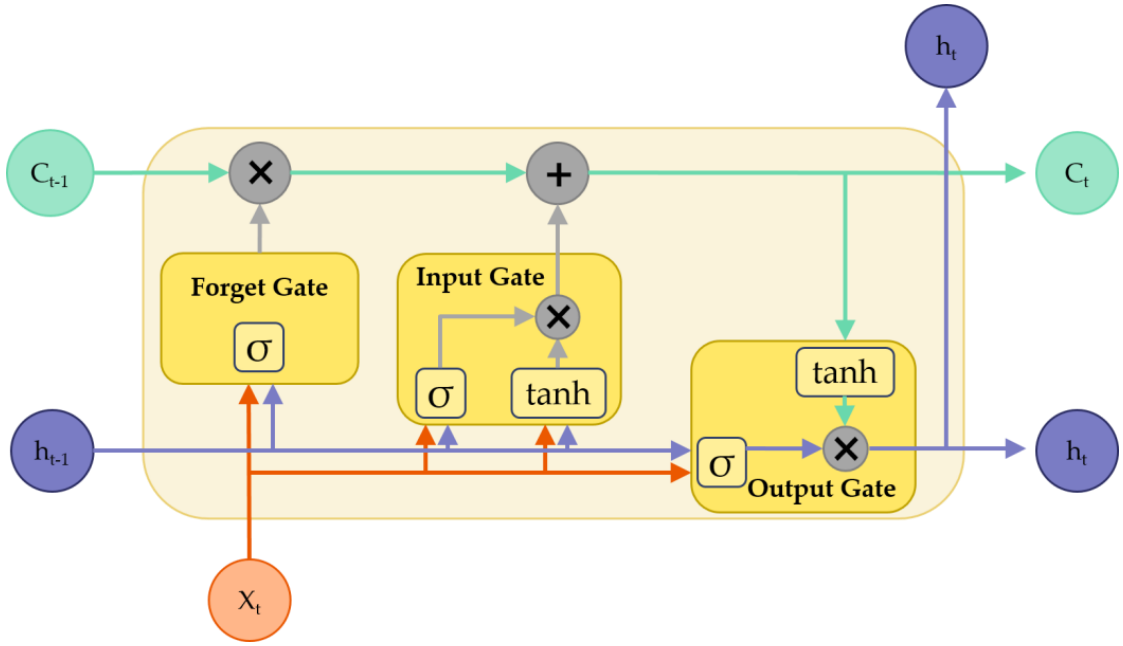

Figure 6. The structure of a Long Short-Term Memory cell.

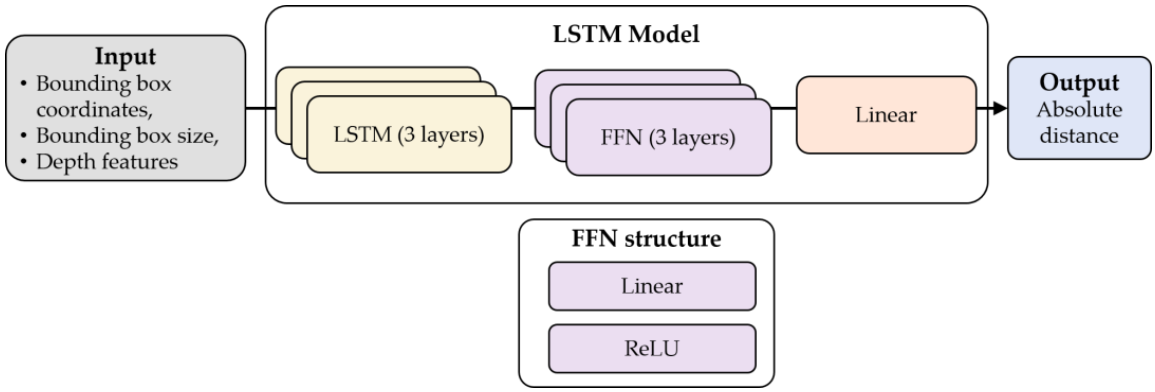

Figure 7. The structure of the proposed Long Short-Term Memory (LSTM) model for distance prediction (FFN: feedforward network; ReLU: rectified linear unit).

3. Experiments

3.1. Data Preprocessing

The present study used the Karlsruhe Institute of Technology and Toyota Institute (KITTI) [41] dataset. The KITTI dataset consists of the class of each object, the coordinates of the bounding box of the object, the angle of the camera for capturing the object, and the distance from the object to the camera. To train the three models for distance prediction in our study, the KITTI dataset was preprocessed.

First, the coordinates of the bounding box of each object in the KITTI dataset were replaced with those identified with the object detector in our framework. The reason for this is that the proposed framework uses the identified bounding box for distance prediction. This study compared the performance of the models trained using the original bounding box and the identified bounding box. Then, the intersection over union (IoU) function was used to identify the overlapping percentage between two bounding boxes. If the overlapping percentage between two bounding boxes was over 70%, then the bounding box of the object farther from the camera was removed. If the overlapping percentage was less than 70%, then the overlapping area was excluded before extracting the depth features for each of the two objects. Lastly, the KITTI dataset was visually inspected, and any object with a mislabeled object distance was excluded, as shown in Figure 8. After preprocessing, an updated dataset with a total of 27,021 objects was obtained. Then, the updated dataset was randomly split into training, validation, and testing datasets at a ratio of 8:1:1, resulting in 21,616 objects for training, 2702 objects for validation, and 2703 objects for testing. Six classes of objects were used in our study: car, truck, person, bicycle, train, and other.

Figure 8. An example of a mislabeled object distance (highlighted in the dotted ellipse) from the Karlsruhe Institute of Technology and Toyota Institute (KITTI) [41] dataset.

3.2. Model Training

The present study implemented and trained the three distance prediction models using PyTorch 1.9.1 on a laptop-based NVIDIA GeForce RTX 3070 GPU. The hyperparameters and their values for training the XGBoost, RF, and LSTM models are shown in Table 1. For training the LSTM model, this study used L1 loss and set the initial learning rate at 0.005. The ReduceLROnPlateau scheduler was used to decrease the learning rate by 0.5 with patience of 10 epochs. The EarlyStopping callback was used to stop training if the validation loss did not improve after 70 epochs.

Table 2 shows the input variables and the output variable used for training the three models. To represent the object class variable, label encoding was used for the XGBoost model, and one-hot encoding was used for the RF model and LSTM model. Except for the class variable, normalization was used to scale the other input variables. This study proposed including the 20% trimmed mean depth of an object to predict the actual distance by excluding the background pixels around an object but within the bounding box of the object. To calculate the 20% trimmed mean depth of an object, this study flattened the depth matrix of the pixels in the bounding box of the object as a depth vector. Then, the depth vector was sorted. After that, the top 10% and bottom 10% of pixels in the sorted depth vector were excluded. Lastly, the mean depth of the remaining pixels was calculated as the 20% trimmed mean depth of the object. The output variable was the ground truth distance from the object to the capturing camera.

Table 1. Hyperparameters and their values for training the eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Long Short-Term Memory (LSTM) models for distance prediction.

Model	Hyperparameter	Value
XGBoost	colsample_bytree	0.9
	gamma	0.3
	learning_rate	0.01
	max_depth	9
	min_child_weight	3
	n_estimators	1000
	reg_alpha	1
	reg_lambda	0.9
	subsample	0.7
	objective	squared_error
RF	n_estimators	500
	learning_rate	0.01
	max_depth	20
	max_features	2
	min_samples_split	2
	min_samples_leaf	1
	criterion	squared_error
LSTM	Input_dim	15
	Hidden_dim(LSTM)	612
	Layer_dim(LSTM)	3
	Hidden_dim(Linear)	612, 306, 154, 76
	Output_dim(Linear)	1
	Bidirectional	False
	Optimizer	Adam
	Activation function	ReLU
	Max epoch	1000
	Batch size	24

Table 2. Input and output variables and their descriptions for training the proposed distance prediction models.

Category	Variable	Description
Input variables	x_min	Minimum x coordinate of a bounding box
	y_min	Minimum y coordinate of a bounding box
	x_max	Maximum x coordinate of a bounding box
	y_max	Maximum y coordinate of a bounding box
	width	Width of a bounding box
	height	Height of a bounding box
	depth_mean	Mean depth of an object
	depth mean_trim	20% trimmed mean depth of an object
	depth_max	Maximum depth of an object
	depth_median	Median depth of an object
	class	Type of an object
Output variable	d	Ground truth distance of an object

3.3. Evaluation

This study used the mean absolute error (MAE) to evaluate the performance of the distance prediction models with the testing dataset. The MAE for the predicted object distance is defined by Equation (1):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|d_i - \hat{d}_i\right| \tag{1}$$

where N is the total number of objects, d_i is the actual object distance, and $\hat{d}_i$ is the predicted object distance.

This study used another five measures to compare the performance of the proposed framework with various other methods. The five measures were the absolute relative error (*AbsRel*),

squared relative difference (*SquaRel*), root mean squared error (*RMSE*), *RMSE log*, and threshold accuracy (*Threshold*), which are defined as follows:

$$AbsRel = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|d_i - \hat{d}_i\right|}{d_i} \tag{2}$$

$$SquaRel = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\left|d_i - \hat{d}_i\right|\right|^2}{d_i} \tag{3}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}||d_i - \hat{d}_i||^2} \tag{4}$$

$$RMSE\ log = \sqrt{\frac{1}{N}\sum_{i=1}^{N}||\log d_i - \log \hat{d}_i||^2} \tag{5}$$

$$Threshold = \%\ \text{of}\ d_i\ \text{s.t.}\max\left(\frac{\hat{d}_i}{d_i}, \frac{d_i}{\hat{d}_i}\right) = \delta < threshold \tag{6}$$

where the threshold usually takes on three values: $\delta < 1.25$, $\delta < 1.25^2$, and $\delta < 1.25^3$.

This study performed on-road evaluation of the proposed distance prediction framework. The evaluation experiment was conducted on a wide road without many cars by varying the object (car) distances from 10 m to 80 m in intervals of 10 m, as shown in Figure 9. A steel tape with a length of 100 m was used to mark the ground truth distances. A cheap webcam was used to record video of the car in front. The webcam was connected to a laptop with an NVIDIA GeForce RTX 3070 GPU installed to run the proposed framework. To simplify the experiment and make sure the measurement accuracy of the ground truth distances, the camera was mounted on a fixed platform. The height of the platform was set to be the same as the platform above the center console of a 10th-generation Honda Accord sedan. The webcam was mounted approximately level with the ground plane. No calibration of the camera was needed in the experiment. After the camera started recording, the front car was driven to the ground truth distances. Meanwhile, the proposed framework took three images form three contiguous video frames captured at each ground truth distance level and predicted the distance of the front car to the camera in real time. The average of the distance values predicted from the three contiguous video frames was calculated and used as the predicted object distance. Then, the accuracy of the predicted object distances was obtained through comparison with the ground truth distances.

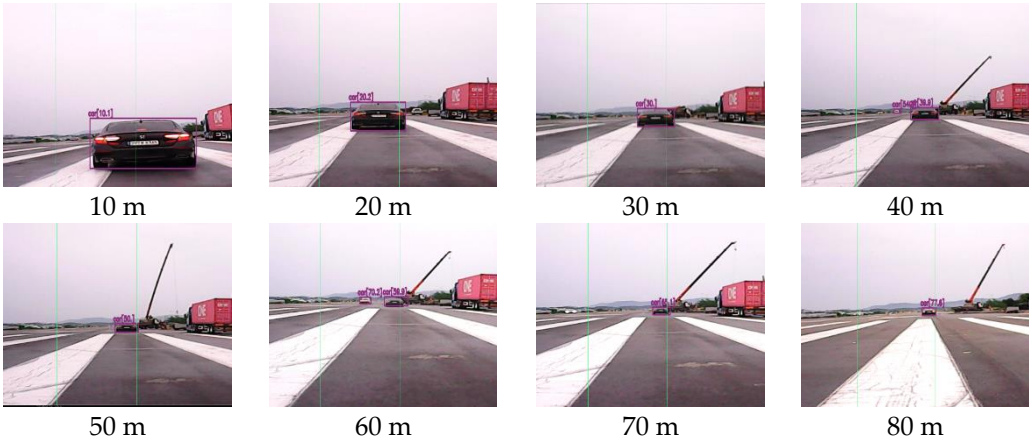

Figure 9. On-road experiment for evaluating the performance of the proposed distance prediction framework at different distance levels.

4. Results and Discussion

Table 3 shows the evaluation results of the three distance prediction models. Among the three models, LSTM outperformed the other models in terms of the MAE. For different object classes, this study found that the XGBoost model showed the best performance in distance prediction for the car and bicycle classes. For the remaining classes, the LSTM model showed the best performance. Therefore, in the proposed distance prediction framework, if a car or a bicycle was detected, the XGBoost model was used for distance prediction. Otherwise, the LSTM model was used. Table 4 shows the evaluation results of the three models at different distance intervals. As the object distance increased, this study found that the error in distance prediction increased for all the three models.

Table 3. Evaluation results of the eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Long Short-Term Memory (LSTM) models for distance prediction in terms of mean absolute error (MAE) for different object classes using the testing dataset.

Model	MAE (m)						
	Car	**Person**	**Bicycle**	**Train**	**Truck**	**Others**	**Overall**
XGBoost	0.2159	0.7366	1.3290	1.8476	2.4005	1.9559	1.2194
LSTM	1.2131	0.6178	1.6292	1.2472	1.9459	1.1650	1.1658
RF	1.3258	0.7664	1.6695	2.1551	2.6382	2.5058	1.3134

Table 4. Evaluation results of the eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Long Short-Term Memory (LSTM) models for distance prediction in terms of mean absolute error (MAE) at different distance intervals using the testing dataset.

Model	MAE (m)							
	0–9 m	**10–19 m**	**20–29 m**	**30–39 m**	**40–49 m**	**50–59 m**	**60–69 m**	**70–80 m**
XGBoost	0.3786	0.6032	0.9749	1.5372	1.9183	2.7571	3.6277	4.1768
LSTM	0.4154	0.5248	0.8868	1.5052	1.9079	2.5899	3.7846	3.3255
RF	0.4079	0.6287	1.0588	1.6363	2.0780	3.0674	3.5624	4.7060

Table 5 shows the evaluation results when using different levels for the trimmed mean object depth, namely 10%, 20%, and 30%. Among the three different levels, using a 20% trimmed mean depth achieved the best performance in distance prediction for the three distance prediction models, except for the RF model. Since our framework will use the XGBoost and LSTM models only, using the 20% trimmed mean depth is recommended.

Table 5. Evaluation results of the eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Long Short-Term Memory (LSTM) models for distance prediction in terms of mean absolute error (MAE) with different levels (10%, 20%, and 30%) of trimmed mean object depth using the testing dataset.

Model	MAE (m)		
	10% Trimmed	**20% Trimmed**	**30% Trimmed**
XGBoost	1.2279	1.2194	1.2258
LSTM	1.1909	1.1658	1.1895
RF	1.2665	1.3134	1.2657

Table 6 shows the evaluation results of the three distance prediction models trained using the ground truth bounding boxes from the KITTI dataset and those trained using the identified bounding boxes with the object detector in the proposed framework. This study found that the latter showed better performance. This is the reason why our models were trained using the identified bounding boxes.

Table 6. Evaluation results of the eXtreme Gradient Boosting (XGBoost), Random Forest (RF), and Long Short-Term Memory (LSTM) models trained using the ground truth bounding boxes and those trained using the identified bounding boxes for distance prediction in terms of mean absolute error (MAE) using the testing dataset.

Model	MAE (m)	
	Trained Using Ground Truth Bounding Boxes	Trained Using Identified Bounding Boxes
XGBoost	1.5130	1.2194
LSTM	1.6205	1.1658
RF	1.5295	1.3134

Table 7 shows the performance comparison results for the KITTI dataset between the proposed framework and various other methods, including one stereo camera-based study that used stereo image pairs to train their network. The proposed framework outperformed the other studies in terms of the five measurements.

Table 7. Performance comparison of the proposed framework with other methods for the Karlsruhe Institute of Technology and Toyota Institute (KITTI) dataset.

Studies	Camera Type	Error Metric				Accuracy Metric		
		AbsRel	SquaRel	RMSE	RMSE log	δ	δ	δ
Zhou et al. [19]	Monocular	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yin and Shi [11]	Monocular	0.147	0.936	4.348	0.218	0.810	0.941	0.977
Liang et al. [6]	Monocular	0.101	0.715	NA	0.178	0.899	0.981	0.990
Shu et al. [14]	Monocular	0.088	0.712	4.137	0.169	0.915	0.965	0.982
Guizilini et al. [13]	Monocular	0.078	0.420	3.485	0.121	0.931	0.986	0.996
Ding et al. [5]	Stereo	0.071	NA *	3.740	NA	0.934	0.979	0.992
Ours	Monocular	0.047	0.116	2.091	0.076	0.982	0.996	1.000

* NA: not available; AbsRel: absolute relative error; SquaRel: squared relative difference; RMSE: root mean squared error.

Table 8 shows the on-road evaluation results of the proposed distance prediction framework. Compared with other studies [7,42], the proposed framework showed the best performance at different distance levels. The time required for distance prediction by the proposed framework was approximately 0.3 sec per frame. Kim [7] reported that the processing time was 0.76 sec per frame.

Table 8. On-road evaluation results of the proposed distance prediction framework at different distance levels.

Distance (m)	Accuracy (%)		
	Proposed Framework	Kim [7]	Kumar et al. [42]
10	98.33	98.0	NA
20	98.67	92.2	NA
30	98.44	91.7	98.02
40	99.50	91.3	NA
50	97.52	91.2	96.32
60	97.47	NA *	NA
70	93.19	NA	NA
80	96.33	NA	95.89

* NA: not available.

A potential limitation of the proposed distance prediction framework is that the accuracy of the distance predictor depends on the accuracy of the object detector and that of the depth estimator. For the distance predictor, this study suggests that if the detected object

is a car or a bicycle, then the XGBoost model is used for distance prediction; otherwise, the LSTM model is used. It is possible that a non-car or non-bicycle object could be falsely detected as a car or a bicycle. In that case, the accuracy of distance prediction could be slightly affected.

To use our proposed framework in a vehicle, any webcam can be used, since the proposed framework does not require a high-end webcam. The webcam needs to be mounted so it is approximately level with the ground plane. The webcam can be mounted at the head of the vehicle. In this case, no calibration is needed. The webcam can be mounted on the platform above the center console of a vehicle or attached to the top of the windshield of the vehicle as well. In this case, simple calibration is needed. The horizontal distance between the head of the vehicle and the camera needs to be measured. The measurement can be performed within one minute using a tape measure or any other distance measurement tools. Then, the measured distance can be input into the proposed framework to subtract the measured distance from the predicted distance, and thus the proposed framework can provide the distance between the front vehicle and the head of the driver's vehicle. A smartphone can be used instead of a webcam for video recording. In this case, the one-site video stream can be sent to a cloud server with the proposed framework installed for distance prediction, and then the predicted distance can be sent back to the smartphone for driving assistance.

ADAS plays a more and more important role in preventing deaths and injuries by decreasing the number of car accidents. Typical ADAS features include adaptive cruise control, forward collision warnings, automatic emergency braking (AEB), pedestrian AEB, rear AEB, lane keeping assistance, blind spot warnings, parking sensor ADAS, and rearview camera ADAS. Based on each ADAS feature, its sensors are mounted at different locations of a vehicle, including the top of the front windshield, the lower front bumper, and the front, rear, and sides of a vehicle. In the U.S., 92.7% of new vehicles had at least one ADAS feature in 2018 [43]. Distance prediction between a driver's vehicle and its surroundings is an essential task for ADAS. The proposed framework can be used for accomplishing the distance prediction task.

5. Conclusions

The proposed framework estimates the distances between one's vehicle and the objects in front of the vehicle from an image captured by a webcam mounted in the vehicle. The object detector in the proposed framework detects the classes and bounding boxes of the objects. The depth estimator in the proposed framework estimates the depth map of the captured image. The depth map is overlaid with the bounding boxes to extract the depth features for each object. If the object is a car or a bicycle, then the XGBoost model is used for predicting the distance between the camera and the object, based on the bounding box and depth features of the object. Otherwise, the LSTM model is used.

In the on-road experiment, the accuracy of the proposed framework for distance estimation was 93.19–99.50% at different distance levels. The processing time was 0.3 sec per frame. The proposed framework outperformed the existing studies in terms of accuracy and efficiency. A limitation of this work is that the experiment was conducted on a wide road without many cars in order to mark the ground truth distances. For future work, the proposed framework needs to be comprehensively evaluated in various road conditions.

Author Contributions: Conceptualization, S.L. and K.H.; methodology, S.L. and K.H.; software, S.L., K.H. and S.P.; validation, S.L. and K.H.; formal analysis, S.L. and K.H.; investigation, S.L., K.H., S.P. and X.Y.; resources, X.Y.; data curation, S.L. and K.H.; writing—original draft preparation, S.L., K.H., S.P. and X.Y.; writing—review and editing, X.Y.; visualization, S.L., K.H., S.P. and X.Y.; supervision, X.Y.; project administration, X.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Oberhammer, J.; Somjit, N.; Shah, U.; Baghchehsaraei, Z. RF MEMS for automotive radar. In *Handbook of Mems for Wireless and Mobile Applications*; Uttamchandani, D., Ed.; Woodhead Publishing Ltd.: Cambridge, UK, 2013; pp. 518–549.
2. Ali, A.; Hassan, A.; Ali, A.R.; Khan, H.U.; Kazmi, W.; Zaheer, A. Real-Time Vehicle Distance Estimation Using Single View Geometry. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Snowmass Village, CO, USA, 1–5 March 2020; pp. 1111–1120.
3. Khader, M.; Cherian, S. *An Introduction to Automotive LIDAR*; Technical Report; Taxes Instruments Incorporated: Dallas, TX, USA, 2018.
4. Raj, T.; Hashim, F.H.; Huddin, A.B.; Ibrahim, M.F.; Hussain, A. A survey on LiDAR scanning mechanisms. *Electronics* **2020**, *9*, 741. [CrossRef]
5. Ding, M.; Zhang, Z.; Jiang, X.; Cao, Y. Vision-based distance measurement in advanced driving assistance systems. *Appl. Sci.* **2020**, *10*, 7276. [CrossRef]
6. Liang, H.; Ma, Z.; Zhang, Q. Self-supervised object distance estimation using a monocular camera. *Sensors* **2022**, *22*, 2936. [CrossRef] [PubMed]
7. Kim, J.B. Efficient vehicle detection and distance estimation based on aggregated channel features and inverse perspective mapping from a single camera. *Symmetry* **2019**, *11*, 1205. [CrossRef]
8. Tram, V.T.B.; Yoo, M. Vehicle-to-vehicle distance estimation using a low-resolution camera based on visible light communications. *IEEE Access* **2018**, *6*, 4521–4527. [CrossRef]
9. Kim, G.; Cho, J.S. Vision-Based Vehicle Detection and Inter-Vehicle Distance Estimation. In Proceedings of the International Conference on Control, Automation and Systems, Jeju, Republic of Korea, 17–21 October 2012.
10. Liu, L.C.; Fang, C.Y.; Chen, S.W. A novel distance estimation method leading a forward collision avoidance assist system for vehicles on highways. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 937–949. [CrossRef]
11. Yin, Z.; Shi, J. GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 1983–1992.
12. Song, Z.; Lu, J.; Zhang, T.; Li, H. End-to-end Learning for Inter-Vehicle Distance and Relative Velocity Estimation in ADAS with a Monocular Camera. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 1–17 June 2020.
13. Guizilini, V.; Ambrus, R.; Pillai, S.; Raventos, A.; Gaidon, A. 3D Packing for Self-Supervised Monocular Depth Estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2485–2494.
14. Shu, C.; Yu, K.; Duan, Z.; Yang, K. Feature-metric Loss for Self-supervised Learning of Depth and Egomotion. In Proceedings of the 16th European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020.
15. Zhang, Y.; Ding, L.; Li, Y.; Lin, W.; Zhao, M.; Yu, X.; Zhan, Y. A regional distance regression network for monocular object distance estimation. *J. Vis. Commun. Image Represent.* **2021**, *79*, 103224. [CrossRef]
16. Zhu, J.; Fang, Y. Learning Object-Specific Distance from a Monocular Image. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3839–3848.
17. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep ordinal regression network for monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 2002–2011.
18. Xu, D.; Ricci, E.; Ouyang, W.; Wang, X.; Sebe, N. Multi-Scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5354–5362.
19. Zhou, T.; Brown, M.; Snavely, N.; Lowe, D.G. Unsupervised Learning of Depth and Ego-Motion from Video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6612–6619.
20. Kreuzig, R.; Ochs, M.; Mester, R. DistanceNet: Estimating Traveled Distance from Monocular Images using a Recurrent Convolutional Neural Network. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–20 June 2019.
21. Felzenszwalb, P.; McAllester, D.; Ramanan, D. A Discriminatively Trained, Multiscale, Deformable Part Model. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
22. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.

23. Viola, P.; Jones, M. Rapid Object Detection Using a Boosted Cascade of Simple Features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Kauai, HI, USA, 8–14 December 2001.
24. Cai, Z.; Vasconcelos, N. Cascade R-CNN: Delving Into High Quality Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
25. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS 2015), Montreal, QC, Canada, 7–12 December 2015.
26. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
27. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
28. Law, H.; Deng, J. CornerNet: Detecting Objects as Paired Keypoints. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 734–750.
29. Zhu, C.; He, Y.; Savvides, M. Feature Selective Anchor-Free Module for Single-Shot Object Detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 840–849.
30. Duan, K.; Bai, S.; Xie, L.; Qi, H.; Huang, Q.; Tian, Q. CenterNet: Keypoint Triplets for Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6569–6578.
31. Zhang, G.; Luo, Z.; Cui, K.; Lu, S. Meta-DETR: Few-Shot Object Detection via Unified Image-Level Meta-Learning. *arXiv* **2021**, arXiv:2103.11731.
32. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 213–229.
33. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Proceedings of the 2017 Conference on Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
34. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv* **2020**, arXiv:2002.05709.
35. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
37. Kim, D.; Ka, W.; Ahn, P.; Joo, D.; Chun, S.; Kim, J. Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth. *arXiv* **2022**, arXiv:2201.07436.
38. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
39. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
40. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
41. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
42. Kumar, G.A.; Lee, J.H.; Hwang, J.; Park, J.; Youn, S.H.; Kwon, S. LiDAR and camera fusion approach for object distance estimation in self-driving vehicles. *Symmetry* **2020**, *12*, 324. [CrossRef]
43. ADAS Statistics: BSW, LDW, ACC & LKA. Available online: https://caradas.com/adas-statistics/ (accessed on 17 November 2022).