

Article

End-to-End Monocular Range Estimation for Forward Collision Warning

Jie Tang  and Jian Li *

College of Intelligence Science, National University of Defense Technology, Changsha 410073, China;
kakaxi314@nudt.edu.cn

* Correspondence: lijian@nudt.edu.cn

Received: 11 September 2020; Accepted: 16 October 2020; Published: 21 October 2020



Abstract: Estimating range to the closest object in front is the core component of the forward collision warning (FCW) system. Previous monocular range estimation methods mostly involve two sequential steps of object detection and range estimation. As a result, they are only effective for objects from specific categories relying on expensive object-level annotation for training, but not for unseen categories. In this paper, we present an end-to-end deep learning architecture to solve the above problems. Specifically, we represent the target range as a weighted sum of a set of potential distances. These potential distances are generated by inverse perspective projection based on intrinsic and extrinsic camera parameters, while a deep neural network predicts the corresponding weights of these distances. The whole architecture is optimized towards the range estimation task directly in an end-to-end manner with only the target range as supervision. As object category is not restricted in the training stage, the proposed method can generalize to objects with unseen categories. Furthermore, camera parameters are explicitly considered in the proposed method, making it able to generalize to images taken with different cameras and novel views. Additionally, the proposed method is not a pure black box, but provides partial interpretability by visualizing the produced weights to see which part of the image dominates the final result. We conduct experiments to verify the above properties of the proposed method on synthetic and real-world collected data.

Keywords: end-to-end learning; range estimation; forward collision warning; convolutional neural networks

1. Introduction

Range estimation, estimating distance from ego to the closest object in front, is the core component of the forward collision warning (FCW) system. Based on the estimated range and time to collision which can be computed by ranges in multi-frame, FCW can be made for automobile robot and advanced driver assistant systems (ADAS) to avoid potential collisions [1–3]. Although radar or LiDAR sensors can provide accurate range measurements, they are too heavy and expensive for small robots and ADAS. Estimating range from a single monocular image is a convenient and cheap solution, which is favored and widely adopted by commercial products, such as the famous Mobileye's ADAS (<https://www.mobileye.com/our-technology/adas/>). In addition to the usage in industrial applications, predicting range from a monocular image is a challenging ill-posed problem with great value for academic research.

Due to the ambiguity of distance estimation from a monocular image, especially on the global scale [4], geometry relationship of perspective projection is often used to facilitate this estimation. Two cues are mostly considered in traditional approaches. One is the known vehicle size, in which the object category is constrained as a vehicle with a fixed size. According to the size of the detected vehicle in the image, the target range can be inferred through perspective projection [5–7]. However,

considering different subcategories of vehicles like car, SUV, bus, and truck, the size of a vehicle has a large variance that could lead to a large error in the estimated range. The other cue is the planar road surface assumption. Typically, an object is detected in the image and the original image is converted to an Inverse Perspective Mapping (IPM) image to achieve distance along the forward direction [8–11]. Compared with the known vehicle size, planar road surface assumption is more general and presents more accurate results [12]. In general, all these traditional approaches need to detect the object first and then estimate the distance to this object by utilizing perspective projection with the chosen cue.

Recently, deep learning methods have achieved great success in the computer vision community. For range estimation, prior works [13–15] have proposed object-specific end-to-end deep learning frameworks. They follow the multi-task learning scheme to simultaneously detect objects and estimate the corresponding range for each object by direct regression. Compared to separately hand-craft designing or optimizing each step in traditional approaches, end-to-end manner of deep learning methods can train all components jointly with the learned features adapting to the task of interest and achieve superior performance. However, like traditional approaches, they can only estimate ranges for objects from some limited given categories, which still does not fulfill the requirement of FCW to respond to various objects (including unseen categories in the training stage).

In contrast to previous methods of mostly estimating ranges to all the specific objects in the image, our method is class-agnostic and only yields range to the closest object in the preset collision region, which is the real requirement of FCW. We express the target range as a weighted sum of a set of potential distances. These distances are from a distance map generated by inverse perspective projection based on intrinsic and extrinsic camera parameters, while the corresponding weights are from the weight map produced by a deep neural network. Unlike prior work [13] where the range is directly regressed with camera information ignored when deploying, camera parameters are explicitly embedded in our distance map generation. As demonstrated by [16–18] that explicitly implementing camera parameters into network can improve generalization capability, while conventional convolutional neural network (CNN) based direct regression methods tend to overfit dataset that can not generalize to new cameras and novel views. At the same time, the whole architecture of our method is differentiable, which also enables the merits of end-to-end learning.

Towards this end, we propose a novel network structure to predict the corresponding weights of the generated distances for range estimation. Specifically, following the planar road surface assumption, we first utilize inverse perspective projection to generate a distance map, in which each pixel representing the distance to a point on road surface. Then, the weight map is generated by a U-Net structure with the same resolution as the distance map. Considering that the convolutional layer can only provide spatially-agnostic operation in local, it is difficult for a CNN structure to learn the point that what we want is the range to the closest object, when multiple objects are in front. Therefore, we use fully convolutional networks as the encoder and decoder parts of the U-Net structure and employ fully connected layers on the encoded features to provide spatially-specific operations and global receptive field. Finally, we mask both the distance map and weight map to just remain pixels in the preset collision region to estimate range as a weighted sum of distances. The whole network is trained in an end-to-end manner with only the target range as supervision, which avoids expensive object-level annotation such as class label and bounding box and encourages the network to be class-agnostic as well.

We process a synthetic dataset to build a dataset with accurate ground truth range for training and testing. In addition to synthetic data, we also collect data in the real world by our autonomous driving vehicle to verify our method. Our method can generalize to objects of unseen categories, different cameras, and novel views. It also presents superior performance when compared with an IPM-based two-step method. Moreover, our method is not a pure black box but provides partial interpretability because the produced weight map can easily be visualized to indicate which part of the image dominates the estimated range.

2. Approach

2.1. Overview

Our method is an end-to-end deep learning architecture that directly estimates range to the closest object in front for forward collision warning (FCW). The pipeline of the proposed method is illustrated in Figure 1. Based on the planar road surface assumption, we use inverse perspective projection to convert the U, V coordinate maps of an image to a distance map, in which each pixel value corresponds to the distance to the road surface along the forward direction. The range to be estimated is formalized as a weighted sum of these distances. The corresponding weights are generated from a weight generation network with a single image and the mask of the collision region as input. The whole network is differentiable and can be supervised only by the ground truth range.

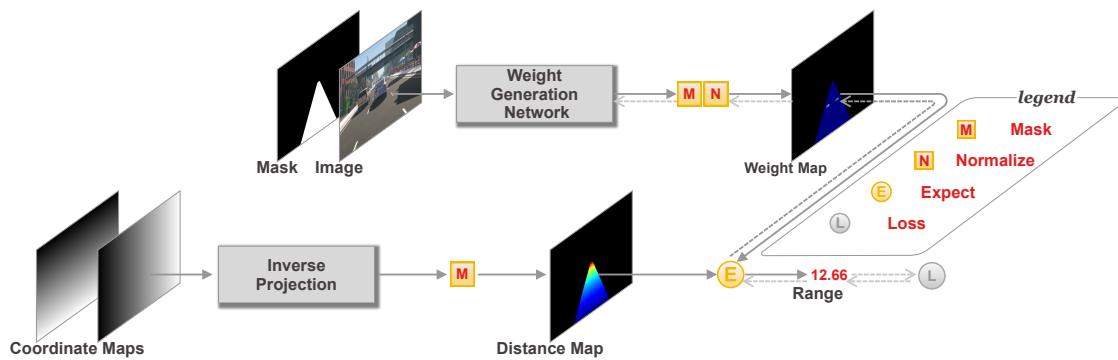


Figure 1. Overview of architecture. The range is represented as a weighted sum of a set of potential distances and the whole architecture consists of weight map generation and distance map generation. The solid lines with arrows represent the forward implementation of our method, while the dashed lines with arrows indicate the loss calculation and back-propagation in the training stage.

2.2. Distance Map Generation

Estimating distance from a single monocular image is severely under-constrained. The estimated distance would have inherent ambiguity, especially on the global scale. Some assumptions are needed to infer distance directly from image coordinates. Our method follows the planar road surface assumption to generate a distance map through inverse perspective projection. The planar road surface assumption assumes that the road surface is a plane and objects are on the road, which is general and has no restriction on the category of the object.

Under the pinhole camera model, the 3D scene is mapped onto a 2D image plane. Thus, each pixel $(u, v)^T$ in an image corresponds to a point $(x, y, z)^T$ in 3D space. The camera coordination system and the world coordinate system in our setting are illustrated in Figure 2. The center point of the world coordinate system is on the road plane and straightly beneath the camera with Z axis perpendicular to the road surface. The correspondence relationship between these two coordinate systems can be expressed by perspective projection:

$$s \cdot (u, v, 1)^T = K \cdot [R \mid T] \cdot (x, y, z, 1)^T, \quad (1)$$

where K and $[R \mid T]$ are intrinsic and extrinsic camera parameters respectively and s is the scalar projective parameter. With the planar road surface assumption, for all the 3D points on the road surface, coordinates z in the world coordinate system equal to 0, which can be removed from Equation (1). Then,

$$s \cdot (u, v, 1)^T = P \cdot (x, y, 1)^T. \quad (2)$$

Here, P is a 3×3 projection matrix. For each pixel $(u, v)^T$ in the image coordinate, we use inverse projection to calculate the corresponding $(x, y)^T$, which indicates a point on the road surface with x representing the distance along the forward direction.



Figure 2. Our autonomous driving vehicle with image coordinate system, LiDAR coordinate system, and world coordinate system in our setting.

Some traditional methods [8–10] use perspective projection to convert the original image to an image in bird’s eye view which is usually called an Inverse Perspective Mapping (IPM) image. Then the distance information is obtained from this interpolated IPM image. In contrast, the proposed method works on the original image plane to generate a distance map with the same resolution as the original image. One example of the generated distance map is visualized in Figure 1. The value on each pixel of distance map is the calculated x from the inverse perspective projection, representing the distance to the road surface. As we are only concerned with the object in front of the ego vehicle, which may cause a forward collision, we set a potential collision region to mask the distance map by limiting the y coordinate according to the width of the ego vehicle and the x coordinate according to the farthest distance we care about. Finally, the masked distance map consists of a set of potential distances for the target range. More details of the mask generation are introduced in Section 3.

For this distance map generation, even though the required intrinsic and extrinsic camera parameters can be offline calibrated, the vehicle motion may cause perturbation on the extrinsic parameters. The extrinsic parameters can also be estimated on-the-fly. For example, in [19], the extrinsic parameters are estimated based on the parallel lane detection. How to obtain accurate extrinsic parameters for inverse projection is out of the scope of this paper. In our method, we assume we have both the accurate intrinsic and extrinsic camera parameters.

2.3. Weight Map Generation

We consider the target range as a weighted sum of a set of distances within the masked region. Thus, the range is obtained by dot product of the distance map and a weight map. The weight map is generated by a deep neural network with the same resolution as the distance map. In principle, any off-the-shelf dense prediction architecture can be adopted for such weight map generation. In our method, as shown in Figure 3, we employ a novel U-Net structure network to generate the weight map by extracting features on multi-resolution.

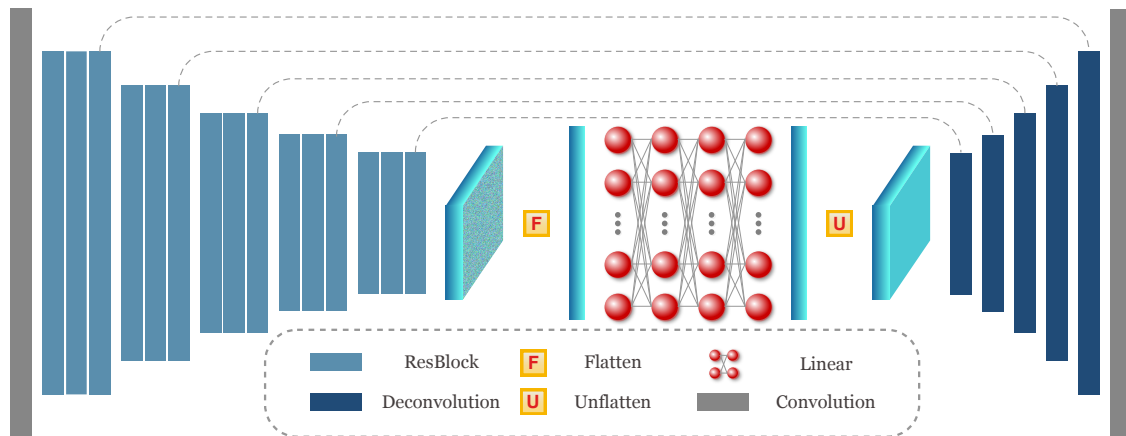


Figure 3. The network structure of weight generation network. The U-Net structure is an encoder-decoder network. The encoder and decoder parts consist of fully convolutional networks. Fully connected layers are applied to the flattened encoded features to provide spatial position information. See the text for more details.

Typically, the U-Net structure network [20] is an encoder-decoder network. We adopt fully convolutional structures for the encoder and decoder parts. For the encoder part, we firstly apply a 5×5 convolution layer on the input images. Then, each time a convolution layer with the stride of 2 is adopted to reduce the resolution of the feature map. For each resolution, 3 stacked ResBlocks [21] are used to extract features. The ResBlock consists of two sequential 3×3 convolution layers with a skip layer. The resolution of the encoded feature is $1/32$ of the original resolution. For the decoder part, we use a sequence of deconvolution layers with the stride of 2 to increase the resolution of the feature map. Finally, a 1×1 convolution layer is used to map the feature map to single-channel output. In this U-Net structure, for each resolution, we employ an additional skip connection to fuse the feature maps of the same resolution from the encoder to the decoder. Each convolution or deconvolution layer is followed by a batch normalization [22] and a ReLU [23] layer. As shown in Figure 1, we concatenate the color image and a mask of preset collision region as the input of the U-Net structure. The mask added here is to force the network to pay attention to the region of interest.

As it is possible to have multiple objects in front, the network needs to learn the point that the desired range is the one to the closest object. However, this point is difficult to learn by a fully convolutional network [24], since the parameter sharing mechanism of convolution layer can only provide spatially-agnostic operation with local receptive field. To make the network aware of the spatial position information, we apply three fully connected layers which can assign unique parameters for each spatial position on the encoded feature. For the encoded feature map, we keep the channel dimension unchanged and only flatten the spatial dimension, i.e., the height and width dimension. Then the linear operation of the fully connected layer works on the flattened feature and is shared among channels. By sharing parameters among channels, these fully connected layers can introduce spatial position information without causing too many parameters. In addition to spatial position information, these fully connected layers also provide global perspective field for the network. We apply a dropout [25], layer normalization [26], and ReLU [23] layers after each fully connected layer. The produced features from these fully connected layers are reshaped back (unflattened) to feature map with 2D spatial dimensions and fed to the decoder part of the network.

In our method, these generated weights work as probabilities of corresponding distances for the expected range. Thus, these weights should be non-negative and their sum should be equal to 1. We adopt a softplus [27] layer on the single-channel output of the U-Net structure to yield a weight map with all the values positive. The weight map is then masked by the preset collision region. Finally, we normalize the masked weights to make them suitable as probabilities.

2.4. End-to-End Learning

Our method learns the range from a single image in an end-to-end manner. This end-to-end learning is optimized by minimizing the difference between estimation and ground truth range. Unlike previous end-to-end object-specific distance estimation methods [13–15] that require expensive object-level annotations as supervision, our method is only trained with ground truth range and has no restriction on object category, which makes it possible to generalize to objects of unseen categories. Meanwhile, we have no direct supervision on the predicted weight map. The weight map is generated by the weight generation network which is only supervised to produce accurate range estimation. Since the weight map is a single channel image, as shown in Figure 1, it can easily be visualized to see which part of the image dominates the estimated range. Thus, our method is not a pure black box but has partial interpretability.

2.5. Training Settings

We adopt mean absolute error (MAE) as the training loss to train our range estimation network. We process a synthetic dataset to build a dataset with accurate ground truth range for training. In total, about 170k samples are used for training. More details about the dataset preparation are introduced in Section 3.1. Color image with bottom-center cropped 320×960 resolution and the corresponding mask of the preset collision region are used as the input of our network. The code is implemented in Pytorch and trained from scratch in a GPU server with 8 Nvidia GTX 2080Ti GPUs. The batch size on each GPU is 8 and the batch norm layers in our network are synchronized across GPUs. We utilize ADAM [28] as the optimizer with an original learning rate of 10^{-3} and weight decay of 10^{-6} . The network is trained for 80 epochs while the learning rate drops to half at the 40th and 60th epochs.

3. Experiment Setup

One of the properties of data-driven methods especially deep learning methods is data-hungry. A large dataset is usually necessary for training. However, it is difficult to obtain accurate range information in the real world, especially considering that it must satisfy the planar road surface assumption adopted in our method. To have a large enough dataset with little noise and error on ground truth range, we process a synthetic dataset to build a dataset for training and testing. We also collect data in the real world by our autonomous driving vehicle with ground truth range from LiDAR to verify our method.

3.1. Synthetic Dataset

The synthetic Apollo dataset (<https://apollo.auto/synthetic.html>) is one of the largest photo-realistic synthetic datasets for autonomous driving currently. The dataset is created by Unity 3D engine with various distinct virtual scenes, including highways, urban, residential, downtown, and indoor parking garage. The dataset also contains plenty of environmental variations, such as different times of day, different weather conditions, different objects, and varied road surface qualities. In addition to the synthesized color image, it provides extensive ground truth data, such as 2D/3D object data, semantic/instance-level segmentation, depth, and 3D lane line data. We choose data in good light conditions (data with time at 9:00, 13:00, and 14:00) and good road surface quality (referred to as ‘No Degradation’ in this dataset) for our experiment.

Camera parameters are required by our method. The intrinsic camera parameters are provided by the dataset. We calculate the extrinsic parameters, which transfer the coordinate from the world coordinate system to the camera coordinate system shown in Figure 2. In particular, we first find the depths of all the pixels whose semantic labels are ‘road’ from the depth image and semantic segmentation image. Based on camera projection, we convert these pixels with depth to a set of 3D points. Then we use RANSAC to fit a plane from these 3D points. Based on the normal of the road plane, we calculate the pitch and roll angles and the height to the road plane of the ego camera. Finally,

we use these angles and height to build the $[R \mid T]$ matrix as the extrinsic camera parameters. In this process, we remove the image from the dataset if we find these 3D points have a sufficient number of outliers of the fitted plane, which indicated that the sample violates the planar road surface assumption.

The target range to the closest object depends on the preset collision region. The collision region is a rectangle space on the road, whose width is the width of the ego vehicle, and height is the farthest distance concerned. The rectangle is located in front of the ego vehicle, and the position may have an offset angle to the forward direction of the camera due to the yaw angle between the camera and vehicle. We project the collision region into image plane to get the potential collision mask. Thus, the collision region is decided by the width of the ego vehicle, the farthest distance concerned, and the offset angle to the forward direction of camera. For this synthetic Apollo dataset, we randomly adjust these three parameters (the vehicle width from 1.5 m to 2.5 m, the farthest distance from 80 m to 90 m, the offset angle from -10° to $+10^\circ$) to provide different collision regions. In this way, a single image may have different target ranges corresponding to different collision regions. It can be seen as data augmentation and forces our method to be robust to different masks.

In the collision region, we take the closest distance among the pixels whose semantic labels are not ‘road’ as the target range. As there are confusing situations in the synthetic Apollo dataset, e.g., leaves on the road, which may not cause a forward collision, we remove the sample from the dataset if the semantic label of this closest-distance pixel is ‘vegetation’. We also remove the sample from the dataset if the semantic label of this closest-distance pixel is ‘terrian’, which may be just a small bump on the road. In Figure 4, we present some examples from the generated dataset. The closest-distance pixel is marked with a red dot. In total, there are 190k samples with color images, camera parameters, collision regions, and ground truth ranges. We randomly select 90% samples as the train set and the remaining as test set.

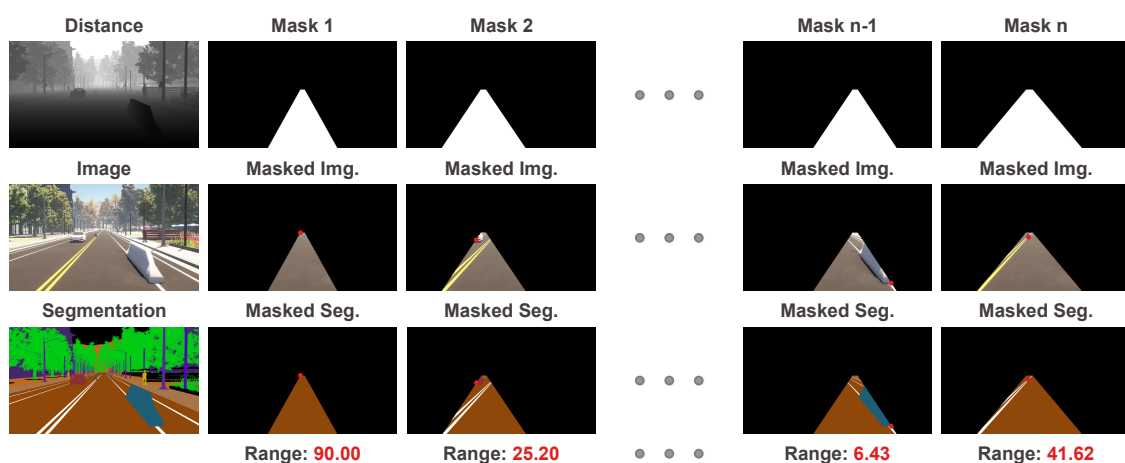


Figure 4. Processed Apollo dataset. These masks represent different preset collision regions. Even for the same color image, different collision regions may induce different target ranges.

3.2. Real-World Data Collection

We also collect data in the real world by our autonomous driving vehicle to verify our method. Our autonomous driving vehicle has one 12 mm lens camera and one 128-line LiDAR mounted as shown in Figure 2. The camera and LiDAR are synchronized with a trigger signal to guarantee LiDAR measures range in the forward direction when the image is captured by the camera. In order to obtain measurements as accurately as possible, our ego vehicle is parked on a flat road to collect data and avoid the effect of ego-motion. In the process of data collection, the scene in front is almost fixed or dynamic objects move at a very low speed. The intrinsic and extrinsic camera parameters (camera coordinate system to LiDAR coordinate system and camera coordinate system to world coordinate system) were calibrated before data collection. The collision region for the data collected by our

autonomous driving vehicle is determined by the vehicle width of 1.8 m, the farthest distance of 85 m, and the offset angle of 0° .

4. Results and Analyses

Our method produces range to the closest object in the front preset collision region for FCW. We demonstrate the properties of our method and compare its performance with a traditional IPM-based method.

4.1. Interpretability

Our method is not a pure black box but has partial interpretability. The estimated range is produced as a weighted sum by a distance map and weight map. The weight map, which is generated by a U-Net structure, is a single-channel image of the same resolution with the input color image. It works as a probability map to yield the expected range. By visualizing the generated weight map, we can take a look at which part of the image the weight map focused on to produce the estimated range.

We illustrate several examples in Figure 5. It is interesting that even no direct object-level supervision such as bounding box is used, our network successfully learns to give a high response on the intersection region of the target object and road surface. Compared to the other part in the preset collision region (the green region in the last row), the distances in the intersection region of the distance map are closer to the target range. Thus, it is reasonable to give larger weights on this intersection region for range estimation. From this view, our method shares the same philosophy with traditional IPM-based methods that the intersection region of the target object and road surface is found for range estimation. However, in contrast to achieving the intersection region by object detection in traditional IPM-based methods, our method implicitly learns to give larger weights on this intersection region in an end-to-end manner towards the true range estimation task.

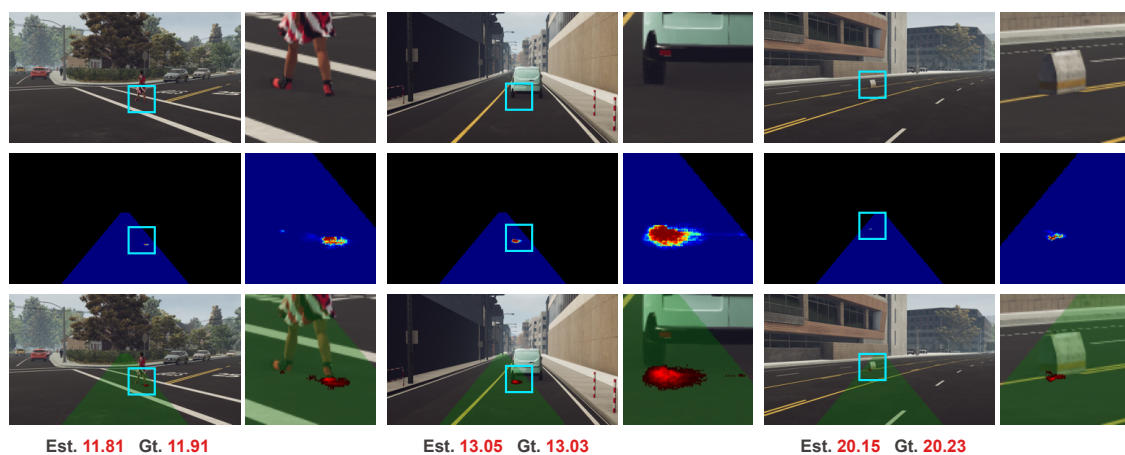


Figure 5. Visualization of generated weight maps. The first row is the color image. The second row is the colorized weighted map. The last row is the color image superimposed on the preset collision region and the colorized weighted map. This interpretable representation of the last row will be used to illustrate experimental results in the following. We use cyan rectangles to highlight the notable regions and zoom in these regions for better visualization.

4.2. Class-Agnostic Property

Previous monocular range estimation methods mostly involve two sequential steps of object detection and range estimation that can only estimate ranges for objects from some limited given categories. Our method is only supervised by the target range and has no restriction on the object category. This encourages our method to be class-agnostic, which can work on objects of various

categories, even unseen categories in the training stage. In Figure 6, we illustrate the mean relative absolute error (MRAE) on the test set. We can see that our method performs well on objects from various classes.

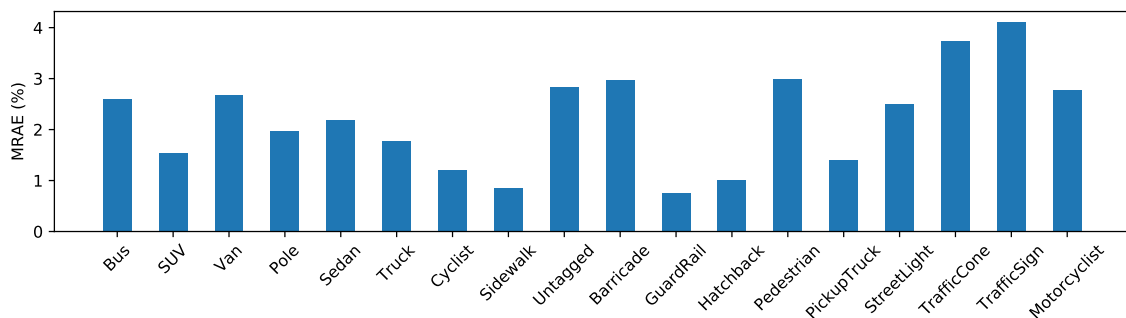


Figure 6. Performance on target objects of various classes.

We illustrate some results in Figure 7. The top row is the results of images from the test set of the processed synthetic Apollo dataset. Our method works well on these examples which are more flexible and rarer than typical objects such as cars and pedestrians. The bottom row shows the results on some real-world collected data. These objects, including fire-extinguisher, cart, and suitcase, are categories that rarely appear in traffic scenes and are not included in the train set. We can see our method can still work on these unseen categories, which demonstrates the class-agnostic property of our method.

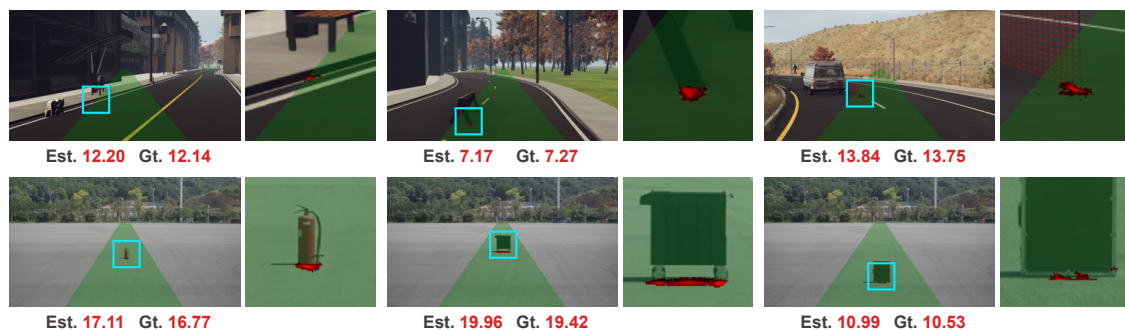


Figure 7. Results on objects of various categories.

4.3. Generalization Capability

Our method is trained on the processed synthetic Apollo dataset. As the intrinsic and extrinsic camera parameters have been explicitly embedded in the distance map generation of our method, our method can generalize to images captured from different cameras and novel views, while methods estimating range by hard regression with camera parameters ignored tends to overfit dataset and can not generalize. We selected some examples from the KITTI dataset [29] and virtual KITTI dataset [30], and show the results in Figure 8. The first row is the paired images from KITTI dataset [29] which are captured in the traffic scene by a stereo camera mounted on an autonomous driving vehicle. The following two rows are the images from virtual KITTI dataset [30], which is a synthetic dataset. The second row is two images captured by cameras mounted with left 15-degree and right 15-degree offsets to the forward direction. The two images in the last row are from cameras with left 30-degree and right 30-degree offsets. We can see our method works well on these six images, even though they are captured with different cameras and different views than images in the train set.

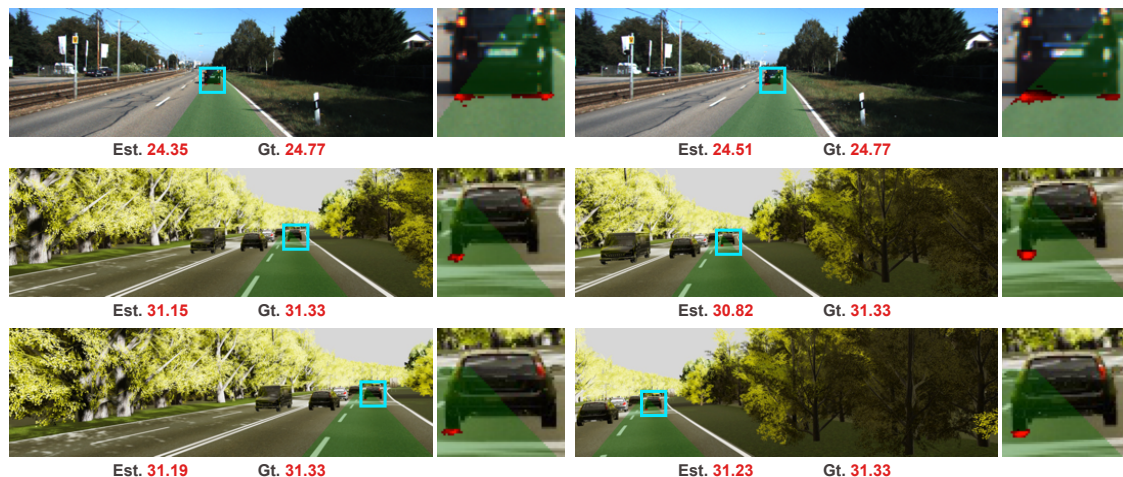


Figure 8. Generalization capability. These six images are from KITTI and virtual KITTI datasets. They are collected with different cameras and views than the training dataset.

4.4. Closest Object

The goal of our method is to estimate the range to the closest object for FCW. When there are multiple objects in the potential collision region, we expect our method to provide the range to the closest object, without being disturbed by other objects that are farther away. In order to encourage our method to learn this point, we apply fully connected layers on the encoded features to provide spatially-specific operations and global receptive field.

We test our method on some examples with multiple objects in front and show results in Figure 9. The top row is the results of one image with three different collision regions. The different collision regions contain different numbers of objects with different target ranges. We also collect data with two pedestrians walking in front shown in the bottom row. For each image, we can see from the superimposed weight maps that our method can focus on the closest object to estimate range.

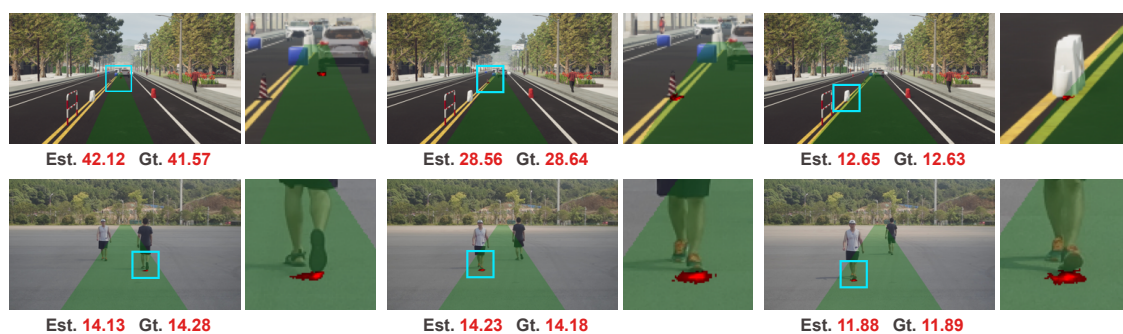


Figure 9. Results on images with multiple objects in front.

4.5. Comparison

As analyzed in Section 4.1, by visualizing the generated weight map, we find our method shares the same philosophy with the traditional IPM-based method. Thus, we compare the performance of our method and traditional IPM-based method on the test set of the processed synthetic Apollo dataset. For the IPM-based method, we first use the released code and model of [31] to detect the car in front which is one of the top-ranked 2D car detection methods in KITTI object detection benchmark (http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=2d). With the same camera parameters as our method, we then project the center point of the bottom side of the bounding box into bird's eye view coordinate and interpolate distance from the IPM image. Since the object category is constrained as car for the object detection model of [31], we only consider the samples

in the test set whose closest object in the preset collision region belongs to the classes of ‘bus’, ‘SUV’, ‘van’, ‘sedan’, ‘truck’, ‘hatchback’, and ‘pickuptruck’. We ignore the sample if the closest car is not successfully detected by [31]. We show the MAE of these two methods in Figure 10. Compared with the IPM-based method, our method has a lower MAE over all the ranges.

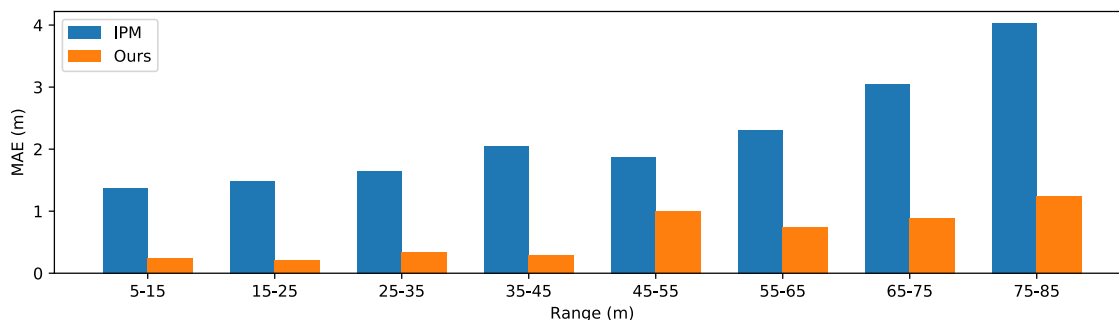


Figure 10. Performance on cars in the test set of the processed synthetic Apollo dataset.

We also collect a data sequence by our autonomous driving vehicle with a car moving in front to evaluate the performance of our method and traditional IPM-based method. Figure 11 depicts the quantitative results. The horizontal axis is the ground truth range obtained from the LiDAR sensor and the vertical axis is the absolute error between the estimate and ground truth range. We can see that compared with the IPM-based method, our method has a better performance, especially when the target car is far away.

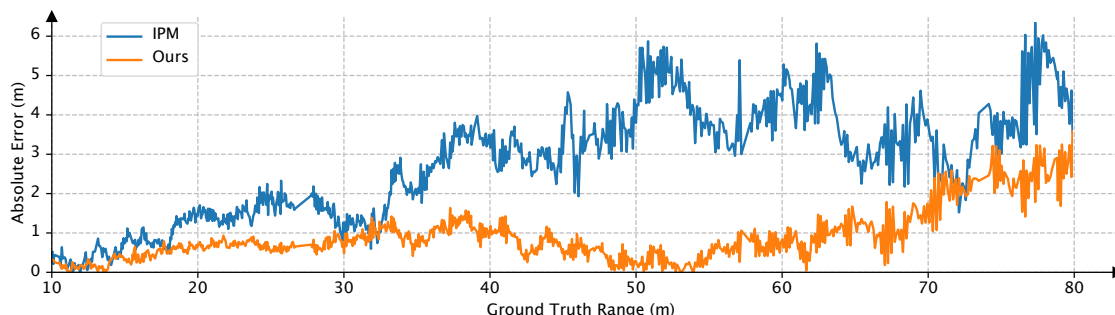


Figure 11. Performance on a data sequence collected by our autonomous driving vehicle.

We illustrate some samples of the results of our collected data in Figure 12. The top row is the results of the IPM-based method and the bottom row is the results of our method. The bounding boxes are drawn with red rectangles for the IPM-based method. We can see that even the target car is successfully detected in the IPM-based method, its performance is still worse than our method with a larger error on the estimated range. We attribute the superior performance of our method to the end-to-end learning. Compared to end-to-end learning whose optimization objective is towards the true task, the IPM based method is a two-step method with the first step optimized to accurate bounding box for object detection which is a proxy task. Object detection tends to produce relatively accurate bounding boxes, for example, satisfying the criteria of $\text{IoU} > 0.7$ in KITTI object detection benchmark. However, it is still difficult to achieve the precise intersection region of the object and road surface from the detected bounding box.

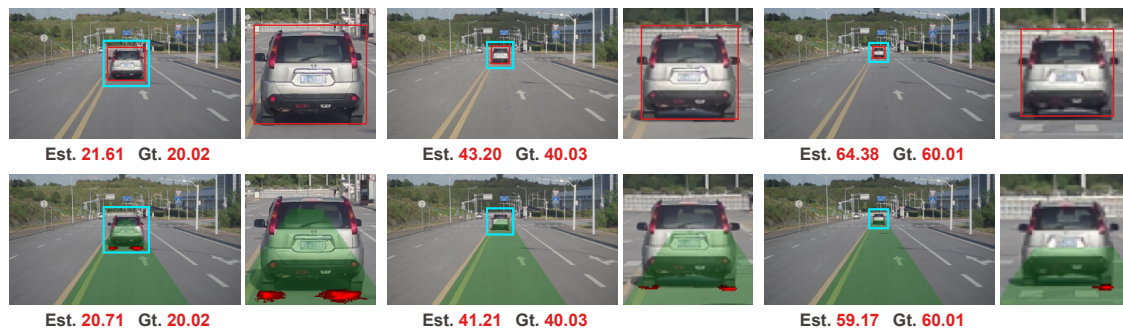


Figure 12. Samples of results in our collected sequence. The results of the Inverse Perspective Mapping (IPM)-based method are shown in the top row. The results of our method are shown in the bottom row.

4.6. Failure Cases

Our method is trained on the train set of the processed synthetic Apollo dataset. In Figure 13, we illustrate the mean absolute error (MAE) on the train set and the test set under different ranges. We can see that distant objects tend to have larger errors.

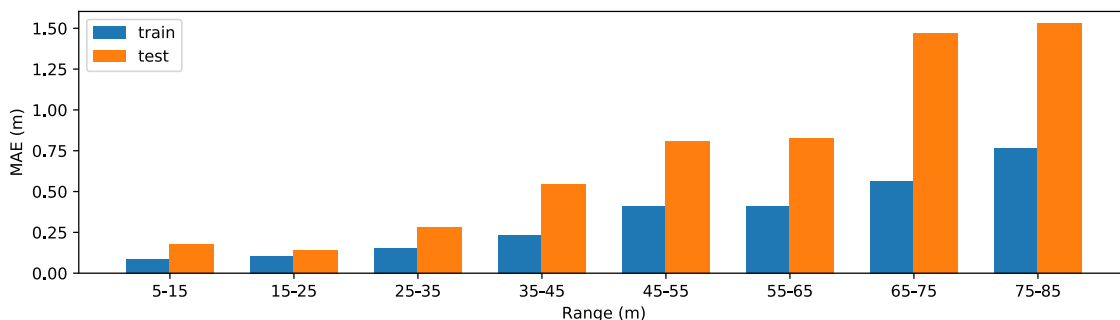


Figure 13. Mean absolute error (MAE) (in m) on the train set and test set under different ranges.

As MAE only demonstrates the mean performance of our method on the dataset, we also plot the δ of our method on the test set in Figure 14. δ means the percentage of samples where the relative error is less a threshold. We can see that for more than 98% of samples, the relative error is less than 0.1. While even the threshold of relative error goes up to 1.0, δ is still less than 100%. This suggests that there are cases in which our method completely fails.

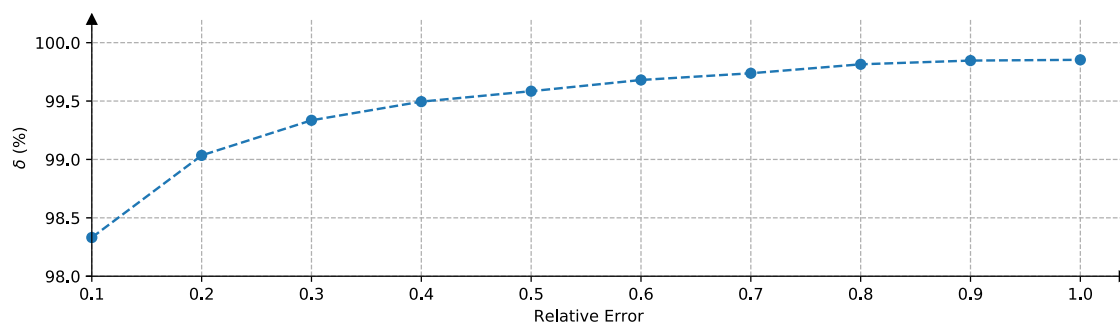


Figure 14. δ corresponding to the threshold of relative error in the test set.

Thanks to the partial interpretability of our method, we can classify and analyze these failure cases. In general, the failure cases of our method can be classified as five types. We illustrate these five types of failure cases in Figure 15. The first type is complex road surface, which may be caused by marks or shadow of trees. As shown in Figure 15a, our method is affected by the shadow on the road

surface and treats it as the target object. The second type is small distant object shown in Figure 15b. The object only occupies a few pixels and is missed by our method. The third type is that the object is almost at the boundary of the preset collision region as shown in Figure 15c. In this ambiguous position, our method is confused whether to respond to this object. The fourth type is that our method does not respond to the closest object, but a farther object. As shown in Figure 15d, it mostly occurs when the perception of the closest object is difficult, such as a very thin pole. The fifth type is that our method responds to multiple objects as shown in Figure 15e. In our method, we employ a U-Net structure network to produce the weight map. Even though we apply fully connected layers on the encoded features to provide spatial position information, the proposed method still does not have strict constraints to ensure that the generated weight map only responds to the single closest object.

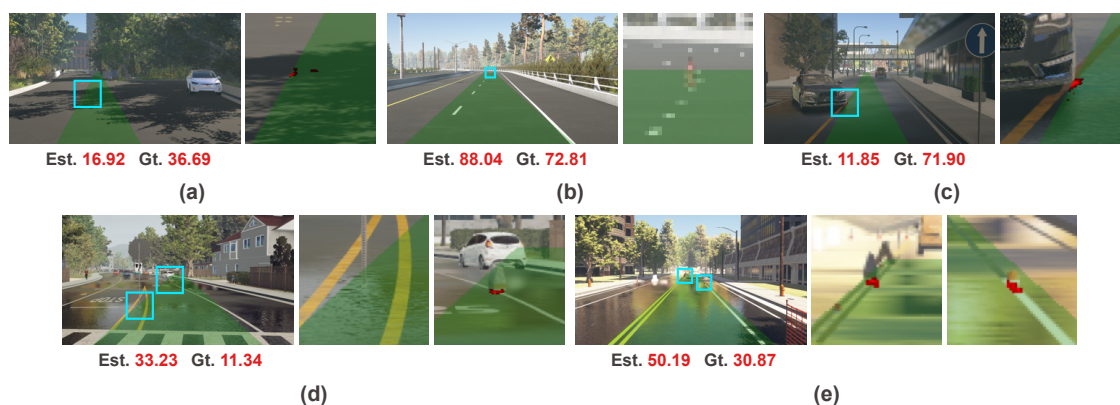


Figure 15. Examples of five types of failure cases of our method. See the text for more details.

5. Conclusions and Future Work

In this paper, we propose a novel end-to-end network to estimate the range to the closest object in front from a monocular image for forward collision warning (FCW). The target range is represented as a weighted sum of a set of potential distances. These distances are generated by inverse perspective projection under the planar road surface assumption and the corresponding weights are produced by a deep neural network. The proposed method is end-to-end trained with only the ground truth range as supervision. Experimental results on synthetic data and real world collected data show that the proposed method has several properties favored by FCW: (1) Class-agnostic property for objects of various categories; (2) generalization capability for images from new cameras and novel views; (3) partial interpretability indicating which part of the image dominates the result; and (4) superior performance to the IPM-based method.

In the future, we plan to collect a large dataset in the real world with accurate ground truth range to fine-tune our model, as the model trained by a synthetic dataset could still easily fail in a real complex scene due to the gap between these two domains. Furthermore, we assume the planar road surface and accurate extrinsic camera parameters in our method. Fitting for non-planar road and estimating extrinsic camera parameters on-the-fly even as a trainable component of the method would be interesting and meaningful extensions of this work.

Author Contributions: Conceptualization, J.T.; methodology, J.T. and J.L.; data acquisition, J.L.; software, J.T.; validation, J.T.; writing—original draft preparation, J.T. and J.L.; writing—review and editing, J.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Natural Science Foundations of China under Grant 61973311.

Acknowledgments: Research on key technologies and platforms for collaborative intelligence driven auto-driving cars was funded by the Science and Technology Development Fund, Macao S.A.R (FDCT) (Project reference number 0015/2019/AKP).

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

References

1. Van Der Horst, R.; Hogema, J. Time-to-collision and collision avoidance systems. In Proceedings of the 6th ICTCT Workshop: Safety Evaluation of Traffic Systems: Traffic Conflicts and Other Measures, Salzburg, Austria, 27–29 October 1993; pp. 109–121.
2. Dagan, E.; Mano, O.; Stein, G.P.; Shashua, A. Forward collision warning with a single camera. In Proceedings of the IEEE Intelligent Vehicles Symposium, Parma, Italy, 14–17 June 2004; pp. 37–42.
3. Chen, Y.L.; Shen, K.Y.; Wang, S.C. Forward collision warning system considering both time-to-collision and safety braking distance. In Proceedings of the 2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, Australia, 19–21 June 2013; pp. 972–977.
4. Eigen, D.; Puhersch, C.; Fergus, R. Depth map prediction from a single image using a multi-scale deep network. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2366–2374.
5. Kim, G.; Cho, J.S. Vision-based vehicle detection and inter-vehicle distance estimation. In Proceedings of the 2012 12th International Conference on Control, Automation and Systems, Jeju Island, Korea, 17–21 October 2012; pp. 625–629.
6. Wu, B.F.; Chen, Y.L.; Chen, Y.H.; Chen, C.J. Real-Time Nighttime Vehicle Detection and Recognition System Based on Computer Vision. U.S. Patent No. 7,949,190, 24 May 2011.
7. Kim, G.; Cho, J.S. Vision-based vehicle detection and inter-vehicle distance estimation for driver alarm system. *Opt. Rev.* **2012**, *19*, 388–393. [[CrossRef](#)]
8. Tuohy, S.; O’Cualain, D.; Jones, E.; Glavin, M. Distance determination for an automobile environment using inverse perspective mapping in OpenCV. In Proceedings of the IET Irish Signals and Systems Conference (ISSC 2010), Cork, Ireland, 23–4 June 2010; pp. 100–105.
9. Mallot, H.A.; Bülthoff, H.H.; Little, J.; Bohrer, S. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biol. Cybern.* **1991**, *64*, 177–185. [[CrossRef](#)] [[PubMed](#)]
10. Rezaei, M.; Terauchi, M.; Klette, R. Robust vehicle detection and distance estimation under challenging lighting conditions. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2723–2743. [[CrossRef](#)]
11. 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](#)]
12. Stein, G.P.; Mano, O.; Shashua, A. Vision-based ACC with a single camera: Bounds on range and range rate accuracy. In Proceedings of the IEEE IV2003 Intelligent Vehicles Symposium, Proceedings (Cat. No. 03TH8683), Columbus, OH, USA, 9–11 June 2003; pp. 120–125.
13. Zhu, J.; Fang, Y. Learning object-specific distance from a monocular image. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October 2019–2 November 2019; pp. 3839–3848.
14. Zhang, Y.; Li, Y.; Zhao, M.; Yu, X. A regional regression network for monocular object distance estimation. In Proceedings of the 2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), London, UK, 6–10 July 2020; pp. 1–6.
15. Zhe, T.; Huang, L.; Wu, Q.; Zhang, J.; Pei, C.; Li, L. Inter-Vehicle distance estimation method based on monocular vision using 3D detection. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4907–4919. [[CrossRef](#)]
16. Facil, J.M.; Ummenhofer, B.; Zhou, H.; Montesano, L.; Brox, T.; Civera, J. CAM-ConvS: Camera-aware multi-scale convolutions for single-view depth. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 11826–11835.
17. Zhao, Y.; Kong, S.; Fowlkes, C. When perspective comes for free: Improving depth prediction with camera pose encoding. *arXiv* **2020**, arXiv:2007.03887.
18. Dijk, T.V.; Croon, G.D. How do neural networks see depth in single images? In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October 2019–2 November 2019; pp. 2183–2191.
19. de Paula, M.B.; Jung, C.R.; da Silveira, L., Jr. Automatic on-the-fly extrinsic camera calibration of onboard vehicular cameras. *Expert Syst. Appl.* **2014**, *41*, 1997–2007. [[CrossRef](#)]
20. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241.

21. 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, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
22. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
23. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the ICML, Haifa, Israel, 21–24 June 2010.
24. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
25. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
26. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
27. Dugas, C.; Bengio, Y.; Bélisle, F.; Nadeau, C.; Garcia, R. Incorporating second-order functional knowledge for better option pricing. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 3–8 December 2001; pp. 472–478.
28. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
29. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [[CrossRef](#)]
30. Gaidon, A.; Wang, Q.; Cabon, Y.; Vig, E. Virtual worlds as proxy for multi-object tracking analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 4340–4349.
31. Ren, J.; Chen, X.; Liu, J.; Sun, W.; Pang, J.; Yan, Q.; Tai, Y.W.; Xu, L. Accurate single stage detector using recurrent rolling convolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5420–5428.

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).