Learning Object-Specific Distance From a Monocular Image

Jing Zhu Yi Fang

NYU Multimedia and Visual Computing Lab, USA
New York University, USA
New York University Abu Dhabi, UAE
{jingzhu, yfang}@nyu.edu

Abstract

Environment perception, including object detection and distance estimation, is one of the most crucial tasks for autonomous driving. Many attentions have been paid on the object detection task, but distance estimation only arouse few interests in the computer vision community. Observing that the traditional inverse perspective mapping algorithm performs poorly for objects far away from the camera or on the curved road, in this paper, we address the challenging distance estimation problem by developing the first end-to-end learning-based model to directly predict distances for given objects in the images. Besides the introduction of a learning-based base model, we further design an enhanced model with a keypoint regressor, where a projection loss is defined to enforce a better distance estimation, especially for objects close to the camera. To facilitate the research on this task, we construct the extended KITTI and nuScenes (mini) object detection datasets with a distance for each object. Our experiments demonstrate that our proposed methods outperform alternative approaches (e.g., the traditional IPM, SVR) on object-specific distance estimation, particularly for the challenging cases that objects are on a curved road. Moreover, the performance margin implies the effectiveness of our enhanced method.

1. Introduction

With the advances in the field of computer vision, visual environment perception, which includes object classification, detection, segmentation and distance estimation, has become a key component in the development of autonomous driving cars. Although researchers have paid a lot of efforts on improving the accuracy of visual perception, they mainly focus on more popular tasks, such as object classification, detection and segmentation [29, 27, 17]. Besides recognizing the objects on the road, it is also important to estimate the distances between camera sensors and the recognized objects (e.g., cars, pedestrians, cyclists), which can provide crucial information for cars to avoid collisions, adjust its speed for safety driving and more importantly, as hints for sensor fusion and path planning. However, the object-specific distance estimation task attracts very few attentions from the computer vision community.

With the emergence of the convolutional neural networks, researchers have achieved remarkable progress on traditional 2D computer vision tasks using deep learning techniques, such as object detection, semantic segmentation, instance segmentation, scene reconstruction [4, 30, 31, 16], but we have failed to find any deep learning application on object-specific distance estimation. One of the main reasons could be the lack of datasets that provides distance for each of the object in the images captured from the outdoor road scene.

In this paper, we focus on addressing the interesting but challenging object-specific distance estimation problem for autonomous driving (as shown in Fig. 1). We have ob-
served that most of the current existing robotic systems or self-driving systems predict object distance by employing the traditional inverse perspective mapping algorithm. They first locate a point on the object in the image, then project the located point (usually on the lower edge of the bounding box) into a bird's-eye view coordinate using camera parameters, and finally estimate the object distance from the bird's-eye view coordinate. Though this simple method can predict reasonable distances for objects that stay close and strictly in front of the camera, it performs poorly on cases that 1) objects are located on the sides of the camera or the curved road, and 2) objects are far away (above 40 meters) from the camera. Therefore, we are seeking to develop a model to address the aforementioned challenging cases with the advantages of deep learning techniques.

Ours is the first work to develop an end-to-end learning-based approach that directly predicts distances for given objects in the RGB images. We build a base model that extracts features from RGB images, then utilizes ROI pooling to generate a fixed-size feature vector for each object, and finally feeds the ROI features into a distance regressor to predict a distance for each object. Though our base model is able to provide promising prediction, it still does not fulfill the precision requirement for autonomous driving. Therefore, we create an enhanced model for more precise distance estimation, particularly for objects close to the camera. Specially, in the enhanced model, we design a keypoint regressor to predict part of the 3D keypoint coordinates \((X, Y)\). Together with the predicted distance \((Z)\), it forms a complete 3D keypoint \((X, Y, Z)\). Leveraging the camera projection matrix, we define a projection loss between the projected 3D point and the ground truth keypoint on image to enforce a correct prediction. Note that the keypoint regressor and projection loss are used for training only. After training, given an image with object (bounding box), the object-specific distance can be directly extracted from the outputs of our trained model. There is no camera parameters intervention during inference.

To validate our proposed methods, we construct an extended dataset based on the public available KITTI object detection dataset [10] and the newly released nuScenes (mini) dataset [1] by computing the distance for each object using its corresponding LiDAR point cloud and camera parameters. In order to quantitatively measure the performance of our work and alternative approaches, we employ the evaluation metrics from depth prediction task as our measurements. We report the quantitative results, and visualize some examples for qualitative comparison. The experimental results on our constructed object-specific distance dataset demonstrate that our deep-learning-based models can successfully predict distances for given objects with superior performance over alternative approaches, such as the traditional inverse perspective mapping algorithm and the support vector regressor. Furthermore, our enhanced model can predict a more precise distance than our base one for objects close to the camera. The inference runtime of our proposed model is twice as fast as the traditional IPM.

In summary, the main contributions of our work are concluded as:

- To address the object-specific distance estimation challenges, e.g., objects far away from the camera or on the curved road, we propose the first deep-learning-based method with a novel end-to-end framework (as our base model) to directly predict distance from given objects on RGB images without any camera parameters intervention.
- We further design an enhanced method with a keypoint regressor, where a projection loss is introduced to improve the object-specific distance estimation, especially for object close to the camera.
- To facilitate the training and evaluation on this task, we construct the extended KITTI and nuScenes (mini) object-specific distance datasets. The experiment results demonstrate that our proposed method achieves superior performance over alternative approaches.

### 2. Related work

Object-specific distance estimation plays a very important role in the visual environment reception for autonomous driving. In this section, we briefly review some classic methods on distance estimation and the advances of deep learning models in 2D visual perception.

**Distance estimation** Many prior works for distance estimation mainly focused on building a model to represent the geometry relation between points on images and their corresponding physical distances on the real-world coordinate. One of the classic ways to estimate distance for given object (with a point or a bounding box in the image) was to convert the image point to the corresponding bird's-eye view coordinate using inverse perspective mapping (IPM) algorithm [28, 25]. Due to the drawbacks of IPM, it would fail in cases that objects are located over 40 meters apart or on a curved road. Another vision-based distance estimation work [13] learned a support vector machine regressor to predict an object-specific distance given the width and height of a bounding box. DistNet [14] was a recent try to build a network for distance estimation, where the authors utilized a CNN-based model (YOLO) for bounding boxes prediction instead of the image features learning for distance estimation. Similar to IPM, their distance regressor solely studied the geometric relation that maps a bounding box with a certain width and height to a distance value. In contrast, our goal is to build a model that directly predicts distances from the learned image features.
Figure 2: The framework of our base model, which consists of three components: a feature extractor to generate a feature map for the whole image, a distance regressor to directly predict a distance from the object specific ROI feature, and a multiclass classifier to predict the category from the ROI feature.

Besides the aforementioned approaches, some other works attempted to address this challenging problem by making use of some auxiliary information. Some marker-based methods [2, 26] first segmented markers in the image then estimated distance using the marker area and camera parameters. Instead of utilizing markers, Feng et al. [8] proposed a model to predict physical distance based on a rectangular pattern, where four image points of a rectangular were needed to compute the camera calibration. They then predicted the distance of any given point on an object using the computed camera calibration. Though prior works are impressive, they require markers or patterns to be put in the image for distance estimation, which limits their generalization for autonomous driving.

2D visual perception Although there is no recent work employing deep learning techniques to learn the robust image features for visual monocular object-specific distance estimation, deep learning techniques have been successfully applied on many other 2D visual perception tasks (e.g., object detection, classification, segmentation, monocular depth estimation) with excellent performance [32, 3, 6, 33]. The series of R-CNN works [12, 11, 24, 15] are the pioneers to boost the accuracy as well as decrease processing time consumption for object detection, classification and segmentation. SSD [20] and YOLO models [22, 23] are also the popular end-to-end frameworks to detect and classify objects in RGB images. Their models could be used to address some of the visual perception tasks for autonomous driving, such as detection and classification, but their models are unable to predict the object distance. Nevertheless, those remarkable works inspired us to build an effective end-to-end model for monocular object-specific distance estimation.

On the other hand, monocular depth estimation could be a problem close to our object-specific distance estimation task. Recently, many researchers have created some supervised and even unsupervised models to predict dense depth maps for given monocular color images with more precise details [7, 18, 19, 9]. Their works are motivating, but they usually cost more memory and processing time no matter if it is for training or testing. For visual perception of autonomous driving, it is more crucial to know the object-specific distance to avoid collisions or fuse multiple sensor information, instead of the dense depth map for the entire scene.

3. Our Approach

Observing the limits of the classic inverse mapping algorithm on distance estimation, we propose a learning-based model for robust object-specific distance estimation. A model that directly predicts the physical distance from given RGB images and object bounding boxes, is introduced as our base model. Moreover, we design an enhanced model with a keypoint regressor for a better object-specific distance estimation.

3.1. Base method

Our base model consists of three components, i.e., a feature extractor, a distance regressor and a multiclass classifier (as shown in Fig. 2).

Feature extractor In our model, a RGB image is fed into an image feature learning network to extract the feature map for the entire RGB image. We exploit the popular network structures (e.g., vgg16, res50) as our feature extractor. The output of the last layer of CNN will be max-pooled and then extracted as the feature map for the given RGB image.

Distance regressor and classifier We feed the ex-
We train the feature extractor, the distance regressor and the classifier simultaneously with loss

\[
\min \mathcal{L}_{\text{base}} = \mathcal{L}_{\text{cla}} + \lambda_1 \mathcal{L}_{\text{dist}}. \tag{3}
\]

We use ADAM optimizer to obtain the optimal network parameters with beta value \( \beta = 0.5 \). The learning rate is initialized as 0.001 and exponentially decayed after 10 epochs. \( \lambda_1 \) is set to 1.0 when training our framework. Note that the classifier network is used during training only. Imposing a prior knowledge of the correlation between the object class and its real size and shape, the classifier encourages our model to learn features that can be leveraged in estimating more accurate distances. After training, our base model can be used to directly predict the object-specific distances given any RGB images and object bounding boxes as input.

### 3.2. Enhanced method

Though our base model is able to predict promising object-specific distance from ROI feature map, it is still not satisfying the precision requirement for autonomous driving, especially for objects close to the camera. Therefore, we design an enhanced method with a keypoint regressor to optimize the base model by introducing a projection constraint, and as a result to enforce a better distance prediction. As shown in Fig. 3, the pipeline of our enhanced model consists of four parts, a feature extractor, a keypoint regressor, a distance regressor and a multiclass classifier.

**Feature extractor** We utilize the same network structure that we use in our base model to extract the RGB image and the object bounding boxes (implying the object locations in the image) into an ROI pooling layer to generate a fixed-size feature vector \( F_i \) to represent each object in the image. The pooled feature then is passed through the distance regressor and classifier to predict a distance and a category label for each object. The distance regressor contains three fully connected (FC) layers (with layers of size \( \{2048, 512, 1\} \) for vgg16, \( \{1024, 512, 1\} \) for res50). A softplus activation function is applied on the output of the last fully connected layer to make sure the predicted distance (denoted as \( D(F_i) \)) is positive. For the classifier, there is a fully connected (FC) layer (with the neuron size equals to the number of the categories in the dataset) followed by a softmax function. Let the output of the classifier be \( C(F_i) \). Our loss for the distance regressor \( \mathcal{L}_{\text{dist}} \) and classifier \( \mathcal{L}_{\text{cla}} \) can be written as:

\[
\mathcal{L}_{\text{dist}} = \frac{1}{N} \sum_{i=1}^{N} \text{smooth}_1(d^*_i - D(F_i)), \tag{1}
\]

\[
\mathcal{L}_{\text{cla}} = \frac{1}{N} \sum_{i=1}^{N} \text{cross-entropy}(y^*_i, C(F_i)), \tag{2}
\]

where \( N \) is the number of objects, \( d^*_i \) and \( y^*_i \) are the ground truth distance and category label for the \( i \)-th object.

**Model learning and inference** We train the feature extractor, the distance regressor and the classifier simultaneously with loss

\[
\min \mathcal{L}_{\text{base}} = \mathcal{L}_{\text{cla}} + \lambda_1 \mathcal{L}_{\text{dist}}. \tag{3}
\]
feature. With the object bounding boxes, we can obtain the object-specific features \( F_i \) using ROI-pooling (see Sec. 3.1 for details).

**Keypoint regressor** The keypoint regressor \( K \) learns to predict an approximate keypoint position in the 3D camera coordinate system. The output of the distance regressor can be considered as the value on the camera Z coordinate, so there are only two coordinate values (X, Y) that need to be predicted by the keypoint regressor, denoted as \( K(F_i) \). It contains three fully connected (FC) layers of sizes \{2048, 512, 2\}, \{1024, 512, 2\} for vgg16 and res50, respectively. Since we do not have the ground truth of the 3D keypoint, we choose to project the generated 3D point \((\text{3D keypoint})\) into an image plane using the camera projection matrix \( P \). Then we compute the errors between the ground truth 2D keypoint \( k_i^* \) and the projected point \((P \cdot [K(F_i), D(F_i)])\). In order to encourage the model to better predict distances for closer objects, we put a weighted regression loss \( L_{\text{dist}} \) as

\[
L_{\text{dist}} = \frac{1}{N} \sum_{i=1}^N \left| \frac{1}{\text{dist}} \right| \left| P \cdot [K(F_i), D(F_i)] - k_i^* \right|_2. \tag{5}
\]

**Distance regressor and classifier** For the distance regressor and classifier, we leverage the same network structure as well as training loss \( L_{\text{dist}} \) (Eq. 1) and \( L_{\text{cla}} \) (Eq. 2) as the base model. The network parameters in the distance regressor are optimized by the projection loss \( L_{\text{Dpoint}} \) as well.

**Network learning and inference** We train the feature extractor, the keypoint regressor, the distance regressor and the classifier simultaneously with loss

\[
\text{min} L_{\text{enhance}} = L_{\text{cla}} + \lambda_1 L_{\text{dist}} + \lambda_2 L_{3\text{Dpoint}}. \tag{5}
\]

We use the same setting for the optimizer, beta value and learning rate as the base model. \( \lambda_1 \), \( \lambda_2 \) are set to 10.0, 0.05. We only use the camera projection matrix \( P \), keypoint regressor and classifier for training. When testing, given a RGB image and the bounding boxes, our learned enhanced model directly predicts the object-specific distances without any camera parameters intervention. We implement our (base and enhanced) models using the popular deep learning platform PyTorch [21] and run them on a machine with Intel Xeon E5-2603 CPU and NVIDIA Tesla K80 GPU.

4. Training data construction

One of the main challenges of training deep neural networks for object-specific distance estimation task is the lack of datasets with distance annotation for each object in the RGB images. Existing object detection datasets only provide the bounding boxes and object category annotations, while dense depth prediction datasets provide pixel-level depth values for each image without any object information. Neither of them provide clear object-specific distance annotations. Therefore, we construct two extended object detection datasets from KITTI and nuScenes (mini) with ground truth object-specific distance for autonomous driving.

**KITTI and nuScenes (mini) dataset** As one of the well-known benchmark datasets for autonomous driving, KITTI [10] provides an organized dataset for object detection task with RGB image, bounding (2D and 3D) boxes, category labels for objects in the images, and the corresponding velodyne point cloud for each image, which is ideal for us to construct a object-specific distance dataset. Similarly, the newly released nuScenes(mini) [1] also contains all the information (i.e., RGB images, bounding boxes, velodyne point clouds) for our dataset construction.

**Object distance ground truth generation** As shown in Fig. 4a, to generate the object-specific distance ground truth for a object in a RGB image, we first segment the object points from the corresponding velodyne point cloud using its 3D bounding box parameters; then sort all the segmented points based on their depth values; and finally exact the \( n \)-th depth value from the sorted list as the ground truth distance for given object. In our case, we set \( n = 0.1 \times \text{ (number of segmented points) } \) to avoid extracting depth values from noise points. Additionally, we project the velodyne points (used for ground truth distance extraction) to their corresponding RGB image planes, and get their image coordinates as the keypoint ground truth. We append the ground truth of the object-specific distance and keypoint to the KITTI / nuScenes(mini) object detection dataset labels, together with the RGB images to construct our dataset.

Since both KITTI and nuScenes(mini) only provide the ground truth labels for the training set in its object detection dataset, we generate the distance and keypoint ground truth for all the samples in the training set. Following the split strategy as [5], we split the samples from KITTI training set into two subsets (training / validation) with \( 1:1 \) ratio. There is a total of 3,712 RGB images with 23,841 objects in the training subset, and 3,768 RGB images with 25,052 objects in the validation subset. All the objects are categorized into 9 classes, i.e., \textit{Car}, \textit{Cyclist}, \textit{Pedestrian}, \textit{Misc}, \textit{Person_sitting}, \textit{Tram}, \textit{Truck}, \textit{Van}, \textit{DontCare}. Our generated ground truth object-specific distances are varied from [0, 80] in meters. Fig. 4b shows the distribution of the generated object-specific distances and the object categories in our entire constructed dataset. We can find that distances are ranged mostly from 5M to 60M, and \textit{Car} is the dominant category in the dataset. For the nuScenes(mini) dataset, we randomly split the samples into two subsets with 1,549 objects in 200 training images and 1,457 objects in 199 validation images. All objects are labeled with 8 categories (\textit{Car}, \textit{Bicycle}, \textit{Pedestrian}, \textit{Motorcycle}, \textit{Bus}, \textit{Trailer}, \textit{Truck}, \textit{Construction vehicle}) and distances varied from 2M to 105M.
5. Evaluation

In this section, we evaluate our proposed models with a comparison to alternative approaches. We train our models on the training subsets of our constructed datasets, while test them on the validation subsets.

**Evaluation metrics** Our goal is to predict a distance for objects as close to the ground truth distance as possible. Therefore, we adopt the evaluation metrics provided by [7], usually used for depth prediction. It includes absolute relative difference (Abs Rel), squared relative difference (Squa Rel), root of mean squared errors (RMSE) and root of mean squared errors computed from the log of the predicted distance and the log ground truth distance (RMSELog). Let \(d^*_i\) and \(d_i\) denote the ground truth distance and the predicted distance, we can compute the errors as:

\[
\text{Threshold: } \% \text{ of } d_i \text{ s.t. } \max\left(\frac{d_i}{d^*_i}, \frac{d^*_i}{d_i}\right) = \delta < \text{ threshold},
\]

- **Abs Relative difference (Abs Rel):** \(\frac{1}{N} \sum_{d \in N} |d - d^*| / d^*\),
- **Squa Relative difference (Squa Rel):** \(\frac{1}{N} \sum_{d \in N} \frac{|d - d^*|^2}{d^*}\),
- **RMSE (linear):** \(\sqrt{\frac{1}{N} \sum_{d \in N} ||d_i - d^*_i||^2}\),
- **RMSE (log):** \(\sqrt{\frac{1}{N} \sum_{d \in N} ||\log d_i - \log d^*_i||^2}\).

(b) Distribution of generated object-specific distances (top), and different object categories in our constructed KITTI-based object-specific distance dataset (bottom).

**Compared approaches** As one of the most classic methods to predict (vehicle) distance in an automobile environment, inverse perspective mapping algorithm (IPM) [28] approximates a transformation matrix between a normal RGB image and its bird’s-eye view image using camera parameters. We adopt the IPM in the MATLAB computer vision toolkit to get the transformation matrices for the RGB images (from validation subset). After projecting the middle points of the lower edge of the object bounding boxes into their bird’s-eye view coordinates using the IPM transformation matrices, we take the values along forward direction as the estimated distances.

Similar to recent work [13], we compute the width and height of each bounding box in the training subset, and train a SVR with the ground truth distance. After that, we get the estimated distances for objects in the validation set by feeding the widths and heights of their bounding boxes into the trained SVR.

For our proposed model, we utilize vgg16 and res50 as our feature extractor for both base and enhanced model. We trained our models for 20 epochs with the batch size of 1 on the training dataset augmented with horizontally-flipped training images. After training, we feed the RGB image with the bounding boxes into our trained models and take the output of the distance regressor as the estimated distance for each object in the validation subset.

**Results on KITTI dataset** We present a quantitative comparison in the constructed KITTI dataset for all the eval-
Table 1: The comparisons of object-specific distance estimation with alternative approaches on the *val* subset of our constructed KITTI-object-detection-based dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>higher is better</th>
<th>lower is better</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \delta &lt; 1.25 )</td>
<td>( \delta &lt; 1.25^2 )</td>
</tr>
<tr>
<td>Support Vector Regressor (SVR) [13]</td>
<td>0.345</td>
<td>0.595</td>
</tr>
<tr>
<td>Inverse Perspective Mapping (IPM) [28]</td>
<td>0.701</td>
<td>0.898</td>
</tr>
<tr>
<td>Our Base Model (res50)</td>
<td>0.782</td>
<td>0.927</td>
</tr>
<tr>
<td>Our Base Model (vgg16)</td>
<td>0.846</td>
<td>0.947</td>
</tr>
<tr>
<td>Our Enhanced Model (res50)</td>
<td>0.796</td>
<td>0.924</td>
</tr>
<tr>
<td>Our Enhanced Model (vgg16)</td>
<td><strong>0.848</strong></td>
<td>0.934</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pedestrian</th>
<th>Car</th>
<th>Cyclist</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Vector Regressor (SVR) [13]</td>
<td>0.129</td>
<td>0.182</td>
<td>0.285</td>
</tr>
<tr>
<td>Inverse Perspective Mapping (IPM) [28]</td>
<td>0.688</td>
<td>0.907</td>
<td>0.957</td>
</tr>
<tr>
<td>Our Base Model (res50)</td>
<td>0.649</td>
<td>0.896</td>
<td>0.966</td>
</tr>
<tr>
<td>Our Base Model (vgg16)</td>
<td>0.578</td>
<td>0.861</td>
<td>0.960</td>
</tr>
<tr>
<td>Our Enhanced Model (res50)</td>
<td>0.734</td>
<td><strong>0.963</strong></td>
<td>0.988</td>
</tr>
<tr>
<td>Our Enhanced Model (vgg16)</td>
<td><strong>0.747</strong></td>
<td>0.958</td>
<td>0.987</td>
</tr>
</tbody>
</table>

In addition to the quantitative comparison, we visualize some estimated object-specific distance using our proposed models, along with the ground truth distance and the predictions using alternative IPM and SVR for comparison in Fig. 6. The SVR results show the difficulties to estimate a distance according to the width and height of a bounding box. IPM usually performs well for the objects close to or strictly in front of the camera, while it generally predicts incorrect distances for objects far away from the camera, such as the cyclist on the urban environment example, the furthest cars on both highway and curved road images. However, both of our models can predict more accurate distances for those objects. The other challenging case is to predict distance for objects on a curved road. IPM fails when vehicles are turning, whereas our models can successfully handle them. Besides, our enhanced model predicts a more precise objects-specific distance with less time. The average inference time of our model (vgg16) is 16.2 ms per image, which is slightly slower than SVR (12.1 ms) but twice as fast as IPM (33.9 ms).
Figure 6: Examples of the estimated distance using our proposed base model (BM) and enhanced model (EM). We also provide ground truth distance (GT), the predicted distances using IPM and SVR for comparison. Our models can successfully predict distances on challenging cases, such as objects over 40 meters or on the curved road.

Table 3: Comparison of (average) object-specific distance estimation on the nuScenes-based (mini) dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>δ₁</th>
<th>δ₂</th>
<th>δ₃</th>
<th>AK</th>
<th>SK</th>
<th>RMSE</th>
<th>RMSE_{log}</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVR [13]</td>
<td>0.308</td>
<td>0.652</td>
<td>0.833</td>
<td>0.504</td>
<td>13.197</td>
<td>18.480</td>
<td>0.846</td>
</tr>
<tr>
<td>IPM [28]</td>
<td>0.441</td>
<td>0.772</td>
<td>0.875</td>
<td>1.498</td>
<td>1979.375</td>
<td>249.849</td>
<td>0.926</td>
</tr>
<tr>
<td>Base Model(res50)</td>
<td>0.310</td>
<td>0.621</td>
<td>0.846</td>
<td>0.466</td>
<td>7.593</td>
<td>15.703</td>
<td>0.492</td>
</tr>
<tr>
<td>Base Model(vgg16)</td>
<td>0.393</td>
<td>0.697</td>
<td>0.914</td>
<td>0.404</td>
<td>5.592</td>
<td>12.762</td>
<td>0.420</td>
</tr>
<tr>
<td>Enhanced Model(res50)</td>
<td>0.367</td>
<td>0.683</td>
<td>0.877</td>
<td>0.340</td>
<td>5.126</td>
<td>14.139</td>
<td>0.433</td>
</tr>
<tr>
<td>Enhanced Model(vgg16)</td>
<td>0.535</td>
<td>0.863</td>
<td>0.959</td>
<td>0.270</td>
<td>3.046</td>
<td>10.511</td>
<td>0.313</td>
</tr>
</tbody>
</table>

6. Conclusion

In this paper, we discuss the significant but challenging object-specific distance estimation problem in autonomous driving. It is the first attempt to utilize deep learning techniques for object-specific distance estimation. We introduce a base model to directly predict distances (in meters) from a given RGB image and object bounding boxes. Moreover, we design an enhanced model with keypoint projection constraint for a more precise estimation, particular for the objects close to the camera. We trained our models on our newly constructed dataset extended from KITTI and nuScenes(mini) with a ground truth distance for each object in the RGB images. The experimental results demonstrate that our base model is able to predict distances with superior performance over alternative approaches IPM and SVR, while our enhanced model obtains the best performance over all the compared methods.

7. Acknowledgement

The authors would like to highly acknowledge the research team from XMotors.ai for their contributions to bring the inspiration for this proposed research direction. We appreciate Dr. Husam Abu-haimed and Dr. Junli Gu for their valuable inputs to initialize this research work. We also particularly thank Dr. Husam Abu-haimed, Dr. Junli Gu, Dr. Dongdong Fu and Dr. Kuo-Chin Lien for their insightful discussion and comments during the rebuttal period.


