machines

MDPI

Article

Monocular 3D Object Detection Based on Uncertainty Prediction of Keypoints

Mu Chen [1,2,3,4,*], Huaici Zhao [1,2,3,*] and Pengfei Liu [1,2,4]

[1] Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China; liupengfei@sia.cn
[2] Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China
[3] University of Chinese Academy of Sciences, Beijing 100049, China
[4] Key Laboratory of Opto-Electronic Information Processing, Chinese Academy of Sciences, Shenyang 110016, China
* Correspondence: chenmu@sia.cn (M.C.); hczhao@sia.cn (H.Z.)

Abstract: Three-dimensional (3D) object detection is an important task in the field of machine vision, in which the detection of 3D objects using monocular vision is even more challenging. We observe that most of the existing monocular methods focus on the design of the feature extraction framework or embedded geometric constraints, but ignore the possible errors in the intermediate process of the detection pipeline. These errors may be further amplified in the subsequent processes. After exploring the existing detection framework of keypoints, we find that the accuracy of keypoints prediction will seriously affect the solution of 3D object position. Therefore, we propose a novel keypoints uncertainty prediction network (KUP-Net) for monocular 3D object detection. In this work, we design an uncertainty prediction module to characterize the uncertainty that exists in keypoint prediction. Then, the uncertainty is used for joint optimization with object position. In addition, we adopt position-encoding to assist the uncertainty prediction, and use a timing coefficient to optimize the learning process. The experiments on our detector are conducted on the *KITTI* benchmark. For the two levels of easy and moderate, we achieve accuracy of 17.26 and 11.78 in AP_{3D}, and achieve accuracy of 23.59 and 16.63 in AP_{BEV}, which are higher than the latest method KM3D.

Keywords: keypoints; uncertainty prediction; monocular 3D detection

Citation: Chen, M.; Zhao, H.; Liu, P. Monocular 3D Object Detection Based on Uncertainty Prediction of Keypoints. *Machines* **2022**, *10*, 19. https://doi.org/10.3390/machines10010019

Academic Editors: Xiaochun Cheng and Daming Shi

Received: 18 November 2021
Accepted: 23 December 2021
Published: 26 December 2021

1. Introduction

The understanding of 3D properties of objects in the real world is critical for vision-based autonomous driving and traffic surveillance systems [1–5]. Compared with a two-dimensional (2D) object detection task, the 3D object detection task involves nine degrees of freedom, in which the length, width, height, and pose of the 3D bounding box need to be detected. Currently, there are three main methods for 3D object detection: monocular 3D object detection, stereo-based 3D object detection and LIDAR-based 3D object detection. Among them, the LIDAR-based and the stereo-based detection methods can usually obtain higher detection accuracy with the provision of reliable depth information. However, the radar system has the disadvantages of high cost, high energy consumption, and short service life. On the contrary, the monocular detection method, which is characterized by low cost and low energy consumption, has received extensive attention and attracted researchers to conduct studies in this field. Therefore, our work focuses on the improvements in monocular 3D object detection techniques.

Monocular 3D object detection takes a single RGB image as input, and outputs the pose and dimension of the object in the real world. Due to the lack of depth information, this process is ill-conditioned, and the ambiguity will occur in the process of inverse projection from the 2D image plane to 3D space. Obviously, compared with stereo-based and LIDAR-based methods, the task of monocular 3D object detection is more challenging. Thanks to the powerful feature extraction and parameter regression capabilities of the neural network, some original monocular 3D object detection pipelines [6,7] regress the 3D

dimension, orientation and position of an object directly by designing a deep convolutional neural network. However, learning 3D spatial information from planar features of 2D images undoubtedly expands the searching space of parameter learning. Therefore, it is difficult for the deep convolution neural networks to learn effectively without additional auxiliary information. In order to address these challenges, recent works attempt to help the networks learn 3D information better by adding extra information. For example, Mono3D [7] takes advantage of masks to realize instance segmentation. MF3D [8] and ROI-10D [9] use depth information as supervision. However, adding extra information for monocular 3D object detection means more annotation information should be obtained. Fortunately, other scholars try to tackle this problem by using the geometric constraints between 3D space and 2D images, which can narrow the searching space for parameter learning and improve the learning efficiency and detection accuracy.

In our work, we adopt a keypoints-based framework to accomplish the 3D object detection task. In detail, our approach predicts the keypoints projected from the center and corner points of the 3D bounding box. Then, the 3D position of a predicted object is solved by minimizing the re-projection error with supervision of 3D ground-truth. Usually, this method requires a long detection pipeline, resulting in prediction errors in the intermediate process. Unfortunately, the solution of the 3D position depends on the results of previous processes, which will cause the errors to be further amplified and affect the final detection results. Notably, Chen et al. [10] propose to utilize the pairwise relationship of the objects as a constraint to characterize the errors that exist in the detection pipeline. This method greatly improves the detection capability. Considering that the accuracy of keypoints prediction will seriously affect the solution of 3D position, we design an uncertainty prediction module to measure the possible errors at the stage of keypoints prediction. Specifically, by dealing with the coordinates of keypoints, we can obtain some 2D boxes that can be used to extract the regions for describing the features of objects. Then, the network outputs the uncertainty of each object through a series of fully connected neural networks with the 2D box attributes and feature regions as input. Finally, we integrate the uncertainty into 3D position loss for joint optimization. Additionally, we also use a setting of the timing coefficient for more effective training. The subsequent experiments show that our improvements can be beneficial to the 3D detection task. In general, our main contributions are as follows:

(a). A method to predict keypoints uncertainty based on multi-clue fusion.
(b). A strategy to optimize the 3D position by jointly considering the uncertainty.
(c). KUP-Net outperforms the previous methods on the kitti dataset.

2. Materials

Mono-based 3D detection: As an issue of great interest in detection technology, monocular 3D object detection suffers from a lack of depth information when only a single RGB image is available as input. At present, there are two strategies for solving this problem. The first is to use instance segmentation, a prior CAD wireframe model, an independent depth estimation network and other means to assist the monocular vision pipeline in learning geometric information. For example, Chen et al. use the ground plane hypothesis in Mono3D [7] to segment instances, which can improve the detection accuracy by obtaining the contours of the detected objects. In AM3D [11], Ma et al. add an independent depth prediction branch for the monocular 3D object detection pipeline, and the predicted depth information can be converted into point clouds to realize a more efficient detection. In addition, the CAD wireframe models are also widely used to assist the monocular 3D object detection task. For example, DeepMANTA [12] ingeniously combines the keypoints method with CAD prior to prediction of the 3D object information. These methods are not based on a pure monocular image to obtain the 3D information of objects. Although they can improve detection accuracy to a certain extent, the annotation work for extra label information is always laborious and cumbersome. On the contrary, the second strategy only uses a single RGB image and the corresponding ground-truth to complete the detection

task, which is different from the above ways. The most representative work is Deep3D [13], in which Mousavian et al. assume that the projection points of the corners in 3D bounding box are all located on the four edges of the 2D bounding box. According to this assumption, they construct the projection constraints based on the 3D–2D relationship of $4^4 = 256$. Inspired by Faster-RCNN [14], M3D-PRN [15] narrows the searching space of geometric information by setting up a series of 3D anchors. Recently, the latest approach RTM3D [16] predicts the projection points of 3D corners based on the well-known 2D object detection framework CenterNet [17]. This concise design causes the outstanding work to become the first real-time algorithm for the monocular 3D object detection task. Subsequently, KM3D [18] adopts a differentiable geometric reasoning module to realize the end-to-end training. Admittedly, the success of these methods is inspiring, but they all ignore the possible errors in the intermediate process due to a long detection pipeline, which has a bad effect on the detection performance.

Uncertainty estimation: When the deep neural network is used to complete various tasks, there will inevitably exist cognitive uncertainty and unpredictable uncertainty [19,20]. In many machine vision tasks, a large number of scholars have begun to consider the impact of uncertainty. For example, in the process of depth regression work, the existence of noise often has a large impact on the final results [19,21]. Regarding this problem, ref. [22] uses uncertainty theory to model the estimation error in depth, which is beneficial to the regression of depth. Similarly, Chen et al. [10] calculate the uncertainty of the distance between objects and the uncertainty of the object location to optimize the prediction of the 3D object position. In addition, for the tasks of trajectory planning [23], pedestrian positioning [24], etc., uncertainty estimation has also been well applied. Our work aims at modeling the uncertainty of keypoints prediction, and then optimizing the position loss under the guidance of uncertainty theory.

3. Methods

In Figure 1, we show the overall structure of our monocular 3D object detector. This detection pipeline follows the one-stage approach [17], which takes RGB images as input and outputs 2D and 3D properties related to the objects. Depending on detection tasks, we can add different components behind the feature map output from the backbone network. For the 2D task, these components mainly include a heatmap, 2D size, and 2D center offset, but the 3D task usually contains a keypoints heatmap, keypoints offset, local orientation, 3D dimension and 3D object confidence. Based on the predicted 3D properties, we follow the practice of [18] and adopt a geometric reasoning module (GRM) to solve the 3D position for objects. Furthermore, we design an uncertainty prediction module for characterizing the errors in the detection pipeline to optimize the 3D object position. Next, we will elaborate on these contents from four aspects: 2D detection, 3D detection, uncertainty prediction module and loss optimization.

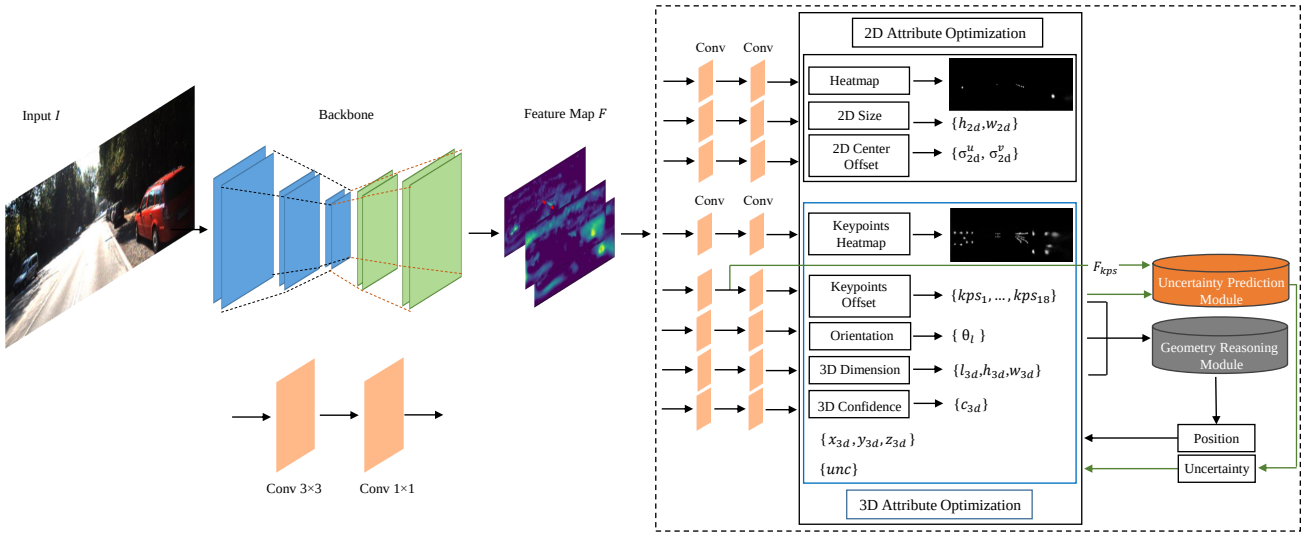


Figure 1. Overview of our network architecture: the backbone is composed of DLA-34 followed by eight sub-task components, the 3D object position is solved by the geometric reasoning module, and the uncertainty module is used to predict the uncertainty for joint optimization in the position loss.

3.1. 2D Detection

Our detection pipeline uses a single RGB image $I \in \mathbb{R}^{W \times H \times 3}$ as input and outputs a global feature map $F \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 64}$ that is used as input to different components of the subsequent detection tasks, where N is a down-sampling factor. Referring to the method of CenterNet [17], we need three sub-functional components to predict the related properties of the 2D bounding box: the object heatmap, 2D box size, and center point offset. Specifically, we apply the convolution kernels of 3 × 3 and 1 × 1 on the F to obtain the object heatmap $H_c \in [0,1]^{\frac{W}{N} \times \frac{H}{N} \times C}$ for determining the category to which the object belongs, where $c = 3$ represents three types of objects to be detected: car, pedestrian, and cyclist. Similarly, the 2D box size $hw_{2d} : \{h_{2d}, w_{2d}\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 2}$ and the center point offset $\widetilde{cs} : \{\sigma^u_{2d}, \sigma^v_{2d}\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 2}$ can be obtained by conducting the same convolution operation.

3.2. 3D Detection

Compared with the 2D detection, the 3D detection is more complicated. As we all know, the projection from 3D space to 2D image plane will result in the loss of depth information. Without auxiliary information, it is ill-conditioned to perform inverse projection from the image plane. Therefore, to restore the more accurate position information for objects, we need to obtain as much 3D information as possible and use geometric constraints to solve the geometric property. In the 3D detection task, the appearance information can be obtained from the feature level, mainly including the object local orientation, keypoints heatmap, keypoints offset, 3D bounding box dimension and 3D confidence. The dimension and orientation properties are visualized in Figure 2.

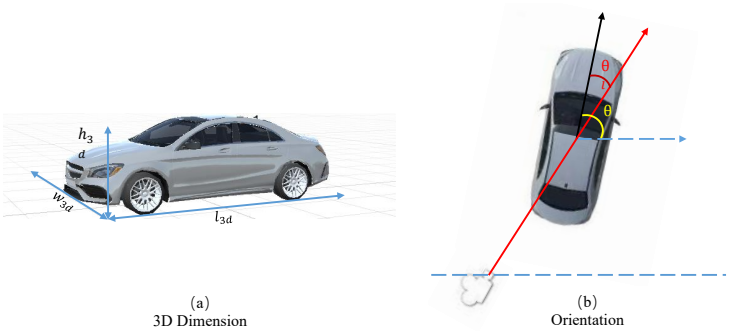


Figure 2. Illustration of the 3D dimension (**a**) and orientation (**b**).

Based on the assumption that vehicles drive on a horizontal road, the global orientation of cars is constant when no motion occurs in the yaw direction. However, with the depth of the object changes in the real world, its local orientation will be changed from the perspective of monocular camera. We follow the suggestion in Deep3D [13] to predict the local orientation $\widetilde{O} : \{\theta_l\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 8}$ for the object. In addition the light angle can be calculated by the camera intrinsic matrix to obtain the global orientation $O : \{\theta\}$. Similar to the operation of the object heatmap prediction in 2D detection, we need to predict the keypoints heatmap $H_{kp} \in [0,1]^{\frac{W}{N} \times \frac{H}{N} \times 9}$, which represents the projection of the eight corner points and center points of the 3D bounding box on the 2D image plane. At the same time, the keypoints offset $\widetilde{kps} : \left\{\widetilde{kps_1^u}, \widetilde{kps_1^v}, \ldots, \widetilde{kps_9^v}\right\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 18}$ are predicted for the subsequent construction of geometric constraints between 3D points and 2D points. Furthermore, the keypoints offset and orientation, dimension and confidence of 3D bounding boxes also need to be acquired for constructing geometric constraints. Therefore, the last two components will be used to obtain the object 3D dimension $D_{3d} : \{l_{3d}, h_{3d}, w_{3d}\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 3}$ and the confidence $S_{3d} : \{c_{3d}\} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 1}$. After the appearance properties of the 3D bounding box have been predicted, it is most important to obtain its geometric property, i.e., its position in 3D space. By using the 3D dimension D_{3d}, orientation O, keypoints offset $\widetilde{kps}$, we can solve the position $\widetilde{T} : \{\widetilde{x_{3d}}, \widetilde{y_{3d}}, \widetilde{z_{3d}}\}$ through a geometric reasoning module proposed in [18,25,26], which can be expressed as follows:

$$\widetilde{T} = \arg\min_{T} \sum_{i}^{9} \left\| f_i(T, D_{3d}, O) - \widehat{kps_i} \right\|_2 \tag{1}$$

where $\widehat{kps}$ is the keypoints coordinate calculated by $\widetilde{kps}$, and the $f(T, D_{3d}, O)$ is:

$$f(T, D_{3d}, O) = K \begin{bmatrix} R(O)^{1\times 3} & T^{1\times 3} \\ 0^T & 1 \end{bmatrix} \mathrm{diag}(D_{3d})\, \mathrm{Cor}$$

$$\mathrm{Cor} = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & -1/2 \\ 1/2 & -1/2 & -1/2 & 1/2 & 1/2 & -1/2 & -1/2 & 1/2 & 0 \\ 1/2 & 1/2 & -1/2 & -1/2 & 1/2 & 1/2 & -1/2 & -1/2 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{2}$$

By solving the pseudo-inverse of SVD, we can obtain the value of $\widetilde{T}$ with constantly approaching the ground-truth $T : \{x_{3d}, y_{3d}, z_{3d}\}$. The details can be seen in the Appendix A.

3.3. Uncertainty Prediction Module

Owing to the prediction uncertainty in neural networks, the prediction results may be subject to errors. Therefore, it is meaningful to take the influence of uncertainty into consideration, which has been confirmed in [20]. In the above Section 3.2, we introduced the 3D detection task, in which the 3D object position $\widetilde{T}$ is solved by keypoints offset $\widetilde{kps}$. Considering that the prediction errors of $\widetilde{kps}$ directly affect the accuracy of $\widetilde{T}$, it is necessary to construct an error characterization about $\widetilde{kps}$ to jointly optimize position loss. Therefore, we model the uncertainty prediction in a probability framework, and the predicted uncertainty is used to measure the accuracy of the 3D position, which can guide the network to train a better detector. We assume that the object position along each axis of the coordinate conforms to the Laplace distribution defined by $\widetilde{T}$ and $\{unc\}$, in which $\widetilde{T}$ and $\{unc\}$ represent the mean and variance, respectively. The more accurate the object position, the closer the $\widetilde{T}$ is to ground-truth. The closer the $\{unc\}$ is to 1, the lower the uncertainty. On the contrary, the uncertainty is higher. As shown in Figure 3, we design an uncertainty prediction module. Specifically, by using the $\widetilde{kps}$, we first solve the nine keypoint coordinates $\{(u_1, v_1), ..., (u_9, v_9)\}$ for each object. Then, these coordinates will be used to obtain a series of diagonal points $\{(u_{min}, v_{min}), (u_{max}, v_{max})\}$ that are constrained within the projection of their corresponding

3D bounding boxes. It should be noted that $u_{\min}, v_{\min}, u_{\max}, v_{\max}$ represent the maximum and minimum values of the nine keypoints along the u direction and the v direction, respectively. Naturally, we can define the 2D keypoints box $Bbox_{kps}$: $\{CT_{kps}, D_{kps}\}$, in which the center coordinate CT_{kps} is $\{u' = \frac{|u_{\max}+u_{\min}|}{2}, v' = \frac{|v_{\max}+v_{\min}|}{2}\}$ and the 2D size D_{kps} is $\{w' = |u_{\max} - u_{\min}|, h' = |v_{\max} - v_{\min}|\}$. Due to the $Bbox_{kps}$ not containing the object feature clue, we use the $Bbox_{kps}$ to extract the region of interest ROI_{kps} with the size of 5×5 from the feature map $F_{kps} \in \mathrm{R}^{\frac{W}{N} \times \frac{H}{N} \times 256}$ that is used to predict the $\widetilde{kps}$. For each ROI_{kps} region, a fully connected neural network maps it into a 128-dimensional feature representation ROI_{128}. Before sending the position clue of 2D keypoints box to an uncertainty prediction module, we encode it from the low dimension to the high dimension for a better characterization. The encoding method is as follows:

$$\begin{gathered} B_{\text{bbox}} = (u', v', w', h') \\ \psi(B_{\text{bbox}}) = \left(\sin\left(2^0 \pi B_{\text{bbox}}\right), \cos\left(2^0 \pi B_{\text{bbox}}\right), \ldots, \sin\left(2^{M-1} \pi B_{\text{bbox}}\right), \cos\left(2^{M-1} \pi B_{\text{bbox}}\right)\right) \end{gathered} \tag{3}$$

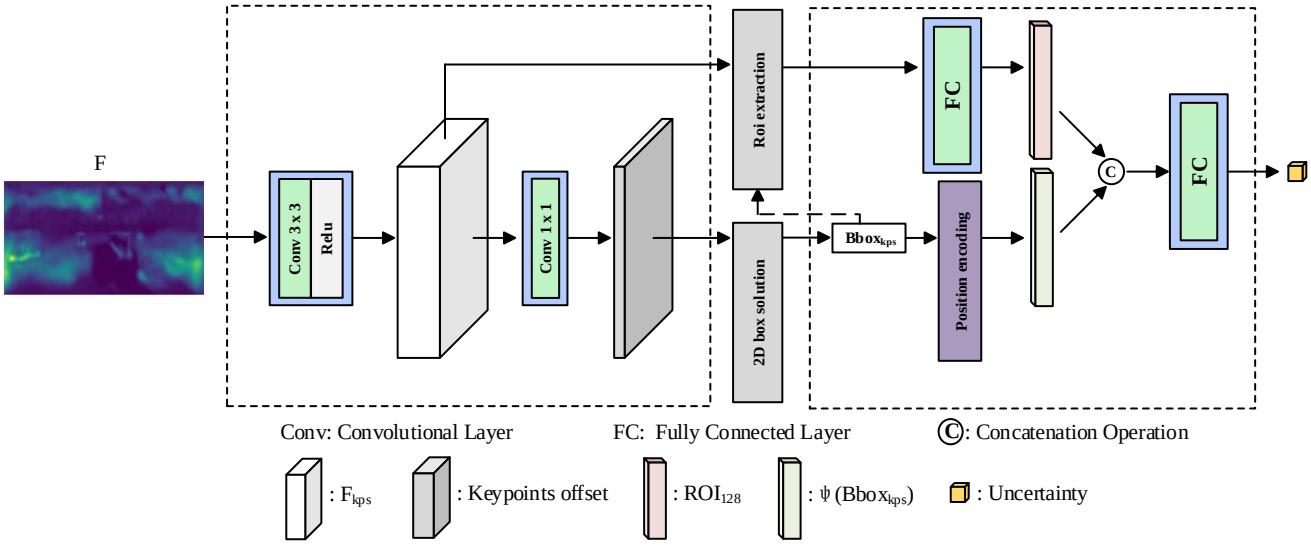


Figure 3. Flow chart of uncertainty prediction.

In order to fuse together the feature clue and box properties in the same dimension, the 2D box properties should also be encoded into a 128-dimensional vector $\psi(B_{bbox})$ as shown in Equation (3), where M is set to 16. Finally, ROI_{128} and $\psi(B_{bbox})$ are concatenated into F_f, and then the uncertainty $\{unc\}$ is output through a fully connected neural network. It should be noted that only the uncertainty along the z axis is predicted for optimization and the reasons can be seen in the ablation study. In Algorithm 1, we give a more concise description of the uncertainty prediction process.

Algorithm 1: The illustration of uncertainty prediction.

Input: Feature map F_{kps}, Keypoints offset $\widetilde{kps}$
Output: Uncertainty unc
1:Coordinate solution: $\widetilde{kps} \rightarrow$ keypoints coordinate:$\{u_1, v_1, \ldots, u_9, v_9\}$
2:Diagonal points acquisition: $\{u_1, v_1, \ldots, u_9, v_9\} \rightarrow \{u_{\min}, v_{\min}, u_{\max}, v_{\max}\}$
3:2D boxes acquisition: $\{u_{\min}, v_{\min}, u_{\max}, v_{\max}\} \rightarrow B_{bbox}$
4:Feature region extraction: $F_{kps}, B_{bbox} \rightarrow ROI_{kps}$
5:Position encoding: $B_{bbox} \rightarrow \psi(B_{bbox})$
6:Feature fusion: $ROI_{kps} \rightarrow ROI_{128}$; concatenate ROI_{128} and $\psi(B_{bbox})$ into F_f
7:Output: $F_f \rightarrow unc$
8:**END**

3.4. Loss Function

To describe the loss function more clearly, we first explain the 2D and 3D detection loss, respectively, and then give the loss function of the whole pipeline.

By calculating the difference value between H_c, hw_{2d}, $\widetilde{cs}$ and their corresponding ground-truth, the 2D loss function L_{2d} can be defined as the three parts of heatmap loss L_c, 2D size loss L_{hw}, and center point offset loss L_{cs}:

$$L_{2d} = \tau_1 L_c + \tau_2 L_{hw} + \tau_3 L_{cs} \tag{4}$$

In order to solve the imbalance problem between positive and negative samples, we follow [17,27,28] to optimize L_c with focal loss. Considering that the 2D size and the center point offset are easy to regress, we choose L1 loss [29] to optimize L_{hw} and L_{cs}.

Similarly, the 3D loss function L_{3d} is composed of six parts: keypoints heatmap loss L_k, keypoints offset loss L_{kps}, 3D dimension loss L_D, local orientation loss $L_{\widetilde{O}}$, 3D confidence loss L_S and position loss L_T:

$$L_{3d} = \gamma_1 L_k + \gamma_2 L_{kps} + \gamma_3 L_D + \gamma_4 L_{\widetilde{O}} + \gamma_5 L_S + \gamma_6 L_T \tag{5}$$

Clearly, the 3D detection task has more regression targets and is more difficult than the 2D detection task. For the term L_k that still has the sample imbalance problem, we optimize it in the same way with L_c. The 3D dimension loss L_D can be better optimized with L1 loss [29]. To optimize L_{kps} more efficiently, we use the depth-guided L1 loss in [18] to dynamically adjust the penalty coefficient. Since the regression of local angle $\widetilde{O}$ is a multi-modal problem, it is more appropriate to choose multi-bin loss [12] to deal with $L_{\widetilde{O}}$. The loss of 3D confidence L_S is optimized by calculating the binary cross-entropy between confidence and the 3D IOU score. For the optimization of 3D position loss L_T, we focus on minimizing the re-projection error under the guidance of heteroscedasticity uncertainty theory in [19]:

$$\mathrm{L}_T = |\widetilde{x_{3d}} - x_{3d}| + |\widetilde{y_{3d}} - y_{3d}| + \left(\frac{\sqrt{2}}{1/\mathrm{unc}} |\widetilde{z_{3d}} - z_{3d}| + \log(1/unc) \right) \tag{6}$$

In constructing the 3D position loss function, we add the prediction uncertainty in a continuous form for joint optimization. The loss function of the whole detection pipeline is defined as:

$$\begin{aligned} L_{\text{Detection}} =& L_{2d} + L_{3d} \\ =& \tau_1 L_c + \tau_2 L_{hw} + \tau_3 L_{cs} + \gamma_1 L_k + \gamma_2 L_{kps} + \gamma_3 L_D + \gamma_4 L_O + \gamma_5 L_S + \gamma_6 L_T \end{aligned} \tag{7}$$

During the training phase, we empirically set the loss coefficients τ_1, τ_2, τ_3, γ_1, γ_2 to 1, and set the γ_3, γ_4 to 4 and 0.4, respectively. The γ_5 is set to e^{-5} at the beginning and changes with the increase in training epochs. For the coefficient of position loss γ_6, we adopt a setting of the timing coefficient, which is similar to that in [30]. This setting makes the position loss added to the whole loss function at the 6th epoch, which can be defined as:

$$\gamma = \begin{cases} 0 & epoch \leq epoch_l \\ e^{-5*(1-(epoch/epoch_h))^2} & epoch_l < epoch < epoch_h \\ 1 & epoch_h \leq epoch \end{cases} \tag{8}$$

where, $epoch_l$ and $epoch_h$ are the lower bound and upper bound for the number of iterations, respectively.

4. Results

4.1. Experimental Setting

As an authoritative dataset, the *KITTI* object detection benchmark [31] is widely applied in the field of 3D object detection to evaluate the performance of detectors. In

the *KITTI* dataset, there are 7481 labeled training images and 7518 unlabeled test images for users. According to the suggestion of [7,8,12,32,33], we divide the dataset into $train_1$, val_1, $train_2$, val_2 and $train$, $test$. This is because the official test set does not give the corresponding annotation information. In our detector, the input images are filled into the size of 1280×384, and the global feature map output through the backbone network is $96 \times 320 \times 64$. Behind the global feature map, eight branches are connected for the 2D detection task and 3D detection task, respectively. The convolution kernels of 3×3 and 1×1 are applied on the global feature to obtain output for each component. Keypoints labels for supervision are obtained by projecting 3D truth values of the left and right images, and we use image inversion, image scaling and other technologies to enhance the dataset. The whole training process is realized on i7-8086K CPU and a single 1080Ti GPU, with the batch size of 8. Moreover, we use an Adam [34] optimizer to optimize the parameters of the whole network. The total number of epochs is 180, and the initial learning rate is set to 1.25×10^{-4}, which is reduced by $5\times$ at 60 and 140 epochs, respectively. In order to demonstrate the performance of our method, we label the detection difficulties in three levels (*Easy*, *Moderate*, *Hard*) based on the level of occlusion, truncation and height of the objects [10,18]. Under the guidance of [13,16], we choose two metrics for 3D detection task: average precision for 3D intersection-over-union(AP_{3D}) and average precision for bird's eye view(AP_{BEV}). If the 2D detection task is needed, average precision for 2D intersection-over-union(AP_{2D}) and average orientation similarity(AOS) are used to evaluate 2D detection performance.

4.2. Experimental Results

Since most of the current works of monocular 3D object detection are devoted to the detection of cars, we first conduct qualitative and quantitative analysis on this category. Then, we show the results of 2D and multi-class detection tasks for a comprehensive evaluation regarding our detector.

4.2.1. Qualitative Results of Cars

To intuitively reflect the effect of our detector, we visualize the 3D detection results of cars. In Figure 4, we choose four scenes to show the visualization of the 3D bounding box and bird's eye view results of our detector. According to the comparison with ground-truth, we can see that our detector can well distinguish the cars and locate them, which proves the efficiency of our approach.

(a) Ground-truth (b) Output (c) Bird's Eye View

Figure 4. Visualization of car category detection in four scenes. In each picture group, we show the ground-truth (**a**), the output (**b**) and the bird's eye view (**c**) from the left to the right.

4.2.2. Quantitative Results of Cars

We select the representative methods in the field of monocular 3D object detection and some of the latest methods to compare with our approach. The performance metrics for the comparison algorithms are the official metric data provided. As shown in Tables 1 and 2, we show the comparisons between our approach and other methods under the evaluation of AP_{3D} and AP_{BEV}. Considering that some methods only use AP^{11} for evaluation, we use AP^{11} instead of AP^{40} in val_1 and val_2 to obtain a more comprehensive comparison result [18]. In the *test*, we choose AP^{40} to obtain a fair comparison result. According to [16], the backbone network can choose *ResNet*-18 [6] and DLA-34 [35], respectively, for the tradeoff of speed or accuracy. Our improvement is not focused on the detection speed, so we only use DLA-34 as the backbone network to achieve higher detection accuracy. Based on the comparison results, we can see that our approach significantly improved detection accuracy at both easy and moderate levels. Therefore, considering the uncertainty in the intermediate process of the detection pipeline plays an effective role in the 3D object detection task.

Table 1. AP_{3D} comparison on val_1, val_2 and *test* with a 0.5 and 0.7 IOU threshold: (a) Extra indicates whether there is additional information to assist detection (b) E, M and, H indicate the levels from Easy, Moderate, to Hard. (c) The red, blue, and cyan denote the highest, second highest, and third highest results respectively.

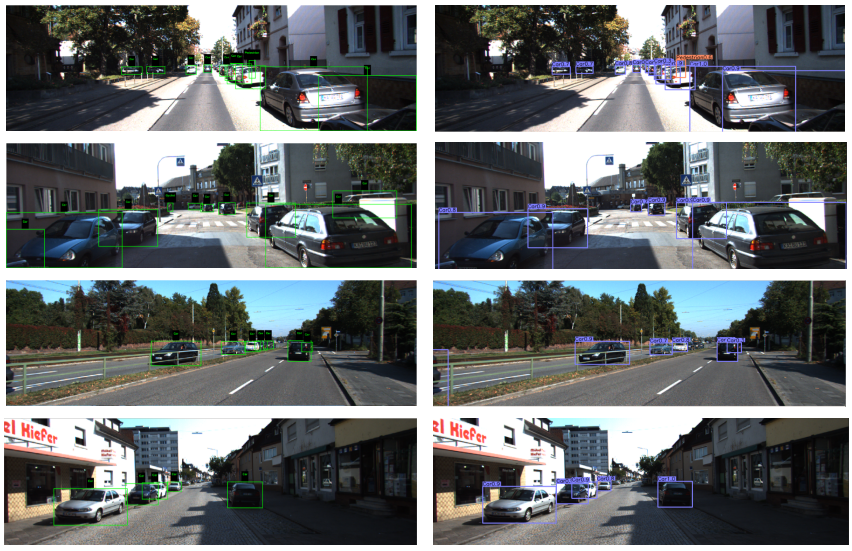
Method	Extra	AP_{3D}@IOU = 0.5 [val_1/val_2]			AP_{3D}@IOU = 0.7 [val_1/val_2/*test*]		
		E	M	H	E	M	H
Mono3D [7]	Mask	25.19/ -	18.20/ -	15.52/ -	2.53 / - / -	2.31 / - / -	2.31 / - / -
ROI-10D [9]	Depth	37.59/ -	25.14/ -	21.83/ -	9.61 / - /12.30	6.63 / - /10.30	6.29 / - /9.39
MF3D [8]	Depth	47.88/45.57	29.48/30.03	26.44/23.95	10.53/ 7.85 / 7.08	5.69 / 5.39 / 5.18	5.39 / 4.73 /4.68
3DOP [33]	Stereo	46.04/ -	34.63/ -	30.09/ -	6.55 / - / -	5.07 / - / -	4.10 / - / -
MonoPSR [36]	Lidar	49.65/48.89	41.71/40.93	29.95/33.43	12.75/13.94/12.57	11.48/12.24/10.85	8.59 /10.77/9.06
M3D-RPN [15]	None	48.96/49.89	39.57/36.14	33.01/28.98	20.27/20.40/14.76	17.06/16.48/ 9.71	15.21/13.34/7.42
GS3D [37]	None	32.15/30.60	29.89/26.40	26.19/22.89	13.46/11.63/ 4.47	10.97/10.51/ 2.90	10.38/10.51/2.47
Deep3DBox [12]	None	27.04/ -	20.55/ -	15.88/ -	5.85 / - / -	4.10 / - / -	3.84 / - / -
MonPair [10]	None	- / -	- / -	- / -	- / - /13.04	- / - / 9.99	- / - /8.65
RTM3D [16]	None	54.36/52.59	41.90/40.96	35.84/34.95	20.77/19.47/13.61	16.86/16.29/10.09	16.63/15.57/8.18
KM3D [18]	None	56.02/54.09	43.13/43.07	36.77/37.56	22.50/22.71/16.73	19.60/17.71/11.45	17.12/16.15/9.92
Ours	None	56.51/54.63	42.75/43.56	36.15/36.02	22.97/23.14/17.26	19.23/20.12/11.78	16.95/16.84/9.51

Table 2. AP_{BEV} comparison on val_1, val_2 and *test* with a 0.5 and 0.7 IOU threshold: (a) Extra indicates whether there is additional information to assist detection (b) E, M and, H indicate the levels from Easy, Moderate, to Hard. (c) The red, blue, and cyan denote the highest, second highest, and third highest results respectively.

Method	Extra	AP_{BEV}@IOU = 0.5 [val_1/val_2]			AP_{BEV}@IOU = 0.7 [val_1/val_2/*test*]		
		E	M	H	E	M	H
Mono3D [7]	Mask	30.05/ -	22.39/ -	19.16/ -	5.22 / - / -	5.19 / - / -	4.13 / - / -
ROI-10D [9]	Depth	46.85/ -	34.05/ -	30.46/ -	14.50/ - /16.77	9.91 / - /12.40	8.73 / - /11.39
MF3D [8]	Depth	55.02/54.18	36.73/38.06	31.27/31.46	22.03/19.20/13.73	13.63/12.17/ 9.62	11.60/10.89/ 8.22
3DOP [33]	Stereo	55.04/ -	41.25/ -	34.55/ -	12.63/ - / -	9.49 / - / -	7.59 / - / -
MonoPSR [36]	Lidar	56.97/55.45	43.39/43.31	36.00/35.47	20.63/21.52/20.25	18.67/18.90/17.66	14.45/14.94/15.78
M3D-RPN [15]	None	55.37/55.87	42.49/41.36	35.29/34.08	25.94/26.86/21.02	21.18/21.15/13.67	17.90/17.14/10.23
GS3D [37]	None	- / -	- / -	- / -	- / - / 8.41	- / - / 6.08	- / - / 4.94
Deep3DBox [12]	None	30.02/ -	23.77/ -	18.83/ -	9.99 / - / -	7.71 / - / -	5.30 / - / -
MonPair [10]	None	- / -	- / -	- / -	- / - /19.28	- / - /14.83	- / - /12.89
RTM3D [16]	None	57.47/56.90	44.16/44.69	42.31/41.75	25.56/24.74/ -	22.12/22.03/ -	20.91/18.05/ -
KM3D [18]	None	62.39/59.35	49.93/45.14	43.73/42.47	27.83/28.87/23.44	23.38/22.87/16.20	21.69/22.55/14.47
Ours	None	62.57/59.73	49.19/49.36	43.61/43.18	28.73/28.16/23.59	23.27/23.41/16.63	21.32/21.68/14.25

4.2.3. Results of 2D and Multi-Class Detection

Besides the results of 3D detection for cars, we also report the ability of our approach to perform 2D and multi-class detection. Similarly, we provide subjective and objective evaluations for the two tasks. In Figures 5 and 6, we qualitatively show the visualization results of 2D detection for cars and 3D detection for pedestrians and cyclists, respectively. Additionally, we also give the corresponding metric values in Tables 3 and 4, in which Table 3 shows the AP_{2D} and AOS for the 2D detection task and Table 4 shows the AP_{3D} and AP_{BEV} for the pedestrian and cyclist detection tasks. Based on these results, we can find that our detector can also be competent for the 2D and multi-class detection tasks.



(a) Ground Truth (b) Output

Figure 5. Visualization of 2D detection for cars in four scenes. In each picture group, we show the ground-truth (**a**) and the output (**b**).

(a) Ground Truth (b) Output

Figure 6. Visualization of multi-class detection for pedestrians and cyclists in four scenes. In each picture group, we show the ground-truth (**a**) and the output (**b**).

Table 3. Evaluation for 2D detection of cars on *test* by AP_{2D} and AOS metrics: E, M, and H indicate the levels from Easy, Moderate, to Hard.

Method	AP_{2D}@IOU = 0.7			AOS		
	E	M	H	E	M	H
AM3D [11]	92.55	88.71	77.78	-	-	-
TLNet [38]	76.92	63.53	54.58	-	-	-
MonoDIS [29]	94.61	89.15	78.37	-	-	-
MonoPSR [36]	93.63	88.50	73.36	93.29	87.45	72.26
M3D-RPN [15]	89.04	85.08	69.26	88.38	82.81	67.08
KM3D [18]	96.44	91.07	81.19	96.34	90.70	90.72
Monopair [10]	96.61	93.55	83.55	91.65	86.11	76.45
Ours	96.59	92.47	83.14	94.28	89.41	84.23

Table 4. Evaluation for multi-class detection of pedestrian and cyclist on $val1$ by AP_{3D} and AP_{BEV} metrics: E, M, and H indicate the levels from Easy, Moderate, to Hard.

Method	AP_{3D}@IOU = 0.5			AP_{BEV}@IOU = 0.5		
	E	M	H	E	M	H
Pedestrian	11.35	10.42	10.37	12.10	11.35	10.46
Cyclist	15.68	11.64	11.03	15.97	11.81	11.14

4.3. Ablation Study

In the above section, we have fully shown the results and efficiency of our method. Next, we will further elaborate on three aspects: timing coefficient, uncertainty prediction mode, and position encoding. By doing this, we can concretely analyze the contribution to each improvement in our work.

4.3.1. Effect of Timing Coefficient

For stable and efficient training, we use a timing coefficient setting for position loss. At the initial training phase, the parameter of the network cannot sufficiently meet the needs of the position solution task, which will lead to a large deviation in position prediction. This large deviation is a disadvantage to the learning and convergence of a multi-task pipeline [22]. Therefore, we propose to use the timing coefficient to decide when to add position loss into the global optimization. Thanks to this strategy, we can provide a set of appropriate initial training parameters for the position optimization. In our work, we optimize the position loss after the fifth epoch, and the loss coefficient changes exponentially with the number of epochs in the following training process. The evaluation details can be seen in Table 5. The accuracy is improved using this setting.

Table 5. Evaluation for the influences of position encoding (Poe) and timing coefficient (Tic) settings: E, M, and H indicate the levels from Easy, Moderate, to Hard.

Poe	Tic	AP_{3D}@IOU = 0.7			AP_{BEV}@IOU = 0.7		
		E	M	H	E	M	H
-	-	10.13	4.58	3.07	15.28	8.51	7.19
-	✓	12.42	6.71	5.14	18.53	11.57	9.65
✓	-	15.73	10.06	8.14	21.64	14.76	12.53
✓	✓	17.26	11.78	9.51	23.59	16.63	14.25

4.3.2. Effect of Uncertainty Prediction Mode

According to the division of the 3D coordinate axis, we provide three optional uncertainty terms to jointly optimize the solution of 3D position, namely unc_x, unc_y and unc_z, which represent the uncertainty along the x, y and z axes, respectively. We provide four combinations of these uncertainties according to the process of 3D–2D projection. As shown in Table 6, we find that applying uncertainty prediction only for z-direction can bring better benefits to the detector. In view of this situation, we think that only the depth information is missing along the z axis in the forward projection process. Therefore, predicting the uncertainty unc_z is the best choice for our detector.

Table 6. Evaluation for the influences of uncertainty unc_x, unc_y, and unc_z settings: E, M, and H indicate the levels from Easy, Moderate, to Hard.

unc_x	unc_y	unc_z	AP_{3D}@IOU = 0.7			AP_{BEV}@IOU = 0.7		
			E	**M**	**H**	**E**	**M**	**H**
✓	✓	-	13.91	8.45	6.58	19.42	12.56	10.39
-	-	✓	17.26	11.78	9.51	23.59	16.63	14.25
-	-	-	16.38	10.82	8.74	22.35	15.41	13.26
✓	✓	✓	16.04	10.39	8.57	21.95	15.02	12.98

4.3.3. Effect of Position Encoding

In the uncertainty prediction module, we use the fusion information of feature clue ROI_{kps} and 2D position clue $Bbox_{kps}$ as input. Obviously, the feature clue is in a high dimension and the 2D position clue is in a low dimension. If we concatenate the two clues directly, the problem of feature mismatching will emerge. Thus, a position encoding on $Bbox_{kps}$ is necessary. From the results in Table 5, it can be seen that the position encoding in Section 3.3 is useful, which enhances the information representation ability and contributes to the prediction of uncertainty.

5. Discussion

Our approach outperforms the latest method in the levels of easy and moderate of AP_{3D} and AP_{BEV}. Especially in the AP_{3D}, the relative improvements of 3.2% and 2.9% were achieved in easy and moderate levels, respectively. Compared with the methods using depth information or CAD models in the inference process, the pure-image-based method is higher ill-posed and more difficult to use to improve the detection accuracy. However, a pure-image-based method usually performs better in the real-time requirement for the tasks of automatic driving and complex environment perception.

Although the method based on pure images exhibits better real-time performance, it is obviously unreliable to fit 3D properties only using deep neural networks. Therefore, the design of the keypoints detection framework adds concise and reliable geometric constraints under the condition of real timeliness. However, in KM3D, the features used to predict keypoints is ambiguous. For example, the keypoints of an object in 2D image plane may appear on the other objects or the background region. This ambiguity of features brings uncertainty to keypoints prediction, influencing the accuracy of the 3D object position. Based on this observation, we model the uncertainty of keypoints prediction, and integrate the uncertainty into the 3D position loss, which is advantageous to the learning of parameters and the convergence of training. Unlike previous works [39,40] that apply uncertainty theory in 2D object detection, Monopair introduces uncertainty estimation into 3D object detection for the first time. However, Monopair is designed on a depth prediction architecture, which cannot perform well in real time. Moreover, Monopair does not encode position clues explicitly when predicting the uncertainty, causing the irrelevant regions to have effects on the prediction results. Accordingly, we fuse the feature clue and position clue defined by the keypoints region. In comparison with the method that only use feature clues, our fusion strategy achieves a richer and clearer feature representation.

6. Conclusions

In our work, aiming at the possible errors in the intermediate process of the monocular 3D object detection pipeline, we have constructed an uncertainty prediction module to optimize the solution of the 3D position property. Experiments on the $KITTI$ dataset show that our method is effective and outperforms some of the latest algorithms. However, our method does not perform well for the detection of hard samples. Based on this problem, we find that the hard samples have more serious occlusion and truncation problems than easy and moderate samples, and therefore a more accurate module is required to predict uncertainty. Furthermore, better results may be achieved by using different training setups, and readers interested in this can make other attempts. In the following work, we will try to develop a richer and more reliable uncertainty constraint strategy for the detection

pipeline by referring to the related methods of automatically encoding the relationship between different objects in natural language processing (NLP) [41–43].

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

Funding: This research is funded by the National Equipment Development Department of China (Grant No. 41401040105) and the National Natural Science Foundation of China (Grant No. U2013210).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the University of Chinese Academy of Sciences for the support.

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

Appendix A

Based on the reference [18], we transform the Equation (2) into a linear system with the shape of $AX = B$, which can be specifically expressed as:

$$\begin{bmatrix} -1 & 0 & kps_1^u \\ 0 & -1 & kps_1^v \\ -1 & 0 & kps_2^u \\ 0 & -1 & kps_2^v \\ -1 & 0 & kps_3^u \\ 0 & -1 & kps_3^v \\ -1 & 0 & kps_4^u \\ 0 & -1 & kps_4^v \\ -1 & 0 & kps_5^u \\ 0 & -1 & kps_5^v \\ -1 & 0 & kps_6^u \\ 0 & -1 & kps_6^v \\ -1 & 0 & kps_7^u \\ 0 & -1 & kps_7^v \\ -1 & 0 & kps_8^u \\ 0 & -1 & kps_8^v \end{bmatrix} \begin{bmatrix} x_{3d} \\ y_{3d} \\ z_{3d} \end{bmatrix} = \begin{bmatrix} \frac{l_{3d}}{2}\cos(\theta) + \frac{w_{3d}}{2}\sin(\theta) - kps_1^u\left(-\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{h_{3d}}{2} - kps_1^v\left(-\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{l_{3d}}{2}\cos(\theta) - \frac{w_{3d}}{2}\sin(\theta) - kps_2^u\left(-\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{h_{3d}}{2} - kps_2^v\left(-\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{l_{3d}}{2}\cos(\theta) - \frac{w_{3d}}{2}\sin(\theta) - kps_3^u\left(\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{h_{3d}}{2} - kps_3^v\left(\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{l_{3d}}{2}\cos(\theta) + \frac{w_{3d}}{2}\sin(\theta) - kps_4^u\left(\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{h_{3d}}{2} - kps_4^v\left(\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{l_{3d}}{2}\cos(\theta) + \frac{w_{3d}}{2}\sin(\theta) - kps_5^u\left(-\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{h_{3d}}{2} - kps_5^v\left(-\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ \frac{l_{3d}}{2}\cos(\theta) - \frac{w_{3d}}{2}\sin(\theta) - kps_6^u\left(-\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{h_{3d}}{2} - kps_6^v\left(-\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{l_{3d}}{2}\cos(\theta) - \frac{w_{3d}}{2}\sin(\theta) - kps_7^u\left(\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{h_{3d}}{2} - kps_7^v\left(\frac{l_{3d}}{2}\sin(\theta) - \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{l_{3d}}{2}\cos(\theta) + \frac{w_{3d}}{2}\sin(\theta) - kps_8^u\left(\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \\ -\frac{h_{3d}}{2} - kps_8^v\left(\frac{l_{3d}}{2}\sin(\theta) + \frac{w_{3d}}{2}\cos(\theta)\right) \end{bmatrix} \tag{A1}$$

where A is the keypoints matrix normalized by the intrinsic camera, X is the 3D object position, and B is the corner points matrix normalized by the intrinsic camera. To solve X, we need to obtain the inverse of matrix A. Since A is not a square matrix, we use the SVD operator to transform A to $U\sum V^T$ to obtain its pseudo inverse A^+, in which U and V are both unitary matrices. By multiplying A^T with A, we can see that $A^TA = V\sum^2 V^T$ and $(A^TA)^-A^TA = E$. Obviously, $A^+ = (A^TA)^-A^T$ and $X = (A^TA)^-A^TB$.

References

1. Li, S.; Yan, Z.; Li, H.; Cheng, K.T. Exploring intermediate representation for monocular vehicle pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021.
2. Yang, B.; Luo, W.; Urtasun, R. PIXOR: Real-time 3D Object Detection from Point Clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
3. Qi, C.R.; Litany, O.; He, K.; Guibas, L.J. Deep Hough Voting for 3D Object Detection in Point Clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019.
4. Peng, W.; Pan, H.; Liu, H.; Sun, Y. IDA-3D: Instance-Depth-Aware 3D Object Detection from Stereo Vision for Autonomous Driving. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
5. Ferryman, J.M.; Maybank, S.J.; Worrall, A.D. Visual surveillance for moving vehicles. *Int. J. Comput. Vis.* **2000**, *37*, 187–197. [CrossRef]
6. 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, 27–30 June 2016.
7. Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Urtasun, R. Monocular 3D Object Detection for Autonomous Driving. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
8. Xu, B.; Chen, Z. Multi-level Fusion Based 3D Object Detection from Monocular Images. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
9. Manhardt, F.; Kehl, W.; Gaidon, A. ROI-10D: Monocular Lifting of 2D Detection to 6D Pose and Metric Shape. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
10. Chen, Y.; Tai, L.; Sun, K.; Li, M. MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020.
11. Ma, X.; Wang, Z.; Li, H.; Zhang, P.; Ouyang, W.; Fan, X. Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019.
12. Chabot, F.; Chaouch, M.; Rabarisoa, J.; Teulière, C.; Chateau, T. Deep MANTA: A Coarse-to-Fine Many-Task Network for Joint 2D and 3D Vehicle Analysis from Monocular Image. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
13. Mousavian, A.; Anguelov, D.; Flynn, J.; Kosecka, J. 3D Bounding Box Estimation Using Deep Learning and Geometry. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
15. Brazil, G.; Liu, X. M3D-RPN: Monocular 3D Region Proposal Network for Object Detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019.
16. Li, P.; Zhao, H.; Liu, P.; Cao, F. RTM3D: Real-Time Monocular 3D Detection from Object Keypoints forAutonomous Driving. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020.
17. Zhou, X.; Wang, D.; Krhenbühl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.
18. Li, P. Monocular 3D Detection with Geometric Constraints Embedding and Semi-supervised Training. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5565–5572. [CrossRef]
19. Kendall, A.; Gal, Y. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? *arXiv* **2017**, arXiv:1703.04977.
20. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016.
21. Liu, C.; Gu, J.; Kim, K.; Narasimhan, S.G.; Kautz, J. Neural RGB®D Sensing: Depth and Uncertainty From a Video Camera. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 10978–10987. [CrossRef]
22. Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight Uncertainty in Neural Networks. In Proceedings of the International Conference on Machine Learning, Lille, France, 7–9 July 2015.
23. Wirges, S.; Reith-Braun, M.; Lauer, M.; Stiller, C. Capturing Object Detection Uncertainty in Multi-Layer Grid Maps. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019.
24. Bertoni, L.; Kreiss, S.; Alahi, A. MonoLoco: Monocular 3D Pedestrian Localization and Uncertainty Estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019.
25. Giles, M. *An Extended Collection of Matrix Derivative Results for Forward and Reverse Mode Algorithmic Dieren Tiation*; Oxford University Computing Laboratory: Oxford, UK, 2008.
26. Ionescu, C.; Vantzos, O.; Sminchisescu, C. Matrix Backpropagation for Deep Networks with Structured Layers. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2965–2973. [CrossRef]
27. Laine, S.; Aila, T. Temporal Ensembling for Semi-Supervised Learning. *arXiv* **2016**, arXiv:1610.02242.
28. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.

29. Simonelli, A.; Bulò, S.R.R.; Porzi, L.; López-Antequera, M.; Kontschieder, P. Disentangling Monocular 3D Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019.
30. Feng, D.; Rosenbaum, L.; Dietmayer, K. Towards Safe Autonomous Driving: Capture Uncertainty in the Deep Neural Network for Lidar 3D Vehicle Detection. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018.
31. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Providence, RI, USA, 16–21 June 2012.
32. Yu, X.; Choi, W.; Lin, Y.; Savarese, S. Data-Driven 3D Voxel Patterns for Object Category Recognition. In Proceedings of the CVPR 2015, Boston, MA, USA, 7–12 June 2015.
33. Chen, X.; Kundu, K.; Zhu, Y.; Ma, H.; Fidler, S.; Urtasun, R. 3D Object Proposals using Stereo Imagery for Accurate Object Class Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 1259–1272. [CrossRef] [PubMed]
34. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
35. Yu, F.; Wang, D.; Shelhamer, E.; Darrell, T. Deep Layer Aggregation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
36. Ku, J.; Pon, A.D.; Waslander, S.L. Monocular 3D Object Detection Leveraging Accurate Proposals and Shape Reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
37. Li, B.; Ouyang, W.; Sheng, L.; Zeng, X.; Wang, X. GS3D: An Efficient 3D Object Detection Framework for Autonomous Driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
38. Qin, Z.; Wang, J.; Lu, Y. Triangulation Learning Network: From Monocular to Stereo 3D Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
39. Choi, J.; Chun, D.; Kim, H.; Lee, H. Gaussian YOLOv3: An Accurate and Fast Object Detector Using Localization Uncertainty for Autonomous Driving. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019.
40. He, Y.; Zhu, C.; Wang, J.; Savvides, M.; Zhang, X. Bounding Box Regression with Uncertainty for Accurate Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
41. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing System, Long Beach, CA, USA, 4–9 December 2017.
42. Zhang, Q.; Yang, Y. ResT: An Efficient Transformer for Visual Recognition. *arXiv* **2021**, arXiv:2105.13677.
43. Alaparthi, S.; Mishra, M. Bidirectional Encoder Representations from Transformers (BERT): A sentiment analysis odyssey. *arXiv* **2020**, arXiv:2007.01127.