Spatio-Temporal Context Modeling for Road Obstacle Detection

Xiuen Wu Fuzhou University xiuen_wu@163.com Tao Wang* Minjiang University twang@mju.edu.cn

Zuoyong Li Minjiang University fzulzytdq@126.com Lingyu Liang* South China University of Technology eelyliang@scut.edu.cn

Fum Yew Ching Universiti Sains Malaysia fyching@student.usm.my

Abstract

Road obstacle detection is an important problem for vehicle driving safety. In this paper, we aim to obtain robust road obstacle detection based on spatio-temporal context modeling. Firstly, a data-driven spatial context model of the driving scene is constructed with the layouts of the training data. Then, obstacles in the input image are detected via the state-of-the-art object detection algorithms, and the results are combined with the generated scene layout. In addition, to further improve the performance and robustness, temporal information in the image sequence is taken into consideration, and the optical flow is obtained in the vicinity of the detected objects to track the obstacles across neighboring frames. Qualitative and quantitative experiments were conducted on the Small Obstacle Detection (SOD) dataset and the Lost and Found dataset. The results indicate that our method with spatio-temporal context modeling is superior to existing methods for road obstacle detection.

1 Introduction

The performance of object detection algorithms has been significantly improved in recent years, largely due to the success of models based on deep convolutional neural networks [2, 7, 9, 17, 18, 19]. In addition, object detection algorithms have been widely used in a spectrum of practical application scenarios. Among them, obstacle object detection in road scenes is a crucial ability for self-driving vehicles. If we can detect the obstacle in front of the vehicle ahead of time and maneuver around it, traffic collisions and consequent injuries, death, and property damage can be avoided.

Despite its significance in driving safety, road obstacle detection is much more challenging than generic object detection. Firstly, road obstacles can be very small especially when viewed from distance. This poses challenges to modern deep learning-based object detectors as they usually operate at fixed levels of the spatial pyramid. While we could potentially train specialized detectors for small-scale object detection, these algorithms would require additional computational budget due to the increased feature resolution. In addition, visual cues from road obstacles can be weak or ambiguous due to motion blur, illumination variation, occlusion, etc. This is particularly the case when the objects are either small or distant, or both. As humans, we overcome these difficulties by constantly focusing on road conditions while driving. An experienced driver will even know what kind of obstacles are more likely to appear at which part of the scene, and pay extra attention to these

^{*}Corresponding authors. Paper accepted by the 4th International Conference on Machine Learning for Cyber Security (ML4CS 2022), Guangzhou, China. The final authenticated version is available online at https://doi.org/10.1007/978-3-031-20096-0_17.



Figure 1: Overview of our method. Given an input sequence of images, our method first integrates the spatial context by building a scene layout model that reasons about the spatial distribution of obstacles and road region. We then calculate the optical flow of the detected object region in two adjacent frames, so as to obtain the object position offset between them. The spatial and temporal contexts are both integrated into the final detection results.

regions accordingly. Therefore, it would be ideal if an object detection method could reason about the presence of obstacles in a similar fashion.

In this work, we aim to improve state-of-the-art road obstacle detectors with spatio-temporal context. For example, by analyzing the spatial context, we can discover that the obstacles are usually located on the road surface and mainly in front of the vehicle. Therefore, we adopt a data-driven approach to extract the location of obstacles and the road layout from training images to build a scene layout model, so that our detector can focus more on the objects on the road surface and eliminate the irrelevant false positives from the background region. In addition, temporal cues are also important for road obstacle detection. Sometimes, an object is detectable in one frame, but will become undetectable in the next frame due to adverse factors such as motion blur, illumination changes, or partial occlusion. Therefore, we could exploit the temporal information in consecutive frames to further assist object detection. Specifically, we use optical flow [1] to transfer object cues from one frame to another. We note that, in our case, the obstacles and the vehicle are both moving while driving. Therefore, when computing the optical flow between the preceding and the following frames, the flow may be too noisy. Therefore, we propose a method based on object region selection, which only calculates the optical flow of the region around the detected objects. This not only removes the complex background, but also reduces the computational complexity. See Fig. 1 for a high-level overview of our method. The contribution of this work is four-fold:

- We propose to integrate spatial and temporal context into state-of-the-art object detection algorithms based on deep learning to improve detection accuracy for road obstacles.
- Based on the spatial context of the obstacles and roads, we construct an interpretable and effective data-driven road scene layout model.
- Based on the temporal context, we track the detected objects via optical flow to assist their detection in the subsequent frames. In particular, we propose an optical flow calculation method based on object region selection.
- We empirically demonstrate the superiority of our method over state-of-the-art object detection methods on the Small Obstacle Detection (SOD) dataset [20] and the Lost and Found dataset [16].

2 Related Work

2.1 Object Detection

State-of-the-art methods for object detection are mainly based on Convolutional Neural Networks (CNNs). On a broad level, these algorithms can be categorized into two-stage detectors and one-stage detectors. The two-stage detectors mainly follow the R-CNN [7] pipeline that includes the object proposal stage and the detection stage. The original R-CNN uses CNNs for object classification on object proposals generated by selective search [22]. However, it is too slow for real-time applications due to the repeated computations of convolutional features for each proposal. Fast R-CNN [6] is faster than R-CNN because it performs feature extraction for object classification only once for all the region proposals. Furthermore, Faster R-CNN [19] combines object proposal and classification into an unified model, so as to improve the efficiency of the network and allow end-to-end training. On the other hand, the YOLO series [9, 17, 18] is a classic example of one-stage detectors, which frames object detection as a regression problem. YOLO can directly predict the bounding boxes and the object categories without proposal generation and region refinement, so it is better suited for real-time applications. In this work, we choose Faster RCNN [19] and YOLOv5 [9] as our baseline object detectors and explore how to improve their results with spatio-temporal context.

2.2 Spatial Context

Modeling the spatial context for object detection is a well-studied problem in computer vision. For instance, Zhang et al. [25] construct the temporal and spatial relationship between the object and the surrounding context through the Bayesian framework. Yao et al. [24] propose a holistic scene understanding model that simultaneously solve the problems of object detection, segmentation and scene classification. Wang et al. [23] propose an efficient scene layout aware object detection method for traffic surveillance. Unlike existing work, we propose a simple but effective method to construct a scene layout model that is tailored to the task of road obstacle detection. In particular, our method considers the spatial distribution of both obstacles and road region, which is not adequately investigated in the literature.

2.3 Temporal Context

There have also been papers on how to exploit the temporal context for video object detection. For example, Kang et al. [10] propose a deep learning framework to solve the problem of general object detection in videos by combining temporal and contextual information. Zhu et al. [26] utilize optical flow to propagate feature across frames to avoid costly feature extraction for non-key frames. Galteri et al. [5] propose a closed-loop framework that uses the object detection results on the previous frame to feed back to the proposal algorithm to improve detection accuracy. Again, our approach differs from these methods as we specifically consider the challenging scenario of road obstacle detection in autonomous driving, and that we propose a method based on object region selection that limits the adverse impact of background noise.

2.4 Road Obstacle Detection

In recent years, a lot of work has been done on road obstacle detection. Kyutoku et al. [11] propose a method based on image subtraction, which mainly uses the difference between the road surface region of the present and past in-vehicle camera images to detect obstacles. Levi et al. [13] propose to treat the obstacle detection problem as a column-wise regression problem, and then use CNN to solve it. Leng et al. [12] present a method that utilizes the U-V disparity map and contextual information to detect obstacles. Our work differs from the methods above in the sense that we integrate a cross-image spatial context model of the obstacles and the road into object detection, in addition to the temporal cues between consecutive frames to more accurately detect objects.

3 Method

In this work, we propose a general framework for integrating the spatio-temporal context into object detection, and it works with any object detection algorithm that outputs bounding boxes. Firstly, the



Figure 2: Scene layout construction. We derive data-driven 2D distributions of obstacles and roads from the training set. (a) Obstacle distribution heat map. (b) Road distribution heat map. (c) The scene layout obtained by combining obstacle distribution and road distribution.

object detection results are obtained through the detector, and then a spatial context score is calculated for the position of each bounding box using the scene layout model. This score is combined with the detection to suppress the false positives in the background. Finally, the object is tracked through the optical flow, so as to provide additional support for object hypotheses in frames with weaker visual cues. The overall process is shown in Fig. 1.

More formally, suppose at time t we have an image I_t as input. Let the object hypothesis be $x \in X$, where X is the object pose space. To simplify the notation, we assume each hypothesis is $x = (x_c, b_s, b_r, o)$ where $x_c = (b_x, b_y)$ is the image coordinate location of the object center, $b_s = (b_w, b_h)$ a scale, b_r an aspect ratio and $o \in O$ a target class. Note that each x now implies a bounding box as well. Object detection algorithms define a scoring function $S_D(x)$ for each valid object hypothesis x. For example, in Faster RCNN, this score is usually obtained via a multi-class softmax score on a convolutional feature map f_{CNN} , i.e., $S_D(x) = \frac{exp(f_{CNN}(x))}{\sum_{o \in O} exp(f_{CNN}(x))}$. We propose an additional scene layout score $S_L(x)$ for any given object hypothesis x. The final detection score is a weighted sum of the two scores:

$$S(x) = S_D(x) + \theta S_L(x) \tag{1}$$

where θ is a hyperparameter for the relative importance between the two terms.

In addition, as the image of the current frame is I_t , we define the image of the next frame as $I_{(t+\delta)}$. Likewise, for a given object hypothesis x in the current frame, the bounding box generated by optical flow tracking in the next frame is denoted as $x_{(t+\delta)}$. For the bounding box generated by optical flow, we define its scoring function in Eqn. 4.

3.1 Scene Layout Construction

We obtain the spatial context information in the road scene by constructing the scene layout, and then use the obtained contextual information to provide regularization to the detection results. For the locations with high probability of obstacles, a higher detection score is given, so as to enhance the confidence of the objects that are difficult to detect on the road. In addition, by modeling the spatial distribution of roads, we can quickly eliminate the false positives outside the road region. Therefore, we build the scene layout based on the spatial distribution of both the obstacles and the road, as outlined below.

3.1.1 Obstacle Distribution

In autonomous driving, the spatial distribution of obstacles shows strong regularity, because we will usually focus on the objects in front of the vehicle as they potentially affect our driving safety. Due to the distance of these objects, they may be small and their visual cues may be weak due to motion blur, illumination, or partial occlusion. Therefore, the detector often gives low confidence for these objects



Figure 3: An example of road distribution. (a) Obstacle distribution heat map. (b) Road region. (c) Road distribution obtained by spreading out the obstacle distribution along transverse and longitudinal directions in the road region.

and they cannot be reliably detected in practice. To address the above problem, we derive statistics on the distribution of obstacles in the training set. We first obtain all the ground truth bounding boxes in the training set and take all their center points to obtain a 2D spatial distribution in the form of a heat map. The results are shown in Fig. 2 (a). The distribution of obstacles in the image shows a strong regularity, which is mainly concentrated in the center of the image. Therefore, we use this distribution as prior information to regularize the object detection results.

3.1.2 Road Distribution

In addition, the road distribution is obtained to further assist the construction of the scene layout model, so that the obstacles in the road can be effectively detected, and the false positives outside the road region can be appropriately ignored. We note that both the SOD dataset and the Lost and Found dataset provide road segmentation annotations, so we could easily obtain the road contour in the training set. To obtain the final road contour, we take several points along the road contour in each image to obtain an average road contour across multiple images. Since the obstacle distribution we obtained in the previous step can be unevenly distributed in the road region, here we propose a method to spread out the obstacle distribution to the entire road region. Specifically, according to the distribution of obstacles, different weights are given along the transverse and longitudinal directions in the road region, respectively. The process is illustrated in Fig. 3. It can be seen that for our daily driving scenarios, the road gradually opens up from far to near, and is roughly in the shape of a pyramid. Also, we note that it is possible to design more complex road models for diverse driving scenarios, but the current model is sufficient for the relatively simple road distribution in the datasets we use in this paper.

3.1.3 Scene Layout Aware Detection

The obstacle distribution and road distribution are added up and then normalized to obtain the final scene layout, as shown in Fig. 2 (c). Afterwards, we can generate a score $S_L(x)$ for each object hypothesis x using the scene layout. Then we can combine the score of object hypothesis x obtained from the object detector $S_D(x)$ with the scene layout score $S_L(x)$ to obtain the final score S(x). Specifically, the definition of $S_L(x)$ is as follows:

$$S_L(x) = \begin{cases} -1 & M(x) < 0.15\\ 0 & 0.15 < M(x) < 0.6\\ \alpha e^{M(x)} + b & M(x) \ge 0.6 \end{cases}$$
(2)

Here $S_L(x)$ is the scene layout score of object x. M(x) is the score based on the location of object x in the scene layout that includes both obstacle and road distributions, α is a variable parameter to adjust the score of the scene layout, and b is a fixed bias value of the scene layout score.

According to Eqn. 2, we assume that when the score M(x) < 0.15, the object is outside the road distribution, which is considered a false positive. When the score is 0.15 < M(x) < 0.6, we consider that the object is within the road region but close to the boundary, so there will be no change to the detection score. When the score $M(x) \ge 0.6$, it is considered that the object appears at a position that we are most likely to detect the obstacles, and the score is increased accordingly.

3.2 Obstacle Tracking with Optical Flow

Here, we use the Lucas-Kanade (LK) method [1] to calculate the optical flow between two consecutive frames, and note that other methods may also be used. Specifically, we consider two frames from time t to $t + \delta$, in order to achieve the tracking of the detected objects. In general, LK method is based on the following three assumptions: (1) the brightness of the tracked part of the object in the scene remains basically unchanged; (2) the motion is relatively slow relative to the frame rate; (3) adjacent points on the same surface in a scene should have similar motion. In addition, the image pyramid is introduced to improve its performance. By reducing the size of the image, the moving speed of the object in the image is relatively reduced, so that the object with faster moving speed can be tracked with better quality.

While detecting and tracking road obstacles, the assumptions above may not be met due to constantly changing motion, illumination, partial occlusion, etc. Therefore, instead of calculating the optical flow for the entire scene, we focus on the vicinity where an object is detected. As such, we can eliminate the unnecessary distractions from other parts of the scene, as illustrated in Fig. 4. The specific steps are as follows:

Step 1. The detector is used to detect the obstacles, obtain the detection results of the preceding and the following frames, and select the detections with scores $S_D(x) > 0.3$ in the preceding frame.

Step 2. To judge whether the detection is missed in the following frame, we define a search area A_r in the following frame by enlarging the detection bounding box of the detected object. The definition of the search area A_r is given as:

$$A_r(x) = (b_x, b_y, \alpha b_w + \tau, \beta b_h + \tau)$$
(3)

where α and β are the adjustment coefficients of the selected area width and height respectively. τ represents the initial size of the area. Here, $\alpha > 1$, $\beta > 1$, and $\tau > 0$ as we only enlarge the bounding box to define the search area.

Step 3. If the obstacle is missed in the following frame (i.e., no bounding box for the same obstacle category within $A_r(x)$), we crop out the the search area given in step 2 in both the preceding and the following frames, and obtain image corners in the cropped area using the Shi-Tomasi corner detection algorithm [21], and select the feature corners located on the detected object.

Step 4. The Lucas Kanade optical flow method is then used to track the feature corners in the cropped area, in order to obtain the offset of the feature corners in the two frames before and after.

Step 5. We transfer the bounding boxes in the previous frame to the following frame through the obtained offset. Again, this happens only when the obstacle is missed in the following frame.

3.3 Spatio-Temporal Aware Detection

Based on the spatial and temporal context modeling above, we combine them in the final inference process. Firstly, the obstacles in the input image are detected by the detector, and then the scene layout model is used to process the detection results to remove the incorrect detections outside the road scene distribution, and strengthen the confidence of small or weak obstacles in the road region (see Sec. 3.1). Secondly, the bounding box of missed detections in the road region is recovered by obstacle tracking and bounding box transfer (see Sec. 3.2). Finally, in order to avoid false positives, we calculate the score of the recovered bounding boxes in combination with the scene layout model. Our final score function is written as follows:

$$S(x_{(t+\delta)}) = S(x) - (\lambda \log M^2(x_{(t+\delta)}) + b)$$

$$\tag{4}$$



Figure 4: Obstacle tracking with optical flow. The orange dashed box indicates the object area used to calculate the optical flow of adjacent frames. (a) The yellow and pink lines represent the offset of the object. (b) The blue box indicates the bounding box overlaid from the preceding frame. The green box indicates the bounding box obtained by optical flow tracking in the following frame. Best viewed in electronically, zoomed in.

where $S(x_{(t+\delta)})$ is the score function for object $x_{(t+\delta)}$ generated by optical flow tracking. $M(x_{(t+\delta)})$ is the location score of $x_{(t+\delta)}$ in the scene layout (see Sec. 3.1). λ is a variable parameter used to adjust the score of the scene layout model. *b* is a bias, giving a fixed initial score to the scene layout model. If the score $S(x_{(t+\delta)}) < 0.3$, we treat the recovered bounding box as invalid and remove it.

4 Experiments

4.1 Datasets

Our experiments are carried out on two public datasets: the Small Obstacle Detection (SOD) dataset [20] and the Lost and Found dataset [16], for evaluating the efficacy of the method. There are 2927 images in the SOD dataset, including 1937, 530 and 460 images in the training, validation, and test set, respectively. It comprises of 15 video sequences in total and utilizes a diverse set of small obstacle instances, and uses different road scenes and different sets of obstacles while recording the train, val and test sequences. Test split is kept to be most challenging in terms of turns, occlusions and shadows to better evaluate the generalization ability. The Lost and Found dataset contains images of small items, such as cargo, wooden strips and toys scattered in the free space in front of the car. Among them, training set and test set contain 1036 and 1068 images.

4.2 Implementation Details

In this work, we use Faster RCNN [19] and YOLOv5 [9] object detection algorithms as our baselines. We use their latest implementations in PyTorch without any changes. In the Faster RCNN algorithm, we use ResNet-50 [8] as the backbone network, SGD as the optimizer, and use a minibatch size of 8. A total of 30 epochs were trained. The learning rate starts from 0.006, and the learning rate decreases to one-third of the original every 5 epochs. We also use the ImageNet [3] pre-trained model for network initialization. In addition, we use feature pyramid networks (FPN) [14] in order to learn high quality multi-scale feature representations. In the YOLOv5 algorithm, we use the yolov516 network structure, and use the MS-COCO [15] pre-trained model to initialize the network weight. SGD is used as the optimizer, and the minibatch size is set to 8. A total of 100 epochs are trained, and the epoch with the best validation results is selected. In addition, for the search area in Eqn. 3, we set $\alpha = 2$, $\beta = 3$, $\tau = 30$ and $\tau = 40$ for SOD and Lost and Found, respectively.

4.3 Experimental Results

4.3.1 Metrics

Following common practice [4, 15], we use average precision (AP) to evaluate the detection algorithm. Specifically, true positives (TP) represent the number of positive objects correctly detected, false positives (FP) represent the number of background regions incorrectly marked as objects, false negatives (FN) represent the number of positive objects not detected, and recall and precision are calculated according to Eqns. 5 and 6.

$$Recall = \frac{TP}{TP + FN}$$
(5)

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

Given recall and precision at varying detection score thresholds, we could plot the precision-recall curve of the test results, and the average accuracy (AP) represents the area between the curve and the coordinate axes. In addition, we evaluate results computed at varying Intersection over Union (IoU) [4] thresholds, i.e., AP values at 50%, 75% IoU and the average of AP from 50% to 95% IoU (at 5% IoU interval) as the evaluation metrics.

4.3.2 Results

Taking Faster RCNN and YOLOv5 as our baselines, the experimental results obtained by adding the scene layout model (SL), the obstacle tracking with optical flow (OF) and the combination of the two methods are shown in Table 1.

Table 1: The ablation study on SOD dataset. **SL:** Scene layout. **OF:** Optical flow tracking. Note that our method improves performance both the Faster RCNN and the YOLOv5 baselines.

Madha d	AD	A D	A D	AD	AD	AD
Method	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L
Faster RCNN	48.4	21.8	24.6	22.8	57.3	23.3
Faster RCNN+SL	50.9	22.2	25.3	23.1	59.9	23.3
Faster RCNN+OF	49.8	22.0	25.0	23.0	57.5	23.3
Faster RCNN+SL+OF	52.6	22.1	25.7	23.4	60.1	23.3
YOLOv5	57.8	27.7	30.7	29.3	57.7	27.3
YOLOv5+SL	58.3	27.8	30.9	29.2	57.9	27.1
YOLOv5+OF	58.9	27.9	31.1	29.4	58.2	28.9
YOLOv5+SL+OF	59.4	27.7	31.2	29.6	58.1	27.9

Taking AP₅₀ as an example, the average precision is increased by 2.5% and 1.4% when the scene layout (SL) model and object tracking with optical flow (OF) method are used separately with Faster RCNN, which demonstrates the efficacy of the two methods. Combining these two methods provides a significant improvement by 4.2%. For YOLOv5, the scene layout model provides a modest improvement of 0.5%. However, object tracking with optical flow provides a considerable improvement of 1.1%. Combining these two methods provides the largest improvement of 1.6%. We also present some qualitative results in Fig. 5, which clearly show that our method is able to eliminate false positives while being able to detect distant obstacles with weak visual support.

In addition, due to the much larger time interval between adjacent image frames in the Lost and Found dataset, the temporal smoothness between two adjacent frames is poor. Therefore, in this dataset, we only use the scene layout model. The quantitative results we obtained are summarized in Table 2. It can be seen that AP_{50} is increased by 0.9% when the scene layout model is used. We also show some qualitative results in Fig. 6, demonstrating the ability of our method to remove false positives, while boosting the scores of obstacles in the scene layout detected by Faster RCNN.

The computational efficiency of our method is also acceptable for practical applications. Taking the baseline Faster RCNN algorithm on SOD dataset as an example, the average inference time per image is about 61ms on a single nVIDIA RTX 3080 Ti. The inference time does not change when we introduce the spatial context, as the parameters in the scene layout model can be calculated in advance and used directly in the inference phase. Then, with the optical flow tracking added, the



Figure 5: Example detection results on the test set of the SOD dataset. Columns: **GT**: Input image with ground-truths overlaid. **Faster RCNN**: Detections with Faster RCNN. **Faster RCNN+SL+OF**: Detections with Faster RCNN+Scene Layout+Optical Flow. **YOLOv5**: Detections with YOLOv5. **YOLOv5+SL+OF**: Detections with YOLOv5+Scene Layout+Optical Flow. Red boxes are false positives, green boxes are true positives. Detection score threshold is 0.05. Best viewed electronically, zoomed in.



Figure 6: Example detection results on the test set of the Lost and Found dataset. Columns: **GT:** Input image with ground-truths overlaid. **Faster RCNN:** Detections with Faster RCNN. **Faster RCNN+SL:** Detections with Faster RCNN+Scene Layout. Red boxes are false positives, green boxes are true positives. Detection score threshold is 0.05. Best viewed electronically, zoomed in.

Table 2: Results with and without scene layout model on Lost and Found dataset.

Method	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L
Faster RCNN	65.4	39.9	38.3	28.7	52.0	67.0
Faster RCNN+SL	66.3	40.4	38.7	28.8	52.7	67.9

processing time increases to 89ms per image. It should be noted that because we adopt the optical flow calculation method based on region selection to eliminate unnecessary computations in a large part of the image, the additional computational budget for the temporal context is reduced to only 28ms per image.

Based on the analysis above, both the spatial context and the temporal context improve final detection performance when used alone, and their combination produces the best detection results. These results validate the efficacy of the proposed method.

5 Conclusion

In this paper, we propose a novel method to detect road obstacles by integrating the spatial and temporal context. Specifically, we derive the spatial distribution of obstacles and the road with a data-driven approach to build a scene layout model, and propose an obstacle tracking method based on optical flow and object region selection to encode temporal smoothness while suppressing the adverse impact of background noise. Our experiments show significant improvements in object detection accuracy compared to state-of-the-art baseline detectors. As a general framework for spatio-temporal context modeling, our method can work with other object detection algorithms not mentioned in this paper. In the future, we plan to further integrate spatio-temporal context modeling into deep models to allow for end-to-end training of a unified model.

Acknowledgments. This work is partially supported by NSFC (61972187, 61703195), Fujian NSF (2022J011112, 2020J02024), the Open Program of The Key Laboratory of Cognitive Computing and Intelligent Information Processing of Fujian Education Institutions, Wuyi University (KLCCIIP2020202), and the Research Startup Fund of Minjiang University (MJY19021). Lingyu Liang is supported by Science and Technology Program of Guangzhou (202102020692), the Open Fund of Ministry of Education Key Laboratory of Computer Network and Information Integration (Southeast University) (K93-9-2021-01), the Open Fund of Fujian Provincial Key Laboratory of Information Processing and Intelligent Control (Minjiang University) (MJUKF-IPIC202102), Guangdong NSF (2019A1515011045) and CAAI-Huawei MindSpore Open Fund.

References

- [1] Jean-Yves Bouguet et al. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel corporation*, 5(1-10):4, 2001.
- [2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6154–6162, 2018.
- [3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.
- [4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
- [5] Leonardo Galteri, Lorenzo Seidenari, Marco Bertini, and Alberto Del Bimbo. Spatio-temporal closed-loop object detection. *IEEE Transactions on Image Processing*, 26(3):1253–1263, 2017.
- [6] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- [7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision* and pattern recognition, pages 580–587, 2014.
- [8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.
- [9] Glenn Jocher, Alex Stoken, Jirka Borovec, Ayush Chaurasia, L Changyu, AV Laughing, A Hogan, J Hajek, L Diaconu, YK Marc, et al. ultralytics/yolov5: v5. 0-yolov5-p6 1280 models aws supervise. ly and youtube integrations. *Zenodo*, 11, 2021.

- [10] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2896–2907, 2017.
- [11] Haruya Kyutoku, Tomokazu Takahashi, Yoshito Mekada, Ichiro Ide, and Hiroshi Murase. On-road obstacle detection by comparing present and past in-vehicle camera images. In *MVA*, pages 357–360. Citeseer, 2011.
- [12] Jiaxu Leng, Ying Liu, Dawei Du, Tianlin Zhang, and Pei Quan. Robust obstacle detection and recognition for driver assistance systems. *IEEE transactions on intelligent transportation systems*, 21(4):1560–1571, 2019.
- [13] Dan Levi, Noa Garnett, Ethan Fetaya, and Israel Herzlyia. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *BMVC*, volume 1, page 4, 2015.
- [14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.
- [15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- [16] Peter Pinggera, Sebastian Ramos, Stefan Gehrig, Uwe Franke, Carsten Rother, and Rudolf Mester. Lost and found: detecting small road hazards for self-driving vehicles. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1099–1106. IEEE, 2016.
- [17] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- [18] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. IEEE Conference on Computer Vision & Pattern Recognition, pages 6517–6525, 2017.
- [19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [20] Aasheesh Singh, Aditya Kamireddypalli, Vineet Gandhi, and K Madhava Krishna. Lidar guided small obstacle segmentation. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 8513–8520. IEEE, 2020.
- [21] Tiziano Tommasini, Andrea Fusiello, Emanuele Trucco, and Vito Roberto. Making good features track better. In Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. 98CB36231), pages 178–183. IEEE, 1998.
- [22] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.
- [23] Tao Wang, Xuming He, Songzhi Su, and Yin Guan. Efficient scene layout aware object detection for traffic surveillance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 53–60, 2017.
- [24] Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In 2012 IEEE conference on computer vision and pattern recognition, pages 702–709. IEEE, 2012.
- [25] Kaihua Zhang, Lei Zhang, Qingshan Liu, David Zhang, and Ming-Hsuan Yang. Fast visual tracking via dense spatio-temporal context learning. In *European conference on computer vision*, pages 127–141. Springer, 2014.
- [26] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2349–2358, 2017.