Perspective Aware Road Obstacle Detection

Krzysztof Lis, Sina Honari, Pascal Fua, and Mathieu Salzmann

***Abstract*—While road obstacle detection techniques have become increasingly effective, they typically ignore the fact that, in practice, the apparent size of the obstacles decreases as their distance to the vehicle increases. In this paper, we account for this by computing a scale map encoding the apparent size of a hypothetical object at every image location. We then leverage this perspective map to (i) generate training data by injecting onto the road synthetic objects whose size corresponds to the perspective foreshortening; and (ii) incorporate perspective information in the decoding part of the detection network to guide the obstacle detector. Our results on standard benchmarks show that, together, these two strategies significantly boost the obstacle detection performance, allowing our approach to consistently outperform state-of-the-art methods in terms of instance-level obstacle detection.**

***Index Terms*—Computer Vision for Transportation, Data Sets for Robotic Vision, Deep Learning for Visual Perception, Object Detection, Segmentation and Categorization.**

I. Introduction

VISION-BASED driving assistance is now commercially available [1] and enables vehicles to plan a path within the predicted drivable space while avoiding other traffic. However, unusual and unexpected obstacles lying on the road remain a potential danger. Since not every vehicle has stereo cameras or a LiDAR sensor to detect them in 3D, much effort has recently been made to achieve detection in a monocular fashion via learning-based strategies. Such road obstacle detection can also be beneficial for robots in novel environments. Given that such objects are non-exclusive, obtaining exhaustive datasets of real images annotated with such obstacles for training purposes is impractical. Hence, many state-of-the-art deep learning approaches [2], [3], [4], [5] rely on synthetically-generated training data, e.g., by cutting out objects and inserting them into individual frames of the Cityscapes dataset.

However, these methods fail to leverage, both while generating training data and performing the actual detection, the predictable perspective foreshortening in images captured by vehicles' front-facing cameras. It is a standard practice [4], [6], [7] to insert objects of arbitrary sizes at any image location in the training data and to detect objects at multiple-scales irrespective of where they appear in the image. This does not exploit the well-known fact that more distant objects tend to be smaller and that, given a calibrated camera, the relationship between real and projected sizes is known.

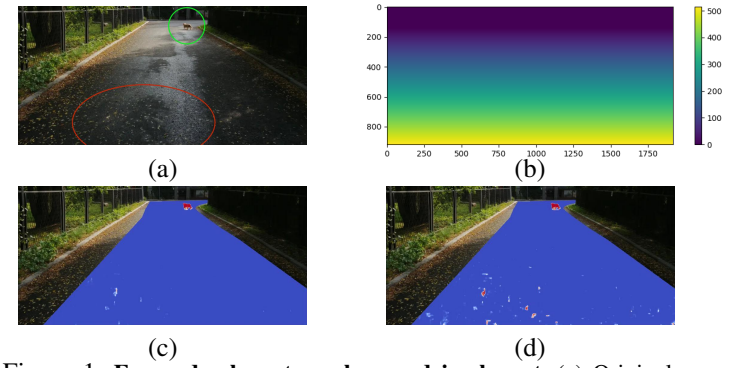


Figure 1: **Far and relevant vs close and irrelevant.** (a) Original image. The green circle denotes a real obstacle far away, and the red circle indicates nearby but harmless leaves. (b) The perspective map indicates, at each pixel, the size in pixels of a hypothetical meter-wide object at that location. (c) Our approach uses the perspective map to distinguish relevant objects from irrelevant ones. It correctly flags in red the pixels of the real obstacle while ignoring the leaves. (d) Without the *perspective aware training set*, a network with a similar architecture flags them all.

In this work, we show that leveraging the perspective information substantially increases performance. To this end, as shown in Fig. 1, we compute a scale map, whose pixel values denote the apparent size in pixels of a hypothetical meter-wide object placed at that point on the road. We then exploit this information in two complementary ways:

- **Perspective-Aware Synthetic Object Injection.** Instead of uniformly injecting synthetic objects into road scenes to synthesize training data, as in [4], [6], [7], we use the perspective map to appropriately set the projected size of the objects we insert.
- **Perspective-Aware Architecture.** We feed the perspective map at multiple levels of a feature pyramid network, enabling it to learn the realistic relationship between distance and size embodied in our training set and in real road scenes.

The bottom portion of Fig. 1 illustrates the benefits of our approach. It not only detects small far-away obstacles but also avoids false alarms arising from small irregularities near the car, such as the leaves here, because their size at this image location does not match that of real threats to the vehicle. Our results show that these strategies together contribute to significantly improving the accuracy of road obstacle detection, particularly in terms of instance-level detection, which is critical for a self-driving car that need to identify all potential hazards on the road.

We evaluate our approach on the *Segment Me If You Can* [8] benchmark's obstacle track and the *Lost&Found* [9] test subset. We demonstrate that it significantly outperforms state-

Manuscript received: September 13, 2022; Revised December 21, 2022; Accepted February 9, 2023. This paper was recommended for publication by by Associate Editor I. Gilitschenski and Editor C. Cadena Lerma upon evaluation of the reviewers' comments. The work was supported in part by the International Chair Drive for All - MINES ParisTech - Peugeot-Citroën - Safran - Valeo.

All authors are with Computer Vision Laboratory, EPFL, Lausanne, Switzerland. (krzysztof.lis@epfl.ch, lis.krzysztof@protonmail.com; sina.honari@gmail.com; pascal.fua@epfl.ch; mathieu.salzmann@epfl.ch).

Digital Object Identifier (DOI): 10.1109/LRA.2023.3245410

of-the-art techniques that use architectures similar to ours, but without explicit perspective handling. The implementation of our method is available at https://github.com/cvlab-epfl/perspective-aware-obstacles.

II. Related Work

A complete overview of state-of-the-art road anomaly detection methods can be found in [8]. In short, many of the most effective monocular methods, as ours, generate synthetic training data to palliate for the lack of a sufficiently diverse annotated road obstacle dataset. We therefore focus on these methods, and then discuss other attempts at exploiting perspective information for diverse tasks.

A. Synthetic Training Data for Obstacle Detection

There is an intractable variety of unexpected objects that can pose a collision threat on roads. To handle this diversity, most existing obstacle detection methods rely on creating synthetic data for training purposes. It is often created from background traffic frames, often from Cityscapes [10], into which synthetic obstacles are inserted.

In [2], the synthetic anomalies are generated by altering the semantic class of existing object instances and synthesizing an image from those altered labels. In [3], this is complemented by adding the Cityscapes *void* regions as obstacles. However, many of the objects exploited by these techniques are located above or away from the road, and the resulting training data only yields limited performance for small on-road obstacles. Our results show that we outperform these methods.

In [7], synthetic obstacles are obtained by cropping random polygons within the background frame and copying their content onto the road, or filling them with a random color. Other methods [4], [6], [11] inject object instances extracted from various image datasets. While this can be done effectively, it remains suboptimal because the objects are placed at random locations, without accounting for their size or for the scene geometry. This is what we address here by explicitly exploiting perspective information, and we demonstrate that it yields a substantial performance boost.

B. Exploiting Perspective Information

Earlier works [12], [13] propose a lightweight sliding-window classifier of drivable space using a pyramid of input patches whose dimension depends on their distance from the horizon. These patches are then rescaled according to their distance to the camera, ensuring that the similar obstacles have similar pixels sizes when presented to the classifier, regardless of the effects of perspective in the original image. This application of perspective information to overcome scale variance is effective, but it can not be easily combined with standard CNNs which operate on the whole image rather than individually rescaled patches.

For any perspective camera, distortion depends on image position. A popular approach to enabling a deep network to account for this in its predictions is to provide it with pixel coordinates as input. In [14], [15], [16], this is achieved by treating normalized pixel coordinates as two additional channels. In [17] the pixel coordinates are used to compute an attention map, to exploit the fact that the class distribution correlates with the image height, for example the *sky* class is predominantly at the top of the image. Another way to implicitly account for perspective effects is to introduce extra network branches that process the image at different scales and fuse the results [18], [19]. However, this strategy, as those relying on pixel coordinates, does not explicitly leverage the perspective information available when working with a calibrated camera, as is typically the case in self-driving.

None of obstacle-detection algorithms explicitly accounts for the relationship between projected object size and distance. This can be done by creating *scale maps* that encode the expected size in the world of an image pixel depending on its position. Scale maps have been used for obstacle and anomaly detection [20], [21]. In [20], the scale information is used to crop and resize image regions before passing them to a vehicle detection network, which then gets to view the cars at an approximately constant scale. This requires running the detector multiple times on the crops. By contrast, our method processes the whole image at once, and the model learns how to leverage the perspective information to adjust the features. In [21], the scale maps are used to rectify the road surfaces, and obstacles are then detected in the rectified views. Unlike these methods, we exploit perspective maps as input to our network, instead of using them for image pre-processing. This prevents the creation of visual artifacts caused by image warping, which yields higher accuracy, as we will show in experiments.

Scale maps have also been extensively investigated for crowd counting purposes [22], [23], [24], [25], [26], [27]. In [22], [23], the models predict perspective information based on observed body and head size. In [24], an unsupervised meta-learning method is deployed to learn perspective maps, which are then used to warp the input images so that they depict a uniform scale as in [25]. In [26], a scale map serves as an extra channel alongside the RGB image and is passed through the backbone feature extractor, whereas in [27] an additional branch is added to the backbone to process the single-channel scale map and to concatenate the resulting features afterwards. In short, perspective information is used during feature computation. In this paper, we follow a different track and incorporate the scale map at different levels of a feature pyramid network. Our experiments show this to be more effective. Furthermore, we argue and demonstrate that, for anomaly detection, incorporating perspective information into the network is not enough; one must also exploit it when synthesizing training data.

C. Fusing RGB and Depth for Road Obstacle Detection

When depth information is available, from stereo camera disparity or RGB-D sensors, it can be fused with the RGB appearance to improve obstacle detection. For example, [28] combines semantic segmentation with stereo-based detections; MergeNet [29] extracts complementary features from RGB-D; RFNet [30]'s two-stream backbone extracts RGB and depth

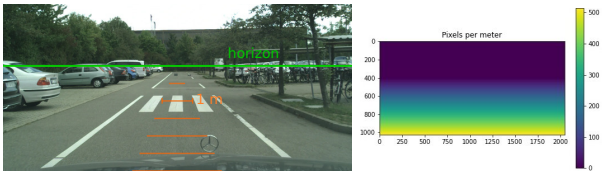


Figure 2: **Perspective map.** *Left:* A 1-meter length overlaid at different image heights on an image from the *Lost&Found* dataset. *Right:* The corresponding perspective map.

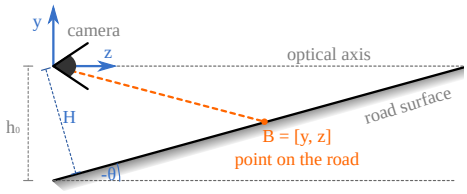


Figure 3: **Building the perspective map.** Geometry of a front-facing camera viewing a planar road surface. Here, the y coordinate of the orange point is negative because it is below the optical axis.

features and uses them to output joint segmentation of known classes and unusual obstacles. Depth (or disparity) contains important geometric cues about the obstacles, which protrude from the road plane, and the above-mentioned methods exploit these cues to detect the obstacles. By contrast, we do not use stereo images and the associated precise scene geometry; our perspective map is generated using a flat-road assumption and contains no information about the obstacles. Our architecture uses perspective as context for analyzing obstacle appearance, by taking it as an extra feature channel without any processing.

III. Approach

Our approach relies on a *perspective map* that captures scale change of objects on the road plane. Therefore, we also refer to it as a *scale map*. In this section, we first describe its construction and then how we use it both to control training data synthesis and as an input to our detection network.

A. Computing the Perspective Map

A perspective map is a scalar field whose value at a given pixel denotes the width in pixels of a hypothetical meter-wide object placed at that point on the road. Fig. 2 depicts one. We compute it from the camera calibration parameters, which are known in a self-driving setup because a vehicle's camera can be calibrated during its production. The camera parameters are f, the camera's focal length in pixels, H, its elevation above the ground in meters, and θ, its pitch angle.

We assume the road to be planar, which, locally, is a good approximation in the majority of real driving scenarios. Let us consider a 3D road point $B = [x, y, z]$ in camera coordinates, which projects onto $[u, v]$ in the image space, as shown in Fig. 3. For simplicity, we denote by $[0, 0]$ the principal point that lies at the center of the image, which is also known in a calibrated camera. Assuming the road to be planar, the pinhole camera model dictates that

$$v = f\frac{y}{z} \ . \tag{1}$$

Figure 4: *Left:* Anchor points distributed along the road surface. We place obstacles at a random subset of anchor points. *Right:* Frame with injected obstacles.

As B lies on the road plane, which is inclined by an angle θ w.r.t. the camera's optical axis, we can write

$$y = z\tan(\theta) - h_0 \ , \tag{2}$$

where $h_0 = \frac{H}{\cos\theta}$, with H being the perpendicular distance from the camera to the road. Solving for z by replacing y in Eq. 1 by its definition in Eq. 2 yields

$$z([u, v]) = h_0\frac{f}{f\tan(\theta) - v} \ , \tag{3}$$

where we indicate that z depends on the pixel location $[u, v]$ by $z([u, v])$. The visible scale is inversely proportional to the z coordinate in the camera frame, and so the scale value $P([u, v])$ in the perspective map $\mathbf{P}$ is equal to

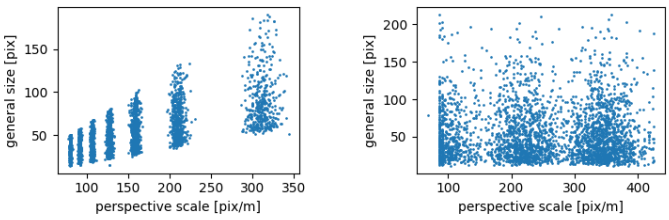
$$P([u, v]) = f\frac{1}{z([u, v])} = \frac{\cos(\theta)}{H}(f\tan(\theta) - v) \ . \tag{4}$$

Note that this requires the pitch angle θ of the camera optical axis with respect to the road surface to be known. When the car is stable on its four wheels, it only depends on how the camera is mounted, which is known. If the pitch changes while driving, an online camera calibration module, e.g., one that relies on vanishing points [31], could be used to update the value of θ. We also assume that various distortions have been corrected so that a pinhole camera model applies.

B. Perspective-Aware Synthetic Object Injection

Collecting a training database of all items that could potentially be left on the road and pose a collision threat is impractical. Effective obstacle detection can thus only be achieved via handling previously unseen objects. To this end, existing methods [4], [7], [6] generate synthetic training frames by injecting objects into the road scenes. However, they use random object sizes and locations. Instead, we leverage the perspective map so that the inserted object sizes are consistent with their locations on the road plane.

a) Placement: We generate a rectangular grid on the road plane, with grid lines being 3.5 meters apart in the direction along the road and 1 meter apart in the width direction. Once this grid is projected onto the image, each grid intersection yields an anchor point offset from the grid point by a random vector whose coordinates are drawn from a zero-centered normal distribution with standard deviation of 0.5 meter. The anchor points are shown in Fig. 4. We then place obstacles at a random subset of these anchor points.



Perspective Aware Uniform [4]

Figure 5: Distribution of injected object sizes. *Left:* Our **Perspective Aware Strategy** selects object sizes based on the perspective map to ensure that objects look smaller when they are further away. There are clusters because we inject objects at discrete grid points on the road-plane projected to the image. *Right:* The **Uniform Strategy** chooses random objects from the whole instance database.

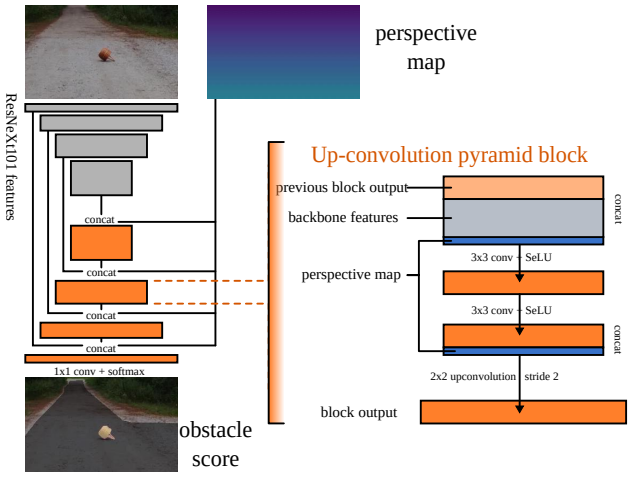


Figure 6: Perspective-aware architecture. The perspective map is injected into the decoding blocks at different resolutions. In each block it is appended twice; first to the backbone features, and second to the intermediate activations preceding the transpose convolution for upsampling.

b) Size: We extract object instances – vehicles, pedestrians, traffic signs and lights – from the Cityscapes dataset [10]. These yield image cut-outs of diverse shapes, ranging from thin poles to wide vehicles. We take an object's *overall pixel size* pix_{obj} to be the average of three values: the square root of its pixel area, and its bounding box width and height. We aim to generate synthetic objects within a range of physical sizes $[\text{ph}_{\text{min}}, \text{ph}_{\text{max}}]$ in meters. To simulate the corresponding visible pixel size of an object seen at an image point $[u, v]$, we multiply the physical range by the scale map value at $[u, v]$: $\text{pix}_{(\text{min,max})}([u, v]) = \text{ph}_{(\text{min,max})} P([u, v])$. Then, we randomly select an object from the training set whose size satisfies $\text{pix}_{\text{min}}([u, v]) \leq \text{pix}_{\text{obj}} \leq \text{pix}_{\text{max}}([u, v])$ and paste it at $[u, v]$. Since we use known classes, such as humans or cars, to later detect unknown classes, such as bottles or tires, we ignore the original size of the known items, and instead choose the object size as a hyper-parameter using the validation-set. Fig. 4-right depicts the resulting object insertion. Note that this does *not* involve scaling the original cut-out objects. Instead, we simply select objects of the appropriate size, thus avoiding scaling artifacts. In Fig. 5, we visualize the resulting relationship between object size and perspective map for both our perspective-aware approach and a uniform injection one. We will compare these two approaches quantitatively.

C. Perspective-Aware Architecture

To distinguish obstacle pixels from the road surface ones, we rely on a U-Net type network architecture, which we train using negative binary cross-entropy loss of pixel classification between the model's prediction and the ground truth segmentation map. The input image is first processed by a ResNeXt101 [32] feature extractor, pre-trained on ImageNet [33] and frozen at training time. We extract four levels of features with increasing receptive fields.

In each block of our up-convolution pyramid, we concatenate the perspective map to the backbone features, and we use it again before the transposed convolution, as depicted in Fig. 6. With such an insertion, the scale information presented to each level of the pyramid can then influence the interpretation of the backbone features at different receptive fields and hence locally adjust the effective receptive field of the detector. In practice, this allows the network to distinguish between distant obstacles and small but harmless irregularities, such as wet patches, leaves, or tile edges, which ought to be ignored. As evidenced below, our perspective-aware architecture shows its full advantage when used together with the perspective-aware object injection.

In our experiments, the perspective map is scaled by $\frac{1}{400}$ before being passed to the network, following the normalization applied in [27], to bring it to an approximate value range between 0 and 1, which improves convergence.

IV. Experiments

In this section, we present the datasets and metrics, and compare our method with the state-of-the-art ones.

A. Datasets

We train our network on Cityscapes. During training, we sample patches of 768 × 384 pixels, with random horizontal flipping. We also perform noise augmentation so that the network generalizes to road surfaces rougher than those found in Cityscapes. We follow the evaluation protocol of the *Segment Me If You Can*[8] obstacle detection benchmark and test our network using the following two datasets.

a) Lost & Found - Test No Known: Lost & Found [9] is an established obstacle dataset captured by placing objects on the road and taking images from an approaching vehicle. The *No Known* variant excludes objects present in the Cityscapes training set, such as pedestrians or bicycles, to focus on the methods' ability to generalize to previously unseen obstacles. It contains 1043 frames with 1709 occurrences of 7 unique lost cargo items placed in 12 parking lot and street scenes. The camera calibration parameters required to compute the perspective map are part of the dataset.

b) RoadObstacle21: RoadObstacles21 is the obstacle track of the recent *Segment Me* benchmark [8]. Like Lost & Found, it contains photos of obstacles placed on roads, but it expands the number of unique objects and the diversity of the

		Road Obstacles 21				Lost&Found - Test No Known			
	requires	Component-level			Pixel-level	Component-level			Pixel-level
Method	OOD	$\overline{F_1}\uparrow$	$\overline{\text{sIoU}}\uparrow$	$\overline{\text{PPV}}\uparrow$	AuPRC $\uparrow$	$\overline{F_1}\uparrow$	$\overline{\text{sIoU}}\uparrow$	$\overline{\text{PPV}}\uparrow$	AuPRC $\uparrow$
Ours	No	**67.1 ± 1.7**	**65.2 ± 0.6**	60.2 ± 2.7	75.2 ± 0.1	**68.6 ± 0.4**	**49.8 ± 0.6**	**87.6 ± 1.2**	**87.4 ± 0.4**
DenseHybrid [11]	Yes	50.7	45.7	50.1	**87.1**	52.3	46.9	52.1	78.7
Maximized Entropy [5]	Yes	48.5	47.9	**62.6**	85.1	49.9	45.9	63.1	77.9
SynBoost [3]	Yes	37.6	44.3	41.8	71.3	48.7	36.8	72.3	81.7
Road Inpainting [4]	No	36.0	57.6	39.5	54.1	52.3	49.2	60.7	82.9
JSRNet [7]	No	11.0	18.6	24.5	28.1	36.0	34.3	45.9	74.2
ODIN [34]	No	9.4	21.6	18.5	22.1	34.5	39.8	49.3	52.9
Image Resynthesis [2]	No	8.4	16.6	20.5	37.7	19.2	27.2	30.7	57.1
Maximum Softmax [35]	No	6.3	19.7	15.9	15.7	10.3	14.2	62.2	30.1
Void Classifier [36]	Yes	5.4	6.3	20.3	10.4	1.9	1.8	35.1	4.8
Mahalanobis [37]	No	4.7	13.5	21.8	20.9	22.1	33.8	31.7	55.0
Embedding Density [36]	No	2.3	35.6	2.9	0.8	27.5	37.8	35.2	61.7
Ensemble [38]	No	1.3	8.6	4.7	1.1	2.7	6.7	7.6	2.9
MC Dropout [39]	No	1.0	5.5	5.8	4.9	13.0	17.4	34.7	36.8

Table I: Obstacle detection scores on RoadObstacle21 and Lost&Found datasets. Both component-level and pixel-level metrics are reported on each dataset. The primary metric is average component detection $\overline{F_1}$ score. *Requires OOD* column indicates if a model is using out of distribution data by training on additional datasets. Other methods train only on Cityscapes dataset.

		Road Obstacles 21		Lost&Found - Test No Known	
Architecture	Object Injection	$\overline{F_1}\uparrow$	AuPRC $\uparrow$	$\overline{F_1}\uparrow$	AuPRC $\uparrow$
1 Ours (perspective-aware)	Ours (perspective-aware)	**67.1 ± 1.7**	75.2 ± 0.1	**68.6 ± 0.4**	**87.4 ± 0.4**
2 No perspective channel	Ours (perspective-aware)	52.5 ± 6.7	70.6 ± 1.0	63.5 ± 3.9	86.0 ± 0.8
3 P-map backbone branch [27]	Ours (perspective-aware)	65.0 ± 2.2	74.6 ± 1.2	63.5 ± 2.4	85.5 ± 0.7
4 P-map along RGB [26]	Ours (perspective-aware)	58.2 ± 4.1	64.6 ± 2.8	66.2 ± 3.5	73.9 ± 8.3
5 XY channels	Ours (perspective-aware)	54.8 ± 4.2	71.1 ± 2.3	63.8 ± 0.8	86.1 ± 0.1
6 Image warping [21], [25]	Ours (perspective-aware)	45.2 ± 0.5	65.5 ± 0.7	20.3 ± 0.6	43.5 ± 1.3
7 Ours (perspective-aware)	Uniform [4]	56.1 ± 1.7	**77.1 ± 0.9**	52.4 ± 2.6	82.1 ± 3.3
8 No perspective channel	Uniform [4]	43.7 ± 6.6	73.3 ± 1.7	48.5 ± 6.4	74.5 ± 10.2
9 P-map backbone branch [27]	Uniform [4]	53.3 ± 4.3	76.8 ± 0.5	55.2 ± 1.5	84.3 ± 0.2
10 P-map along RGB [26]	Uniform [4]	50.8 ± 1.6	69.0 ± 2.4	50.6 ± 5.8	79.7 ± 1.5
11 XY channels	Uniform [4]	51.9 ± 1.4	75.7 ± 0.5	53.7 ± 2.1	78.7 ± 1.7

Object size [m]		Road Obstacles 21 - Validation		Lost&Found - Train	
ph_{min}	ph_{max}	$\overline{F_1}\uparrow$	AuPRC $\uparrow$	$\overline{F_1}\uparrow$	AuPRC $\uparrow$
0.1	- 0.3	59.3 ± 3.4	80.2 ± 1.6	**66.1 ± 1.1**	77.5 ± 0.5
0.25	- 0.55	65.1 ± 3.0	95.7 ± 0.7	62.5 ± 0.3	**89.4 ± 0.8**
0.5	- 0.9	**65.6 ± 1.4**	**96.6 ± 0.4**	57.5 ± 0.1	87.6 ± 0.4
0.75	- 1.25	49.5 ± 1.7	94.2 ± 0.1	50.3 ± 2.4	86.8 ± 1.4
all	sizes [4]	56.1 ± 1.7	77.1 ± 0.9	52.4 ± 2.6	82.1 ± 3.3

Table II: Ablation study. Left) We compare different variants of utilizing the perspective map and show their impact while using either uniform or our perspective-aware object injection. Right) Effect of the size of the injected training objects.

scenes to include more road textures and weather conditions. It comprises 327 frames containing 388 occurrences of 31 unique objects placed in 8 scenes. There are no camera calibration parameters. We therefore estimated them as follows. We assume $f = 2265$pix, that is, the same focal length as in the Cityscapes training set, and $H = 1.5$m because the dataset was captured using handheld cameras. We estimate the camera's pitch angle by approximating the horizon level - the image-space position of the road plane's vanishing line. In the considered datasets, the camera has no side-to-side roll, so we assume the line to be horizontal. The sides of the road and not regular enough to fit lines to them, but with the images depicting forward views along the roads, the horizon is slightly above the end of the visible road. Hence, we first segmented the road using the semantic segmentation PSP network [40], and then took the approximate horizon level to be 16 pixels above its uppermost edge. Given v_{horiz}, the number of pixels between the image midpoint and the horizon level, the pitch angle is retrieved as

$$\theta = \tan^{-1}(\frac{v_{\text{horiz}}}{f}) \ .$$

Such an estimate is obviously very rough, but it is sufficient to inform our model of the scale changes on the road, as we will empirically show when comparing to other variants of our model that do not leverage perspective information.

B. Metrics

The *Segment Me* benchmark [8] measures the methods' performance at both pixel and component levels. The pixel classification task involves distinguishing pixels belonging to obstacles from those of the road surface. It is primarily evaluated with the area under the precision-recall curve (AuPRC). However, pixel metrics give more importance to nearby and big obstacles than to distant or small ones, because of the image area they occupy. For the purpose of driving safety, it thus is more relevant to reason in terms of obstacle instances and their detection regardless of distance and image size. This is addressed by component-level metrics, such as $\overline{F_1}$, $\overline{\text{sIoU}}$ and $\overline{\text{PPV}}$, proposed in [8].

C. Quantitative Evaluation

In Table I, we compare our approach to the state-of-the-art methods featured in the *Segment Me* benchmark. The *requires OOD (out of distribution)* column indicates whether the method was trained using additional data beyond the commonly-used Cityscapes training set, for example by using objects from COCO [41] as obstacles. Our method outperforms the baselines in terms of instance metrics, and only performs worse than [5] and [11] on two metrics, which leverages OOD, thereby demonstrating the good generalization of our approach without resorting to extra training data.

D. Ablation study

1) Impact of Perspective: In Table II-left, we report the results of an ablation study in which we altered either our architecture to use the perspective map in different ways or the perspective-aware synthetic object insertion strategy.

In particular, we consider the following variants:

- *Ours (perspective-aware)*: Our full architecture using the perspective map as described in Section III-C.
- *No perspective channel* : This variant omits the perspective map from our complete architecture.
- *P-map backbone branch* : We provide the perspective map as input to the network but process it in a feature extractor separately from the RGB feature extractor, as in the architecture of [27].

- *P-map along RGB*: We provide the perspective map along with RGB as input to the network, following [26]. In this variant, we unfreeze the backbone feature extractor weights during training to let the network train on the perspective inputs.
- *XY channels* : We provide the image coordinates as two additional channels instead of the perspective map, applying the idea from [14], [17].
- *Image warping*: We follow the idea of [21], [25] and use the perspective map to transform the image into a top-down view of the road.

For synthetic object insertion, we consider two variants; one with perspective-aware object injection as described in Section III-B, and another with uniform object insertion that injects objects uniformly from the full object pool, without restricting it to the objects whose size are inversely proportional to the location where they will be placed. In this variant, the obstacles are placed uniformly in image space rather than on a grid in the road plane. This strategy is identical to [4].

In each setup we provide the mean and standard deviations over three training runs. As observed in Table II-left, using our perspective-aware objection injection together with our perspective-aware network architecture yields the best performance (row 1), and dropping each one of them reduces the accuracy (rows 2, 7, 8). It is worth noting that using the *P-map backbone branch* architecture with our perspective-aware object insertion (row 3) also yields close, yet still inferior results, which indicates that, while there can be other ways of using the perspective map in the architecture, the perspective-aware object insertion plays a key role. The *No perspective channel* variant shows a noticeable drop (row 7), indicating that the network benefits from exploiting the perspective maps. While using the XY image coordinates (row 5) yields reasonable results in *Lost&Found*, whose camera angle matches that of the training set, its performance drops when faced with the different camera setup of *Road Obstacles*. The *P-map along RGB* and *Image warping* variants also show a significant drop (rows 4, 6), which indicates that the way the perspective map is used plays a role in the network accuracy. In particular, the performance of *Image warping*, where the network operates on the warped image, is much lower.

The results show that training with a uniform injection strategy (rows 7-11) yields a much lower $\overline{F_1}$ score than with our perspective-aware approach (rows 1-5). However, the pixel classification AuPRC values of the uniform strategy tend to be higher. We observe that the networks trained with the uniform object insertion technique often better segment the large objects (predicting more pixels on the object), but miss small objects and introduce small false positive instances. By contrast, the higher $\overline{F_1}$ scores obtained with our perspective-aware injection approach evidence the resulting networks' reliability in detecting obstacles more accurately and avoiding false-positives, which is more critical for self-driving cars.

We show qualitative examples in Fig. 7. The first two columns evidence how the perspective map helps the model to distinguish distant obstacles from nearby harmless details, which can span similar pixel sizes in the image. In the rightmost column, we show a difficult case where Max-entropy [5] obtains higher AuPRC but lower $\overline{F_1}$ than us. While such models can segment more pixels of the found objects, they entirely miss some of the objects on the road. Overall, our network detects the obstacles more reliably, and learns to ignore small irregular regions.

This effect is quantified in Fig. 8 where we plot the number of false-positives and true-positives as a function of the distance to the camera, estimated as the inverse of the perspective map. Using our training set and architecture strongly decreases the number of nearby false-positives compared to a system without those contributions, and slightly increases the number of correctly detected obstacles.

2) Object size: In Table II-right, we show how the chosen range $\text{Obj}_{[\text{min,max}]}$ for synthetically injected objects affects detection performance. To avoid overfitting to the benchmarks, we performed this study using public validation sets, *Road Obstacles 21 - Validation* and *Lost & Found - Train*, featuring different obstacles and scenes than the test sets.

As described in Section III-B, the minimum and maximum sizes in meters are multiplied by the local perspective-map value at the site of injection, to determine how big the injected object can be in pixels. We then select at random an object fitting within this pixel range. The results in the table indicate that there is no size range that ideally fits all the circumstances; the small 0.1-0.3 meter range is best for the *Lost & Found - Train* set, while the 0.5-0.9 m range prevails in *Road Obstacles 21 - Validation*, presumably due to these ranges matching the typical object sizes in those datasets. We choose the intermediate 0.25-0.55 meter range, behaves well in both datasets. One might conclude that including objects of all sizes would be best for generalization, but that would prevent expressing the perspective-size relationship, as shown in the bottom row. Indeed, such a strategy, used in [2], yields lower performance than our narrower ranges.

Inference Speed. Our network achieves inference at 12.1 frames of size 1920×1080 per second on an Nvidia V100 GPU; it can be further sped up by network distillation, quantization or TensorRT.

V. Conclusion

We have shown that perspective-aware obstacle injection to generate training data, together with incorporating perspective information in the decoding stages of a network outperforms the state-of-the-art road obstacle detection methods. Our results indicate that the perspective information can guide the model to reduce false positives for small nearby irregularties while still detecting small and far-away objects.

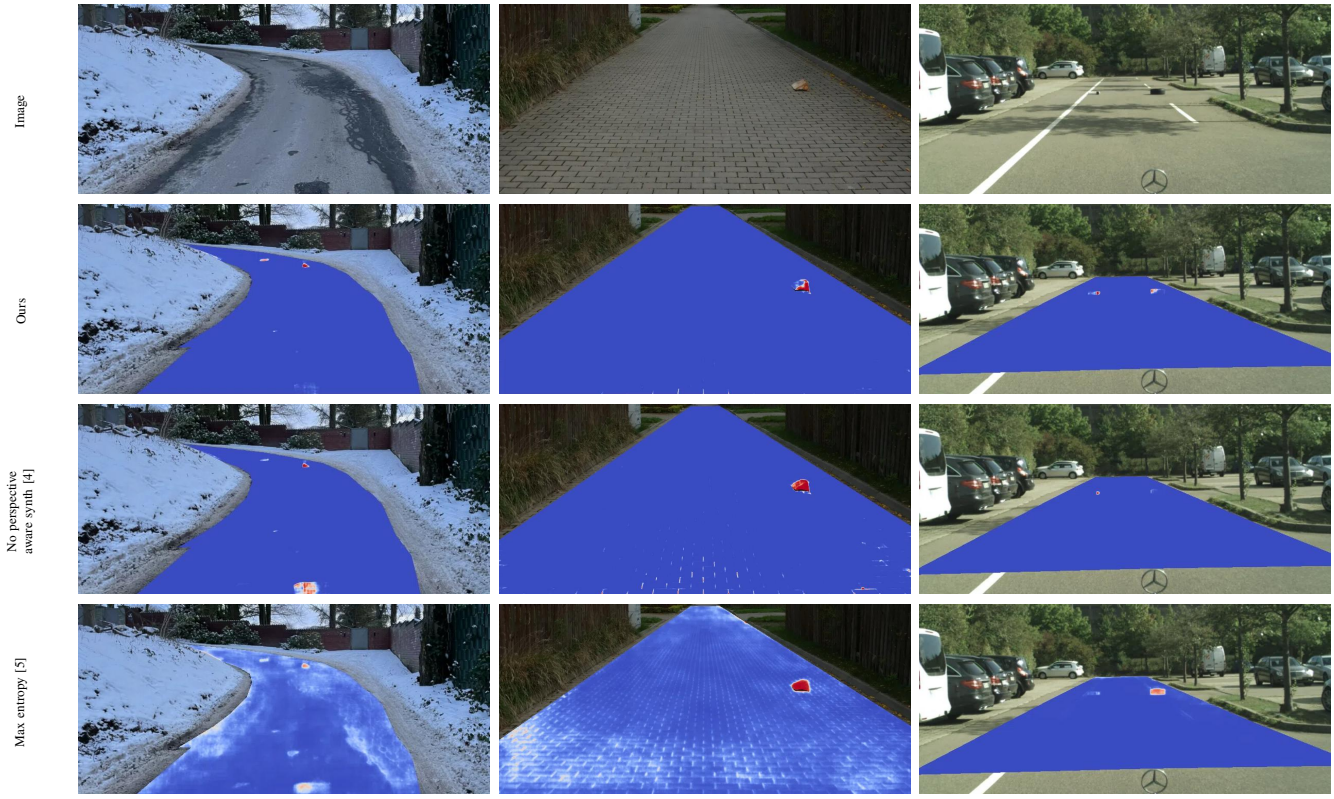


Figure 7: *Left, center:* Perspective information guides our detector to ignore nearby small irregularities on the road surface, while the variants without perspective map and perspective-aware object insertion exhibit false positives in that area. The nearby false-positives and distant obstacles are of similar pixel sizes, so the perspective map allows differentiating between them. *Right:* Our method finds both obstacle instances despite imperfect segmentation. While Max entropy [5] achieves a better pixel-classification score by perfectly segmenting the bigger object, it misses the smaller object.

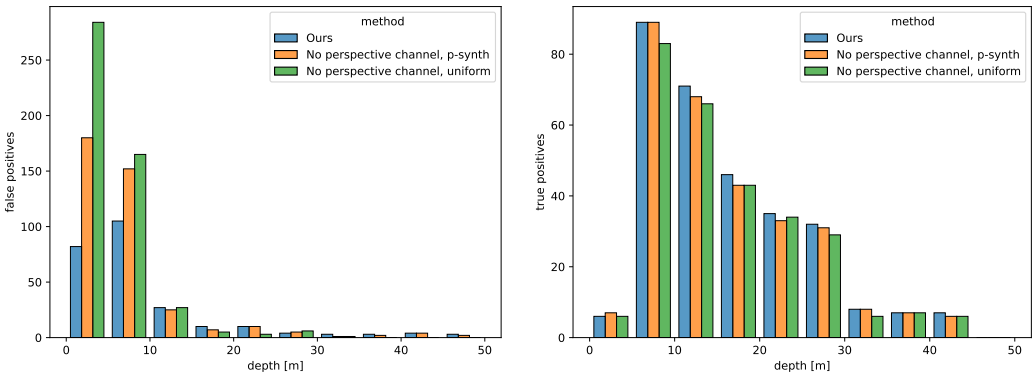


Figure 8: Number of false positives (*left*) and true positives (*right*) as a function of the distance from the camera for the *Obstacle Track - test* dataset. Our training set and architecture (Ours) yield much fewer nearby false-positives and slightly more true-positives than a variant without the perspective map (No perspective channel, p-synth) or one trained with the uniformly-injected synthetic obstacles (No perspective channel, uniform). FP and TP are calculated for an IOU threshold of 0.5.

REFERENCES

[1] "AutoNation Drive Editors' Guide to Lane Assist Systems," https://www.autonationdrive.com/research/best-cars-with-lane-assist.htm, 2021.

[2] K. Lis, K. Nakka, M. Salzmann, and P. Fua, "Detecting the Unexpected via Image Resynthesis," in *International Conference on Computer Vision*, 2019.

[3] G. Di Biase, H. Blum, R. Siegwart, and C. Cadena, "Pixel-Wise Anomaly Detection in Complex Driving Scenes," in *Conference on Computer Vision and Pattern Recognition*, June 2021.

[4] K. Lis, S. Honari, P. Fua, and M. Salzmann, "Detecting Road Obstacles by Erasing Them," in *arXiv Preprint*, 2020.

[5] R. Chan, M. Rottmann, and H. Gottschalk, "Entropy Maximization and Meta Classification for Out-Of-Distribution Detection in Semantic Segmentation," in *International Conference on Computer Vision*, 2021.

[6] P. Bevandić, I. Kreso, M. Orsic, and S. Segvić, "Simultaneous Semantic Segmentation and Outlier Detection in Presence of Domain Shift," in *German Conference on Pattern Recognition*, 2019.

[7] T. Vojir, T. Šipka, R. Aljundi, N. Chumerin, D. O. Reino, and J. Matas, "Road Anomaly Detection by Partial Image Reconstruction with Segmentation Coupling," in *International Conference on Computer Vision*, October 2021.

[8] R. Chan, K. Lis, S. Uhlemeyer, H. Blum, S. Honari, R. Siegwart, P. Fua, M. Salzmann, and M. Rottmann, "Segmentmeifyoucan: A Benchmark for Anomaly Segmentation," in *Advances in Neural Information Processing Systems*, 2021.

[9] P. Pinggera, S. Ramos, S. Gehrig, U. Franke, C. Rother, and R. Mester, "Lost and Found: Detecting Small Road Hazards for Self-Driving Vehicles," in *International Conference on Intelligent Robots and Systems*, 2016.

[10] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Conference on Computer Vision and Pattern Recognition*, 2016.

[11] M. Grcic, P. Bevandic, and S. Segvic, "DenseHybrid: Hybrid Anomaly Detection for Dense Open-Set Recognition," in *European Conference on Computer Vision*, 2022.

[12] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, J. Han, B. Flepp, U. Muller, and Y. LeCun, "Online Learning for Offroad Robots: Using Spatial Label Propagation to Learn Long-Range Traversability," in *Robotics: Science and Systems Conference*, vol. 11, 2007, p. 32.

[13] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning Long-Range Vision for Autonomous Off-Road Driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, 2009.

[14] M. Liu, W. Buntine, and G. Haffari, "Learning How to Actively Learn: A Deep Imitation Learning Approach," in *Annual Meeting of the Association for Computational Linguistics*, 2018.

[15] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep Image Prior," in *Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[16] Y. Lyu and X. Huang, "Road Segmentation Using CNN with GRU," in *arXiv Preprint*, 2018.

[17] S. Choi, J. T. Kim, and J. Choo, "Cars Can't Fly Up in the Sky: Improving Urban-Scene Segmentation via Height-Driven Attention Networks," in *Conference on Computer Vision and Pattern Recognition*, 2020.

[18] X. Li, Z. Jie, W. Wang, C. Liu, J. Yang, X. Shen, Z. Lin, Q. Chen, S. Yan, and J. Feng, "Foveanet: Perspective-Aware Urban Scene Parsing," in *Conference on Computer Vision and Pattern Recognition*, 2017.

[19] C. Huynh, A. Tran, K. Luu, and M. Hoai, "Progressive Semantic Segmentation," in *Conference on Computer Vision and Pattern Recognition*, 2021.

[20] S. Bai, Z. He, Y. Lei, W. Wu, C. Zhu, M. Sun, and J. Yan, "Traffic Anomaly Detection via Perspective Map Based on Spatial-Temporal Information Matrix," in *Conference on Computer Vision and Pattern Recognition*, 2019.

[21] C. D. Prakash, F. Akhbari, and L. J. Karam, "Robust Obstacle Detection for Advanced Driver Assistance Systems Using Distortions of Inverse Perspective Mapping of a Monocular Camera," *Robotics and Autonomous Systems*, vol. 114, pp. 172–186, 2019.

[22] A. Chan, Z. Liang, and N. Vasconcelos, "Privacy Preserving Crowd Monitoring: Counting People Without People Models or Tracking," in *Conference on Computer Vision and Pattern Recognition*, 2008.

[23] M. Shi, Z. Yang, C. Xu, and Q. Chen, "Revisiting Perspective Information for Efficient Crowd Counting," in *Conference on Computer Vision and Pattern Recognition*, 2019.

[24] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-Scene Crowd Counting via Deep Convolutional Neural Networks," in *Conference on Computer Vision and Pattern Recognition*, 2015, pp. 833–841.

[25] Y. Yang, G. Li, Z. Wu, L. Su, Q. Huang, and N. Sebe, "Reverse Perspective Network for Perspective-Aware Object Counting," in *Conference on Computer Vision and Pattern Recognition*, 2020.

[26] W. Liu, M. Salzmann, and P. Fua, "Context-Aware Crowd Counting," in *Conference on Computer Vision and Pattern Recognition*, 2019.

[27] W. Liu, K. Lis, M. Salzmann, and P. Fua, "Geometric and Physical Constraints for Drone-Based Head Plane Crowd Density Estimation," *International Conference on Intelligent Robots and Systems*, 2019.

[28] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, and C. Rother, "Detecting Unexpected Obstacles for Self-Driving Cars: Fusing Deep Learning and Geometric Modeling," in *IEEE Intelligent Vehicles Symposium*, 2017.

[29] K. Gupta, S. A. Javed, V. Gandhi, and K. M. Krishna, "MergeNet: A Deep Net Architecture for Small Obstacle Discovery," in *International Conference on Robotics and Automation*, 2018.

[30] L. Sun, K. Yang, X. Hu, W. Hu, and K. Wang, "Real-time fusion network for RGB-D semantic segmentation incorporating unexpected obstacle detection for road-driving images," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5558–5565, 2020.

[31] K. Chaudhury, S. Diverdi, and S. Ioffe, "Auto-Rectification of User Photos," in *International Conference on Image Processing*, 2014.

[32] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in *Conference on Computer Vision and Pattern Recognition*, 2017.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A Large-Scale Hierarchical Image Database," in *Conference on Computer Vision and Pattern Recognition*, 2009.

[34] S. Liang, Y. Li, and R. Srikant, "Enhancing the Reliability of Out-Of-Distribution Image Detection in Neural Networks," in *International Conference on Learning Representations*, 2018.

[35] D. Hendrycks and K. Gimpel, "A Baseline for Detecting Misclassified and Out-Of-Distribution Examples in Neural Networks," in *International Conference on Learning Representations*, 2017.

[36] H. Blum, P.-E. Sarlin, J. Nieto, R. Siegwart, and C. Cadena, "Fishyscapes: A Benchmark for Safe Semantic Segmentation in Autonomous Driving," in *International Conference on Computer Vision*, October 2019.

[37] K. Lee, K. Lee, H. Lee, and J. Shin, "A Simple Unified Framework for Detecting Out-Of-Distribution Samples and Adversarial Attacks," in *Advances in Neural Information Processing Systems*, 2018, pp. 7167–7177.

[38] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles," in *Advances in Neural Information Processing Systems*, 2017.

[39] J. Mukhoti and Y. Gal, "Evaluating Bayesian Deep Learning Methods for Semantic Segmentation," in *arXiv Preprint*, 2018.

[40] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *Conference on Computer Vision and Pattern Recognition*, 2017.

[41] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick, "Microsoft COCO: Common Objects in Context," in *European Conference on Computer Vision*, 2014, pp. 740–755.