Physics-Based Rendering for Improving Robustness to Rain
Supplementary

Shirsendu Sukanta Halder
Inria, Paris, France
shalder@cs.iitr.ac.in

Jean-François Lalonde
Université Laval, Québec, Canada
jflalonde@gel.ulaval.ca

Raoul de Charette
Inria, Paris, France
raoul.de-charette@inria.fr

https://team.inria.fr/rits/computer-vision/weather-augment/

1. Field of view of a drop in a sphere

We estimate the field of view (FOV) of a drop when projected on a sphere to compute the radiance and chromaticity of each streak, as detailed in sec. 3.3 of the main paper. Despite its motion, we make the assumption of a constant field of view within a given exposure time. This is acceptable due to the short exposure time used here (i.e. 2ms for Kitti, 5ms for Cityscapes). For each drop, the simulator outputs the start position (i.e. shutter opening) and end position (i.e. shutter closing) in both the 3D camera-centered and the 2D image coordinate frames.

We refer to the figure 1 for a geometrical illustration of the following. Let us consider an imaged drop D, having 3D start position $\mathbf{X_0}$ and end position $\mathbf{X_1}$. We first compute $\mathbf{X} = \frac{\mathbf{X_0}+\mathbf{X_1}}{2}$ the *assumed constant* position for which we will estimate the corresponding FOV. The position being camera-centered, the drop viewing direction is therefore $\mathbf{d} = \frac{\mathbf{X}}{||\mathbf{X}||}$.

We compute the equation of the plane P going through $\mathbf{X}$ and orthogonal to the viewing direction $\mathbf{d}$:

$$P = \mathbf{d}_x + \mathbf{d}_y + \mathbf{d}_z - \mathbf{d} \cdot \mathbf{X} = 0\,, \tag{1}$$

where $\cdot$ is the dot product and select a random vector $\mathbf{u}$ (with $||\mathbf{u}|| = 1$) lying on P. Accounting for the field of view of the drop $\theta \approx 165°$ (according to [6]), we compute an arbitrary vector $\mathbf{v}$ on the viewing cone *through* the drop

$$\mathbf{v} = \mathbf{d} \cdot \mathbf{R_u}(\theta/2)\,, \tag{2}$$

with $\mathbf{R_u}(\theta/2)$ the 3x3 general rotation matrix of angle $\theta/2$ about vector $\mathbf{u}$. We use $\theta/2$ because the cone being symmetric along the viewing direction, the complete cone field of view obtain is θ. The set V' of vectors forming the viewing cone through the drop is obtained by the rotation of $\mathbf{v}$ all around the viewing direction. Formally,

$$V' = \{\mathbf{v} \cdot \mathbf{R_d}(\alpha) \mid \forall\, \alpha \in [0, 2\pi[\}\,, \tag{3}$$

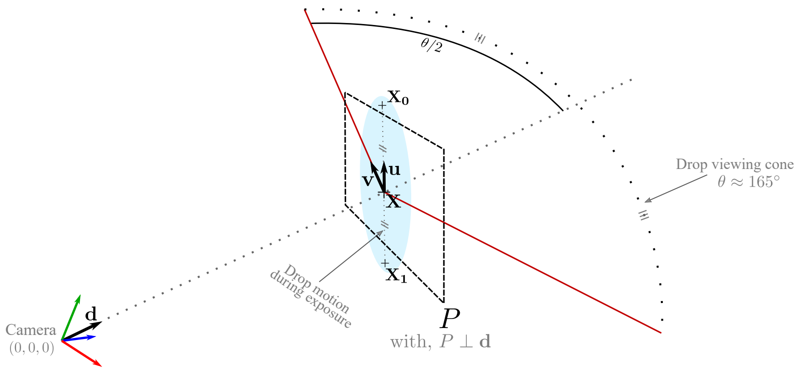

Figure 1. Geometrical construction to compute a drop FOV. Considering $\mathbf{X_0}$ and $\mathbf{X_1}$ the drop position at shutter opening and closing, respectively. We assume a constant drop position $\mathbf{X} = \frac{\mathbf{X_0}+\mathbf{X_1}}{2}$ (during the exposure time, a few milliseconds). Note that we drew only a slice of the drop FOV for simplicity but a full 3D visualization would show a full 3D cone. The drop FOV *in* the environment map is the projection of the 3D drop FOV on the scene sphere of constant distance (refer to text for details).

with $\mathbf{R_d}(\alpha)$ the rotation matrix of α around vector $\mathbf{d}$. In practice, V' is a finite set of radially equidistant vectors (for computational reason we use $|V'| = 20$).

To compute the coordinates of the drop FOV in the environment, we assume a projection sphere S of radius 10m. Hence, we compute the set $Q = \{\phi(S, \mathbf{v}') \mid \forall\, \mathbf{v}' \in V'\}$ of points where vectors intersect the environment sphere, considering only the positive viewing direction axis. Given that the sphere is centered to the camera position and all drops 3D positions are expressed in the camera referential, the intersection $\phi(S, \mathbf{v}')$ of a vector $\mathbf{v}'$ and sphere S of radius S_ρ is straight-forward with

$$\begin{aligned}
\phi(S, \mathbf{v}') &= \mathbf{v}' + t\mathbf{d} \,\text{with,} \\
t &= \frac{-b + \sqrt{b^2 - 4ac}}{2a}\,, \\
a &= \mathbf{d}_x^2 + \mathbf{d}_y^2 + \mathbf{d}_z^2\,, \\
b &= 2(\mathbf{d}_x\mathbf{v'}_x + \mathbf{d}_y\mathbf{v'}_y + \mathbf{d}_z\mathbf{v'}_z)\,, \\
c &= \mathbf{v'}_x^2 + \mathbf{v'}_y^2 + \mathbf{v'}_z^2 - S_\rho^2\,.
\end{aligned} \tag{4}$$

Having computed Q, the finite set of 3D positions intersecting our environment sphere S, the set Q' of spherical coordinates (azimuth, altitude) are obtained from simple Cartesian to spherical mapping, and directly translated to the environment map. Thus, Q' is the projection of the drop FOV on the environment map.

Accounting for implementation details, one may note that Q' is a discrete representation of the drop field of view contours. In practice, a polygon filling algorithm is used to obtain the drop FOV F, which we use for computing the photometry of a rainstreak (cf. sec. 3.3.2 of the main paper).

2. Compositing a rain streak with different exposure time

In their seminal work, Garg and Nayar [4] demonstrated that the streak appearance is closely related to the amount of time τ a drop stays on a pixel. It is thus important to account for the difference of exposure time in the streak appearance database [5] when adding rain to existing images. Given that the appearance database does not provide enough calibration data to recompute the exact original τ_0, we estimate it using observations made in appendix 10.3 of [6]. The latter states that for a constant exposure time τ can be safely approximated with $\sqrt{a}/50$ (a the drop diameter, in meters), which we use to compute τ_0 according to simulation settings in [5].

Using the notation defined in eq. (6) from the main paper, the radiance of streak S' is corrected with

$$S' \frac{\tau_1}{\tau_0}\,, \tag{5}$$

where τ_1 is the time the current drop stays on a pixel, as obtained in a streak-wise fashion by the physical simulator. Noteworthy, [6] also emphasizes that for a given streak the changes of τ across pixels are negligible, so τ can safely be assumed constant.

Finally, after normalization, the alpha of each streak is scaled according to τ_1 and the targeted exposure time T. According to Garg and Nayar equations (cf. eq. (18) from [6]), the composite rainy image is an alpha blending of the background image I_{bg} and the rain layer I_r. For pixel $\mathbf{x}$ corresponding to $\mathbf{x}'$ in the streak coordinates, it leads to:

$$\begin{aligned}
I_{rainy}(\mathbf{x}) &= \alpha I_{bg}(\mathbf{x}) + I_r(\mathbf{x}')\,, \\
&= \frac{T - S'_\alpha(\mathbf{x}')\tau_1}{T} I(\mathbf{x}) + S'(\mathbf{x}')\frac{\tau_1}{\tau_0}\,.
\end{aligned} \tag{6}$$

3. Rendering fog

Rendering fog was done in [15] but we include this in our framework as well to provide more comprehensive weather augmentation capabilities.

Fog results of the formation of small suspended water droplets (10^{-3}–10^{-2} mm) at the surface of the earth and is defined by its extinction coefficient β, closely related to the optical thickness of the fog. For an observer, the Koschmieder law define the maximum visibility in fog as: $V = -\ln(C_T)/\beta$, where C_T is the minimum identifiable contrast (typically 0.05 for humans [10]). As reference, moderate/dense fog has a maximum visibility distance of 375 m ($\beta = 8$) and 37.5 m ($\beta = 80$), respectively. Regardless of the fog intensity, fog can be accurately simulated as a volumetric effect since any light ray intersect a large number of particles [9].

The common strategy for fog simulation is to assume the visual attenuation to be constant through time and space [15]. However, this doesn't account for the spatially varying extinction coefficient of the fog which may change drastically at the few-meter scale. Instead, we use the heterogeneous fog volume described in [7] which vary spatially ensuring a continuous gradient.

The resulting radiance $L(d)$ when imaging a scene point at distance d (m) in fog is the sum of the directly transmitted radiance D and the airlight radiance A:

$$L(d) = D(d) + A(d) = L_0 e^{\beta d} + L_\infty (1 - e^{-\beta d}) \tag{7}$$

where L_∞ is the airlight radiance at the horizon, L_0 the original scene radiance (i.e. the radiance in clear weather) and β (m^{-1}) the extinction coefficient function of the fog thickness. Consequently, the radiance of a scene point decreases as its distance from the observer increases. Inversely, airlight radiance increases with the path length producing the bright whitish appearance of fog.

Given a clear weather image I (i.e. without optical extinction), the foggy version I_{fog} from depth map d is straightforward from eq. (7). For pixel $\mathbf{x}$:

$$I_{\text{fog}}(\mathbf{x}) = I(\mathbf{x}) e^{\beta(t) d(\mathbf{x})} + L_{\text{fog}}(1 - e^{-\beta(t) d(\mathbf{x})}) \,, \tag{8}$$

where $\beta(t)$ is the extinction β spatially varying along the ray path for heterogeneous fog [7], from the imaging sensor to the scene point imaged, that is $t \in [0, d(\mathbf{x})]$.

4. Qualitative results on rainy images

Visual outputs of qualitative semantic segmentation on nuScenes [1] are reported in fig. 2. To produce ground truths, available on the project page, frames were hand labeled using Cityscapes [2] encoding. The finetuning process is described in Sec. 6 of the main paper. Note that as mentioned, only Cityscapes augmented with our rain rendering pipeline is used in the training. Subsequently, the network did *not see any* nuScenes data at training stage.

Despite the different Cityscapes/nuScenes imaging setups, when PSPNet [20] is finetuned with our synthetic rain data the semantic segmentation - though not perfect - is significantly more robust to real rain (AP is 25.6% finetuned versus 18.7% untuned). This advocates the usefulness of our rain rendering pipeline to increase robustness of real rainy applications. Noteworthy, labels such as road (purple) or car (blue) that are important for robot navigation, exhibit a significant increase (rows 1, 2 fig. 2) with our finetuning.

5. Qualitative results with rain augmentation

In the figs. 3, 4 for object detection and figs. 5, 6 for semantic segmentation, we show additional qualitative results of both our rain and fog generation and the evaluation of the most representative algorithms on the two tasks of object detection and semantic segmentation.

5.1. Object detection

Note that only detections with confidence $\geq$0.7 are displayed, and that we consider car/truck/van/train as a single class of vehicles. Almost every algorithm experiences a steady decrease in the confidence of the predicted bounding box but overall object detection is rather robust to light/moderate fog and rain events. In thick fog and high rains, the majority of the algorithms like DSOD [16], RFCN [3], SSD [8] and YOLOv2 [11] suffer a decrease in the number of detected objects.

5.2. Semantic segmentation

For semantic segmentation, we use the same color-coding as stated by the Cityscapes dataset [2]. The figures displayed give a clear indication of the complete breakdown of all algorithms especially in case of 100+ mm/hr rain, which we believe is due to the interference due to the streaks. The predicted semantics in case of heavy rains are in complete dissonance when compared to the spatial coherence of the clear weather image as well as there are a lot of erroneous label predictions. On the other hand, in fog, there are relatively less incorrect labels but the accuracy is inversely proportional with the distance. This is not too surprising since, with increasing distance, objects are relatively less visible which causes the algorithms to miss them. In rain, the conjunction of sharp close-by streaks and distant optical attenuation lead to a less predictable outcome.

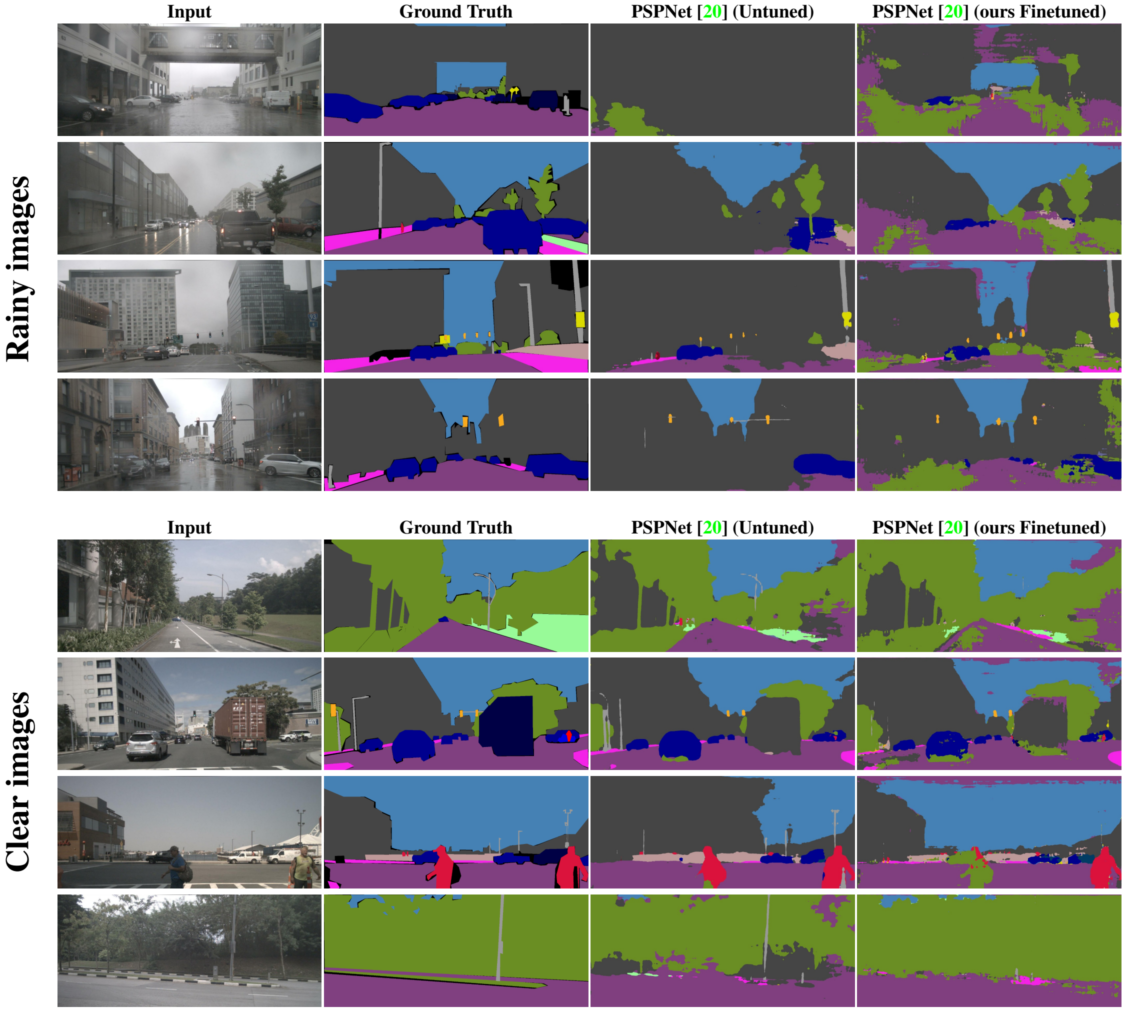

Figure 2. Semantics segmentation on rainy images and clear images from the nuScenes dataset [1] (cropped for visualization). For each image, we show the ground -truth annotated segmentation map, original PSPNet [20] performance (untuned) and the performance of PSPNet performance when finetuned with our rain rendering pipeline.

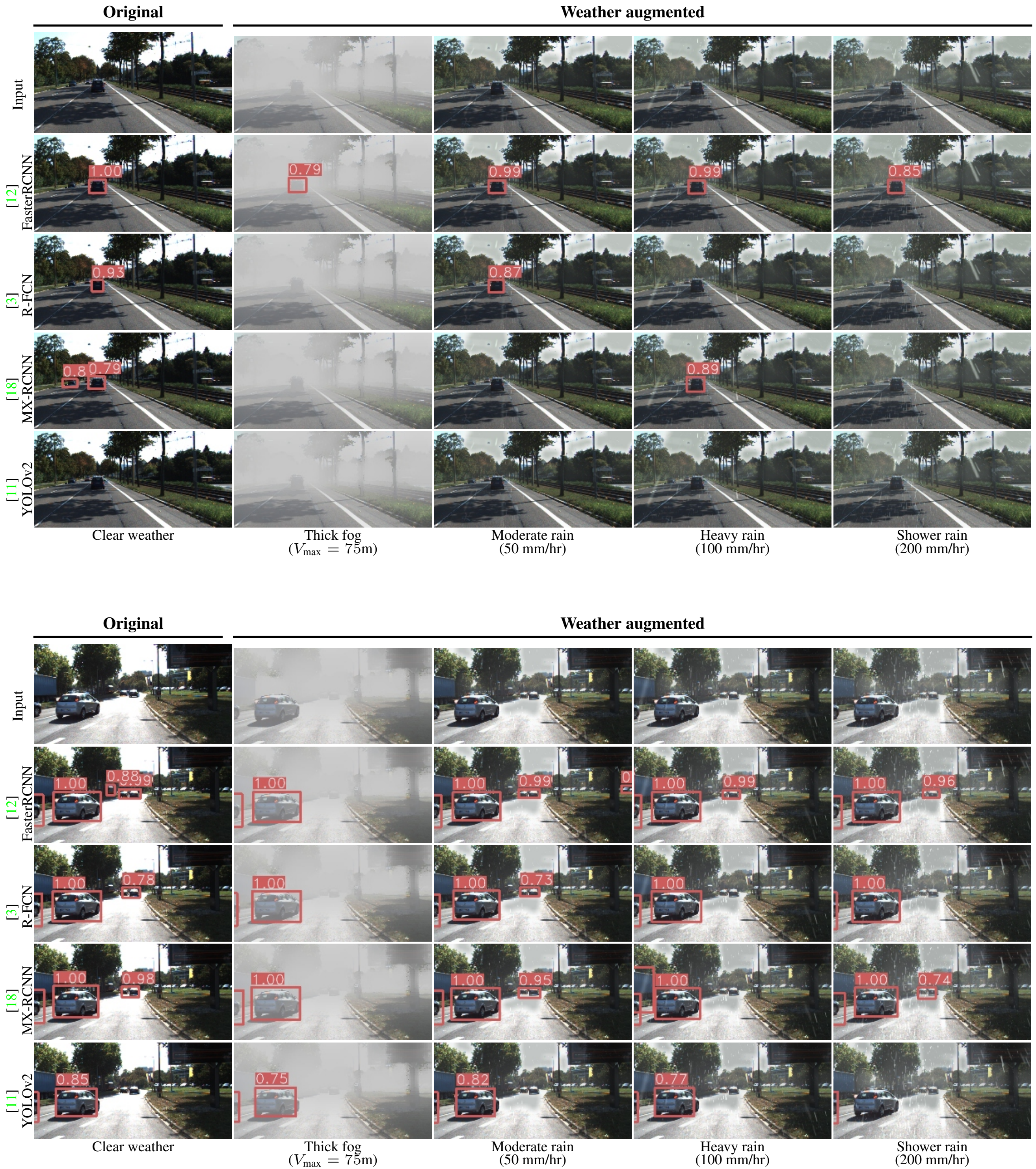

Figure 3. (Qualitative evaluation of object detection on our weather augmented Kitti dataset (cropped for visualization). From left to right, the original image (clear) and four weather augmented images. Most object detectors such as MX-RCNN [18] or R-FCN [3] fail at detecting objects in extreme rain conditions (200 mm/hr), while Faster R-CNN [12] shows robustness to rain but decrease its detection confidence. YoloV2 [11] experiences a decrease in confidence as well as failures in detecting objects with increase in rain intensity. Only detections with confidence ≥ 0.7 are shown.

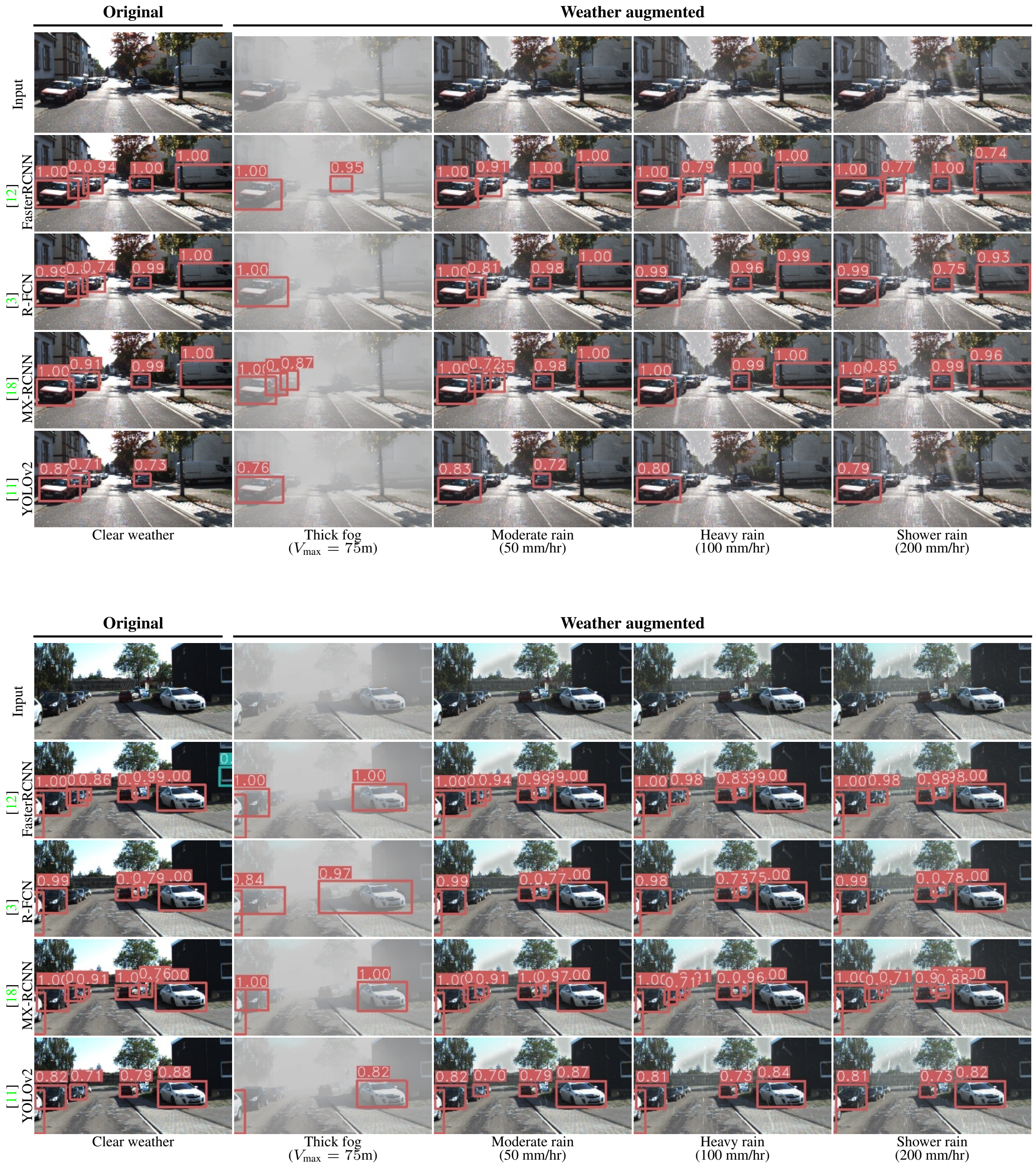

Figure 4. Qualitative evaluation of object detection on our weather augmented Kitti dataset (cropped for visualization). From left to right, the original image (clear) and four weather augmented images. In addition to a gradual decrease in the confidence of the bounding box when thresholded, the number of objects detected using R-FCN [3] or Faster R-CNN [12] subsequently decrease with rain. This behaviour is especially visible in the fog and 100+ mm/hr rain of both the set of images. With heavy rain YOLOv2 [11] output confidence of bounding boxes around objects decrease and faraway objects are not detected. Only detections with confidence ≥ 0.7 are shown.

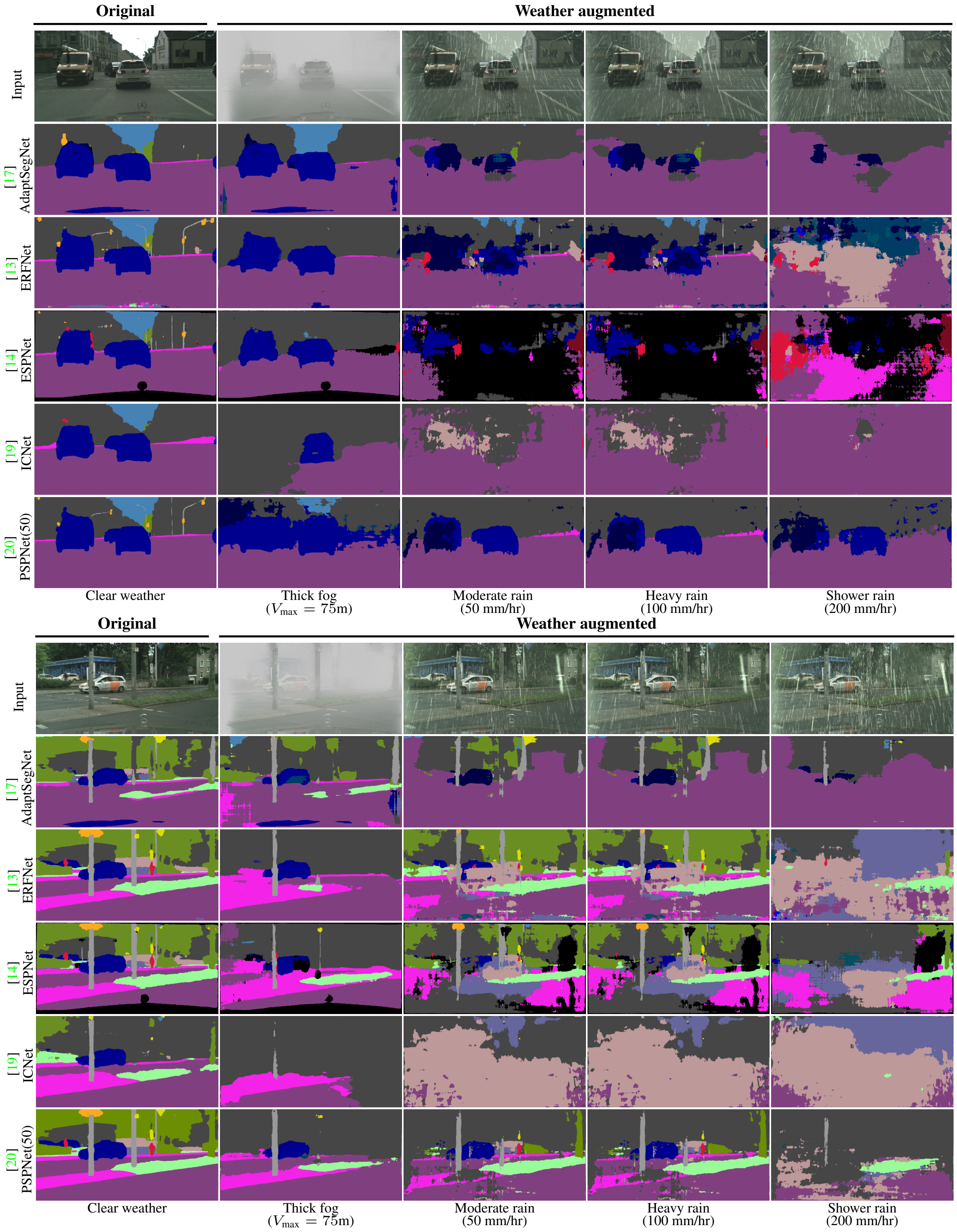

Figure 5. Qualitative evaluation of semantic segmentation on weather augmented Cityscape dataset (cropped for visualization). From left to right, the original image (clear) and four weather augmented images. It is clearly visible that in cases of moderate and high rains (50+ mm/hr) all algorithms predict wrong labels at many regions in the image scene. The semantic predictions also tend to neglect some object labels when there are occlusions caused by dense fog and increased rainfall rates which imply optical attenuation for both and high frequencies pattern for rain only. ESPNet [14], ERFNet [13] and ICNet [19] experience a complete break down as visible in both the set of images.

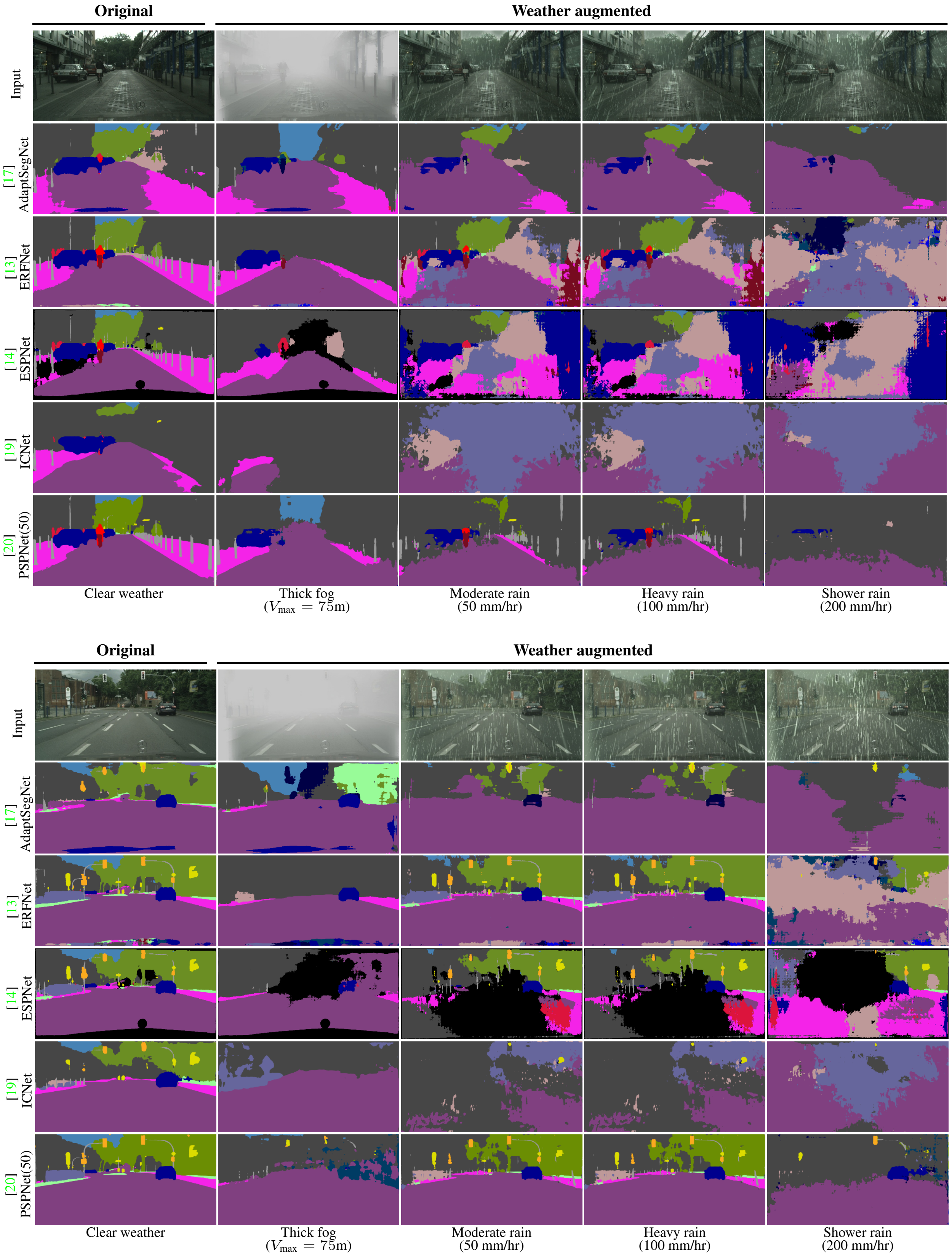

Figure 6. Qualitative evaluation of semantic segmentation on weather augmented Cityscape dataset (cropped for visualization). From left to right, the original image (clear) and four weather augmented images. It is clearly visible that in cases of moderate and high rains (50+ mm/hr) all algorithms predict wrong labels at many regions in the image scene. In the bottom set, ERFNet [13], ESPNet [14] and PSPNet(50) [20] still output valid labels in shower rain in the upper left/right part of the image. This is probably because less streaks are visible.

References

[1] Holger Caesar and et al. nuscenes: A multimodal dataset for autonomous driving. *preprint arXiv:1903.11027*, 2019. 3, 4

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 3

[3] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, pages 379–387, 2016. 3, 5, 6

[4] Kshitiz Garg and Shree K Nayar. When does a camera see rain? In *IEEE International Conference on Computer Vision*, volume 2, pages 1067–1074. IEEE, 2005. 2

[5] Kshitiz Garg and Shree K Nayar. Photorealistic rendering of rain streaks. In *ACM Transactions on Graphics (SIGGRAPH)*, volume 25, pages 996–1002. ACM, 2006. 2

[6] Kshitiz Garg and Shree K Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1):3–27, 2007. 1, 2

[7] Sule Kahraman and Raoul de Charette. Influence of Fog on Computer Vision Algorithms. Research report, Inria Paris, Sept. 2017. 3

[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, 2016. 3

[9] Srinivasa G Narasimhan and Shree K Nayar. Vision and the atmosphere. *International Journal of Computer Vision*, 48(3):233–254, 2002. 2

[10] Ales Prokes. Atmospheric effects on availability of free space optics systems. *Optical Engineering*, 48(6):066001, 2009. 2

[11] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 5, 6

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. 5, 6

[13] Eduardo Romera, José M Alvarez, Luis M Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018. 7, 8

[14] Mohammad Rastegari Sachin Mehta, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *European Conference on Computer Vision*, 2018. 7, 8

[15] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, Sep 2018. 2, 3

[16] Zhiqiang Shen, Zhuang Liu, Jianguo Li, Yu-Gang Jiang, Yurong Chen, and Xiangyang Xue. Dsod: Learning deeply supervised object detectors from scratch. In *IEEE International Conference on Computer Vision*, 2017. 3

[17] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 7, 8

[18] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5, 6

[19] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *European Conference on Computer Vision*, 2017. 7, 8

[20] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 4, 7, 8