Semantic Cameras for 360-Degree Environment Perception in Automated Urban Driving

Andra Petrovai and Sergiu Nedevschi, *Member, IEEE*

***Abstract*— The European UP-Drive project addresses transportation-related challenges by providing key contributions that enable fully automated vehicle navigation and parking in complex urban areas, which results in a safer, inclusive, affordable and environmentally friendly transportation system. For this purpose, the project consortium developed a prototype electrical vehicle equipped with cameras and LiDARs sensors that is capable to autonomously drive around the city and find available parking spots. In UP-Drive, we created an accurate, robust and redundant multi-modal environment perception system that provides 360° coverage around the vehicle. This paper summarizes the work of the project related to the surround view semantic perception using fisheye and narrow field-of-view semantic virtual cameras. Deep learning-based semantic, instance and panoptic segmentation networks, which satisfy requirements in accuracy and efficiency have been developed and integrated into the final prototype. The UP-Drive automated vehicle has been successfully demonstrated in urban areas after extensive experiments and numerous field tests.**

***Index Terms*— Automated driving, environment perception, image segmentation.**

I. Introduction

THE future of mobility is the automation of individual transportation, which will bring major social, economical and ecological benefits. Automated electric transportation will alleviate challenges regarding massive urbanization and persistent traffic congestion, and a more efficient coordination of vehicles in traffic will address the problem of climate change with reduced greenhouse gas emissions. Moreover, automated driving means increased safety on roads by removing human error. Car-sharing will become more attractive and the mobility of the aging or disabled population will be improved. However, due to the lack of maturity of key technologies especially in the context of complex urban areas, full vehicle automation is a long-time vision and an ongoing effort from academia and the automotive industry.

The H2020 European UP-Drive project [4] addresses these technological challenges by developing a fully automated electrical vehicle that is able to safely navigate in a complex urban environment with use cases such as robot taxi or automated valet parking service. Solving automated driving in urban areas is an extremely difficult challenge, and in order to accommodate the limited 4-year time frame of the project, the area of application has been restricted to urban 30 km/h zones and mapped areas.

Fig. 1. The UP-Drive VW e-Golf prototype performs fully automated driving in urban areas. Semantic environment perception is achieved with five semantic cameras that constantly monitor 360° around the vehicle.

The UP-Drive vehicle is capable of automatically finding empty parking spots in urban areas, using a long-term semantic map shared between multiple agents, is able to autonomously drive to the available parking spot, park in and also to drive to any pick-up place around the city. Creating an advanced system for automated driving and parking requires development and research of key technologies: robust surround view environment perception, accurate life-long metric localization and mapping, scene understanding and aggregation of semantic data over a cloud-based infrastructure.

The goal of perception in autonomous driving applications is to detect, track, classify and represent the objects in the driving environment. The key elements to achieve these tasks are a redundant, robust, accurate and multi-modal sensory system providing a 360° coverage of the vehicle surrounding. For a robust perception, redundancy must be ensured at algorithmic level, but also at sensory level, with different types of sensors providing geometric, semantic, motion information. Towards this goal, the UP-Drive vehicle is equipped with fisheye cameras that provide 360° coverage, front narrow field of view cameras for increased depth range and 360° LiDARs for 3D perception.

For safe navigation in urban areas, we develop a robust and fast 360° 2D perception solution with semantic cameras that is deployed on the UP-Drive vehicle. The motivation to develop such a semantic perception solution stems from a key element

Manuscript received 27 July 2020; revised 8 April 2021 and 8 October 2021; accepted 9 February 2022. Date of publication 14 March 2022; date of current version 11 October 2022. This work was supported in part by the EU H2020 UP-Drive Project under Grant 688652 and in part by the Integrated Semantic Visual Perception and Control for Autonomous Systems (SEPCA) Project under Grant PN-III-P4-ID-PCCF2016-0180. The Associate Editor for this article was Y. Chen. *(Corresponding author: Andra Petrovai.)*

The authors are with the Department of Computer Science, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania (e-mail: andra.petrovai@cs.utcluj.ro; sergiu.nedevschi@cs.utcluj.ro).

Digital Object Identifier 10.1109/TITS.2022.3156794

of our 3D perception system: the low-level representation of the environment that is built by fusing semantic, instance and geometric information. In order to achieve this, the proposed semantic virtual cameras need to cover the same 360° area around the vehicle, in the near and far depth range, as the LiDAR sensors. From the multitude of sensors mounted on the vehicle, the camera system is best at capturing the semantics of the environment: images provide rich appearance information such as color and texture which makes the semantic segmentation in the 2D space the most robust. The goal of our 2D perception system is to detect static road infrastructure, but also to detect dynamic objects such as road users. In our holistic panoptic image segmentation approach, each pixel in the image is uniquely assigned a semantic label and also an instance identifier. The panoptic information is further used by the low-level sensor fusion module, in which the semantic and instance information extracted from images is associated to the 3D point clouds from LiDARs. The augmented 3D point cloud is further processed to classify 3D objects using the semantic information and to refine raw 3D bounding boxes based on the instance identifiers. The classified 3D objects provided by the perception module are further processed by other modules in the software stack, such as scene understanding, motion planning, maneuver prediction, decision making or traffic flow prediction [7]. Modeling the interaction between traffic participants is highly correlated with their semantics since assumptions about their behavior could be implemented.

This paper describes the 360° 2D semantic environment perception system, more specifically, the proposed deep learning-based semantic virtual cameras that provide semantic, instance and panoptic segmentation by processing images from four fisheye and one narrow field-of-view frontal camera. Our semantic cameras need to meet requirements of high accuracy and low processing time in order to enable fully automated navigation of the vehicle in urban areas. In order to keep the processing time reasonable (100-150 ms) given the limited hardware resources, we perform semantic segmentation on all five images and detect instances only from the frontal fisheye and narrow field-of-view images, since the frontal area offers the most relevant information for navigation. We also develop an original panoptic segmentation solution [13] which unifies semantic and instance segmentation of the front fisheye image and increases the segmentation accuracy. The instance and panoptic segmentation can be easily extended to other views, when using more powerful hardware.

To summarize, our main contributions are the following:

- We describe the complete process of developing a 360° 2D semantic environment perception system that can be integrated into a fully automated software stack and discuss the challenges associated with bringing a lab prototype to operate in real-world conditions. To the best of our knowledge, this is the first work that thoroughly presents each step of the process from a practical perspective: from camera setup to network design and implementation, dataset creation, network training and testing, network deployment, and software integration. We also highlight the problems that we encountered and our solutions towards developing a fully working system in the real environment that can operate reliably and fast on limited hardware with low power consumption.
- Our 360° 2D semantic environment perception system meets requirements of high accuracy and low inference time, while processing images from five cameras: four fisheye and one narrow field-of-view camera.
- We propose deep learning-based semantic virtual cameras that provide pixel-level semantic, instance and panoptic information. Our novel panoptic segmentation module unifies semantic and instance segmentation such that each pixel is uniquely assigned a semantic and instance label. Our panoptic module increases the segmentation accuracy and facilitates the low-level fusion with the 3D point clouds as part of the 3D perception system.
- Our solution has been deployed on a prototype vehicle and integrated into the fully automated software. We provide implementation details and thoroughly describe the integration process. We also perform extensive experiments on our UP-Drive dataset and numerous field tests with the prototype vehicle in urban environment.
- We find that, from a practical perspective, detecting distant objects is important especially when driving at high speeds. Fisheye cameras limit our detection range due to its wide-angle lenses. We propose using high resolution images, as well as network optimization and quantization, to extend the detection range, while keeping the processing time low. Another important finding is that by adding a narrow field-of-view camera to our system, we can extend the detection range for pedestrians three times compared to fisheye cameras.

II. Related Work

In this section, we review semantic, instance and panoptic segmentation networks from the literature.

A. Semantic Segmentation

Semantic segmentation partitions an image into meaningful segments, which share a common representation. Each pixel in the image receives a semantic class that belongs to either *stuff* or *things* categories. *Stuff* classes represent amorphous and uncountable elements in the scene, that usually have repetitive texture, but not a fixed shape or size. In the driving environment, examples of *stuff* classes include road, sidewalk, building, nature. On the other hand, *things* classes define objects that can be counted and have a specific shape. Road users belong to this category: vehicles, pedestrians and cyclists. Fully Convolutional Networks (FCN) [31] are widely used in the semantic segmentation task, due to their state-of-the-art results on public benchmarks, such as Cityscapes [12], Mapillary Vistas [32] or COCO [29]. In FCNs, fully-connected layers from a CNN are replaced with convolutional layers. Two important aspects should be considered for accurate pixel level classification: how to capture context cues and how to maintain details of finer scales such as shapes and boundaries. [8], [9] enlarge the receptive field of classification CNNs by adopting dilated convolutions in the last residual blocks, while providing a higher output resolution. Deformable

convolutions [14] avoid dilation stride engineering by learning filters with adaptive receptive fields. Spatial Pyramid Networks (PSPNet) [50] capture global image information by employing parallel pooling operations and fusing features at different scales. [8] proposes Atrous Spatial Pyramid Pooling (ASPP) with parallel dilated convolutions. The drawback of dilated convolutional neural networks is that they have a high memory footprint, since they generate a high output stride of $8\times$.

Another popular CNN architecture for semantic segmentation is the encoder-decoder [26], [43], [46]: the encoder network learns hierarchical feature representations, while the decoder network upsamples feature maps to the input image resolution and recovers spatial information. In [44], the authors propose an encoder-decoder network for free space segmentation that processes RGBD data, in the context of a real-time multi-sensor perception framework for automated vehicles.

The end-to-end training in the multi-task setting can improve accuracy compared to separately trained networks for each task and leads to a solution that can generalize better [20]. Therefore, semantic segmentation has been coupled with object detection and instance segmentation tasks in [13], [21], [48], where the authors extend the state-of-the-art two-stage object detector and instance segmentation network Mask R-CNN [17] with a semantic segmentation head. These two-stage networks provide accurate results but are difficult to train due to the high number of design parameters. The complexity of the inference pipeline makes deployment and optimization with deep learning inference engine such as TensorRT [2] difficult to achieve.

Automated driving requires real-time performance of semantic segmentation algorithms. From the above mentioned methods, the encoder-decoder architectures achieve the best trade-off between accuracy and latency. Therefore, we employ the efficient ERFNet [43] network for semantic segmentation of fisheye images, which delivers high quality segmentation at reduced computational costs.

B. Instance Segmentation

Instance segmentation predicts a semantic mask and an instance identifier for each object such that we can differentiate between objects belonging to the same class. Classification is performed at instance-level, which means that object masks could overlap. All pixels from an instance mask have the same semantic class and the same instance identifier. Instance segmentation approaches follow in general two directions: segmentation of candidate regions or segmentation of instances without proposals.

The most representative solution for proposal-based methods is Mask-RCNN [17], which has demonstrated outstanding performance on public benchmarks [12], [29]. The authors extend the two-stage object detector Faster R-CNN [42] with a mask prediction head, which performs segmentation inside detected 2D bounding boxes. Mask R-CNN achieves scale invariance by introducing the RoiAlign operation which samples a fixed number of features inside each candidate box. Cascade R-CNN [6], Non-local networks [47], PANet [30] bring improvements to Mask R-CNN at different stages of the pipeline and provide accurate masks, but at reduced inference speed and are not suitable for real-time processing. With the introduction of specialized losses such as Focal Loss which addresses the class imbalance problem or by using training tricks and multi-scale predictions [41], single-stage object detectors manage to obtain on-par accuracy with two-stage detectors while being significantly faster. RetinaMask [15] integrates a mask prediction head on top of RetinaNet [28], improves the loss function and includes hard examples in training, thus increasing the detection performance while keeping the computational cost of the detector network.

Proposal-free methods perform semantic segmentation and cluster pixels belonging to the same instance based on similarity measures. Kendall *et al.* [20] and Neven *et al.* [33] propose learning an offset vector for each pixel that points to the instance centroid. PersonLab [34], CornerNet [24] introduce keypoint guided instance segmentation, while Deep Polygon Transformer [25] and DeepSnake [36] formulate instance segmentation as the problem of fitting a polygon around the object. TensorMask [10] employs 4D Tensors and demonstrates advantages over 3D Tensors with increased computational costs. DWT [5] models the energy of the watershed transform with CNNs, but cannot handle objects separated into multiple parts. Instance segmentation can be also viewed as a graph partition problem in which the total score of edges connecting different components is maximized, but these types of methods [16], [23] are currently time-consuming due to the complexity of the graph space.

Proposal-based methods achieve top-performing results for instance segmentation and single-stage approaches enable fast inference and are suitable for automated driving applications. We employ RetinaMask [15] in our image segmentation module, due to its high accuracy, reduced computational costs and more importantly the reduced complexity of the inference pipeline, which allows for further acceleration with deep learning inference engines [2] and eases the deployment in the perception software.

C. Panoptic Segmentation

Panoptic segmentation provides a unified semantic and instance representation. It performs dense pixel classification into *things* and *stuff* classes and assigns an instance identifier to every *things* pixel in the image. Panoptic segmentation can be achieved by simultaneously solving two other tasks: semantic and instance segmentation. Kirillov *et al.* has formally defined the task of panoptic segmentation in [22] and has proposed a baseline approach in [21]. They integrate a semantic segmentation head in the Mask R-CNN framework [17] and propose merging heuristics for instance and semantic segmentation that solve overlaps and semantic class mismatches. [39], [48] propose an end-to-end trainable network that employ semantic and instance logits to build the panoptic output which is directly learned inside the network. [40] achieves state-of-the-art results by designing a multi-scale fusion layer that facilitates information flow within the network. Although these

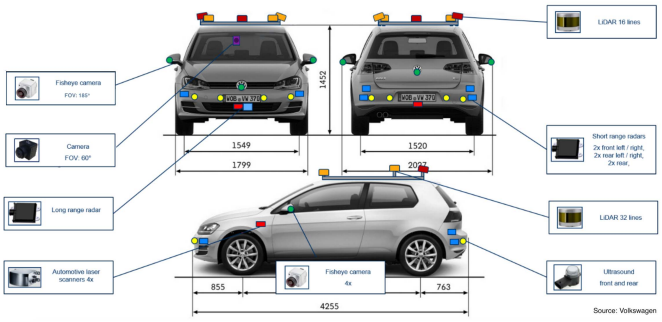

Fig. 2. Test vehicle setup. The vehicle is equipped with 5 cameras, 11 LiDARs and 7 radars.

methods provide accurate results, their high latency makes them unsuitable for use in the automated driving context.

We propose an original panoptic segmentation fusion scheme [13] that corrects semantic classification using instance classes and improves instance masks by propagating instance identifiers on semantic masks. The advantage of our method is that it is fast, it provides improved and unified semantic and instance segmentation output and can be easily integrated as a post-processing step on top of any image segmentation networks.

III. Test Vehicle Setup

The test vehicle platform for the UP-Drive project is the fully electric VW e-Golf. Three different types of sensors have been mounted on the vehicle in order to guarantee a 360° 3D multimodal environmental perception: cameras, radars and LiDARs. Five externally synchronized cameras are integrated into the sensor setup in order to cover the near and far-range surround view of the vehicle: one 60° field-of-view camera located behind the windshield and four fisheye cameras. The front and rear fisheye cameras are mounted horizontally near the vehicle emblem, while the left and right cameras are mounted on each side mirror and are tilted downwards. With four wide 185° field-of-view cameras, the system offers 360° coverage around the vehicle with some overlap between neighboring cameras. The fisheye system of cameras delivers color images of resolution 1280×800, JPEG compressed at 30 frames/second.

The fisheye system of cameras have wide-angle lenses and an equivalent short focal length, that determine a large extent of the scene to be captured. The apparent size of objects in the images is smaller compared to narrow-field of view cameras. Thus, the fisheye cameras limit the detection range of segmentation algorithms. From our experiments, we obtained robust pedestrian segmentation with fisheye images up to only 25 meters. Detecting distant objects is important especially when driving at high speed. In order to overcome this problem, we introduced a narrow-field of view RGB camera in the setup, mounted behind the windshield. The camera has a 60° horizontal field-of-view, it delivers 1928×1208 resolution images with a framerate of 30 frames/second and extends the detection range three times. The complete sensor suite can be visualized in Figure 2.

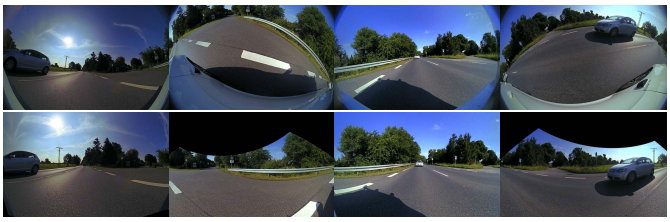

Fig. 3. Results of the fisheye images unwarping process. First row: fisheye images, second row: cylindrical projection of the fisheye images. From left to right: front view, right view, rear view and left view.

IV. 2D Perception With Semantic Cameras

In this section, we describe our 360° 2D semantic environment perception system, as seen in Figure 4. First, we provide details about image pre-processing steps such as undistortion and fisheye unwarping in Section IV-A. Next, we describe our virtual semantic cameras that provide semantic, instance and panoptic segmentation in Section IV-B. We provide details about the datasets used for training and testing, networks architecture, training procedure and the original solution for panoptic segmentation [13]. Finally, network integration and deployment is discussed in Section IV-B.5.

A. Image Undistortion and Unwarping

The raw fisheye images are not used in practice by our perception system because structures and objects in the image are highly distorted due to the wide-angle lenses. Therefore, we apply image undistortion and unwarping in order to obtain a more suitable representation of the scene in the image. Unwarping is the process of backprojecting the fisheye image onto a virtual projection surface. In our case [45], we adopt cylindrical unwarping which generates images with large horizontal field-of-view (HFV) and small distortions, while also preserving the orientation of vertical lines. In the image unwarping process, we reduce the horizontal field-of-view from 185° to 160°. We provide a visual comparison of fisheye and unwarped fisheye images in Figure 3.

B. Deep Semantic, Instance and Panoptic Segmentation

Deep learning extends the problem of classification to representation learning and has shown superior results to algorithms based on hand-crafted features in many computer vision tasks, including semantic, instance and panoptic segmentation. In order to achieve accurate and robust results, the deep neural networks require a large, consistent and high-quality dataset of labeled images, generated manually, semi-automatically or even automatically. Real-time performance is supported by the new generation of GPU devices and advanced studies in neural network optimization, allowing high inference speed with low computational costs on the new low-power GPUs. Considering the advantages of deep learning and the advances in technology, deep neural networks based solutions are suitable for the automated driving perception software.

1) Image Segmentation Dataset: For training and evaluating the proposed deep neural network for semantic and instance segmentation on fisheye images, we employ our internal

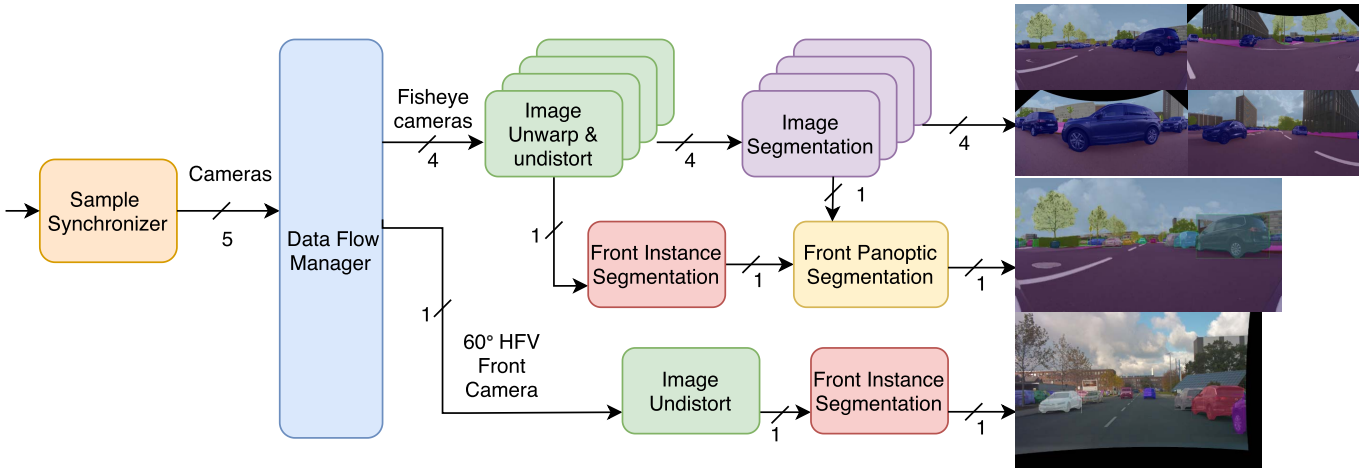

Fig. 4. Our semantic perception system processes input images from five cameras: four fisheye surround-view cameras that cover 360° around the vehicle and one front narrow field-of-view camera. The Data Flow Manager receives the five synchronized image samples and outputs the best temporally aligned set with the beginning of the perception processing cycle. The four fisheye images are first unwarped and undistorted and semantic segmentation follows. For the front unwarped fisheye image, we also perform instance segmentation and implement a fusion between semantic and instance segmentation to finally obtain the panoptic output. Due to time constraints, the narrow field-of-view image is processed only by the instance segmentation algorithm.

UP-Drive dataset. The lack of publicly available datasets for 360° perception, also identified by [49], has driven us to record and annotate our own dataset. To fuel interest towards 360° perception, [49] has recently published sensory data with a different configuration than ours, but which also covers the surround view: left, right fisheye cameras, and front, rear stereo cameras.

Our UP-Drive dataset is large and has been designed to capture a wide variety of outdoor weather and lighting conditions. The data was recorded by driving the car in several cities in northern Germany but also on highways and country roads. Recordings were performed in daytime and account for different lighting conditions, from morning to afternoon. Sequences were acquired in a time span of several months in three seasons: spring, summer and autumn. The diversity of weather conditions has been taken into consideration, therefore the data was recorded in sunny, cloudy weather but also in heavy rain. Lens flare, but also lens distortions from rain drops have been captured. From all the recordings, 19562 non-sequential frames were selected for semantic and instance-level annotation. Images cover the surrounding view of the vehicle: front, left, right and rear. There are 5111 front view images, 4684 left view images, 4800 right view images, 4967 rear view images. The UP-Drive dataset was labeled using similar methodology with the Cityscapes dataset [12]. Pixel-level semantic segmentation annotation is provided for all images into 23 classes, and also instance-level labels for a subset of 6 classes that represent traffic participants. We split the dataset into the training set, with 15782 images and the validation set, with 3780 images.

The image dataset of narrow field-of-view images is relatively less diverse and smaller than the fisheye dataset. It contains 1869 images which are labeled with pixel-level instance masks for 6 classes. We train our network on 1495 images and validate on 374.

2) Semantic Segmentation: We implement a Fully Convolutional Neural Network (FCN) [31] for semantic segmentation of the four fisheye images. In order to correct the distortions introduced by the wide-angle lenses, the fisheye images are undistorted and unwarped at both train and inference time. State-of-the-art semantic architectures are either very deep or wide or employ complex layers at the cost of higher memory usage and higher execution time. Computational resources should be considered when developing perception systems for automated driving since the algorithms must run in real-time on low power hardware in the vehicle. For our segmentation network, we adopt ERFNet [43], an efficient network that achieves a good trade-off between accuracy and efficiency. The network has an encoder-decoder architecture, where the encoder extracts image features at different scales and the decoder combines the features in a higher resolution representation. The building block of ERFNet is the factorized residual layer. This layer represents a 1D non-bottleneck residual module that decomposes a 2D kernel into a linear combination of 1D kernels. In this design, each 3×3 convolution is transformed into 3×1 and 1×3 convolutions. The number of parameters is reduced with 33% when using a kernel size of 3. At the same time, the network is much more memory efficient and faster while having an increased capacity which results in a high accuracy of segmentation similar to more complex models. The feature extractor encodes features at three scales: $1/2$, $1/4$, $1/8$ from the original input resolution by stacking residual 1D non-bottleneck blocks having dilated convolutions. A high output resolution is important in order to preserve detailed information and small objects. Dilation in convolutional layers has proven an efficient mechanism for capturing multi-scale context, which is essential for correct classification. The lightweight decoder is formed of 1D non-bottleneck blocks and recovers spatial and semantic information from the last layer of the encoder.

We implement our model in the PyTorch [35] framework on a system with four Tesla V100 GPUs. The network is trained for 150 epochs with a batch size of 12 images per GPU and a polynomial learning rate decay starting from 0.0025.

The cross entropy loss function is optimized with Adam optimizer. We crop the images to 1280×640 and apply random horizontal flipping augmentation and random left-right translation. The network is initialized with pretrained weights on the Cityscapes dataset [12].

3) Instance Segmentation: The deep instance segmentation network needs to be accurate and efficient and at the same time the architecture should allow optimization and deployment with deep learning inference engines such as TensorRT [2]. In order to keep the processing time of the entire perception system low, we perform instance segmentation only on the front fisheye image and on the narrow-field of view image, since the area in the front of the ego-vehicle contains the most important information about the environment needed for safe navigation.

In the first iterations of the project, we developed a two-stage Mask R-CNN-based solution for semantic and instance segmentation [13]. Two-stage networks propose object candidates in the first stage, while in the second stage, candidate bounding boxes are regressed and classified and a binary mask is predicted for each object. Due to the complexity of the inference pipeline, optimization of the network with TensorRT [2] was hard to achieve. Although the network provides accurate results, the network is computational expensive and cropping and downsampling the fisheye input images from 1280×800 to 512×256 was required in order to reduce the inference time. The difficulties we encountered with network acceleration, deployment and the high inference time even at a smaller resolution, motivated us to search for a more efficient single-shot instance segmentation network with a simplified inference pipeline.

Our final solution for instance segmentation employs RetinaMask [15], which extends the state-of-the-art single-shot object detector RetinaNet [28] with a Mask R-CNN type of instance mask prediction head. For the feature extraction backbone, we use the ResNet-50 [18] with a 5-level Feature Pyramid Network (FPN) [27]. FPN allows multi-scale object detection by encoding multi-resolution representations from $1/4$ to $1/64$. The FPN follows the original implementation [27] with 256 feature maps and 5 anchor scales. A bounding box regression and classification head with four convolutional layers is appended to each level of the pyramid. The bounding box predictions are aggregated, filtered and distributed to layers in the FPN. Next, the ROIAlign [17] operation samples the same number of features (14×14) from each predicted bounding box, which are finally processed by the mask prediction head with four convolutional layers and one transposed convolution. Finally, a $[1 \times 1]$ convolution generates the final class-wise masks of size 28×28.

We train two instance segmentation networks on fisheye and narrow-field of view images from the UP-Drive dataset. The network is pretrained on Microsoft COCO dataset [29] and Cityscapes dataset [12]. We set the batch size to 16 images and train for $30k$ iterations with the base learning rate of 0.01, which is decreased by 10 at $20k$ iteration. Optimization of the loss function is done using Stochastic Gradient Descent (SGD). Fisheye images are cropped to 1280×640 and then scaled during training with the shorter edge of the image randomly sampled from $[480, 640]$. Narrow-field of view images are resized to 832×416 and multi-scale training is performed at scales in $[320, 416]$. We apply random horizontal flipping augmentation.

4) Panoptic Segmentation: We propose an original solution for panoptic segmentation in which we fuse the semantic and instance segmentation output by applying a novel fusion scheme, which efficiently solves instance-level overlaps and conflicts between semantic classes. Our solution is derived from the following observations: instance segmentation masks are more raw due to their low resolution (28×28) and errors at object border could be observed especially in the case of large objects, while semantic segmentation provides good delimitation between *things* and *stuff* pixels but confuses classes belonging to the same category. In our implementation, we use three categories: vehicles, humans and two-wheeled and six classes: bicycle, motorcycle, bus, car, truck, person. In the first step of the fusion process, we divide pixels into *things* and *stuff* based on the semantic segmentation result. In the panoptic output, *stuff* pixels receive the semantic class from semantic segmentation. In the case of *things* classes, masks provided by instance and semantic segmentation may be misaligned or their semantic classes may be different. To establish the semantic class of each *things* pixel, we consider the class provided by the object detector and use the instance mask to guide a pixel-to-pixel matching. The class label and the instance label of a pixel given by the instance segmentation is preserved only if it is consistent with the semantic category of that pixel from the semantic segmentation. The instance mask pixels which are not matched and correspond to either *things* pixels or other categories in the semantic segmentation output are deleted. After the matching process, we may have *things* pixels that were not covered by any instance mask. In this case, they are connected to the closest previously labeled pixel with a direct semantic path. This extension can be achieved with a breadth-first-search region growing algorithm. The semantic class and instance identifier are propagated, resulting in a more stable object level classification in comparison with pixel level classification. In the case of *things* pixels which did not receive an instance identifier after the region growing process, due to the fact that they were isolated, we generate a new instance identifier and preserve the semantic class from semantic segmentation. We employ a threshold over the segment area in order to avoid introducing false positive segments. The fusion process is depicted in Figure 6.

In UP-Drive, unifying semantic and instance segmentation in the form of panoptic segmentation, which ensures a unique semantic and instance label per pixel, is important in order to increase the segmentation accuracy, but also in the broader context of 3D object detection. Without using the panoptic fusion module, a pixel could have multiple mismatched semantic labels and multiple instance identifiers, since an instance segmentation network provides overlapping instances. This means that a 3D point would be associated with multiple semantic and instance labels and further processing steps would be necessary in order to solve semantic class mismatches and overlaps. Having a unique semantic and instance

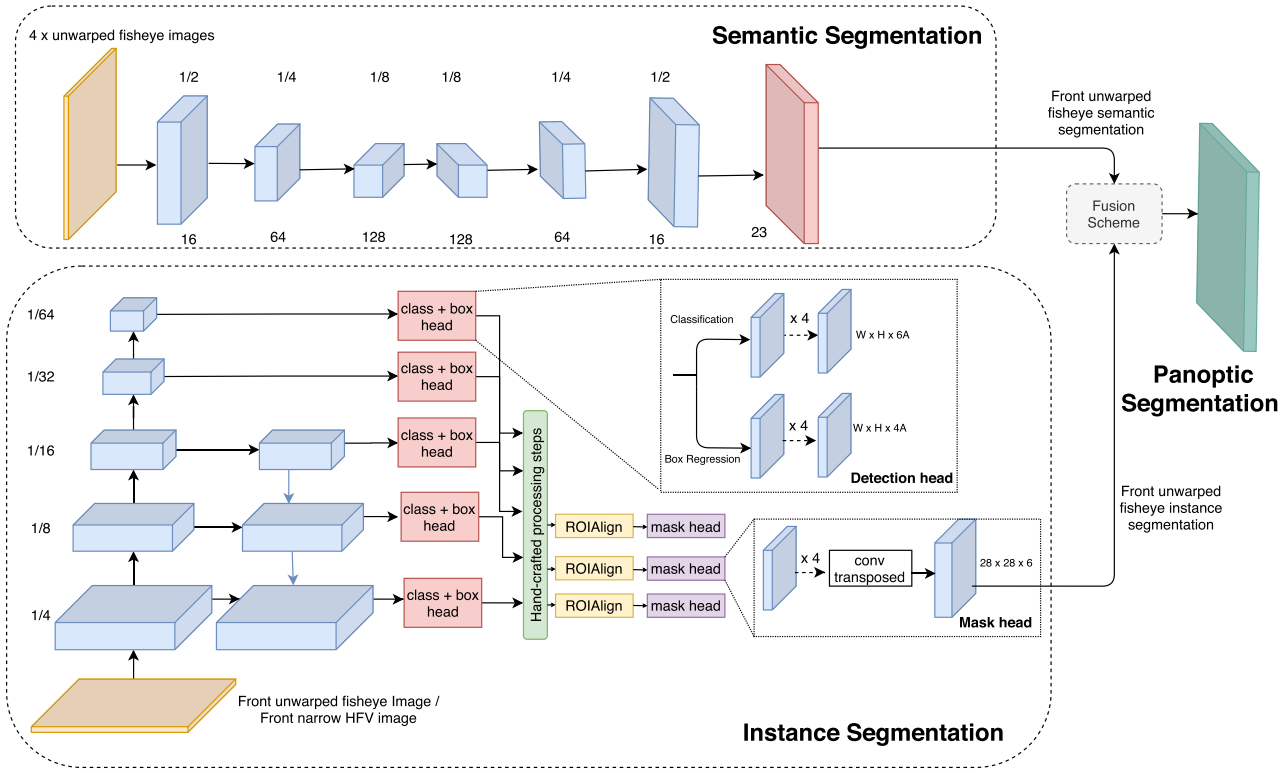

Fig. 5. Network architecture for panoptic segmentation on the front unwarped fisheye image. Semantic segmentation and instance segmentation output is fused using a novel merging scheme. The lightweight encoder-decoder semantic segmentation network also processes left, right, back fisheye images. The single-shot instance segmentation network processes the front unwarped fisheye image and the front narrow field-of-view image.

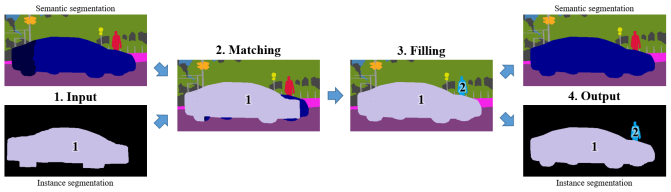

Fig. 6. Fusion of semantic and instance segmentation into a unified panoptic segmentation output. (1) Input: semantic segmentation (where a part of the car is missclassified as truck), instance segmentation (mask for car is slightly misaligned and cropped, and the pedestrian behind the car is not detected). (2) Matching: category-wise pixel matching (3) Filling: region growing with instance semantic label and instance ID propagation on the semantic mask (4) Panoptic segmentation.

label per pixel simplifies subsequent processing steps on the 3D augmented point cloud.

5) Network Integration and Deployment: We integrate the segmentation network, the two instance segmentation networks (one for fisheye and one for narrow HFV images) and the panoptic segmentation fusion algorithm, as seen in Figure 5 into our perception software running in ADTF [1]. The panoptic fusion algorithm is implemented in C++ and CUDA. For integrating the semantic and instance networks, we employ the powerful TensorRT library [2] that generates a high-performance runtime engine which can be easily loaded into the C++/CUDA project. TensorRT brings another benefit: it performs network optimization and quantization, which drastically reduces the inference time.

Since TensorRT library cannot import PyTorch models, we first convert the semantic segmentation PyTorch model into the ONNX format using the ONNX Parser [3]. Next, a network object is created in TensoRT and populated with the input from the ONNX model. The Builder component of the library takes the TensorRT network and generates a deployment-ready engine that is optimized for the target platform, in our case NVIDIA GTX 1080 GPU. The generated engine which performs the inference is loaded in the ADTF perception project. In order to reduce the inference time, in the case of semantic segmentation of fisheye images, we create a batch of four images (surround view) and forward them simultaneously through the network. Moreover, in the build phase, the library optimizes the layer graph by eliminating layers whose output is not used, fuses convolution, bias and ReLU operations, aggregates operations and merges concatenation layers. The semantic segmentation network is also quantized into INT8, resulting in four times faster inference speed at full image resolution.

Instance segmentation networks cannot be directly optimized with TensorRT in the same manner as the segmentation network, due to hand-crafted operations such as ROIAlign, filtering of candidate boxes and their assignment to a specific FPN layer. Moreover, the Non-Maxima Suppression (NMS) operation has to be implemented in order to remove overlapping boxes. Therefore, we divide our network into three parts: the backbone with object detection heads, hand-crafted operations (filter out boxes with low confidence, select top 1000 boxes from each FPN layer, apply NMS, select top 50 scoring bounding boxes, ROIAlign) and finally the mask prediction head. The backbone, object detection and mask

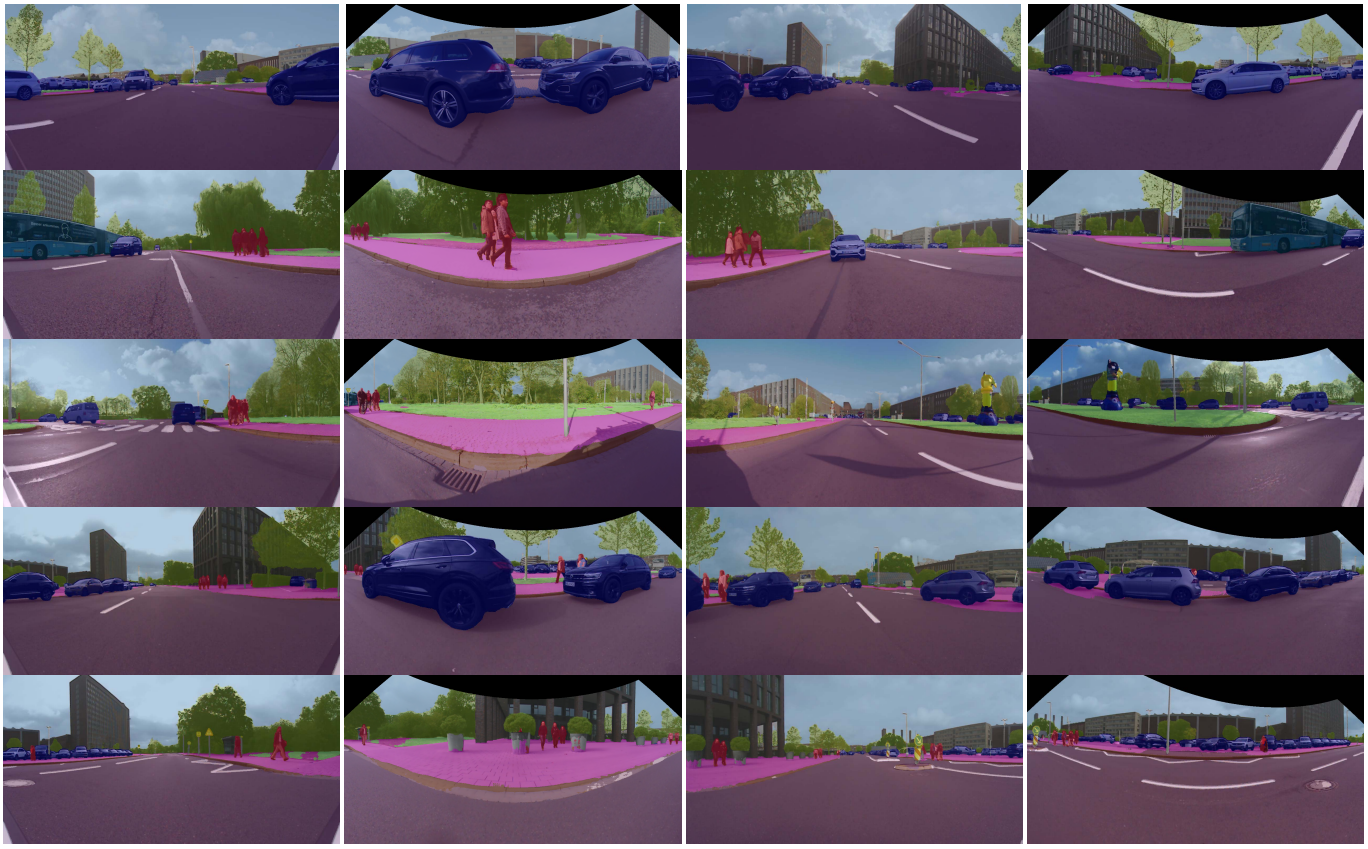

Fig. 7. Semantic segmentation of unwarped fisheye images. We process four images from the fisheye 160° horizontal field-of-view cameras which provide 360° coverage around the vehicle. Each camera views a different direction around the vehicle: front, right, rear and left.

prediction heads are converted to ONNX format, since they contain operations natively implemented in both ONNX and TensorRT. On the other hand, we implement hand-crafted operations as Plugin types of layers in TensorRT using native CUDA. Finally, we generate an optimized engine for instance segmentation with FP32 precision and we obtain almost two times faster inference time.

The fusion module for panoptic segmentation is implemented as a fast post-processing step and is running in 5 ms on the NVIDIA GTX 1080 GPU on the 1280 × 640 front unwarped image It has been integrated with the semantic and instance segmentation modules in the ADTF framework.

V. Experiments

In this section we provide experimental results for 2D semantic, instance and panoptic segmentation on the UP-Drive dataset.

A. Evaluation Setup

1) Evaluation Metrics: We evaluate semantic segmentation using standard mIoU (mean Intersection over Union) metric. For instance segmentation the AP@[.5:.05:.95] (Average Precision over classes and 10 IoU levels from 0.5 to 0.95 with a step size of 0.05) is used.

2) Inference Time: We report the inference time of the networks, measuring the forward pass and all the necessary post-processing steps (e.g. NMS). The execution time is measured on a NVIDIA GTX 1080 GPU with batch size of 1 unless otherwise stated.

TABLE I

Evaluation of the Semantic Segmentation Network on Fisheye Images Corresponding to Front, Left, Back, Right Views. 1280 × 640 - INT8 Is Integrated Into the Final Solution

Resolution	mIoU	Inference time (ms)
Custom CUDNN-based framework		
1280 × 640	67.87	38
512 × 256	64.52	15
TensorRT optimization		
1280 × 640 - FP32	67.87	20
1280 × 640 - INT8	**65.10**	**9**

B. Image Segmentation Results

In Table I we analyze the performance of the semantic segmentation network using different resolutions. In the first stage of the project, we developed a custom CUDNN-based framework that runs PyTorch models natively on GPU, due to the lack of integration possibilities with C++ projects. The inference time is measured using our custom CUDNN-based framework, but is equivalent to the inference time in the PyTorch framework. We train the semantic segmentation network using unwarped fisheye images from all four views. The advantage of using the same model for all images is that the inference can be accelerated by processing the batch of 4 images simultaneously. With 512 × 256 we obtain

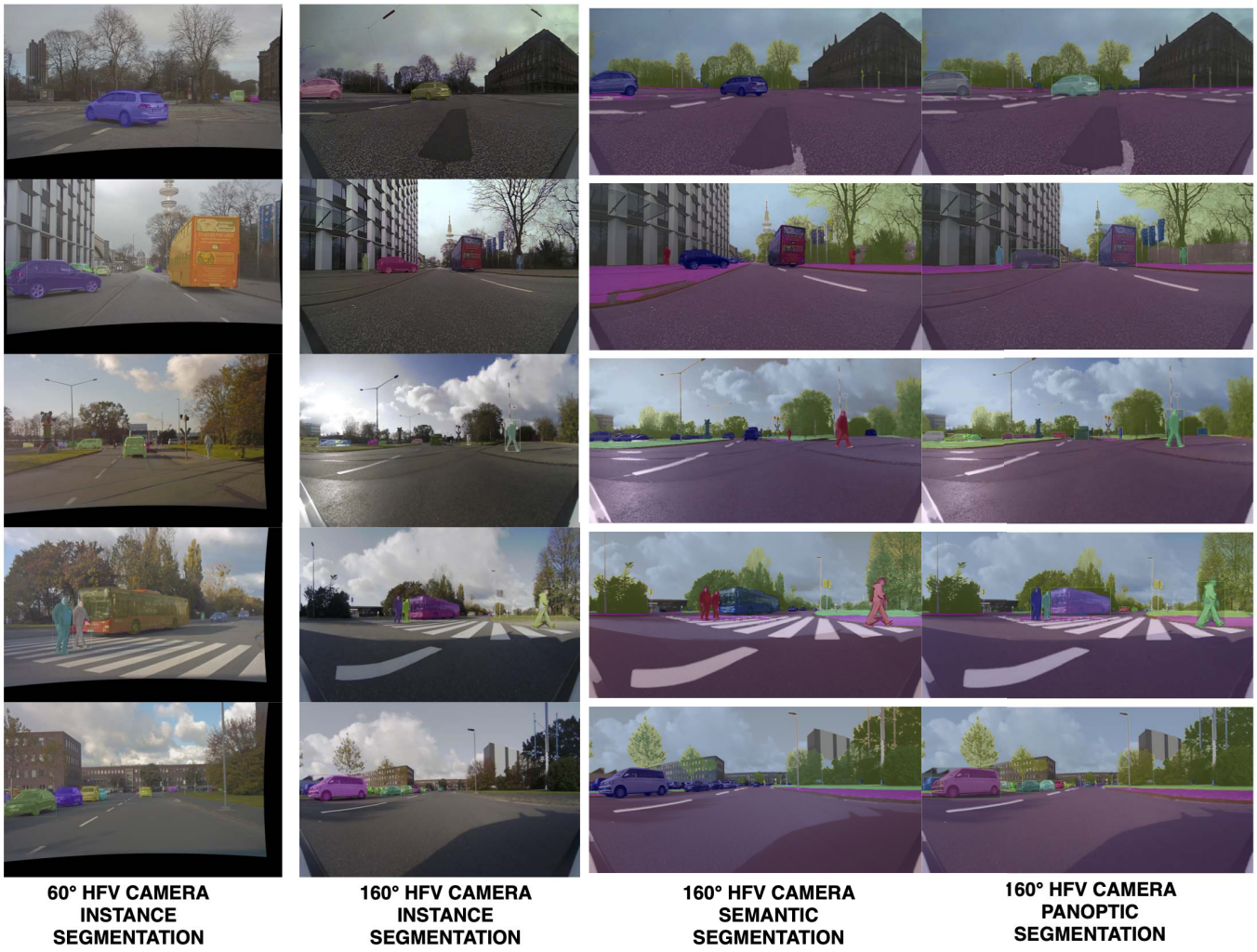

Fig. 8. The front area of the vehicle is covered by two cameras: a narrow 60° horizontal field-of-view camera which provides instance segmentation at increased depth and a wider 160° horizontal field-of-view camera, which provides instance, semantic and panoptic segmentation for the near-range.

64.52 mIoU and an inference time of 15 ms per image, and 60 ms for all four view images. The full resolution yields a more than 3% increase in accuracy, but it is computational expensive and not suitable for our system. In the first iterations of the project, we integrated into the system the model trained on 512 × 256 images by using the CUDNN-based custom framework. We achieved a compromise between accuracy and processing speed by lowering the resolution. The release of the TensorRT library allows us to apply network optimization and reduce the inference time, while processing the full resolution image 1280 × 640. TensorRT also provides routines for calibration for lower precision (INT8). After graph optimization and by using FP32 precision, we obtain high accuracy at 67.87% but with reduced inference time of 20 ms/image. INT8 precision degrades accuracy by 2.7% but provides significant performance improvements, all four unwarped fisheye images are semantically segmented in 36 ms. Finally, we adopt 8-bit inference with network quantization by calibrating the network graph on 300 images and integrate the INT8 optimized model in the perception software.

In the early stages of the project, we developed an original multi-task network based on Mask R-CNN for instance and semantic segmentation of the unwarped fisheye images, which we name MTN Panoptic [13]. We extend the instance segmentation network with a novel semantic segmentation head, which learns multi-scale features using Feature Pyramid Networks (FPN) [27] and an Atrous Spatial Pyramid with parallel dilated convolutions [8]. In Table II we present the results. We train with the original 1280 × 640 resolution and obtain 31.3% mAP for mask prediction and 66.4% segmentation mIoU. The two-stage network is computational expensive, running in 171 ms for full resolution and in 68 ms for lower resolution, and does not answer the imposed time and accuracy constraints of the 2D perception system. We do not integrate this solution but we decided to investigate one-stage networks such as RetinaMask, that provide similar accuracy, but with reduced inference time.

We train and evaluate the one-stage RetinaMask instance segmentation network on all four unwarped fisheye images images (front, left, right, back) and present the results in Table III. We experiment with three different resolutions: 1280 × 640, 832 × 416, 640 × 320. The network is accelerated with the TensorRT library using FP32 precision for increased quality. Compared to the two-stage Mask R-CNN

TABLE II

EVALUATION OF THE MASK R-CNN BASED SEMANTIC AND INSTANCE SEGMENTATION NETWORK ON FISHEYE IMAGES CORRESPONDING TO FRONT, LEFT, BACK, RIGHT VIEWS

Resolution	mAP box	mAP mask	mIoU	Inference time (ms)
1280 × 640	37.8	31.3	66.4	171
512 × 256	29.7	25.2	61.8	68

TABLE III

EVALUATION OF THE INSTANCE SEGMENTATION NETWORK ON FISHEYE IMAGES CORRESPONDING TO FRONT, LEFT, BACK, RIGHT VIEWS. 1280 × 640 - FP32 IS INTEGRATED INTO THE FINAL SOLUTION

Resolution	mAP box	mAP mask	Inference time (ms)
1280 × 640 - FP32	**36.8**	**30**	**66**
832 × 416 - FP32	30.1	24.8	44
640 × 320 - FP32	26.4	21.6	33

TABLE IV

EVALUATION OF THE INSTANCE SEGMENTATION NETWORK ON NARROW FIELD-OF-VIEW IMAGES CORRESPONDING TO FRONT VIEW. 832 × 416 - FP32 IS INTEGRATED IN THE FINAL SOLUTION

Resolution	mAP box	mAP mask	Time (ms)
960 × 604 - FP32	30.2	22.8	58
832 × 416 - FP32	**28.7**	**21.4**	**44**
640 × 320 - FP32	23.4	18.3	33

based network, we observe a slight decrease in accuracy for 1280 × 640 resolution from 31.3% mask mAP to 30% mask mAP, but the inference time is reduced 2.5 times to 66 ms. Adopting the largest resolution available of 1280 × 640 is critical for fisheye images, where the apparent size of objects is small.

We also train and evaluate an instance segmentation network on images from the front camera with a narrow, 60° HFV. Results are in Table IV. In order to achieve the trade-off between processing speed and quality, we optimize the network with FP32 and adopt 832 × 416 resolution, with 21.4% mask mAP and 44 ms inference time.

In Table V, we present ablation studies of each segmentation module on the front unwarped fisheye images at 1280 × 640. By using the INT8 model for the semantic segmentation, the mIoU computed for all four views images is 65.1%, while the mIoU for the front view images is 65.9%. The unified panoptic segmentation improves the semantic segmentation with almost 1%, from 65.9% mIoU to 66.8% mIoU. Moreover, the panoptic module increases the instance segmentation mAP with 0.3%. In terms of panoptic quality, we obtain 42.4% PQ for all classes, 42.2% PQ for *stuff* classes and 43% PQ for *things* classes.

We measure the inference time of the entire 2D perception pipeline in Table VI. The image unwarping is performed for the four fisheye images, while the image undistort for all five images. This is a fast processing step and runs in 5 ms on the GPU. Semantic segmentation of all four unwarped fisheye images using the INT8 quantized network takes 36 ms. Instance segmentation of the front unwarped fisheye image is the most costly operation with 66 ms. We reduce the resolution of the front 60° horizontal field-of-view image to reduce the inference time to 44 ms. The entire pipeline runs in 157 ms. When all the segmented images are available, the 3D point cloud is projected into the images in order to

TABLE V

ABLATION STUDIES OF EACH SEGMENTATION MODULE ON THE FRONT VIEW UNWARPED FISHEYE IMAGE. WE EVALUATE SEMANTIC SEGMENTATION, INSTANCE SEGMENTATION AND PANOPTIC SEGMENTATION ON 1280 × 640 IMAGES

Model	mIoU	mAP mask	PQ	PQ_{st}	PQ_{th}	Time (ms)
Semantic segmentation	65.9	-	-	-	-	9
Instance segmentation	-	30	-	-	-	66
Panoptic fusion	66.8	30.3	42.4	42.2	43	5
Panoptic segmentation	66.8	30.3	42.4	42.2	43	80

TABLE VI

TIME EVALUATION OF THE ENTIRE 2D SEMANTIC PERCEPTION SYSTEM

Module	Inference time (ms)
Image unwarping and undistort × 5	6
Semantic segmentation × 4	36
Instance segmentation front 160° HFV image	66
Panoptic fusion front 160° HFV image	5
Instance segmentation front 60° HFV image	44
Total	157

build the enhanced semantically segmented 3D point cloud, which is further processed by high-level functions for 3D object detection and classification. By running only image unwarping, the semantic segmentation on the four unwarped fisheye images and instance segmentation on the front 60° HFV image, the pipeline runs in 86 ms on one NVIDIA GTX 1080 GPU.

In Table VII we provide a comparison with state-of-the-art networks for semantic and instance segmentation on the Cityscapes [12] validation set. We compare the mean Intersection over Union, mask mean Average Precision and the unoptimized inference time. We select the ResNet-50 variant of the networks, which we also use, for fair comparison. Our proposed solution was developed in the context of a system that can be deployed on a fully automated vehicle, which imposed constraints such as real-time processing and high accuracy. In order to balance efficiency and accuracy, we select lightweight and fast networks and process lower resolution Cityscapes images with a size of 1024 × 512. Compared to the other methods, our solution is the fastest with 68 ms, and achieves comparable semantic segmentation mIoU with networks using higher resolution images. However, we observe that our instance segmentation results are less accurate, since detection and segmentation of small objects in the downsampled image is more difficult. From a practical point of view, considering the imposed constraints, we obtain a good trade-off between speed and accuracy, competitive results at much lower computational costs.

In Figure 9 we provide a visual comparison between the front 160° HFV and the 60° HFV instance segmentation. Figure 9 presents the case of a pedestrian on the left-hand side sidewalk. Up to 20 meters, the pedestrian in visible only in the front unwarped fisheye image. By processing unwarped fisheye images, we obtain robust instance segmentation of pedestrians up to 25 meters. At 25 meters, the pedestrian is visible and is detected in both images. At distances more than 25 meters, the pedestrian has a very small size in the unwarped fisheye image and is not detected, however the pedestrian is detected in the 60° HFV image. With the use of both cameras, we provide an increased detection range up to 75 meters.

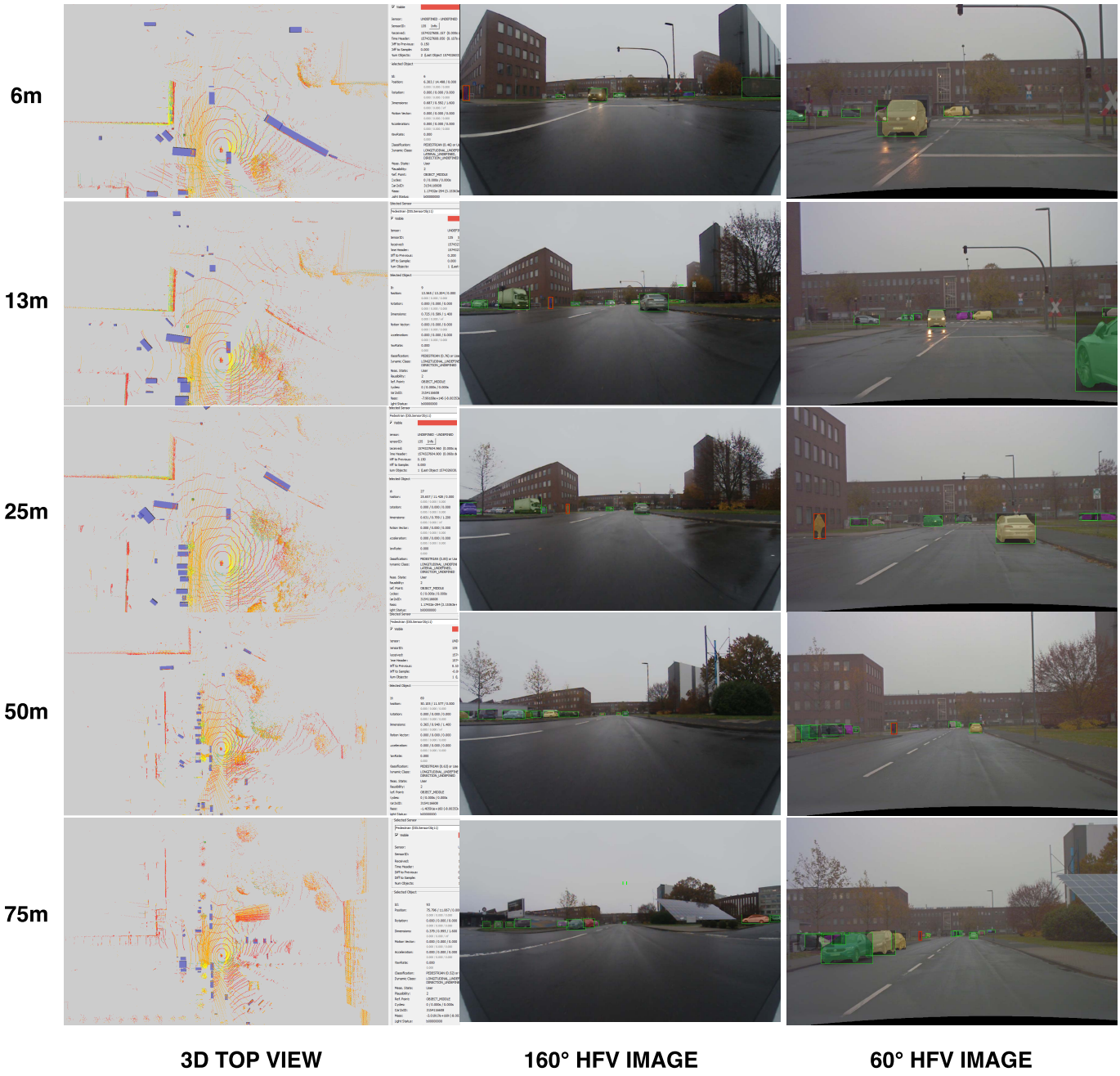

Fig. 9. Comparison between wide and narrow field of view instance segmentation. A pedestrian is marked with a green box in the 3D top view image and a red bounding box in the wide and narrow field of view images. On the first column, we provide the bird's eye view of the 3D point cloud with detected objects. Best viewed in color and zoom.

TABLE VII

EVALUATION OF SEMANTIC AND INSTANCE SEGMENTATION COMPARED TO STATE-OF-THE-ART NETWORKS ON THE CITYSCAPES *validation* SET. INFERENCE TIME IS MEASURED ON GTX 1080TI GPU. FOR ENTRIES MARKED WITH * WE APPROXIMATE THE INFERENCE TIME BASED ON THEIR REPORTED SPEED

Method	Resolution	mIoU	mAP mask	Inference time (ms)
MTN Panoptic [13]	1024 × 512	71.0	28.6	84
Panoptic-FPN [21]	2048 × 1024	75.0	32.0	-
UPSNet [48]	2048 × 1024	75.2	33.3	236
MTN Panoptic [13]	2048 × 1024	76.0	36.4	224
Prototype Panoptic [38]	2048 × 1024	76.9	-	117*
DenseBox [19]	2048 × 1024	77.0	29.8	141*
Panoptic-DeepLab [11]	2048 × 1024	80.5	35.3	167*
Ours	1024 × 512	74.6	27.2	68

VI. Conclusion and Lessons Learned

Building a fully automated vehicle prototype is challenging from multiple perspectives: selection of the most proper sensor suite, development of fast and robust algorithms that provide accurate results in real-world scenarios, creation of quality and complete datasets for training deep learning algorithms, but also integration of the algorithms and deployment on the vehicle.

Our semantic cameras setup perceive 360° around the vehicle. Our first solution used only fisheye cameras mounted in all four directions. However, the segmentation and detection range was limited to the near range around the vehicle. In the case of pedestrians, the segmentation was robust up to 25 meters, which is suitable only for very low-speed driving and parking maneuvers. For the left, right and back view, detection in the near range provides sufficient information for maneuver prediction and decision making even in the case of higher-speed driving. However, detection and segmentation in the far-range is necessary especially on the front view. In our final solution, we concluded that both fisheye cameras and narrow field-of-view cameras for front view are necessary to cover the near and far range. Also, another important aspect is that the detection range can be increased by processing higher resolution images.

During the 4-year course of the project, we observed a very fast evolution of methods, tools and frameworks used. In the early stages of the project, we employed the Torch framework for training our networks, but as time passed, more comprehensive and advanced frameworks have emerged: PyTorch, Tensorflow etc. The Torch framework did not provide integration possibilities of the inference functions in the C++ project, therefore we developed our custom C++/CUDA framework for inference. Later, the launch of the TensorRT library by NVIDIA has provided us several benefits: easy network optimization that enables us to use larger image resolutions with a reduced processing time, and also it eases the integration with C++/CUDA/Python projects. TensorRT was adopted in the later stages of the project. However, not all architectures can be easily optimized, for example, in the case of two-stage architectures, the backbone and network heads can be optimized, but there are many hand-crafted processing steps that need to be implemented on the GPU and integrated with the TensorRT optimized network parts. Choosing a single-stage network brings major benefits in terms of speed, but it can also be more easily optimized and integrated.

The quality and completeness of the training dataset is important and very much effort has been directed towards building a consistent and comprehensive dataset. The activities associated with creating the dataset have been: recording data in diverse weather conditions (including adverse weather) and different seasons, organization and integration of data collected from various sequences, selecting relevant and interesting scenarios in order to ensure the diversity of the training set, selecting images with rare classes and corner cases. Images have been annotated for semantic and instance segmentation in three iterations. In the first iteration, a batch of 1k images from all four views have been annotated in-house using a semi-automatic annotation tool that we developed for this purpose [37]. Next, the network trained with the available dataset has been deployed on the UP-Drive vehicle and it has been tested on real-world scenarios in the urban environment, in order to detect cases where the network does not perform well. Two more iterations of data annotation have followed in which the dataset has been extended to 20k images and multiple image augmentation techniques have been introduced. The lesson learned is that realistic synthetic data generation, automatic annotation, semi-supervised and unsupervised learning have to be investigated.

In the UP-Drive project we have successfully developed an automated vehicle that is able to safely navigate urban areas. We provided a modular deep learning based solution for environment perception based on fisheye and narrow field-of-view semantic cameras with semantic, instance and panoptic segmentation capabilities. In this paper, we present the challenges we encountered in developing the semantic environment perception system because of the high requirements in accuracy, robustness and real-time performance. We investigated multiple solutions, motivated our final design choices, presented details regarding integration of the segmentation modules with the software and finally we discussed the learned lessons during the 4-year course of the project.

References

[1] *Automotive Data and Time-Triggered Framework (ADTF)*. Accessed: Jul. 27, 2020. [Online]. Available: https://www.elektrobit.com/products/automated-driving/eb-assist/adtf/

[2] *Nvidia Deep Learning Tensorrt Library*. Accessed: Jul. 27, 2020. [Online]. Available: https://developer.nvidia.com/tensorrt

[3] *Open Neural Network Exchange (ONNX)*. Accessed: Jul. 27, 2020. [Online]. Available: https://onnx.ai/

[4] *Urban Parking and Driving H2020 European Project*. Accessed: Jul. 27, 2020. [Online]. Available: https://up-drive.ethz.ch/

[5] M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5221–5229.

[6] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6154–6162.

[7] C. Chen, Z. Liu, S. Wan, J. Luan, and Q. Pei, "Traffic flow prediction based on deep learning in Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3776–3789, Jun. 2020.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2017.

[9] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 801–818.

[10] X. Chen, R. Girshick, K. He, and P. Dollar, "TensorMask: A foundation for dense object segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2061–2069.

[11] B. Cheng *et al.*, "Panoptic-deepLab," 2019, *arXiv:1910.04751*.

[12] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[13] A. D. Costea, A. Petrovai, and S. Nedevschi, "Fusion scheme for semantic and instance-level segmentation," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 3469–3475.

[14] J. Dai *et al.*, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.

[15] C.-Y. Fu, M. Shvets, and A. C. Berg, "RetinaMask: Learning to predict masks improves state-of-the-art single-shot detection for free," 2019, *arXiv:1901.03353*.

[16] N. Gao *et al.*, "SSAP: Single-shot instance segmentation with affinity pyramid," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 642–651.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.
[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[19] R. Hou *et al.*, "Real-time panoptic segmentation from dense detections," 2019, *arXiv:1912.01202*.
[20] R. Cipolla, Y. Gal, and A. Kendall, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7482–7491.
[21] A. Kirillov, R. Girshick, K. He, and P. Dollar, "Panoptic feature pyramid networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6399–6408.
[22] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Oct. 2019, pp. 9404–9413.
[23] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother, "InstanceCut: From edges to instances with MultiCut," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5008–5017.
[24] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 734–750.
[25] J. Liang, N. Homayounfar, W.-C. Ma, Y. Xiong, R. Hu, and R. Urtasun, "PolyTransform: Deep polygon transformer for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9131–9140.
[26] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1925–1934.
[27] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.
[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[29] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2014, pp. 740–755.
[30] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.
[31] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[32] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4990–4999.
[33] D. Neven, B. D. Brabandere, M. Proesmans, and L. Van Gool, "Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8837–8845.
[34] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy, "PersonLab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 269–286.
[35] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.
[36] S. Peng, W. Jiang, H. Pi, X. Li, H. Bao, and X. Zhou, "Deep snake for real-time instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8533–8542.
[37] A. Petrovai, A. D. Costea, and S. Nedevschi, "Semi-automatic image annotation of street scenes," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 448–455.
[38] A. Petrovai and S. Nedevschi, "Real-time panoptic segmentation with prototype masks for automated driving," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1400–1406.
[39] A. Petrovai and S. Nedevschi, "Multi-task network for panoptic segmentation in automated driving," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 2394–2401.
[40] L. Porzi, S. R. Bulo, A. Colovic, and P. Kontschieder, "Seamless scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8277–8286.
[41] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[42] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[43] E. Romera, J. M. Álvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized ConvNet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Jan. 2018.
[44] B. Shahian Jahromi, T. Tulabandhula, and S. Cetin, "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles," *Sensors*, vol. 19, no. 20, p. 4357, Oct. 2019.
[45] R. Varga, A. Costea, H. Florea, I. Giosan, and S. Nedevschi, "Super-sensor for 360-degree environment perception: Point cloud segmentation using image features," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–8.
[46] P. Wang *et al.*, "Understanding convolution for semantic segmentation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1451–1460.
[47] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.
[48] Y. Xiong *et al.*, "UPSNet: A unified panoptic segmentation network," in *Proc. CVPR*, 2019, pp. 8818–8826.
[49] Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, "EU long-term dataset with multiple sensors for autonomous driving," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020.
[50] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.

Andra Petrovai received the M.S. degree in computer vision and artificial intelligence from the Technical University of Cluj-Napoca (TUCN), Cluj-Napoca, Romania, in 2016, where she is currently pursuing the Ph.D. degree. Her research interests include image segmentation, deep learning, environment perception, and automated vehicles.

Sergiu Nedevschi (Member, IEEE) received the M.S. and Ph.D. degrees in electrical engineering from the Technical University of Cluj-Napoca (TUCN), Romania, in 1975 and 1993, respectively. From 1976 to 1983, he was a Researcher with the Research Institute for Computer Technologies Cluj-Napoca. In 1983, he joined TUCN. In 1998, he was appointed as a Full Professor of computer science, he founded and since then he has been leading the Image Processing and Pattern Recognition Research Center. He was the Head of the Computer Science Department from 2000 to 2004, the Dean of the Faculty of Automation and Computer Science from 2004 to 2012, and the Vice-Rector with Scientific Research, TUCN, from 2012 to 2020. He was involved in more than 80 research projects, being the coordinator of 62 of them. The industrial cooperation with important automotive players such as Volkswagen AG, Robert Bosch GmbH, and SICK AG, and research institutes such as VTT, INRIA, was achieved through funded research projects. He has published more than 400 scientific papers and has edited over 20 volumes, including books and conference proceedings. His research interests include image processing, pattern recognition, computer vision, machine learning, intelligent, and autonomous vehicles.