

SELMA: SEmantic Large-scale Multimodal Acquisitions in Variable Weather, Daytime and Viewpoints

Paolo Testolina*, *Student Member, IEEE*, Francesco Barbato*, *Student Member, IEEE*, Umberto Michieli, *Graduate Student Member, IEEE*, Marco Giordani, *Member, IEEE*, Pietro Zanuttigh, *Member, IEEE*, Michele Zorzi, *Fellow, IEEE*

Abstract—Accurate scene understanding from multiple sensors mounted on cars is a key requirement for autonomous driving systems. Nowadays, this task is mainly performed through data-hungry deep learning techniques that need very large amounts of data to be trained. Due to the high cost of performing segmentation labeling, many synthetic datasets have been proposed. However, most of them miss the multi-sensor nature of the data, and do not capture the significant changes introduced by the variation of daytime and weather conditions. To fill these gaps, we introduce SELMA, a novel synthetic dataset for semantic segmentation that contains more than 30K unique waypoints acquired from 24 different sensors including RGB, depth, semantic cameras and LiDARs, in 27 different weather and daytime conditions, for a total of more than 20M samples. SELMA is based on CARLA, an open-source simulator for generating synthetic data in autonomous driving scenarios, that we modified to increase the variability and the diversity in the scenes and class sets, and to align it with other benchmark datasets. As shown by the experimental evaluation, SELMA allows the efficient training of standard and multi-modal deep learning architectures, and achieves remarkable results on real-world data. SELMA is free and publicly available, thus supporting open science and research.

Index Terms—Synthetic dataset, CARLA, autonomous driving, domain adaptation, semantic segmentation, sensor fusion.

I. INTRODUCTION

Recent advances in the automotive sector have paved the way toward Connected Intelligent Transportation Systems (C-ITSs) to achieve safer and more efficient driving. Not only can C-ITSs reduce the number of traffic accidents (up to 90%, according to some estimates [1]) or improve traffic management via smart platooning, cruise control and/or traffic light coordination, but it holds the promise to improve fuel economy and contribute to a 60% fall in carbon emissions [2]. Overall, C-ITSs represent a huge market of more than 7 trillion USD [3], hence stimulating significant research efforts.

To these goals, future connected vehicles will be equipped with heterogeneous sensors, including Light Detection and Ranging (LiDAR) and RGB camera sensors, able to provide an accurate perception of the environment. In particular, LiDARs generate a 3D omnidirectional representation of the environment in the form of a point cloud, and stand out as the

most accurate sensors for geometry acquisition under several weather and lighting conditions [4]. On the other side, RGB cameras offer advantages like cheaper price, higher resolution and higher frame rate than LiDARs, even though they suffer from severe sensitivity to illumination and visibility conditions [5]. In this sense, sensor fusion appears as a promising solution to provide more robust scene understanding, at the expense of the additional processing overhead for collecting and combining observations from multiple sensors [6].

However, autonomous driving tasks, in particular semantic segmentation (SS) and Vehicle-to-Everything (V2X) communication, raise several challenges [7], [8], also in view of the complex and dynamic environment in which autonomous vehicles move and operate. In these regards, machine learning (ML) and deep learning (DL) represent valuable tools to address these issues and optimize driving decisions [9]. However, these techniques require the availability of massive amounts of labeled data for proper training, whose acquisition and labeling is extremely expensive and time consuming. Hence, existing open-source datasets, like Waymo [10], Cityscapes [11], and KITTI [12], are scarce and generally lack diversity. Moreover, many datasets are too small to capture the many challenges of the urban scenario, do not encompass multiple (and diverse) sensors, and come with unlabeled scenes, undermining the training of ML models [13].

To fill these gaps, the scientific community has been investigating the usage of synthetic (computer-generated) datasets, where the full control of the data generation pipeline is delegated to simulations, hence ensuring lower costs, greater flexibility and larger quantity than real-world data [14]–[18]. Notably, simulations facilitate data acquisition in different conditions and scenarios, and considering diverse sets of sensors. An open-source simulator to generate synthetic data is CAR Learning to Act (CARLA) [19], which includes urban layouts, a wide range of environmental conditions, vehicles, buildings and pedestrians models, and supports a flexible setup of sensors. At the time of writing, several synthetic datasets exist for SS in autonomous driving [14], [20]–[23], where only [23] is based on CARLA. These datasets, however, present limitations. In particular, samples are generally captured in a limited number of settings, in similar viewpoints, weather, lighting, and daytime conditions, and often from a single sensor. Moreover, they do not provide end-users with fine-grained control over the weather setup or even the same semantic class set as common benchmarks, like Cityscapes [11].

To overcome these limitations, in this paper we present

*: P. Testolina and F. Barbato are primary co-authors.

All authors are with the University of Padova. The work of F. Barbato, U. Michieli and P. Zanuttigh was in part supported by the SID 2020 project “Semantic Segmentation in the Wild.” The work of P. Testolina was supported by Fondazione CaRiPaRo under grants “Dottorati di Ricerca” 2019.

Manuscript received xxx, 2022; revised xxx, 2022.

SEmantic Large-scale Multimodal Acquisitions in Variable Weather, Daytime and Viewpoints (SELMA), a new multimodal synthetic dataset for autonomous driving, built using a modified version of the CARLA simulator. Our dataset stands out as one of the largest, most complete and diverse datasets to ensure adequate design, prototyping, and validation of autonomous driving models, in particular to solve complex tasks like SS. Specifically, our SELMA dataset consists of:

- Data acquired in 30909 independent locations from 7 RGB cameras, 7 depth cameras, 7 semantic cameras, and 3 LiDARs coupled with semantic information. The multimodal setup of SELMA promotes complementary and diversity of data, and permits higher accuracy and performance of learning tasks [24].
- Acquisitions generated in variable weather, daytime and viewpoint conditions, and across 8 maps, for a total of 216 unique settings. To this aim, the CARLA simulator has been modified to increase the photo-realism of the weather conditions, and to maximize the visual variability, e.g., by adding parked bikes or traffic lights and signs.
- Semantic labeling for both camera and LiDAR data into 36 distinct classes, with complete overlap with the training set of common benchmarks like Cityscapes [11], obtained by modifying the source code of the simulator.

We validate the accuracy and realism of our dataset starting from a set of baseline experiments, and show that DL models for semantic understanding trained on our dataset outperform the same models trained on competing synthetic datasets when tested on a real (i.e., non-simulated) domain.

The dataset is freely available for download¹, thus supporting open science and stimulating further research in the field of autonomous driving.

The remainder of this paper is organized as follows. In Sec. II we describe the existing real and synthetic datasets related to our work. Sec. III presents the CARLA simulator and the additions we introduced to acquire the data. The SELMA dataset is then described in detail in Sec. IV, while Secs. V and VI show some numerical results validating the accuracy of models trained on our dataset. Finally, in Sec. VII we provide the conclusions and some future research directions.

II. RELATED WORK

The development of DL architectures seen in recent years goes along with the design of extensive datasets, needed for their optimization. One computer vision (CV) task where such advancements have been particularly significant is scene understanding, which evolved in several sub-tasks, each requiring appropriate data for training. Among them, three tasks are worth mentioning, given the strong push they provided to datasets design: image classification, object detection and semantic segmentation. The first sparked the generation of widely used datasets, e.g., ImageNet [25]. The second and third tasks have been widely applied to many problems, and especially to autonomous driving systems. Here, vehicles require accurate recognition of the surrounding environment to

TABLE I: Comparison among the most popular SS datasets. White rows refer to real datasets, dark grey refers to synthetic datasets, and light grey refers to a combination of the two. Best in **bold**, runner-up underlined. T: Type, R: Real, M: Mixed, S: Synthetic, BB: Bounding Boxes, W: Weathers, ToD: daytime, AA: Anti-Aliasing, †: estimated depth, *: random.

Name	T	Cams	Depth	LiDAR	BB	Labels	SS/BB CS			W	ToD	AA	Positions	
							RGB	LiDAR	all train classes					
A2D2 [43]	R	6	-	5	✓	1	1	38	38	12	-	-	41280	
ACDC [44]	R	1	-	X	X	1	-	19	19	19	<u>3</u>	<u>2</u>	4006	
ApolloScape [37]	R	6	2†	2	✓	1	1	36	22	12	*	*	143906	
Argo [45]	R	9	2†	2	✓	-	-	15	15	5	-	-	N/A	
BDD 100k [33]	R	1	X	X	✓	1	-	<u>40</u>	19	19	* [*]	-	10000	
CamVID [27]	R	1	X	X	✓	1	-	32	11	11	-	-	701	
Cityscapes [11]	R	2	2†	X	✓	1	-	35	19	19	-	-	25000	
DarkZurich [46]	R	1	X	X	X	X	-	19	19	19	-	<u>3</u>	8779	
DRIV100 [47]	R	1	X	X	X	X	-	19	19	19	-	-	N/A	
IDD [34]	R	1	X	X	X	X	1	-	34	25	19	-	-	10003
KITTI [28]	R	2	4†	1	✓	X	X	8	8	8	-	-	N/A	
Mapillary [32]	R	1	X	X	✓	1	-	66	66	19	* [*]	-	25000	
NTHU [48]	R	1	X	X	X	X	-	13	13	13	-	-	12800	
Nuscenes [35]	R	6	X	1	✓	X	1	23	23	-	-	-	40000	
RainCouver [49]	R	1	X	X	X	X	-	3	3	-	<u>1</u>	<u>3</u>	N/A	
SemanticKITTI [39]	R	-	-	1	X	1	-	1	28	20	15	-	-	43552
Nightcity [50]	R	1	X	X	X	1	-	19	19	19	-	<u>1</u>	4297	
CS Fog [51]	M	2	2†	X	✓	1	-	35	19	19	<u>1</u>	-	5000	
CS Rain [52]	M	2	2†	X	✓	1	-	34	18	18	<u>1</u>	-	5000	
GTA5 [20]	S	1	X	X	X	1	-	35	19	19	-	*	24966	
IDDA [23]	S	1	1	X	X	1	-	24	24	16	<u>3</u>	-	X	16000
SYNTHIA [21]	S	1	1	X	X	1	-	23	16	16	* [*]	X	9400	
SELMA (Ours)	S	<u>7</u>	7	<u>3</u>	✓	7	3	36	19	19	<u>9</u>	<u>3</u>	✓	30909

appropriately plan driving actions. This translated into a wide range of real and synthetic datasets to support the training of autonomous driving applications [26]. In this work, we focus on the semantic segmentation task, generally recognized as the most challenging of the three. The most popular SS datasets existing in the literature and their characteristics are reported in Table I.

a) *Real datasets*: Given the high complexity and cost of labeling, most wide-scale real datasets tend not to provide the ground truth, e.g., SS labels or bounding boxes, thus limiting their use in tasks like semantic segmentation and object detection. One of the first works to introduce labeled SS samples in the context of autonomous driving was CamVid [27], which consists of over 700 images labeled in 32 classes. Based on this, a huge effort was made by the creators of KITTI [28]–[31] to provide the first multimodal (stereo RGB and LiDAR) dataset for road scenes. This dataset, unfortunately, consists of only a small subset of 200 SS training images. The next fundamental step was the acquisition of the Cityscapes [11] dataset, which was the first collection of labeled samples large enough to support training of deep architectures to a satisfactory level. It includes 5 000 finely-labeled samples and 20 000 coarsely-labeled samples captured in several German cities, and has become an important benchmark for the segmentation task. Recent works focus more on the volume [32]–[34] and variability [35]–[38] of data. Even more recently, researchers are supporting the advent of LiDAR sensors, and some datasets have been generated accordingly [39]–[42].

b) *Synthetic datasets*: To circumvent the cost involved in the labeling of large-scale datasets, particularly those for SS, many synthetically-generated datasets have been proposed over the years. The first two important benchmarks are GTA5 [20] and SYNTHIA [21], both introduced in 2016. The former was generated exploiting the homonymous game, and provides 25 000 samples of realistic high-quality images. The semantic labels are provided in the same class set as Cityscapes [11], although they were inferred by the authors

¹The SELMA dataset is available at <https://scanlab.dei.unipd.it/app/dataset>.

from secondary shader data, and the classes assigned to objects are not always consistent. The latter was the first dataset to provide depth ground truth for each of its 9 000 samples. The class set is different than that used by Cityscapes, and the overlap is limited to 16 classes (see Appendix C1 for details on the class splits). A third important synthetic dataset is Virtual KITTI [14], [22] which, like its real counterpart, focused heavily on the multimodal aspect. It was the first to provide ground truth optical-flow and instance segmentation data, in addition to color, depth and semantic. More recently, the IDDA dataset [23] was introduced to address the lack of weather conditions variability in available datasets. It was developed using CARLA [19] and includes semantic labels (with an overlap of 16 classes with Cityscapes), depth and RGB data.

From Table I we can see that, among the synthetic datasets, SELMA is the only one to provide labeled data for multiple LiDAR and camera sensors. Moreover, it is the only one that provides multiple weather conditions while supporting the full Cityscapes [11] class set, as opposed to IDDA. Even more, it is the only one that provides 3D bounding boxes. Finally, compared to GTA5 [20], i.e., the only competitor to provide anti-aliased color images, SELMA provides more samples, and considers a much higher variability of setups and sensors.

III. SIMULATOR SETUP

In this section we describe the CARLA simulator for generating synthetic automotive data (Sec. III-A, and the changes we introduced for SELMA (Sec. III-B).

A. The CARLA Simulator

The CARLA simulator is used to generate synthetic data relative to autonomous driving systems. It is designed as an open-source layer over Unreal Engine 4 (UE4) to provide high-quality rendering, realistic physics based on the NVIDIA PhysX engine, and basic Non-Player Character (NPC) logic [19]. Reproducible and reliable physics simulations, as well as realistic and synchronized sensor data, can be obtained through the CARLA Application Programming Interface (API). Hereby, we briefly report the main characteristics of release 0.9.12, which was the starting point for the customization we made to meet the desired characteristics of the dataset, as detailed in Sec. III-B.

a) Unreal Engine Models: CARLA offers a wide variety of carefully designed UE4 models for static (e.g., buildings, vegetation, traffic signs) and dynamic objects (e.g., vehicles and pedestrians), sharing a common scale, and with realistic sizes. In release 0.9.12, the blueprint library includes the model of 24 cars, 6 trucks, 4 motorbikes, 3 bikes, each with customizable colors, and 41 pedestrian models of different ethnicity, build, and dressed with a wide variety of clothes. Furthermore, 8 towns (Town01-07 and Town10HD) were carefully designed by the CARLA team using more than 40 building models. Each town has its unique features and landmarks, thus offering 8 simulation environments with diverse visual characteristics.



Fig. 1: Desk view at three different times of the day.



Fig. 2: Samples in 9 variable weather conditions at Noon.

b) Sensors: Data from the simulated world can be retrieved through a number of different sensors (see Appendix A for a detailed list of supported sensors), that can be placed at an exact location and a given rotation, and attached to a parent actor, thus following its movements with a rigid or spring-arm-like behavior. Sensors data can be collected at each simulation step. When working with multiple, high-resolution sensors, synchronous mode is required to guarantee that the GPU completes the rendering and delivers the data to the client before updating to the next simulation step. Thus, the sensor acquisition rate is the same for all.

c) Weather Conditions and Daytime: Leveraging the underlying UE4 graphics, CARLA offers a variety of *daytime* and *weather* conditions. The combination of daytime and weather will be referred to as *environmental conditions* in the rest of the paper. Such conditions differ in the position and color of the sun, and in the intensity and color of diffuse sky radiation (daytime), as well as ambient occlusion, fog, cloudiness, and precipitation (weather). In release 0.9.12, there are 14 predefined environmental conditions, obtained by the combination of two daytimes (Noon and Sunset), and seven weather conditions (Clear, Cloudy, Wet, WetCloudy², SoftRain, MidRainy, HardRain).

B. Customization

To enhance the quality of the collected data, we customized the source code of CARLA, as detailed in the following.

First, we adjusted the parameters of the predefined environmental conditions, modifying the weather scattering and fog properties, and the position of the sun, to maximize the diversity between the environmental conditions and their photo-realism. Then, we introduced the Night daytime (Fig. 1) and the Mid Fog and Hard Fog weather conditions (Fig. 2).

²Wet and WetCloudy indicate that the road is more reflective and contains puddles. Notably, the former (latter) specifies that observations are acquired in clear (cloudy) sky.

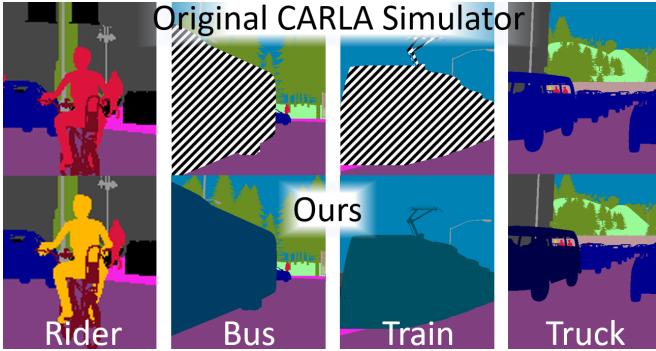


Fig. 3: Comparison between the original version of the CARLA simulator and that with our modifications. Dashed regions indicate classes that were originally missing in CARLA. Notice that our implementation now distinguishes riders from people and trucks from cars.

Thus, the number of daytimes and weather conditions has been increased from 2 and 7 to 3 and 9, respectively, bringing the total number of environmental conditions to 27.

Second, we modified the CARLA semantic classes, to make them compatible with existing benchmark datasets, and added new vehicle models to increase the class diversity. Specifically, the remapping of the classes was done in the source code to affect both the semantic LiDAR and the semantic camera. We introduced the *Train* class, adding a train and a tram model, and we added two bus and two truck models to the existing classes. Then, our modifications to the source code allowed us to introduce the *Rider* class, adding the corresponding tag and separating the rider from its bike/motorbike tag. A visual example is reported in Fig. 3. Finally, as of release 0.9.12, the parked vehicles were not labeled correctly. Exploiting the CARLA API, we removed the corresponding map layer, saving the location information to place vehicles with the correct tag in the exact same position.

Then, the UE4 content was modified to meet the strict requirements that we set for the SELMA dataset. Namely, bikes could only exist along with their rider on board, which prevented parked bikes to be deployed. Nonetheless, bikes are amongst the main road actors, and CV algorithms greatly benefit from visual variability. Indeed, we deemed fundamental for our dataset to include the bike class in all the contexts. Therefore, we added hundreds of parked bikes into the existing CARLA maps. Similarly, the number of traffic lights and signs in the default towns did not reflect their distribution in a real setting, thus possibly compromising the learning ability of the algorithms. Thus, we distributed tens of additional traffic lights and signs in every town. The final class distribution matches that of real-world reference datasets such as Cityscapes, as shown in Fig. 4. The customized simulator is freely available³

IV. SELMA DATASET DESIGN

In this section we present our SELMA dataset, with a focus on the acquisition setup (Sec. IV-A) and splits (Sec. IV-B).

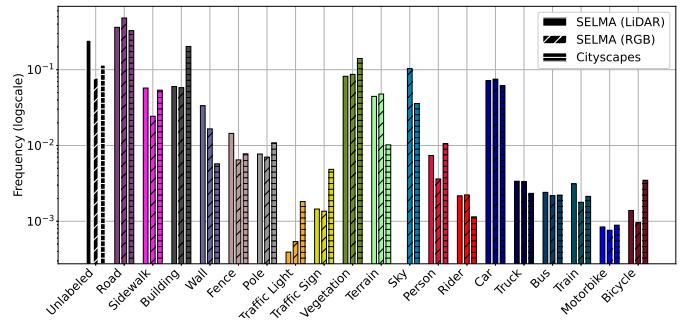


Fig. 4: Class distributions in the SELMA dataset.

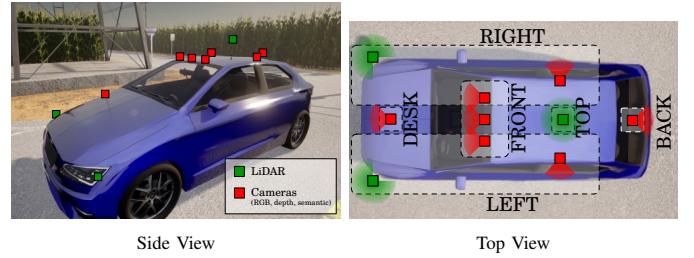


Fig. 5: Side and top views of the sensor setup in SELMA. RGB, depth and semantic cameras are co-located in 7 spots. LiDARs are placed in 3 locations.

A. Acquisition Setup

We designed the acquisition pipeline to exploit the full potential of CARLA, while maximizing the diversity of the acquired data. Acquisitions were made equipping a vehicle, named the ego vehicle, with a full sensor suite depicted in Fig. 5, consisting of:

- 7 RGB cameras, with the post-processing effects enabled, a 90-degree horizontal field of view (FoV), and a native resolution of 5120×2560 , which is downsampled to 1280×640 to achieve a $\times 4$ anti-aliasing enhancement. The post-processing effects include vignette, grain jitter, bloom, auto exposure, lens flare and depth of field. RGB images are saved in JPEG format.
- 7 depth cameras with a 90-degree horizontal FoV and 1280×640 resolution. For the depth images, anti-aliasing is not required and would compromise depth information. Depth images are saved in PNG format.
- 7 semantic cameras, that have the exact same attributes of the depth cameras.
- 3 semantic LiDARs, each with 64 vertical channels, generating 100 000 points per second, with a range of 100 meters. Point clouds are saved in PLY format.

The different camera types are co-registered at 7 different locations, and are set up to have the same FoV and resolution, so that their data can be easily matched. The variability of viewpoints and maps is shown in Fig. 6.

Furthermore, we compute the 3D surface normals at each pixel of the image acquired by the desk camera for all samples of the default random split (see Sec. IV-B). To do so, we employ a state-of-the-art differential technique as done in [53], [54]. While the result is an approximation of the true normals,

³Our customized version of CARLA will be made available upon acceptance.

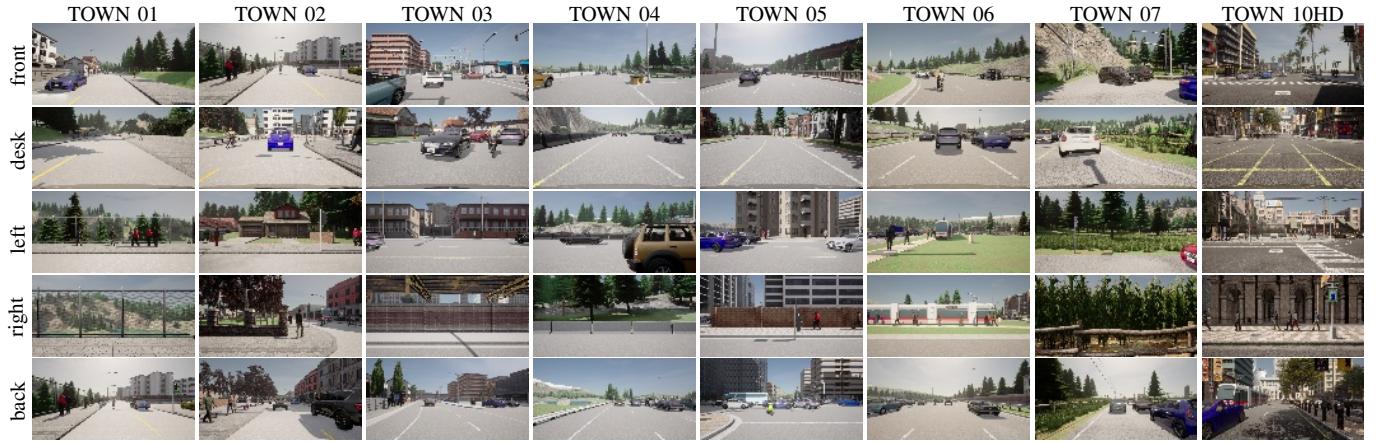


Fig. 6: Randomly sampled images from the SELMA dataset in clear noon setup, demonstrating its diversity. Rows show different cameras, while columns show different (synthetic) towns (thus settings).

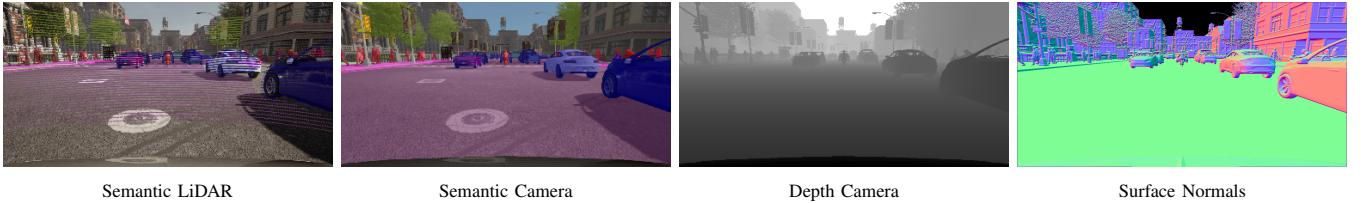


Fig. 7: Sample acquisitions with semantic LiDAR, semantic camera overlaid to the RGB samples, depth camera, and surface normals.

TABLE II: Number of waypoints (WPs) per town. Data are acquired at each waypoint, independently for all environmental conditions.

Town ID	01	02	03	04	05	06	07	10HD	Total
# of WPs	1634	756	3636	8565	6250	6579	1922	1568	30909

the overall precision is very high, as can be seen in the last column of Fig. 7, thanks to the detailed (i.e., synthetic ground truth) and dense depth maps used for the estimation procedure.

Data were acquired in 30 909 independent locations, across 8 virtual towns as reported in Table II and in Appendix A. The locations were selected extracting a list of waypoints at a given distance. For our dataset, we selected points on the roads on every lane and junction, at the distance of 4 meters, which was chosen after several empirical tests as it offered the best trade-off between area coverage and acquisition diversity. The full list of waypoints with their ID is provided with the dataset for each town. At each position, the ego vehicle is created, traffic is generated around it, and pedestrians are randomly placed on the sidewalks. After one second of simulation for the transient to end, the sensors are fired simultaneously, and their data retrieved and saved. The server is then reset and the simulation goes on with the following waypoint.

The same process is repeated in 27 different environmental conditions. These include 3 daytimes (i.e., Noon, Sunset and Night) and 9 weather conditions (i.e., Clear, Cloudy, Wet, Wet and Cloudy, 2 Fog intensities and 3 Rain intensities). Traffic and pedestrians are generated randomly at every iteration, thus the same waypoint simulated under different environmental conditions presents different traffic conditions. We refer to the combinations of environmental conditions and towns as scenes. Our dataset consists of 216 scenes, obtained considering all the sensors and the complete combinations of the available weather and daytime conditions, viewpoints, and towns.

B. Splits

Exploiting the fine-grained control on environmental conditions offered by SELMA, we designed some default splits. Particularly, we considered 6 different weather distributions: Random (SELMA default), Mostly Clear (MC), Noon, Night, Rain and Fog. For more details on the splits, we refer the interested readers to Appendix B.

The Random split contains samples from all weather conditions and daytimes, sampled according to the probability distributions reported in Table III. Most of the samples come from high-visibility weather conditions: Clear, Wet (road), Cloudy and WetCloudy make up for 75% of the split.

In order to preserve the separation among training, validation and test samples, the splits are provided in CSV format, which allows to easily assign a given weather condition to a sample (and to override it, if needed). The samples separation was done according to an 80:10:10 split rule for training, validation and test, respectively.

V. EXPERIMENTAL VALIDATION

In this section we carefully analyze and validate our SELMA dataset. We start with a series of baseline experiments which serve as a reference benchmark for future studies (Sec. V-A). Then, we analyze the thematic subsets of our dataset (Sec. V-B). To conclude, we show how different sensors can be employed jointly to improve the final segmentation accuracy (Sec. V-C), and report some experiments exploiting multiple viewpoints (Sec. V-D).

TABLE III: Probability distributions of environmental conditions in the different splits.

Split	Noon	Sunset	Night	Clear	Cloudy	Wet Road	Wet Road and Cloudy	Mid Fog	Hard Fog	Soft Rain	Mid Rain	Hard Rain
SELMA/Noon/Night	50/100/0%	25/0/0%	25/0/100%	35%	20%	10%	10%	3.5%	3.5%	6%	6%	6%
Mostly Clear/Rain/Fog	50%	25%	25%	25/0/0%	25/0/0%	25/0/0%	0/0/50%	0/0/50%	0/34/0%	0/33/0%	0/33/0%	

TABLE IV: mIoU of baseline SS architectures on Cityscapes (CS) and subsets of SELMA for both RGB images (first 7 columns) and depth (last column).

	CS	SELMA	Noon	Night	MC	Fog	Rain	Depth
DeeplabV2 [55]	67.4	68.9	72.3	68.2	69.9	68.0	68.7	73.4
DeeplabV3 [56]	68.2	70.7	71.7	68.4	70.8	69.0	67.8	72.7
FCN [57]	64.8	68.2	71.1	66.1	69.1	64.5	66.8	73.7
PSPNet [58]	65.3	68.4	71.2	67.2	69.8	66.8	69.0	73.6
UNet [59]	36.8	36.2	41.8	35.7	36.3	28.6	37.8	28.8

A. Baseline Experiments

The first set of experiments is designed to provide a series of benchmark results for the SELMA dataset. The results in Table IV show the performance achieved employing different baseline SS architectures, i.e., UNet [59], FCN [57], PSPNet [58], DeepLab-V2 [55], [60], DeepLab-V3 [56]. All the networks are trained with SGD with momentum of rate 0.9. The learning rate was decreased according to a polynomial decay of coefficient 0.9, starting from 2.5×10^{-4} . The batch size was set to 3 and the weight decay to 10^{-4} . The results for the individual classes are reported in Appendix C1.

a) *RGB*: Initially, we perform a series of experiments using the RGB images from Cityscapes and from the SELMA desk camera in different environmental conditions. First, we observe that the UNet architecture achieves poor results, since it is unable to deal with the large visual variability of our SELMA dataset. The other architectures share the same encoder module, i.e., a ResNet-101, and they all achieve similar mean Intersection over Union (mIoU) performance. Overall, DeepLab-V2 and V3 offer the best performance; for the sake of the performance-complexity trade-off we decided to employ DeepLab-V2 in all the following experiments. The highest accuracy is obtained with the SELMA Noon split, as RGB images are easier to segment. On the contrary, Night, Fog and Rain decrease the accuracy of models.

b) *Depth*: We run the same experimental evaluation using a single input channel representing the depth of the scene. More precisely, since the range of true values is extremely unbalanced and their distribution is highly skewed, we normalize and rescale the depth values: starting from the original depth produced by the simulator, normalized to 1, we compute its fourth root to compress the high-distance information and expand the low-distance information. This is necessary as the sky is marked with the maximum distance possible, and overshadows the other pixels. Then, we rescale and shift the values to the $[-1, 1]$ range. Also in this case, the best performing architectures achieve comparable performance, as shown in the rightmost column of Table IV. The models trained on depth images can more easily segment objects of different classes, outperforming the results achieved on RGB samples.

c) *LiDAR*: Table V reports the LiDAR SS results obtained with RangeNet++ [62] with two backbones

TABLE V: Baseline LiDAR SS methods on SELMA point clouds. Results on the CS label split, removing sky.

Architecture	mIoU
RangeNet++ [62] (SqueezeSeg-V2)	61.9
RangeNet++ [62] (DarkNet-21)	67.4
Cylinder3D [61]	80.3
DLV2 [55] on spherical projections	57.3

TABLE VI: mIoU fusion performance.

Method	mIoU
RGB	68.9
Grayscale	68.0
Depth	73.4
RGBD	72.4
RGBD @layer1	74.3

(SqueezeSeg-V2 [63] and DarkNet-21 [64]) and Cylinder3D [61]. Furthermore, we report the results obtained by flattening the point cloud to an RGB image via spherical projection and employing DeepLab-V2 to segment it. All the backbones are trained with batch size of 4 for 40 epochs with early stopping enabled. The other learning parameters are left to the default values provided in the respective codebases. We can observe that Cylinder3D outperforms the other architectures, achieving an outstanding mIoU of 80.3.

Fig. 8 reports the qualitative results for the best segmentation architectures, i.e., the DeepLab-V2, for the RGB and the depth samples, and Cylinder3D for the point clouds. Comparing the RGB and depth-based prediction, we can appreciate that the latter offers great improvements in the recognition of far, small and challenging items in the background, such as poles, traffic lights and traffic signs. However, the use of geometric information leads to uncertainty in the prediction of traffic lights and signs, which are mixed up in the depth prediction, but not in the RGB one. On the other hand, looking at the point cloud segmentation, we can appreciate the great overall precision, as expected given the high quantitative score. Nevertheless, some artifacts are still present due to the prediction based solely on geometric information, e.g., the sidewalk region on the left is partially confused for ground in proximity of the vegetation class. Another interesting artifact lies in the geometrical arrangement of the errors. Due to the intrinsic working principle of Cylinder3D [61], we observe that most of the errors are propagated along the same radial coordinates. For instance, we see that the network predicts ground in spite of sidewalk for a few consecutive scans in a couple of regions denoted by light blue rectangles. Finally, the prediction performance of semantic labels is poor for small classes such as traffic signs or lights, which are often confused for poles.

B. Thematic Subsets

Then, to highlight the capability of SELMA to incorporate different visual domains, we define 6 subsets, the so-called *thematic* splits, sampling images with specific daytime or weather condition, as mentioned in Sec. IV-B.

The mIoU results for the splits are reported in Fig. 9, where we report the supervised accuracy in the diagonal elements and the source-only accuracy (i.e., trained on the source domain

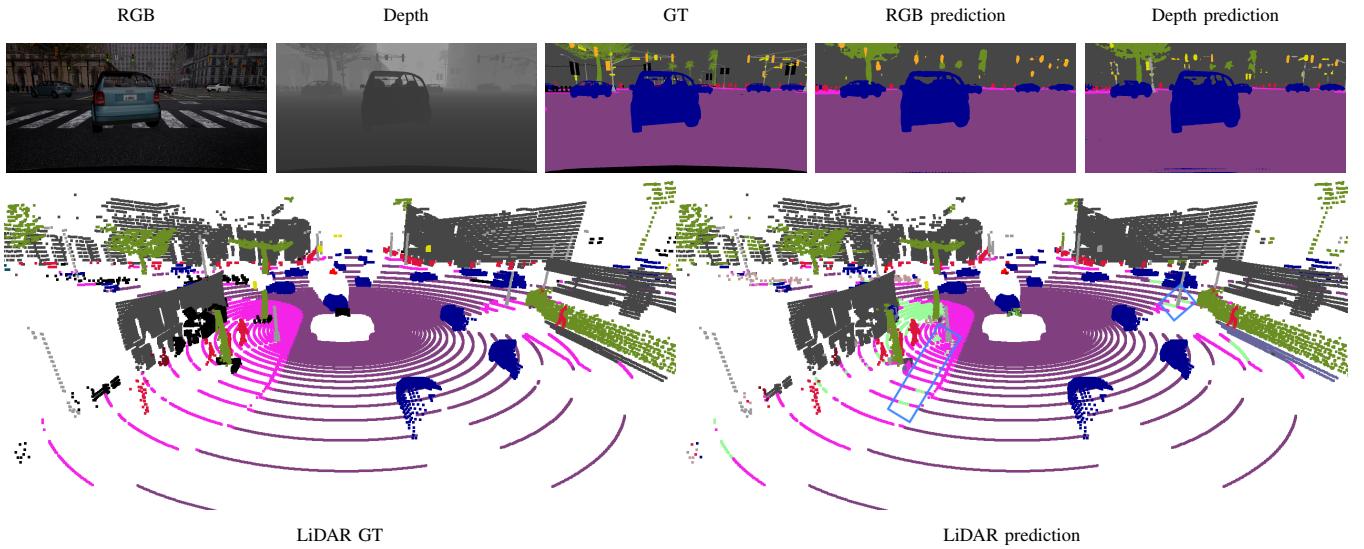


Fig. 8: Qualitative results for the SS task from RGB images (DeepLab-V2 [55]), depth maps (DeepLab-V2 [55]) or point clouds (Cylinder3D [61]).

Source	Target					
	-SELMA	-SELMA Noon	-SELMA Night	SELMA MostlyClear	SELMA Rain	-SELMA Fog
SELMA	68.9	70.6	65.1	69.6	67.7	65.2
SELMA Noon	60.9	71.3	44.9	62.0	60.7	58.7
SELMA Night	54.2	48.9	67.6	55.6	57.7	42.6
SELMA MostlyClear	68.3	70.6	64.1	69.9	65.1	63.7
SELMA Rain	67.9	69.9	62.3	68.2	68.7	61.8
SELMA Fog	62.2	63.0	61.4	63.9	57.9	66.3

Fig. 9: mIoU results on thematic splits sub-sampled from the complete SELMA dataset.

and tested on the target domain) on off-diagonal elements. Here, we can appreciate that training and testing on the same visual domain give almost always the highest mIoU (diagonal elements), except for some cases where the target domain is much easier than the source domain, as is the case for the SELMA Noon as target dataset.

In absolute terms, the hardest subsets (i.e., lowest supervised accuracy) are SELMA Night, Fog, and Rain, respectively. Adapting the source knowledge acquired from a subset containing a single daytime (e.g., Noon or Night) proves to be less robust to domain variability at test time, rather than adapting knowledge from a subset containing multiple daytime domains (e.g., Rain or Fog). Indeed, in the first case we can observe lower off-diagonal accuracy scores compared to the second case.

C. Fusion Experiments

To prove the importance of SELMA as a multimodal dataset, we show how we can improve the segmentation quality by coupling acquisitions from different sensors. Indeed, different sensors have variable performance depending on the visibility conditions of the scenes, and can be used jointly to leverage understanding scores. To highlight this, we report some experiments in Table VI. Notably, using RGB images allows to achieve an mIoU of 68.9, while using the depth alone

TABLE VII: mIoU multi-view results. DeepLabV2 [55] is trained on samples from the desk camera and tested on other cameras.

Target	mIoU
Desk	68.9
Front Left	66.6
Front	66.6
Front Right	66.6
Left	66.2
Back	65.9
Right	62.2

could improve the mIoU to 73.4. As a comparison, using the grayscale version of the image (single channel) we achieve an mIoU of 68.0, which is lower than the result on RGB images, as expected.

Building a combined RGB and depth representation at the input level (denoted as RGBD, i.e., an input of 4 channels), we achieve an mIoU of 72.4, which is higher than using RGB alone, but lower than using depth alone. Hence, we argue that simply combining the input representations as they are provided is not enough to increase the SS accuracy. Therefore, we include an additional convolutional layer (followed by batch normalization and ReLU activation function) to extract features from RGB and depth samples separately. Then, we concatenate the outputs and feed the result as the input of the first layer of the ResNet101. We denote this fusion method as “RGBD @layer1” in Table VI. With this simple provision, we could achieve an mIoU of 74.3, an improvement of 5.4 points with respect to using RGB alone, and by 0.9 points with respect to using depth alone. We believe that future research could leverage the multimodal design of SELMA to extensively investigate a wide range of fusion strategies for different sensors, such as RGB, depth and LiDAR.

D. Multi-View Experiments

As baseline experiments for the multi-view aspect of our dataset, we consider an architecture trained on the desk camera and tested on the available points of view. This permits to verify the intra-domain shift caused by the variable camera viewpoints, which we expect to be more significant in the cameras facing different directions (like Left, Right and Back). The quantitative results of our experiment are reported in Table VII. The mIoU scores confirm our expectations, i.e., all front facing cameras have minimal performance degradation (2.3% mIoU) and are all similar to each other due to their reciprocal proximity (recall Fig. 5). The second best adaptation is achieved by the left-facing camera (with 66.6% mIoU, only 0.4% less than the front cameras), and we argue that this

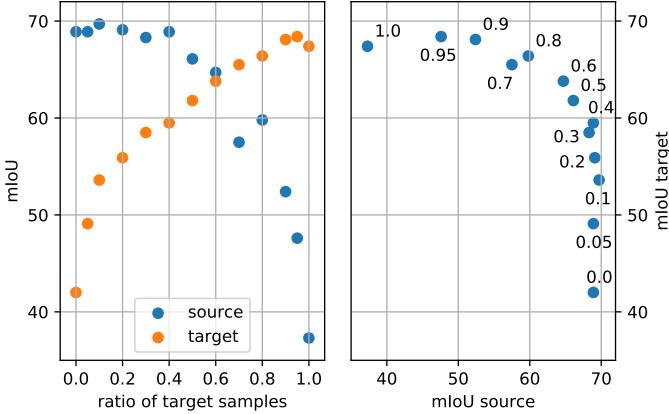


Fig. 10: Accuracy (mIoU) with images sampled either from the Cityscapes dataset (target) or from our SELMA dataset (source).

depends on the right-side driving simulated scenario, meaning that the point of view of the left camera is less occluded than its right-facing counterpart, which is much closer to the buildings on the road side. This assumption is verified by looking at the right-facing camera score, which is the lowest among the cameras, with a significant loss of 6.7% mIoU compared to the desk camera. Finally, the back-facing camera performs similarly to its left-facing counterpart, achieving a score of 65.9% which is only 0.3% lower than the latter.

VI. EXPERIMENTS ON REAL-WORLD DATASETS

In the last set of experiments we validate our SELMA dataset on different SS models. We run an extensive evaluation by training the DeepLab-V2 segmentation architecture on the SELMA dataset and testing it on different real-world datasets.

A. Training with a Mixture of Synthetic and Real Data

To start, we show in Fig. 10 the mIoU accuracy on source and target sets when training the segmentation network on samples drawn either from the target real-world domain, i.e., the Cityscapes dataset (with probability r), or from the source domain, i.e., our SELMA dataset (with probability $1 - r$). Adding as few as 5% to 10% of data from a different domain improves domain generalization, i.e., the network can perform well on both domains. Even more, we highlight that a 5% of samples from SELMA can improve the performance on the target domain from 67.4% to 68.4%. Similarly, 10% of target samples improve the performance on the source domain from 68.9% to 69.7%. The per-class accuracy is reported in the Appendix C1.

To analyze the performance gain when considering imprecise labels, we trained the same architecture using a mixture of SELMA and coarsely-annotated samples from Cityscapes. We achieved an mIoU score of 59.5%, which is 7.9% lower than the fully supervised training on Cityscapes, and 2.3% lower than the mixed training score when $r = 0.5$. This demonstrates that we can bridge the domain gap with few coarsely-supervised samples in the target domain, thus reducing the cost for accurate labeling.

Source	Target	mIoU										
		SELMA	GTAS	IDDA*	SYNTHIA*	ACDC	Cityscapes	Cityscapes Fog	Cityscapes Rain*	IDD*	Mapillary	NightCity
SELMA	SELMA	68.9	39.5	54.3	31.4	27.9	42.0	38.0	37.9	38.3	35.6	20.7
GTAS	GTAS	31.3	68.4	36.3	28.6	29.7	40.4	36.0	31.0	43.5	44.3	19.1
IDDA*	IDDA*	41.3	34.4	74.7	30.1	22.0	34.6	28.0	28.5	31.6	34.3	13.5
SYNTHIA*	SYNTHIA*	27.3	28.8	40.3	69.1	19.6	28.6	24.1	24.1	26.9	28.7	14.0
ACDC	ACDC	32.5	35.4	38.7	25.3	59.3	45.7	40.9	41.5	37.9	43.9	32.4
Cityscapes	Cityscapes	37.3	42.9	44.3	30.3	41.3	67.4	55.8	47.6	45.5	50.1	24.9
Cityscapes Fog	Cityscapes Fog	34.6	40.5	37.1	26.1	35.1	62.4	66.0	55.6	40.3	42.8	20.6
Cityscapes Rain*	Cityscapes Rain*	27.3	28.8	29.6	22.2	29.9	42.5	44.3	49.2	25.8	31.4	16.1
IDD*	IDD*	35.6	39.5	38.7	29.6	32.2	43.2	37.0	35.4	67.1	44.7	21.4
Mapillary	Mapillary	41.3	51.8	49.0	36.9	48.0	58.8	36.5	44.9	52.6	63.5	31.7
NightCity	NightCity	26.0	27.3	32.6	22.3	32.6	36.9	28.0	25.8	33.3	34.5	48.5

Fig. 11: mIoU performance for different synthetic and real-world datasets. The network is trained on a source domain (rows) and tested on a target domain (columns). Off-diagonal elements correspond to the presence of domain shift. The asterisk (*) indicates that a different label set is employed for testing (see Appendix C1 for further details).

B. Training with Synthetic Data Only

We then considered an unsupervised setup where no data from the target dataset were used for the training. We performed an extensive validation in the presence of domain shift, whose results are reported in Fig. 11: we considered four synthetic datasets (in red) and seven real-world datasets (in blue) with extremely variable time and weather conditions. We trained a DeepLab-V2 architecture (with ResNet-101 as backbone) on each dataset, and performed the testing on all the domains. Values on the diagonal correspond to training and testing performed on the same dataset, i.e., standard supervised training, while values off the diagonal correspond to training on a source domain (on the rows) and testing on a different target domain (on the columns). As expected, the values on the diagonal are larger than the others, since there is no domain shift.

In general, the mIoU performance depends on the source domain where training is performed. For example, a source training on SELMA performs well on IDDA, and vice-versa, since the rendering engine is common for the two datasets. Also, a source training on Cityscapes performs well on Cityscapes Fog or Cityscapes Rain, and vice-versa, since the city-level domain is common for the two datasets.

On the other side, training on datasets which only account for daytime images, such as Cityscapes, SYNTHIA or IDDA, struggles to generalize to nighttime images, e.g., sampled from the NightCity dataset. Training on statistically variable datasets, such as SELMA or Mapillary, can greatly improve the generalization capabilities in challenging domains.

Furthermore, we can observe that source training on the SELMA dataset outperforms the other synthetic datasets, on the widely-used real-world Cityscapes dataset. Indeed, source knowledge acquired on SELMA transfers well to Cityscapes, achieving 42.0% of mIoU, higher than transferring knowledge

	SELMA Noon	SELMA Night	SELMA MostlyClear	SELMA Rain	SELMA Fog	GTA5	IDDA*	SYNTHIA*	ADCD	Cityscapes	Cityscapes Fog	Cityscapes Rain*	IDD*	Mapiillary	NightCity	
To Synthetic	59.1	53.6	45.1	58.6	57.4	52.3	34.6	41.3	28.7	31.6	36.2	33.9	29.8	37.1	41.5	29.4
To Real	34.3	32.6	22.1	33.4	31.9	28.7	34.9	27.5	23.7	43.1	47.5	46.1	34.2	40.1	48.0	34.2
From Synthetic	53.6	56.2	48.9	54.8	51.9	49.3	37.6	49.2	33.1	24.1	37.2	32.4	31.8	34.1	32.9	17.0
From Real	33.5	36.3	26.9	34.9	30.4	29.5	38.0	38.6	27.5	39.8	51.0	44.1	42.9	43.2	44.4	27.9
Avg Synthetic	56.4	54.9	47.0	56.7	54.6	50.8	36.1	45.2	30.9	27.8	36.7	33.2	30.8	35.6	37.2	23.2
Avg Real	33.9	34.4	24.5	34.2	31.1	29.1	36.4	33.0	25.6	41.4	49.3	45.1	38.5	41.7	46.2	31.1

Fig. 12: mIoU performance results taken from the heatmap in Fig. 11, and aggregated by synthetic or real domain. The asterisk (*) indicates a different label set (see Appendix C1).

from GTA5 (40.4%), IDDA (22.0%) or SYNTHIA (28.6%). Remarkably, SELMA outperforms both GT5A (which is the most popular dataset for this task) by 1.6% of mIoU, and IDDA (i.e., the most similar dataset based on the same graphic engine) by an outstanding 20% of mIoU.

Finally, in Fig. 12 we show the results averaged according to synthetic or real domains. First, we can appreciate that source domains have better performance on other source domains with respect to target ones, and so do target domains. This is to be attributed to different textures, colors, and brightness rendered by the synthetic graphic engines versus the true target properties of real-world datasets. In general, we observe that acquiring source knowledge on our SELMA dataset (or its subsets) leads to much higher accuracy scores (e.g., 34.3% from SELMA) on both source and target domains, rather than IDDA [23] (27.5%) or SYNTHIA [21] (23.7%) datasets. Overall, SELMA achieves similar scores than acquiring source knowledge from the baseline GTA5 dataset.

VII. CONCLUSIONS

In this paper we presented SELMA, a synthetic dataset with driving scenes that contain a large amount of labeled samples acquired considering several different sensors, weather, daytime and viewpoint conditions. The experimental evaluation shows that SELMA allows to efficiently train deep learning models for scene understanding tasks in the autonomous driving context, achieving a good generalization to real-world data. The SELMA dataset is publicly available and can be downloaded for free, in the hope that it will be useful to the scientific community.

The availability of large-scale multimodal acquisitions in variable weather, daytime and viewpoints in SELMA promotes research towards key challenges, for example scene understanding for autonomous driving applications like multimodal sensor exploitation, domain generalization from synthetic datasets to real scenes, and autonomous driving in adverse weather conditions.

APPENDIX

In this appendix we include some additional implementation details and some experiments supporting the evaluation of our proposed dataset (SELMA). In particular, we start by reporting an overview of the CARLA [19] simulator in Appendix A, to better underline the elements that needed modification



Fig. 13: Samples in the 27 environmental conditions (9 weather and 3 daytime conditions).

to support the full scope of SELMA’s contributions. Then, Appendix B present in detail the thematic environmental splits included in SELMA, which are briefly introduced in Sec. IV-B. Finally, we report a more detailed version of the results presented in Sec. V.

A. CARLA Simulator

Overview: CARLA is built as a client-server architecture. The server handles the Unreal Engine simulation, i.e., the rendering, the physics computation, the world and actor states. The client offers an API (in Python and C++) that handles over part of the control on the simulation to the user. Specifically, when enabling the synchronous mode, the server waits for a control signal from the client, i.e., a *tick*, before updating the simulation. Together with a fixed time-step length, this allows to obtain reproducible and reliable physics simulations, as well as realistic and synchronized sensor data. Namely, the main elements of the Python API are

- the *Client*, that works as a communication interface to the server. It allows to access and modify the high-level simulation settings, e.g., the loading of the World and of the Traffic Manager, and the connection to the server.
- the *World*, that controls the large-scale settings of the environment such as the Weather and the Map, the client-server synchronization, and the life cycles of the Actors.
- the *Actors* are NPCs that can be controlled via the API. They include Pedestrians, Vehicles, Sensors, Traffic Signs and Traffic Lights.

Specifically, through the API it is possible to send commands and meta-commands to the server and receive the sensor readings. Commands and meta-commands control the Actors (e.g., steering and braking of Vehicles) and the behavior of the server (e.g., resetting the simulation, changing the weather and visibility conditions), respectively.

Sensors: Namely, there are 14 sensors available in CARLA: Collision detector, Depth camera, GNSS sensor, IMU sensor, Lane invasion detector, LIDAR sensor, Obstacle Detector, Radar sensor, RGB camera, RSS sensor, Semantic LIDAR sensor, Semantic Segmentation camera, DVS camera, and Optical Flow camera.

We report here additional details of the sensors used for the creation of the dataset:

- **RGB camera:** the camera offers a classical color view of the environment, as rendered by default in UE4. Camera parameters include horizontal field of view, lens aperture, lens distortion, image resolution, gamma value, motion blur, etc. A set of post-processing effects increasing the realism can be enabled with the `enable_postprocess_effects` flag: vignetting, grain jitter, bloom, auto exposure, lens flare and depth of field.
- **Depth camera:** each pixel of the image encodes the distance of the captured object from the camera (also known as depth buffer or z-buffer) to create a depth map of the environment in the FoV of the sensor.
- **Semantic segmentation camera:** every pixel of the image is labelled with the identifier of the object class it represents. When created, each object is tagged with its class identifier, thus making the segmentation exact and effortless, a key advantage over real datasets. This data is used as ground truth when training the segmentation models.
- **Semantic LiDAR:** LiDARs can produce a three-dimensional (3D) representation of the surrounding environment in the form of a point cloud. Generally, information encoded in each point includes its 3D coordinates and the back-scattered intensity. The CARLA semantic LIDAR does not capture intensity, but registers the semantic ground-truth of the object hit by the raycast. From the API, it is possible to set the number of channels N_c , the range, the vertical and horizontal FoV, the rotation frequency and the number of points per second P_s . The rotation is simulated computing the horizontal rotation angle of the LIDAR in a frame. The point cloud is generated through ray-casting at each time step, thus obtaining a number of points per channel per step P equal to $P = P_s / (\text{FPS} * N_c)$. A point cloud contains the points generated in a 1/FPS interval, during which the physics are not updated. Thus, all the points in a point cloud capture the same, static representation of the scene.

Towns: For convenience, we report here the town description that can be found in the CARLA documentation:

- **Town01:** A basic town layout with "T junctions" between roads.
- **Town02:** Similar to Town01, but smaller.
- **Town03:** The most complex town, with a 5-lane junction, a roundabout, unevenness, a tunnel, and more.
- **Town04:** An infinite loop with a highway and a small town.
- **Town05:** Squared-grid town with cross junctions and a bridge. It has multiple lanes per direction. Useful to deal with lane changes.
- **Town06:** Long highways with many highway entrances and exits. It also has a Michigan left.
- **Town07:** A rural environment with narrow roads, barns and hardly any traffic lights.
- **Town10HD:** A city environment with different environ-

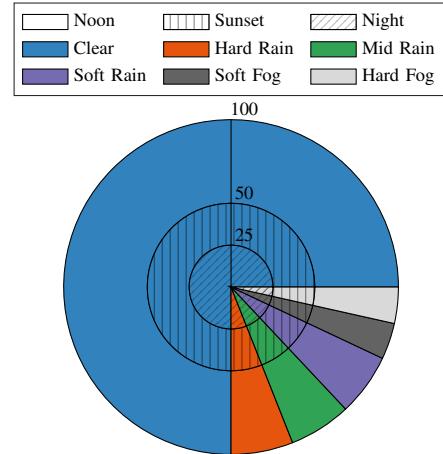


Fig. 14: Distributions of weather conditions and daytime in the SELMA-Rand split.

ments such as an avenue or promenade, and more realistic textures.

Besides the UE4 model files, the OpenDrive and a pointcloud representation are available. Specifically, the OpenDrive file contains all the information on the road topology, that can be used by the CARLA API to generate *Waypoints*, i.e., 3D directed points with attributes associated to OpenDrive properties.

B. Thematic Subsets of SELMA

Six thematic SELMA splits were designed sampling images from specific daytime or weather conditions. The splits are composed as follows:

- **SELMA:** with a slight abuse of notation, this split refers to the random split, whose probability densities are defined in Sec. IV-B and summarized in Fig. 14.
- **Noon** and **Night** splits share the same weather distribution defined for the random split, but the daytime is sampled from Noon and Night samples, respectively.
- **MostlyClear** split shares the same daytime distribution of the random split, whereas the weather samples on non-rainy and non-foggy images with a larger skew towards clear weather ($P[\text{Clear}] = 0.25$, $P[\text{Cloudy}] = 0.25$, $P[\text{WetCloudy}] = 0.25$, $P[\text{Wet}] = 0.25$).
- **Rain** split shares the same daytime distribution defined for the random split, but the weather is sampled from a rainy scenes ($P[\text{SoftRain}] = 0.34$, $P[\text{MidRain}] = 0.33$, $P[\text{HardRain}] = 0.34$).
- **Fog** split shares the same daytime distribution defined for the random split, but the weather is sampled from the foggy scenes ($P[\text{MidFog}] = 0.50$, $P[\text{HardFog}] = 0.5$).

C. Additional Experimental Results

1) **Supervised Learning:** In this paragraph, we report the per-class IoU results of the baseline semantic segmentation architectures.

RGB. Table VIII reports the per-class performance of the considered baseline semantic segmentation architectures:

TABLE X: Per-class and mean IoU results for the supervised training and testing on the same domain with multiple different LiDAR segmentation techniques.

	road	swalk	building	wall	fence	pole	night	tsign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
RangeNet++ [62] (SqueezeSeg-V2)	95.3	78.5	82.0	86.7	66.8	54.4	23.3	41.1	81.6	66.5	-	61.6	41.5	90.5	49.5	67.7	62.8	39.2	25.3	61.9
RangeNet++ [62] (DarkNet-21)	96.6	82.2	88.0	91.9	79.4	59.3	31.5	51.9	86.8	77.3	-	66.0	50.4	90.0	43.5	62.4	78.1	42.9	34.2	67.4
Cylinder3D [61]	96.7	65.8	85.7	79.0	87.5	75.6	87.9	91.2	88.5	48.9	-	95.0	89.5	98.9	84.5	55.3	42.2	88.7	84.6	80.3
DLv2 [55] on spherical projections	91.7	62.9	77.9	69.3	49.2	44.2	23.2	46.0	80.8	55.0	-	56.6	43.3	83.8	61.2	61.7	57.3	46.5	20.0	57.3

TABLE XI: Detailed view of the label sets considered.

	road	swalk	building	wall	fence	pole	night	tsign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	Count
City19	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	19
Idd17	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	17
Synthia16	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	16
Idda16	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	16
Idda-Synthia-15	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	15
Synthia-13	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	13
Idda-Synthia-12	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	12

UNet [59], FCN [57], PSPNet [58], DeepLab-V2 [55], [60], DeepLab-V3 [56]. The highest accuracy is obtained with the SELMA Noon split, as the RGB images acquired under favorable weather conditions are easier to segment. On the contrary, the presence of night, fog and rain, degrades the accuracy of the models. As we can observe, our results are more balanced across the classes with respect to the Cityscapes split.

Overall, DeepLab-V2 and DeepLab-V3 achieve the best performance. For the sake of the performance-complexity trade-off we decided to employ DeepLab-V2 for all the other experiments. In Table IX we show the per-class IoU scores training DeepLab-V2 on our datasets (first group), on common synthetic benchmarks (second group), and on real-world datasets (last group). For each dataset, we evaluate the trained model on the labels splits most commonly considered in the literature. Table XI reports the names of the classes of each label set. The most widely used label set is City19, which is also the most complete. For the sake of fairness with respect of all the datasets, in Table IX we test our models on all the possible label sets. Our SELMA demonstrates par, or often higher, performance compared to other synthetic and real benchmarks due to its extreme variability in daytime and weather conditions, as well as its large-scale property with more than 30,000 unique (labeled) samples.

Depth. Then, we run the same experimental evaluation using a single input channel representing the depth of the scene. We observe that the best performing architectures offer comparable performance and that the final mIoU is higher than mIoU obtained on RGB samples, since we have true (synthetic) depth maps at disposal. The results are reported as last block of Table VIII.

LiDAR. Table X presents the results obtained with two LiDAR semantic segmentation models, i.e., RangeNet and Cylinder3D [61], and with DeepLab-V2 [55] trained on LiDAR samples converted to spherical images. Note that the *sky* class is not included in the LiDAR samples, as no signal is backscattered from the sky. Cylinder3D [61] scores the best result, with an impressive 80.3% mIoU. Its performance is

consistent among classes, with the exception of *terrain*, *bus* and *train*, lower than 60%.

TABLE XII: Accuracy (mIoU) with images sampled either from the target Cityscapes dataset (r) or from our source dataset ($1 - r$).

r	SELMA	Cityscapes
0.0	68.9	42.0
0.05	68.9	49.1
0.1	69.7	53.6
0.2	69.1	55.9
0.3	68.3	58.5
0.4	68.9	59.5
0.5	66.1	61.8
0.6	64.7	63.8
0.7	57.5	65.5
0.8	59.8	66.4
0.9	52.4	68.1
0.95	47.6	68.4
1.0	37.3	67.4

2) *Domain Generalization:* To assess the domain generalization properties from of the proposed dataset, we run an extensive comparison with different datasets and using DeepLab-V2 as segmentation architecture to tackle real datasets either with labeled sample from the real-world domain or without.

In Table XII we show the mIoU accuracy on source and target sets when training the segmentation network on samples drawn either from the target domain (with probability r) or from the source domain (with probability $1 - r$). Adding as few as 5% to 10% of data from a different domain greatly helps in improving domain generalization, i.e., the network can perform well on both domains. Even more, we highlight that a 5% of source samples can significantly improve the performance on the target domain from 67.4% to 68.4%. Similarly, 10% of target samples improve the performance on the source domain from 68.9% to 69.7%.

In Fig. 11, we reported an extensive evaluation to investigate the domain generalization properties of models trained on a source domain with respect to a target domain. We show in Table XIII the per-class IoU results for the source only training on each domain and testing on the Cityscapes dataset.

Finally, we report in Table XIV the per-class IoU results for the supervised training on desk camera and testing on the different points of views.

REFERENCES

- [1] D. J. Fagnant and K. Kockelman, “Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations,” *Transportation Research: Policy and Practice*, vol. 77, pp. 167–181, Jul 2015.
- [2] G. Silberg, R. Wallace, G. Matuszak, J. Plessers, C. Brower, and D. Subramanian, “Self-driving cars: The next revolution,” *White Paper, KPMG LLP & Center of Automotive Research*, p. 36, 2012.
- [3] L. M. Clements and K. M. Kockelman, “Economic effects of automated vehicles,” *Transportation Research Record*, vol. 2606, pp. 106–114, 2017.

TABLE XIII: Per-class IoUs for the source only training on each domain and testing on Cityscapes.

	per-class IoUs																mIoUs on different label sets								
	road	swalk	building	wall	fence	pole	light	tsign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	city19	iddal16	synthia16	synthia13	idda-synthia15	idda-synthia12
Cityscapes [11]	97.1	77.4	89.2	50.1	46.2	45.2	48.9	61.4	89.5	55.2	92.2	69.9	46.5	91.4	66.9	75.2	60.3	52.2	65.8	67.4	67.4	68.6	73.6	68.2	73.5
SELMA	91.1	50.2	80.4	24.8	11.5	32.0	13.1	31.3	82.3	24.9	76.5	51.9	28.0	76.5	17.6	21.9	17.7	29.3	36.9	42.0	46.3	46.1	51.5	47.7	54.0
Noon	88.5	52.1	81.9	21.8	14.3	29.2	16.4	31.2	83.0	26.0	78.7	54.7	27.6	69.2	15.5	9.1	21.9	32.0	35.3	41.5	46.4	45.3	50.7	47.7	54.2
Night	88.1	38.7	66.5	8.5	9.0	30.7	5.5	26.7	70.0	14.5	8.1	41.4	24.1	65.0	7.0	13.8	10.2	24.3	33.5	30.8	34.7	34.6	38.9	36.0	41.0
MostlyClear	87.8	52.7	80.8	20.2	11.6	32.3	14.7	30.6	82.7	27.4	77.7	53.1	29.9	69.5	12.8	20.7	17.4	28.6	38.8	41.5	46.2	45.7	51.4	47.4	53.9
Rain	89.0	44.6	79.4	21.2	9.5	30.7	13.0	31.3	81.4	21.5	75.6	49.8	26.6	72.1	9.9	19.5	13.0	27.3	36.1	39.6	44.3	44.2	49.7	45.8	52.2
Fog	77.9	47.9	75.2	11.6	9.9	27.9	12.1	28.9	79.1	18.2	67.1	47.8	23.5	60.1	3.9	11.3	11.8	25.5	35.2	35.5	40.5	40.1	45.5	42.0	48.4
GTA5 [20]	70.3	21.9	82.5	29.3	25.4	31.7	32.5	16.6	84.4	41.1	82.6	52.1	7.9	78.3	33.2	34.0	7.9	25.9	9.9	40.4	43.3	42.8	46.1	43.4	47.1
IDDA [23]	88.4	49.8	78.0	15.8	8.7	35.2	19.8	9.1	81.3	21.8	69.5	48.8	23.4	71.3	-	-	-	15.3	21.8	34.6	41.1	39.8	44.3	42.4	48.0
SYNTHIA [65]	43.3	18.6	78.0	14.0	0.1	32.7	5.4	15.9	81.2	-	80.0	55.7	19.2	63.5	-	0.0	-	8.0	23.3	28.4	33.7	33.7	37.9	35.9	41.0
ACDC [44]	82.8	46.1	81.0	26.9	22.1	33.6	27.6	41.9	85.1	37.9	80.8	53.8	25.5	76.9	30.2	35.8	13.7	16.8	49.5	45.7	49.3	49.1	54.1	50.0	55.7
Cityscapes Fog [51]	96.9	75.5	87.0	43.5	37.5	41.4	39.1	56.4	87.2	52.0	88.9	65.6	42.6	89.5	57.9	71.6	48.4	44.0	60.7	62.4	63.0	64.2	69.6	63.7	69.5
Cityscapes Rain [52]	93.4	58.9	87.3	9.1	9.6	34.9	42.7	57.8	89.2	61.1	91.8	63.0	48.1	87.1	20.4	75.3	0.0	5.1	67.3	52.7	56.7	57.5	66.7	56.4	66.0
IDD [34]	88.9	21.9	84.6	30.5	24.9	37.5	6.1	30.7	86.8	-	90.6	61.6	34.8	83.6	43.0	44.1	-	15.3	36.1	43.2	45.9	48.6	52.7	48.9	53.4
Mapillary [32]	93.9	60.5	87.2	48.4	39.5	42.3	41.0	53.3	88.4	46.4	90.7	64.0	36.2	89.9	54.8	64.4	13.5	43.7	59.0	58.8	61.5	62.7	67.1	62.5	67.3
NightCity [50]	92.3	48.9	73.4	22.7	14.2	22.4	6.7	16.8	77.6	23.7	52.1	47.9	14.6	81.3	23.8	40.3	5.7	7.2	30.0	36.9	39.5	40.5	45.3	40.5	45.7

TABLE XIV: Per-class IoUs for the supervised training on the desk camera and testing on the different point of views.

	road	swalk	building	wall	fence	pole	light	tsign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
Desk	99.1	87.5	83.9	83.5	57.3	47.1	36.8	61.1	78.4	77.2	89.4	63.9	62.8	88.7	71.6	48.9	72.3	62.0	37.7	68.9
Front Left	96.7	76.7	84.3	78.8	53.8	45.4	43.7	58.0	76.9	73.8	92.8	62.8	56.4	89.6	65.1	48.5	69.2	58.7	33.6	66.6
Front	96.4	75.1	84.2	79.0	54.1	45.3	43.9	58.0	77.0	73.8	92.8	63.5	56.9	89.7	65.8	48.8	69.3	58.5	33.5	66.6
Front Right	96.1	73.5	84.1	79.0	54.5	45.2	44.6	57.6	77.0	73.7	92.8	64.0	56.9	89.6	65.8	48.9	70.1	58.5	33.4	66.6
Left	94.8	76.9	87.0	82.7	46.4	54.7	41.3	48.6	81.4	78.9	91.5	66.6	56.0	86.1	51.0	46.5	71.1	64.1	32.8	66.2
Back	97.3	80.9	84.6	82.8	53.9	46.7	34.1	46.6	78.6	76.7	93.7	64.4	55.6	88.1	56.7	49.6	69.3	60.3	31.7	65.9
Right	81.1	53.0	84.6	81.2	61.7	55.4	38.2	45.6	83.7	78.2	89.6	77.2	52.7	83.0	28.0	26.2	68.5	62.5	31.7	62.2

- [4] Y. Li and J. Ibanez-Guzman, “LiDAR for autonomous driving: The principles, challenges, and trends for automotive LiDAR and perception systems,” *IEEE SP Magazine*, vol. 37, no. 4, pp. 50–61, 7 2020.
- [5] F. Secci and A. Ceccarelli, “On failures of RGB cameras and their effects in autonomous driving applications,” in *IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, 2020.
- [6] V. Rossi, P. Testolina, M. Giordani, and M. Zorzi, “On the Role of Sensor Fusion for Object Detection in Future Vehicular Networks,” in *Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*, 2021.
- [7] M. Giordani, A. Zanella, T. Higuchi, O. Altintas, and M. Zorzi, “Performance study of LTE and mmWave in vehicle-to-network communications,” in *17th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2018.
- [8] ———, “On the Feasibility of Integrating mmWave and IEEE 802.11p for V2V Communications,” in *IEEE 88th Vehicular Technology Conference (VTC-Fall)*, 2018.
- [9] S. Zhang, J. Chen, F. Lyu, N. Cheng, W. Shi, and X. Shen, “Vehicular communication networks in the automated driving era,” *IEEE Comm. Magazine*, vol. 56, no. 9, pp. 26–32, 9 2018.
- [10] P. Sun, H. Kretschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, “Scalability in perception for autonomous driving: Waymo open dataset,” in *CVPR*, 2020, pp. 2446–2454.
- [11] 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 *CVPR*, 2016, pp. 3213–3223.
- [12] A. Geiger, P.Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *IJRR*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [13] J. Mao, M. Niu, C. Jiang, H. Liang, J. Chen, X. Liang, Y. Li, C. Ye, W. Zhang, Z. Li *et al.*, “One million scenes for autonomous driving: Once dataset,” *arXiv:2106.11037*, 2021.
- [14] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, “Virtual worlds as proxy for multi-object tracking analysis,” in *CVPR*, 2016, pp. 4340–4349.
- [15] M. Toldo, A. Maracani, U. Michieli, and P. Zanuttigh, “Unsupervised domain adaptation in semantic segmentation: a review,” *Technologies*, vol. 8, no. 2, 6 2020.
- [16] F. Barbato, M. Toldo, U. Michieli, and P. Zanuttigh, “Latent space regularization for unsupervised domain adaptation in semantic segmentation,” in *CVPRW*, 2021.
- [17] F. Barbato, U. Michieli, M. Toldo, and P. Zanuttigh, “Road scenes segmentation across different domains by disentangling latent representations,” *arXiv:2108.03021*, 2021.
- [18] U. Michieli, M. Biasetton, G. Agresti, and P. Zanuttigh, “Adversarial learning and self-teaching techniques for domain adaptation in semantic segmentation,” *IEEE T-IV*, vol. 5, pp. 508–518, 9 2020.
- [19] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *ACRL*, 2017, pp. 1–16.
- [20] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, “Playing for data: Ground truth from computer games,” in *ECCV*, 2016, pp. 102–118.
- [21] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” in *CVPR*, 2016, pp. 3234–3243.
- [22] Y. Cabon, N. Murray, and M. Humenberger, “Virtual KITTI 2,” 2020.
- [23] E. Alberti, A. Tavera, C. Masone, and B. Caputo, “IDDA: a large-scale multi-domain dataset for autonomous driving,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5526–5533, 10 2020.
- [24] D. Lahat, T. Adali, and C. Jutten, “Multimodal data fusion: an overview of methods, challenges, and prospects,” *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1449–1477, 9 2015.
- [25] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, “Imagenet: A large-scale hierarchical image database,” in *CVPR*, 2009, pp. 248–255.
- [26] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Gläser, F. Timm, W. Wiesbeck, and K. Dietmayer, “Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges,” *IEEE T-ITS*, vol. 22, no. 3, pp. 1341–1360, 3 2021.
- [27] G. J. Brostow, J. Fauqueur, and R. Cipolla, “Semantic object classes in video: A high-definition ground truth database,” *PRL*, vol. 30, no. 2, pp. 88–97, 1 2009.

- [28] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the KITTI vision benchmark suite,” in *CVPR*, 2012.
- [29] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *IJRR*, 1 2013.
- [30] J. Fritsch, T. Kuehnl, and A. Geiger, “A new performance measure and evaluation benchmark for road detection algorithms,” in *ITSC*, 2013.
- [31] M. Menze and A. Geiger, “Object scene flow for autonomous vehicles,” in *CVPR*, 2015.
- [32] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder, “The Mapillary vistas dataset for semantic understanding of street scenes,” in *ICCV*, 2017, pp. 4990–4999.
- [33] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, “BDD100K: A diverse driving dataset for heterogeneous multitask learning,” in *CVPR*. Computer Vision Foundation / IEEE, 2020, pp. 2633–2642.
- [34] G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, and C. Jawahar, “IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments,” in *WACV*. IEEE, 2019, pp. 1743–1751.
- [35] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuScenes: A multimodal dataset for autonomous driving,” in *CVPR*, 2020, pp. 11 621–11 631.
- [36] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet, “Level 5 perception dataset 2020,” <https://level-5.global/level5/data/>, 2019, ©Woven Planet Holdings, Inc. 2019.
- [37] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, “The apolloscape open dataset for autonomous driving and its application,” *IEEE T-PAMI*, 10 2020.
- [38] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, “Trafficpredict: Trajectory prediction for heterogeneous traffic-agents,” in *AAAI*, vol. 33, 7 2019, pp. 6120–6127.
- [39] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences,” in *ICCV*, 2019.
- [40] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, “Semantic3d.net: A new large-scale point cloud classification benchmark,” in *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-1-W1, 2017, pp. 91–98.
- [41] Y. Pan, B. Gao, J. Mei, S. Geng, C. Li, and H. Zhao, “SemanticPOSS: A Point Cloud Dataset with Large Quantity of Dynamic Instances,” *arXiv:2002.09147*, 2020.
- [42] P. Jiang and S. Saripalli, “LiDARNet: A Boundary-Aware Domain Adaptation Model for Point Cloud Semantic Segmentation,” *arXiv:2003.01174*, 2021.
- [43] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühllegg, S. Dorn, T. Fernandez, M. Jänicke, S. Mirashi, C. Savani, M. Sturm, O. Vorobiov, M. Oelker, S. Garreis, and P. Schuberth, “A2D2: Audi Autonomous Driving Dataset,” 2020. [Online]. Available: <https://www.a2d2.audi>
- [44] C. Sakaridis, D. Dai, and L. Van Gool, “ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding,” in *ICCV*, 2021, pp. 10 765–10 775.
- [45] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, “Argoverse: 3d tracking and forecasting with rich maps,” in *CVPR*, June 2019.
- [46] C. Sakaridis, D. Dai, and L. V. Gool, “Guided curriculum model adaptation and uncertainty-aware evaluation for semantic nighttime image segmentation,” in *ICCV*, 2019.
- [47] H. Sakashita, C. Flothow, N. Takemura, and Y. Sugano, “DRIV100: In-the-wild multi-domain dataset and evaluation for real-world domain adaptation of semantic segmentation,” *arXiv:2102.00150*, 2021.
- [48] Y.-H. Chen, W.-Y. Chen, Y.-T. Chen, B.-C. Tsai, Y.-C. Frank Wang, and M. Sun, “No more discrimination: Cross city adaptation of road scene segmenters,” in *ICCV*, 2017, pp. 1992–2001.
- [49] F. Tung, J. Chen, L. Meng, and J. J. Little, “The raincouver scene parsing benchmark for self-driving in adverse weather and at night,” *IEEE RAL*, vol. 2, no. 4, pp. 2188–2193, 2017.
- [50] X. Tan, K. Xu, Y. Cao, Y. Zhang, L. Ma, and R. W. Lau, “Night-time scene parsing with a large real dataset,” *IEEE T-IP*, vol. 30, pp. 9085–9098, 10 2021.
- [51] C. Sakaridis, D. Dai, and L. Van Gool, “Semantic foggy scene understanding with synthetic data,” *IJCV*, vol. 126, no. 9, pp. 973–992, 2018.
- [52] X. Hu, C.-W. Fu, L. Zhu, and P.-A. Heng, “Depth-attentional features for single-image rain removal,” in *CVPR*, 2019, pp. 8022–8031.
- [53] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia, “Unsupervised learning of geometry from videos with edge-aware depth-normal consistency,” in *AAAI*, vol. 32, no. 1, 2 2018.
- [54] V. Guzilini, J. Li, R. Ambrus, and A. Gaidon, “Geometric unsupervised domain adaptation for semantic segmentation,” in *ICCV*, 2021, pp. 8537–8547.
- [55] 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 T-PAMI*, vol. 40, pp. 834–848, 2018.
- [56] L. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv:1706.05587*, 2017.
- [57] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *CVPR*, 2015, pp. 3431–3440.
- [58] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *CVPR*, 2017, pp. 2881–2890.
- [59] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *MICCAI*, 2015, pp. 234–241.
- [60] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *ECCV*, 2018, pp. 833–851.
- [61] X. Zhu, H. Zhou, T. Wang, F. Hong, W. Li, Y. Ma, H. Li, R. Yang, and D. Lin, “Cylindrical and asymmetrical 3d convolution networks for LiDAR-based perception,” *IEEE T-PAMI*, 7 2021.
- [62] A. Milioto, L. Mandtler, and C. Stachniss, “Fast Instance and Semantic Segmentation Exploiting Local Connectivity, Metric Learning, and One-Shot Detection for Robotics,” in *ICRA*, 2019.
- [63] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, “SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud,” in *ICRA*, 2019, pp. 4376–4382.
- [64] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv:1804.02767*, 2018.
- [65] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, “The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes,” in *CVPR*, 2016, pp. 3234–3243.



Paolo Testolina received the B.Sc. in Information Engineering in 2016 and the M.Sc. with Honors in Telecommunication Engineering in 2018 from the University of Padova. He is currently working towards his Ph.D. in Information Engineering at the same university. His research focuses mainly on mmWave networks, from channel modeling to link layer simulation, traffic modeling, and vehicular networks.



Francesco Barbato received the M.Sc. degree in Telecommunication Engineering with Honors from the University of Padova in 2020. He is currently a Ph.D. student in the same University. His research focuses on unsupervised domain adaptation, multimodal learning and continual learning applied to computer vision tasks, particularly to SS for autonomous vehicles.



Umberto Michieli received his Ph.D. in Information Engineering from the University of Padova in 2021. Currently, he is a Postdoctoral Researcher and Adjunct Professor at the same University. He spent research periods at Technische Universität Dresden and Samsung Research UK. His research lies at the intersection of foundation AI problems applied to semantic understanding. In particular, he focuses on domain adaptation, continual learning, coarse-to-fine learning and federated learning.



Marco Giordani received his Ph.D. in Information Engineering in 2020 from the University of Padova, Italy, where he is now a Postdoc and Adjunct Professor. He visited NYU and TOYOTA, and was awarded with the IEEE VTS “Daniel E. Noble Fellowship Award” in 2018 and the IEEE ComSoc EMEA Outstanding Young Researcher Award in 2021. His research focuses on protocol design for 5G/6G networks.



Pietro Zanuttigh received his Ph.D. at the University of Padova in 2007. Currently he is an associate professor at the Department of Information Engineering. He works in the computer vision and machine learning fields, with a special focus on domain adaptation and incremental learning in semantic segmentation, 3D acquisition with ToF sensors, depth data processing, sensor fusion and hand gesture recognition.



Michele Zorzi is with the Information Engineering Department of the University of Padova, focusing on wireless communications. He was Editor-in-Chief of IEEE Wireless Communications from 2003 to 2005, IEEE Trans. on Communications from 2008 to 2011, and IEEE Trans. on Cognitive Communications and Networking from 2014 to 2018. He served ComSoc as a Member-at-Large of the Board of Governors from 2009 to 2011 and from 2021 to 2023, as Director of Education and Training from 2014 to 2015, and as Director of Journals from 2020 to 2021.