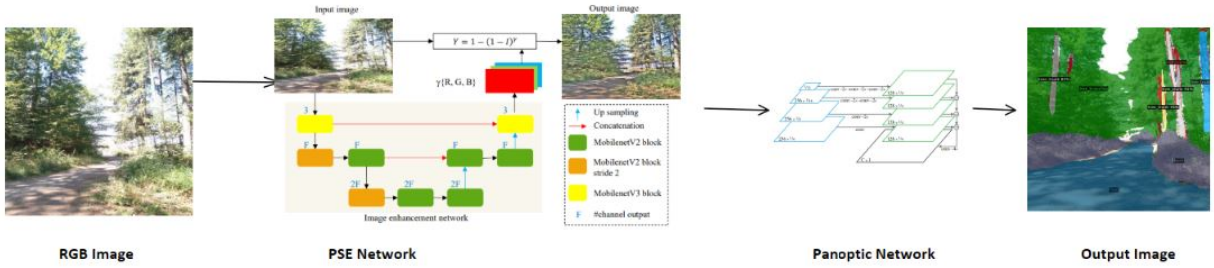
Robotics Research Lab
Department of Computer Science
University of Kaiserslautern-Landau

Master Thesis



Implementation of Panoptic Segmentation for off-road Autonomous driving

KOUSHIK SAMUDRALA

December 30, 2023

Master Thesis

Implementation of Panoptic Segmentation for off-road Autonomous driving

Robotics Research Lab
Department of Computer Science
University of Kaiserslautern-Landau

KOUSHIK SAMUDRALA

Day of issue : 01.07.2023
Day of release : 30.12.2023

First Reviewer : Prof. Dr. Karsten Berns
Supervisor : M.Sc Pankaj Deoli

Hereby I declare that I have self-dependently composed the Master Thesis at hand. The sources and additives used have been marked in the text and are exhaustively given in the bibliography.

December 30, 2023 – Kaiserslautern

Koushik Samudrala
Digitally signed by Koushik Samudrala
Date: 2023.12.30 12:04:46 +01'00'

(KOUSHIK SAMUDRALA)

Acknowledgements

I would like to express my sincere gratitude to everyone who provided their valuable support during my thesis. My heartful thanks go to my supervisor, M.Sc. Pankaj Deoli, for his invaluable support, guidance, and mentoring during my thesis. His profound knowledge and subject matter expertise helped me to finish this work.

I would also like to especially thank Prof. Dr. rer. nat. Karsten Berns, for providing me with this wonderful topic for my thesis at Robotics Research Lab.

Lastly, I would like to thank my family for their support and also my friends who pushed me to conquer this academic milestone.

Abstract

This thesis addresses the challenges faced in the implementation of Panoptic Segmentation for off-road environments, especially inside forests for the autonomous driving of vehicles without any human assistance. The off-road environment has challenging problems in terms of feature detection, illumination conditions, variability in the scenes and textures, etc. Perception systems are the eyes of autonomous vehicles and using their true potential can bring down the costs of the vehicles significantly. In recent years, autonomous driving research has been limited to Urban environments and datasets are generated to mimic the traffic conditions in these Urban settings. In this work, our goal is to investigate the possibility of extending the capability of the vehicle to maneuver autonomously in an off-road condition.

This work aims to implement a Panoptic segmentation model for autonomous off-road vehicles. To support this development, a new dataset named RPTU-Forest was collected using a multi-spectral camera, and the RGB images were captured with varying illuminations and exposure conditions. This dataset is collected only for the specific use case of navigation of a Unimog inside the forest environment with the help of Panoptic segmentation performed by a model trained on these scenes involving several challenges of illumination and shadows.

A comparative study on several available deep learning approaches for panoptic segmentation is done and the best of them is chosen for this task implementation using the collected RPTU-Forest dataset. The problem of over-exposure is addressed which helps the panoptic segmentation model in performing better with predictions.

Shadow removal techniques are investigated to address the darker regions in the images captured and also to help the panoptic model in terms of better feature extractions. Evaluation is performed on the trained Panoptic model and testing is done with several other unseen images by the network. Both Qualitative and Quantitative analyses of the model predictions are done in this work. A comparison of metrics obtained is done with other state-of-the-art (SOTA) approaches available for panoptic segmentation.

Contents

List of Figures 4
List of Tables 6

1 Introduction **7**
1.1 Autonomous driving 7
1.1.1 Urban Environments 9
1.1.2 Off-road environments 10
1.2 Environment Perception 11
1.2.1 Challenges 12
1.3 Motivation 13
1.4 Task 14
1.5 Contribution 14

2 Background **16**
2.1 Artificial Intelligence 16
2.1.1 Machine Learning 16
2.1.2 Deep Learning 18
2.2 Environment Segmentation 20
2.2.1 Semantic segmentation 20
2.2.2 Loss functions for semantic segmentation 22
2.2.3 Instance Segmentation 23
2.2.4 Evaluation of Instance Segmentation 25
2.3 Sensors for environment perception 26
2.3.1 Cameras including Stereo cameras 26
2.3.2 Multi-spectral Cameras 27
2.3.3 LiDAR 28
2.4 Approaches for environment segmentation 29
2.4.1 Segmentation using only images 29
2.4.2 Segmentation using only Point cloud 29
2.4.3 Sensor fusion approaches 30

3 Related works **32**
3.1 Panoptic Segmentation . . . 32
3.2 Available approaches for implementing panoptic segmentation . . . 34
3.2.1 Top-Down Approaches . . . 35
3.2.2 Top-down approaches(One-stage) . . . 38
3.2.3 Bottom-Up Approaches . . . 39
3.2.4 Single-Path approaches . . . 41
3.3 Works done in Urban environments . . . 43
3.3.1 Panoptic-DeepLab architecture . . . 44
3.4 Works done in off-road environments . . . 48

4 Approach and Implementation **50**
4.1 Dataset . . . 50
4.1.1 RPTU-Forest Panoptic dataset . . . 53
4.2 Baseline Network Architecture . . . 56
4.2.1 PanopticFPN Architecture . . . 56
4.3 Approaches to improve model predictions . . . 59
4.3.1 Shadow-Removal and exposure correction . . . 60
4.3.2 Exposure correction . . . 65
4.4 Training . . . 67
4.5 Evaluation Metrics for Panoptic Segmentation . . . 71
4.5.1 Segment matching . . . 72
4.5.2 PQ Computation given matches . . . 73

5 Experiments and Results **75**
5.1 Qualitative Performance . . . 75
5.1.1 Shadow-Removal . . . 75
5.1.2 Image exposure correction . . . 83
5.1.3 Panoptic Segmentation . . . 85
5.2 Quantitative Peformance . . . 89

6 Conclusion **92**
6.1 Summary and Discussion . . . 92
6.2 Further work and Improvements . . . 93

Bibliography **95**

List of Figures

1.1 Driving levels of automation for on-road vehicles as per (SAE-International 14) 9

1.2 Systems present for achieving driving in urban environments(Campbell 10) 10

1.3 Overview of various sensors mounted along with their positions for 360° environment perception. (Wendt 18) 12

2.1 Machine learning generic classification Algorithm. (Geitgey 14) 17

2.2 Machine learning generic classification Algorithm. (Terra 23) 18

2.3 Simple representation of a neural network elements. (Pathmind 23) 18

2.4 Figure showing convolution Operation. 19

2.5 Convolution Encoder-Decoder structure.(Barla 21) 21

2.6 U-Net structure.(Barla 21) 22

2.7 Mask R-CNN framework.(He 17) 25

2.8 Simple PR curve.(Liu 23) 26

2.9 Camera Image Processing pipeline.(Shahian Jahromi 19) 27

2.10 MultiSpectral camera.(Sequoia 21) 28

2.11 PointRCNN network structure.(Shi 19) 30

2.12 Early Fusion approach pipeline.(Shahian Jahromi 19) 31

3.1 Differences between (b) semantic segmentation (class labels per pixel), (c) Instance Segmentation (Class label per Object Mask), and (d) Panoptic Segmentation(Class label per pixel+instance masks per object).(Kirillov 19b) 33

3.2 Classification of Image Panoptic Segmentation methods.(Li 22) 35

3.3 Baseline of Panoptic-segmentation in two-staged model.(Li 22) 36

3.4 Instance segmentation in two-staged approach using Mask R-CNN architecture.(He 22) 36

3.5 EfficientPS network architecture is shown in red and two-way FPN in purple, blue, and green. The semantic segmentation network is in yellow and the instance segmentation network is in orange. The fusion block is shown at the end.(Mohan 21) 37

3.6 UPSNet baseline architecture.(Xiong 19) 38

3.7 SpatialFlow structure.(Chen 20) 39

3.8 DeeperLab structure.(Yang 19) 40

3.9 Pixel consensus voting for panoptic segmentation(PCV) structure.(Wang 20a) 41

3.10 Panoptic Fully Connected Network architecture.(Li 21) 42

3.11 Image showing the functioning of Kernel Update procedure in K-Net architecture.(Zhang 21) 42

3.12 Image showing the different dilation rates and their effect on the filters that are used for feature extraction(Chen 17b) 45

3.13 Mechanism of Spatial Pyramid Pooling module where features are extracted once and pooled at later stages.(Chen 17b) 46

3.14 Panoptic-DeepLab network architecture along with dual ASPP and dual decoder modules.(Cheng 20) 46

3.15 Groundtruth Annotations for FinnWoodland Dataset where the first row is the RGB Images captured during dataset creation and Groundtruth annotations for tasks like instance, semantic, and panoptic segmentation.(Lagos 23) 49

4.1 Samples of Yamaha-CMU-Off-Road Dataset. (Maturana 18) 53

4.2 Class labels in RPTU-Forest Panoptic Dataset. 54

4.3 Figure showing COCO-Panoptic data format structure 55

4.4 Figure showing panoptic predictions of COCO dataset(top) and Cityscapes dataset predictions(bottom) from (Kirillov 19a). 57

4.5 Figure showing PanopticFPN architecture blocks (a) FPN backbone for feature extraction at multiple scales (b) Mask R-CNN based on FPN for instance segmentation and (c) additional dense prediction branch for semantic segmentation. (Kirillov 19a). 57

4.6 Semantic segmentation dense-prediction branch added to FPN (Kirillov 19a). 58

4.7 SPA-Former network architecture.(Zhang 22). 61

4.8 ShadowFormer network architecture.(Guo 23). 62

4.9 Figure showing the cycle-consistency and Mask-guided Cycle consistency constraints from (Hu 19) 64

4.10 Figure showing the architecture of Mask-ShadowGAN with all the losses used for learning to remove shadows from shadow images. (Hu 19) 64

4.11 PSENet structure with main modules as reference image generator, pseudo-GT generator, and enhancement network. (Nguyen 23a) 66

4.12 Groundtruth data used by the RPN and box head modules. (Honda 20) . . 68

4.13 Data Loader from Detectron2. (Honda 20) 69

4.14 example of a dataset_dict returned through a data loader in Detectron2. (Honda 20) 70

4.15 Sample categories file defined for training PanopticFPN using Detectron2. 70

4.16 sample RGB image and its panoptic segmentation mask given as input during training of PanopticFPN model 71

4.17 Inferene pipeline of this panoptic segmentation. 71

4.18 Figure showing predicted and GT Panoptic segments of an image and IoU score of greater than 0.5 is considered Positive prediction and therefore matched.(Kirillov 19b) 73

5.1 Sample ISTD dataset images 76

5.2 Feature Maps visualization of MaskShadowGAN Generator module performed after convolution block, downsampling, residual block, upsampling block, output layer. 78

5.3 Training loss curves of ISTD Dataset on MaskShadowGANs Generator and discriminator 79

5.4 Test inference results of MaskShadowGAN model trained on ISTD dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model. 80

5.5 Image showing the PSENet inference on SICE dataset trained checkpoint. the left image is our input, the middle is the gamma mapping applied input image, and the rightmost is the inverted image on which gamma mapping is applied which is our output as described in (Nguyen 23b). 84

5.6 Image showing the PSENet training loss curves trend on SICE Dataset. The X-axis represents the iterations and the Y-axis represents the loss values. The top curve is the total loss curve. The bottom blue loss curve is the reconstruction loss and the purple color curve represents the variational loss trend as described in the section 4.3.2. 84

5.7 Image showing the Labelme polygon-based annotation of ground truth for panoptic segmentation on the RPTU-Forest dataset. 85

5.8 Sample input image visualization along with annotations used by the model for training. 86

5.9 Inference on the test set of RPTU-Forest dataset sample. 87

5.10 Training loss curve trend for Panoptic segmentation task. 87

5.11 Images showing the panoptic predictions on the images with and without shadows. 88

5.12 Figure showing Panoptic predictions without exposure correction on left and with exposure correction in right. 89

5.13 Figure showing Panoptic evaluation metrics trend curves. th represents things, st represents stuff classes. 89

5.14 Table showing the final losses modified during training of MaskShadowGAN network on RPTU-Forest panoptic dataset 90

5.15 Table showing the evaluation metrics of Panoptic Segmentation trained using PanopticFPN network architecture. St denotes the stuff class and th denotes the things classes in the above table. 91

5.16 Table showing the evaluation metrics of Panoptic Segmentation trained using EfficientPS network architecture on Cityscapes Dataset. St denotes the stuff class and th denotes the things classes in the above table. 91

List of Tables

2.1 Sample images of our Forest dataset captured using Sequoia Multispectral camera. 28

2.2 Segmentation visualizations in an AV, which can help scene understanding: (a) Instance segmentation where each instance is differentiated with varying color; (b) Semantic segmentation output in which each class is labeled separately.(Muhammad 22) 29

4.1 Urban datasets for Image segmentation samples for different datasets along with annotations representation. 52

4.2 Sample Images from Freiburg Forest dataset from (Valada 17). 52

4.3 RPTU-Forest dataset sample RGB images and their ground-truth annotation visualizations using labelme 54

5.1 Inference results of SPA-Former on random test images of the forest environment from the ISTD dataset. The top row shows original shadow images, and the bottom row shows the shadows removed image predictions by the model. 77

5.2 Freiburg forest test images inference results of Shadow-Former model trained on ISTD dataset. The top row shows original shadow images and the bottom row of images shows the shadows removed image predictions from the model. 77

5.3 Test inference results of MaskShadowGAN model trained on Multi dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model.. 81

5.4 Test inference results of MaskShadowGAN model trained on Freiburg-forest dataset for 200 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model. 82

5.5 Test inference results of MaskShadowGAN model trained on RPTU-forest dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model. 83

1. Introduction

From the invention of the wheel to the advanced use of technology in making vehicles run on their own, humans have achieved so much in the past 100 years. But autonomous vehicle concepts are not new trends either, as in the 1980 project such as DARPA[1] exists (Pomerleau 88). Integration of computer vision, Neural networks, sensors such as LIDAR, and control technologies like robot control are attempted (Pomerleau 88).Autonomous driving of a Chevrolet van that is named NAVLAB on the highways is achieved in ALVINN(Pomerleau 88).

1.1 Autonomous driving

As the name suggests, autonomous driving is the ability of the vehicle to drive on its own by sensing the surrounding environment with the aid of sensors mounted on it and avoiding human involvement during driving, making critical decisions. The vehicle can perform all the tasks similar to its traditional vehicle counterpart without an experienced driver involved(Synopsys 23). The Society of Automotive Engineers (SAE) uses the term automated driving instead of autonomous as the word resembles the capacity of the vehicle to make self-decisions and not necessarily follow the driver's command as instructed. In contrast, automated systems resemble the systems that accept input commands from the driver and execute actions as per the instructions given (Synopsys 23).

According to (Wienrich 23), SAE in its standard J3016 defined 6 levels of automation for driving in on-road situations, and the functions of the driver at various levels are as follows:

[1]Defense Advanced Research Projects Agency of the U.S. Department of Defense.

Level 0: No Driving Automation

It is the basic form of driving in which the driver has full control and should perform the braking, acceleration, and steering tasks irrespective of some automated control systems that support occasionally and without any further means of assistance systems involved.

Level 1: Driver Assistance

In this mode, the driver is backed up with certain assistance systems for his disposal to aid him whenever needed such as Lane assistance or Adaptive cruise control (ACC) and the driver still has full control of the vehicle and can override the assistance systems manually if needed.

Level 2: Partial Automation

This mode is an upgrade of level 1 where we can see the intelligent control systems combined into a single system to take over the essential needs of the driver at all times. However, full control and attention of the driver are needed at all times to monitor the vehicle's driving.

Level 3: Conditional Driving Automation

This level of automation enables the driving task to be temporarily overtaken by the driver completely and the driver can engage in other leisure activities of his/her own. There exists a warning time after which the driver has to take the driving role again until the system achieves a functional state again.

Level 4: High Driving Automation

vehicles can drive autonomously at this level without the need for a driver at all and can ride without any passengers inside them. An example of this level is that a vehicle can park on its own without any driver being present inside the vehicle.

Level 5: Partial Automation

The full level of automation in which there is essentially no need for any steering wheel or gas pedal or braking and the driver also becomes a passenger here in this level. (Wienrich 23)

The overall levels of driving automation (SAE-International 14) are depicted in Figure 1.1.

Advanced Driver assistance systems are actively deployed into passenger cars today with a combination of multiple sensors ranging from simple cameras to LiDAR(Light Detection and Ranging), RADAR(Radio Detection and ranging), and also various control systems like automatic parking systems, adaptive cruise control, lane detection modules, and

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	No Automation	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	Driver Assistance	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial Automation	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	Conditional Automation	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	High Automation	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	Full Automation	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

Figure 1.1: Driving levels of automation for on-road vehicles as per (SAE-International 14)

several other such systems which aim to increase the road safety eventually. Big technology giants like Google and Tesla have shown some major developments in autonomous driving vehicles recently. Tesla's new Autopilot (Tesla 17) has a total of 8 cameras and a powerful processing computer that enables it to visualize the surrounding environment up to 250 meters and a 360° view is analyzed all the time, which is a difficult task even for a human driver to achieve and it includes several other driver assistance features like ACC, lane departure warning, AutoSteer, Smart Summon (Tesla 17). Despite all these advancements in computer vision through the usage of several sensors, the perception of the environment remains a task at hand and needs an extensive amount of research in this area. Even though there are visible proofs of autonomous driving vehicles on roads these days, the off-road environment poses a significant challenge for autonomous driving due to the dynamic conditions involved which require a lot of active research.

1.1.1 Urban Environments

Urban environments are ideal for autonomous driving because of well-paved roads, and lane markings (Campbell 10). It is easy for computer vision systems to work on these cues, and with the help of multiple sensor inputs, a better understanding of the surrounding environment of the vehicle can be achieved. New advances enabled autonomous vehicles to sense local surroundings, detect objects, and classify them and can also reason the surroundings change over time and plan their motions by obeying the road rules (Campbell 10). The third round of the DARPA Grand Challenge (DGC) shifted its focus to urban driving where 6 vehicles were able to navigate autonomously and complete the 96km distance successfully using the route network definition file with known prior routes and using GPS locations of paths and detailed descriptions about the environment inside

that file (Campbell 10). To summarize, all teams divided the task into subsystems as shown in Figure 1.2 (Campbell 10).

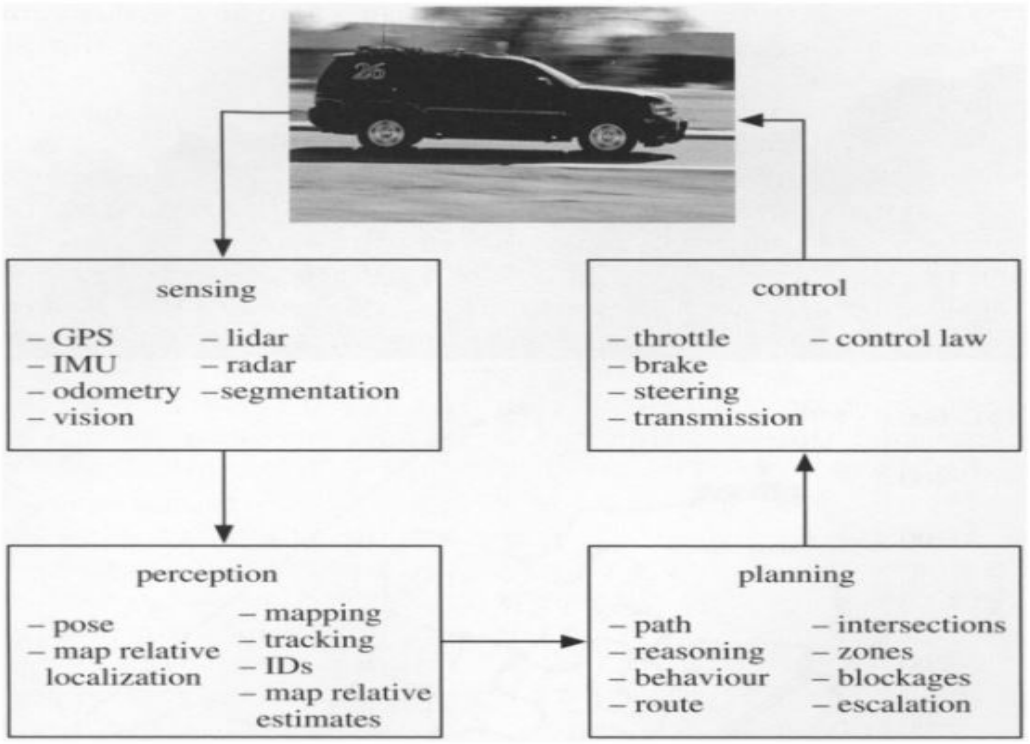


Figure 1.2: Systems present for achieving driving in urban environments(Campbell 10)

As the vehicles are employed with multiple sensors for navigation, a sensing module collects the data from these sensors which can be multiple in the case of autonomous driving and the perception module is essential for the understanding of the surrounding environment with inputs collected such as pose, map and the planning module is responsible for navigation purposes such as path planning, trajectory planning, behavior estimation of surrounding vehicles and the control module is the one that actuates several driving elements inside the vehicle for driving autonomously in the absence of a human driver to actuate (Campbell 10).

1.1.2 Off-road environments

Autonomous driving in an Urban environment has the main goal of transporting people and goods safely between two points, (Berns 11). Off-road autonomous navigation has diversified goals and is also difficult to achieve due to the challenging conditions posed by the environment (Berns 11). Perception of an off-road environment requires the fusion of inputs from different sensors such as cameras, LiDAR, RADAR, GPS(Global Positioning System), and INS(Inertial Navigation System) to estimate the uncertainty in off-road scenarios and is the main challenge for many researchers and engineers who are trying

to achieve this autonomous driving in off-road conditions (Berns 11). In the case of autonomous off-road driving, perception sensors are crucial sensors for bringing accurate estimation of surroundings (Stavens 12). These sensors focus on the detection of obstacles in the path (Stavens 12). In addition, we need to consider the terrain roughness because unlike on-road there are no paved roads and clear lanes for the vehicle to travel smoothly at any speeds (Stavens 12). For a robot to maneuver autonomously in off-road terrain, estimating terrain irregularities in front is essential as with proportion to speed, shock-induced in the vehicle increases (Stavens 12). The typical Off-road environments also pose the challenges of dense vegetation, uneven pathways, and sensor interference due to obstacles such as dust, tree branches, bushes, etc. The lack of clear pathways and the varying nature of the appearance of the environment in different seasons also need to be considered in off-road settings.

1.2 Environment Perception

Autonomous vehicles also need to have an eye on surrounding vehicles and obstacles on the road like curbs, intersections, pedestrians, and sensors are these eyes installed in the vehicles (Van Brummelen 18). Sensors in autonomous vehicles are broadly classified into two main categories: internal and external sensors mounted on the vehicle (Zhu 17). Internal sensors are the ones responsible for measuring the internal state of the ego vehicle such as oil pressure, velocity, engine, and fuel status of the vehicle (Zhu 17). External sensors are responsible for perceiving the surrounding conditions of vehicles such as road lanes, other vehicles approaching and pedestrians (Zhu 17). An overview of typical sensor positions mounted on autonomous vehicles is shown in Figure 1.3. A multitude of sensors are used in typical vehicles to understand the surroundings to a full extent.

Autonomous vehicles mainly rely on external sensors such as vision (Cameras), distance measurement sensors(radar, LiDAR), and ultrasonic sensors to perceive the environment and are considered as primary sensors that enable autonomous driving (Yeong 21). cameras are the typical visual sensors that can capture the surrounding environment's rich features of color and texture details (Yeong 21). They are the most cost-effective sensors available for perception tasks (Yeong 21). cameras are of different types such as monoculars, stereo, fisheye, pinhole, etc. Stereo cameras have the additional ability to estimate depth information by using the disparity between the two camera lenses present (Yeong 21). Fish eye cameras are the common sensors used these days due to their capability to provide a 360° view of the surroundings (Yeong 21).

LiDAR is an active sensor that is used to measure distances (Campbell 18). It is a time-of-flight (TOF) based active sensor, which functions by emitting modulated pulses of light, and the reflection time of these pulses is used to measure the distance (Campbell 18). There are a wide variety of long-range LiDARs these days, that can measure distances

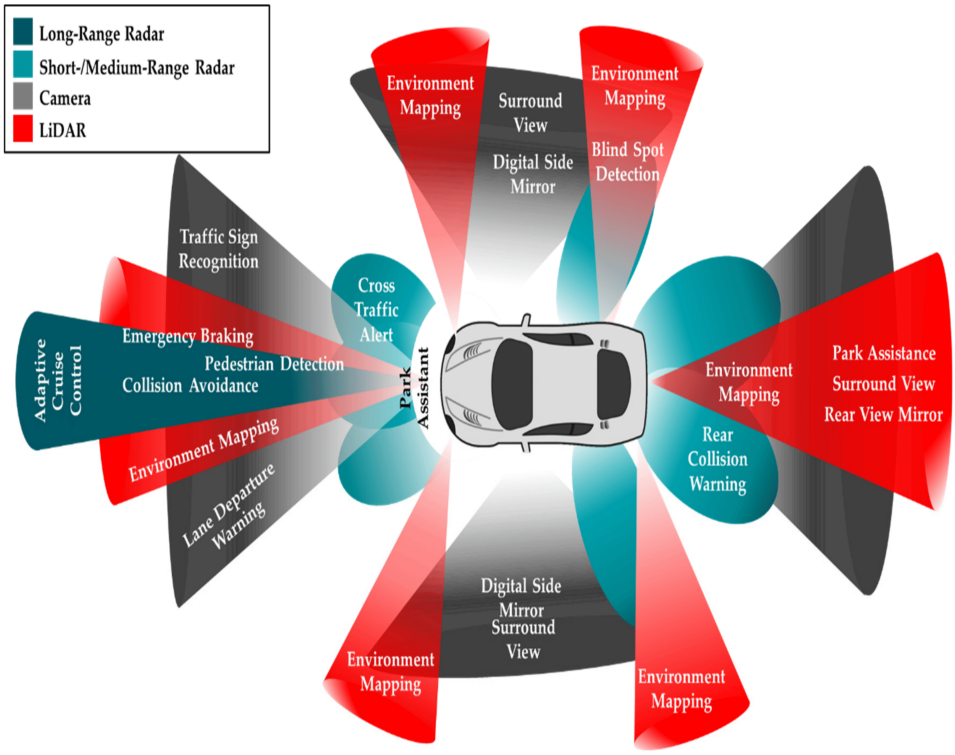


Figure 1.3: Overview of various sensors mounted along with their positions for 360° environment perception. (Wendt 18)

up to 250m (Campbell 18). LiDAR generates a 3D representation of the surrounding environment scene structure in the form of point clouds (Campbell 18). Radar is another typical active sensor used for the detection of distances of objects around the vehicle (Shahian Jahromi 19). It works by emitting electromagnetic waves and measures distances by implementing the Doppler effect (Shahian Jahromi 19). These sensors typically need a huge computational head for the processing of sensor data and in more cases, their fusion is required to efficiently implement the perception of the environment.

1.2.1 Challenges

Despite rapid advancements, sensor technologies still need a lot of improvement and are facing three major challenges: perception problems due to bad weather, perception problems due to challenging lightning, and human perception problems in terms of automotive sensors (Van Brummelen 18). In the case of poor weather like rain, or snow the road markings are obscured by snow, dust, or sometimes worn off (Van Brummelen 18). It poses difficulty for the vision-based systems which solely work by detecting them and navigating the vehicle (Van Brummelen 18). LiDAR also faces difficulty in the case of dense snow known as phantom obstacles, which obstruct the reflection of the laser (Van Brummelen 18). During rain, the splashes of water are often misinterpreted as object detections (Van Brummelen 18). For camera systems, fog covers the camera views

(Van Brummelen 18). Radar works irrespective of weather conditions but relying the object detection purely on radar measurements is not an ideal choice as it has limitations in the detection of markings and classifications (Van Brummelen 18). A potential solution to this problem is the only advancement in biological vision algorithms that mimic human vision as we humans are capable of driving vehicles using only our eyes (Van Brummelen 18).

Another major problem for perception sensors is the varying lighting conditions either getting confused with lens flares, tail light reflections, or mistaking shadows as object detections also one possible solution to tackle the night mode of driving is to use thermal imaging cameras for low-light conditions (Van Brummelen 18). Navigation can be carried out in some vehicles by using the prior information of key points in the predefined maps and previously explored locations by the vehicle (Van Brummelen 18). In this way, the varying lighting conditions faced during navigation are avoided (Van Brummelen 18). However, this is not an optimal approach due to the environment's dynamic nature inside forests for off-road scenarios (Van Brummelen 18). When LiDAR is used, this problem can be avoided as it uses no external light for functioning (Van Brummelen 18). However, the obstacles present in the forest environment like bushes, dense vegetation, and dust may create interference problems for the sensor (Van Brummelen 18). Thus, a fusion of multi-modalities of sensors can be helpful to overcome these varying lighting conditions (Van Brummelen 18).

The major problem that needs to be addressed to see autonomous vehicles on-road is the safety of passengers and the robustness of the sensor systems that are being deployed into the vehicle (Van Brummelen 18). Public awareness of these sensor systems and the rare misfortunes faced in such autonomous vehicles recently made the public more doubtful (Van Brummelen 18). These need to be addressed properly and there is an evident improvement needed in sensors and more sophisticated algorithms are yet to be developed to tackle this challenge (Van Brummelen 18).

1.3 Motivation

Urban automated driving can be seen up to level 2/3 today. But in off-roads where the conditions change rapidly from scene to scene the task of achieving autonomous driving is difficult. Several traffic scene-related driving datasets are publicly available to train the deep learning models for computer vision tasks to perform scene segmentation which helps the vehicles in scene parsing and making decisions of navigation accordingly. But for Off-road environment driving, there are very few datasets available in comparison to the Urban counterpart and the ground truth annotations are available in a limited quantity for these datasets to train the network in an end-to-end manner. Thus we intend to develop a model that can perform Panoptic segmentation of scenes in an off-road environment such as a forest to enable our Unimog to drive through the forest autonomously.

1.4 Task

As discussed above, there is very little work being carried out to perform Panoptic Segmentation in off-road environments due to a lack of datasets. Without these datasets, it is nearly impossible to train any model to learn the features present in these scenes automatically. The challenges posed by the forest environments make it difficult for any model to learn the features of objects inside the forest. Because of diversified terrains, it is very challenging for the cameras to stay stable while capturing the scene information. Forest environments are covered in dense vegetation, obstacles in scene capture such as tree branches, and due to this limited visibility, it is often difficult to make decisions for the perception systems. For navigation planning, the perception systems should act more cleverly than in the case of the urban environment as there exist no paved pathways inside forests. The harsh environment inside forests can hinder the sensor's performance due to inference from the dust, vegetation, and uneven surfaces for active sensors like LiDAR and radar. Dynamic behavior of weather should also be adaptable by the perception systems. When the camera is the only sensor employed for environment perception, it should be more reliable in tackling the above-mentioned difficulties posed by the environment.

Thus we need to find a solution to mitigate these difficulties and train a lightweight model designed to perform the panoptic segmentation in off-road environments. Also, it should be easier to deploy this model onto real-time vehicles and thus we can use it in the vehicles. In this thesis, the above-mentioned problems are tackled and the results are presented.

1.5 Contribution

The contributions from this thesis are as follows:

- Using a Multi-Spectral camera Sequoia, a dataset is collected for a specific forest environment named as RPTU-Forest Panoptic dataset, which captures the scenes with variations in scenes.
- The whole dataset is annotated manually by deciding the classes inspired by the existing Forest type dataset named the Freiburg-Forest dataset.
- A panoptic segmentation model of the PanopticFPN network is trained using this forest dataset and inference is performed.
- Shadow Removal techniques are investigated to optimize the panoptic model performance.
- Image exposure correction techniques are experimented with again to provide performance enhancement for the panoptic model.

- The final-trained models are combined into a single pipeline to get the final predictions directly from the input image rather than executing multiple networks.
- Evaluation of training and testing times are measured on the Panoptic segmentation model.

2. Background

2.1 Artificial Intelligence

As per (GoogleAI 23), Artificial Intelligence(AI) is a branch of science that focuses on enabling machines to mimic human intelligence, with applications across various domains such as computer science, data analytics, medicine, and many other engineering domains. The growing attention towards AI, these days leads to its usage to enhance productivity, reduce human efforts in performing repetitive nature jobs, reduce human errors, and increase safety with continuous operation. AI is majorly classified into two categories: weak AI and strong AI. Weak AI is where machines perform operations based on the instructions fed and prior training performed. Strong AI is where the machines can independently think on a human level and make decisions to perform the tasks assigned to them. Strong AI is not yet in use due to legislation conflicts, however, the demand for weak AI is growing rapidly. (GoogleAI 23).

But how do machines achieve this intelligence? There are two various approaches to mimic the human level of thinking for computers and those are Machine learning and a sub-field of machine learning i.e. Deep learning which are discussed below.

2.1.1 Machine Learning

Machine learning(ML) is one of the subdivisions of Artificial Intelligence that uses algorithms, in general, to read and analyze the data fed to them and then extract the logical pattern from the data and make predictions on the new unseen data without the essence of explicit coding to achieve the task (NVIDIA 23). The coding task to solve the problem is replaced with simply feeding the input data for the algorithm through which algorithms figure out the hidden logic inside the data and make predictions on its own (Geitgey 14). consider an example of a classification algorithm that can classify the data into different

classes based on the types available and we can reuse the same algorithm for the task of classification of emails into spam or not-spam without any changes to the code as depicted in the figure 2.1 below and ML has multiple such generic algorithms defined in it (Geitgey 14).

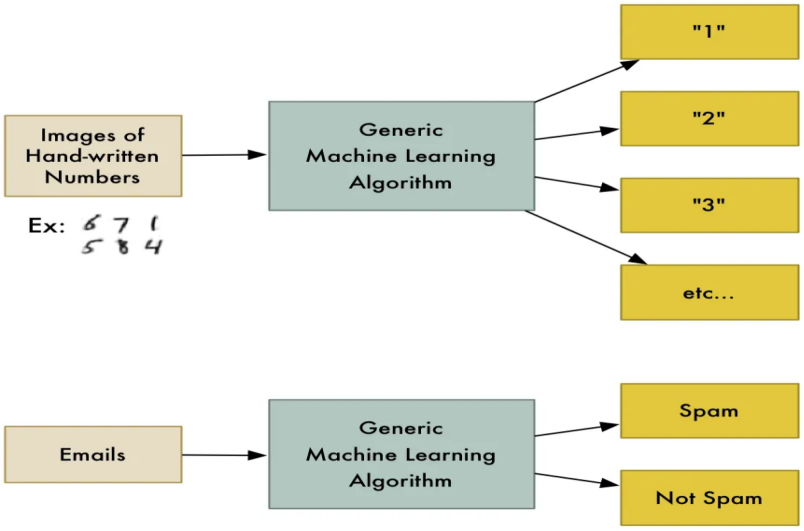


Figure 2.1: Machine learning generic classification Algorithm. (Geitgey 14)

Ml algorithms are mainly of two categories: Supervised and unsupervised learning and the difference between the two types is the supervision that is taken into account when extracting the hidden patterns inside the data (Raicea 17). In supervised learning, the supervision is provided as labels or target outputs that are fed along with the input samples and the algorithm predicts iteratively until it predicts the output close to the given target output of the input data (Raicea 17). Supervised learning generally follows the mapping of Input data(X) to the output label(Y) with the help of mapping function 'f' as described in the equation 2.1 where the mapping is done through analysis of hidden patterns inside the input data we feed to the algorithm.

$$Y = f(X) \tag{2.1}$$

In general, Supervised learning algorithms are divided into Classification and Regression problems (Terra 23). Classification involves the mapping of input to be done into discrete output variables e.g. classification of spam/no-spam messages, and regression problems address the mapping of input into continuous output variables such as market trends, price predictions (Terra 23) and the difference between them is depicted in the figure 2.2.

On the contrary, Unsupervised learning doesn't associate with the target labels fed to the algorithm for prediction, and it is the task of the algorithm to predict the hidden

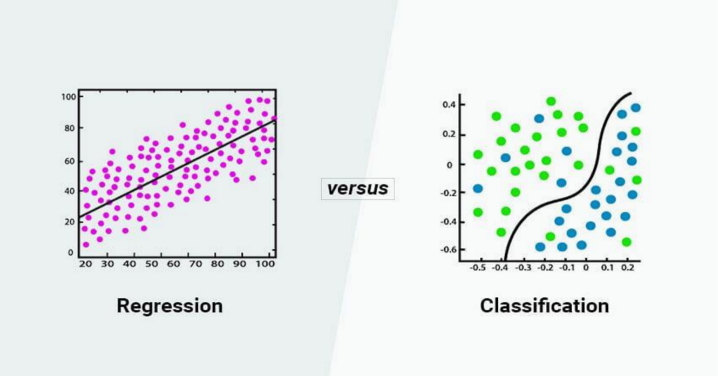


Figure 2.2: Machine learning generic classification Algorithm. (Terra 23)

relation among input data samples and group them based on the common features it detects (Raicea 17).

2.1.2 Deep Learning

Deep learning is another approach for achieving artificial intelligence and is a sub-module of machine learning that achieves the prediction of output from the given input using neurons as a medium which are inspired by the human neural systems and thus utilize the neural networks instead of algorithmic approach (Raicea 17). There exist essentially three main layers: the input layer, hidden layer, and output layer (Raicea 17). The input layer is where the input is fed to the network, the hidden layer is the main layer in which the mapping is done and where the main computation is held, and the output layer simply predicts the output of this network and the word Deep comes from the number of hidden layers present in the network (Raicea 17). The main advantage of these neural networks is that they can adapt to the changes to the input data and thus change in input doesn't translate directly to the change in output (Chen 23). The simplest form of a neural network functioning and elements present is shown in figure 2.3.

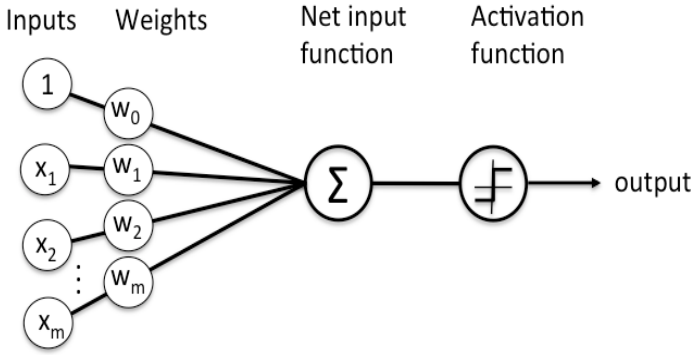


Figure 2.3: Simple representation of a neural network elements. (Pathmind 23)

Each element in neural network layers is referred to as a node which is essentially a computational unit that performs computation of multiplication of input data with weights or correspondences that can either amplify or dampen the significance of input in predicting the output (Pathmind 23). This product is passed through the activation function which decides whether this node output is significant to predict output or we can ignore this node output (Pathmind 23). Thus finally in the output layer, we can get the desired output that is mapped from the input, and based on the defined loss function, the weights are updated until the proper mapping is done (Pathmind 23). Deep refers to the depth or simply number of hidden layers present between input and output and generally if the number of hidden layers is more than three, then the network is called deep neural network (Pathmind 23). with increasing depth of the network, more complex features are learned by the network because of a combination of features learned from the previous layers and this capability enables the neural networks to handle any complex data structures including high dimensional data or Images, text, audio as well with the help of feature extraction automatically without external need as in the case of machine learning (Pathmind 23).

This thesis's most important category of neural networks is the convolutional neural networks (CNN), which are prominently useful in dealing with image data as input. Due to Fully connected layers, artificial neural networks would not scale well to large images (Karpathy 18). Due to parameter-sharing techniques, CNNs have proven to be the best solutions for image data classification tasks (Karpathy 18). CNNs are also invariant to translation i.e. small orientation changes in the input don't affect the output predictions and this feature makes the CNNs useful in dealing with images. In the place of weights in normal neural networks, we use filters which are essentially another version of weights and we lay these filters on top of the input image and slide the window of weights onto overall image locations. This convolution operation is depicted in figure 2.4, where weights are shared across receptive fields. Each filter has the same depth as input volume channels and output volume has the same depth as the total number of filters.

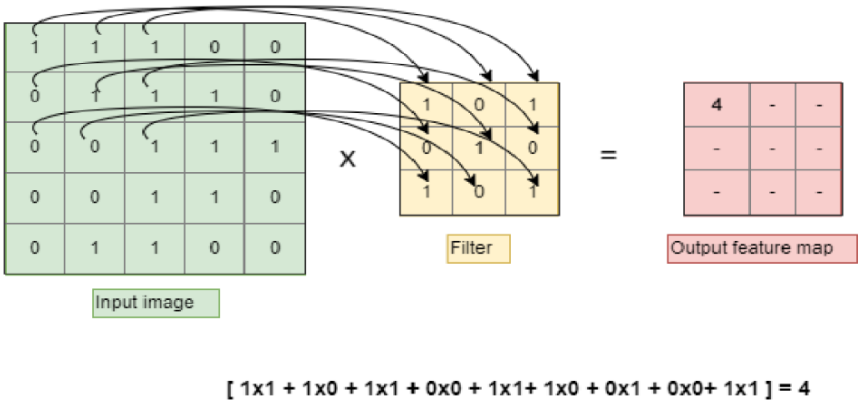


Figure 2.4: Figure showing convolution Operation.

2.2 Environment Segmentation

Segmenting different parts of an image for a better understanding of the scene is an essential concept in Computer Vision(CV). In CV, this image segmentation enables the machines to segment the available objects in the scene into individual instances, and thus each pixel is assigned to a class which helps us to easily analyze the scene we are dealing with (Barla 22). This task involves essentially classification, detection, and assigning labels to pixels that belong to the same objects in the scene (Barla 22). Image segmentation is one of the areas of CV in which small regions of the image are grouped into their respective class label and it is an extension of Image classification which performs localization of objects present inside the image as well as outlines the object pixels inside an image (Bandyopadhyay 21). Traditional CV Image classification tasks generally require only an encoder network, however, for segmentation tasks, we require both an encoder and a decoder architecture (Bandyopadhyay 21). The encoder block is responsible for the representation of input latent space into a feature vector and the decoder decodes this information and forms segmentation maps that describe the locations of objects inside the image along with their outlines (Bandyopadhyay 21).

Image segmentation tasks are mainly classified into three major categories: Semantic, Instance, and Panoptic segmentation which will be described in detail below (Bandyopadhyay 21).

2.2.1 Semantic segmentation

Semantic Segmentation is defined as the task of classifying each pixel present in the image into a class label and during this classification, other information and context are often not considered (Bandyopadhyay 21). This task usually follows three steps such as classifying an object in the image, finding the object, and drawing a bounding box around it through localization and clustering the pixels in the localized image by a segmentation mask (Barla 21). The task is simply a classification of regions of the image into classes and segmenting those regions separately from the rest of the regions with the help of overlaying segmentation mask (Barla 21). For achieving semantic segmentation, the first step is to extract the features from the input image to derive meaningful correlations, and in CV, CNN performs this task and is commonly used (Barla 21).

A fully Convolutional Network(FCN) is the simplest form of a CNN that can be altered to achieve semantic segmentation (Barla 21). Initial layers of this network can be used for feature extractions and final prediction layers are altered to achieve segmentation by replacing the fully connected layers with deconvolutional layers to up-sample the image to its original size which is compressed during the feature extraction stage (Barla 21). This feature extraction by downsampling is generally referred to as an encoder and CNN that is used for upsampling is referred to as a decoder and the visual representation looks like

below in figure 2.5 (Barla 21). There is some information loss during the final upsampling due to 1x1 convolution and usage of this convolution leads to loss of spatial information in the decoder (Barla 21). This causes problems for final upsampling using the little data and for this reason, there are proposals such as FCN8, and FCN16 where they use information from previous pooling layers for upsampling (Barla 21).

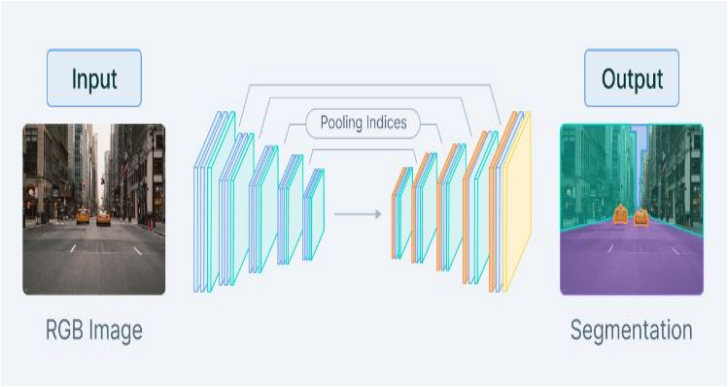


Figure 2.5: Convolution Encoder-Decoder structure.(Barla 21)

U-net is another commonly used approach for semantic segmentation which is a slight modification to FCN to tackle the information loss that arises due to loss in spatial resolution of the input with convolution and pooling operations (Barla 21). This loss of spatial context can be addressed using skip connections inside the encoder-decoder structure in U-net (Barla 21). Encoder block downsamples the input image resolution gradually and represents the latent space in a feature format (Barla 21). The Decoder block uses this compressed feature vector and starts upsampling this representation back to the input image size by utilizing a series of deconvolution operations (Barla 21). FCN normally uses final extracted features for upsampling whereas this U-net utilizes the skip connections which are designed to address the spatial information loss and are used for upsampling purposes (Barla 21). This architecture of U-net enables the concatenation of high-level features and low-level features using skip connections as shown in figure 2.6 and this more spatial context is retained for the model to deliver finer results (Barla 21).

Deeplab (Chen 17a) is the modern approach developed by Google that uses CNN for feature extraction and unlike U-net which concatenates the convolutional layer output with its counterpart deconvolutional layer output, Deeplab uses features from the last convolutional block before upsampling (Barla 21). Instead of deconvolutional blocks for upsampling, the deep lab uses atrous convolutions (Barla 21). Atrous convolution is also known as dilated convolution (Chen 17a). The spacing between the kernel weights in the filter enables enlarging the field of view of convolutional filters so that additional context can be learned (Chen 17a). Atrous convolution overcomes the problems of Deep Convolutional Neural Networks with loss in the image feature resolution due to multiple subsequent convolutions and pooling operations involved (Chen 17a). Due to this loss of

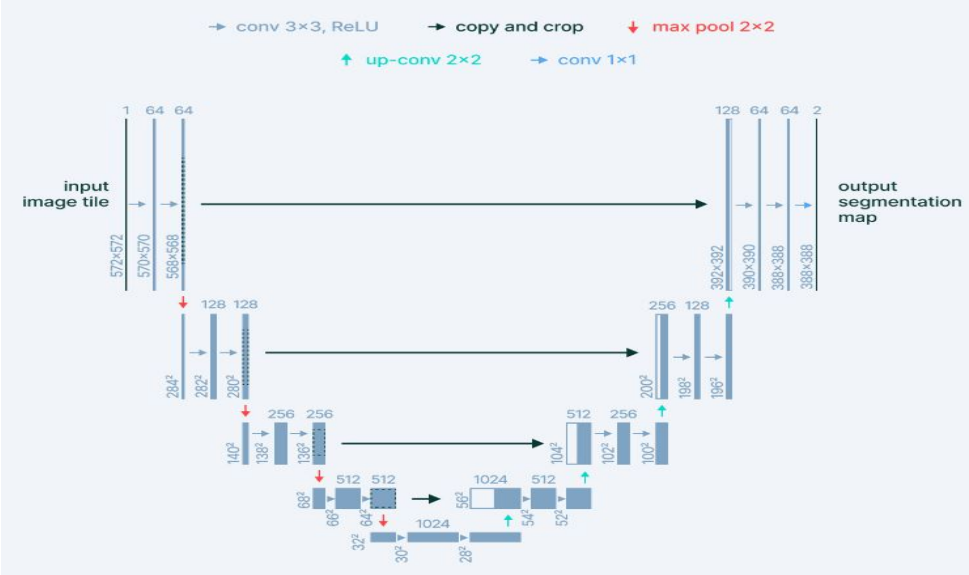


Figure 2.6: U-Net structure.(Barla 21)

input image resolution, the model can only learn the abstract representation of the image and is unable to deliver the dense predictions (Chen 17a). Atrous convolution addresses this loss of spatial information by adjusting the distance between the kernel weights, a large receptive field is achieved and thus more spatial context is retained (Chen 17a). Due to not learning any additional parameters, it is computationally fast enough and the model trained using this convolution can predict dense representations (Chen 17a).

2.2.2 Loss functions for semantic segmentation

Loss functions are essential for any neural network to optimize the predictions we get from the network and for semantic segmentation there are some main loss functions we can use based on the fact that semantic segmentation is nothing more than the classification task (Barla 21).

Pixel-wise Softmax with cross entropy is the mostly used loss for the task of semantic segmentation and it is calculated as the log loss where the comparison of each pixel between output and ground truth and summation is done over all the classes available and is shown below in the equation 2.2 (Barla 21).

$$-\sum_{i} y_i \cdot \log(p_i) \tag{2.2}$$

If there exists any class imbalance in the dataset used, then the cross-entropy loss will fail in certain circumstances, as the negative likelihood estimation tends to shift in favor of the majority class present in the dataset, and thus the loss calculation is biased (Barla 21). It is advisable to use Focal loss which is just a modification to the entropy loss by adding

some new terms as 1-pt where pt is nothing but the example we are considering and a gamma term to reduce the overall loss function by weighing the contribution of easy examples and hard examples for estimating the loss and is defined as below in equation 2.3 (Barla 21).

$$\text{Focal Loss}(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \tag{2.3}$$

For estimation of overlapping between pixels of different classes between predictions and ground truth, we use the Dice loss which utilizes the dice coefficient that ranges from 0-1 where 1 resembles perfect overlap between pixels, and the loss term is denoted as below in the equation 2.5 (Barla 21).

$$\text{Dice Loss}(p, g) = \frac{2 \cdot |p \cap g|}{|p| + |g|} \tag{2.4}$$

Thus semantic segmentation with its capability has varied application fields such as in Self-driving cars, medical image analysis, Aerial image processing, and more (Barla 21).

2.2.3 Instance Segmentation

Semantic segmentation only assigns class labels to the pixels belonging to the same category but it can't discriminate the individual instances of the same class and this is done exactly by the instance segmentation. For better scene understanding, counting the number of objects belonging to a category is a piece of useful information. Instance segmentation is the technique used for identifying instances that belong to the same class and differentiate the boundaries of objects from one another so that each object can be identified separately (Bandyopadhyay 22). Instance segmentation involves the steps of detection of objects, segmenting them from each other object and classifying the classes to which they belong and compared to semantic segmentation and object detection (identifying all the instances of a particular class present inside the image and drawing a bounding box around objects), instance segmentation produces better output (Bandyopadhyay 22).

Object detection identifies the location of any object inside the image using a bounding box, but it does not deliver any additional information about the object (Bandyopadhyay 22). Semantic segmentation works on a pixel-level segmentation, where each pixel in the image is assigned to a class label and it predicts the multiple objects belonging to the same class as a single class label (Bandyopadhyay 22). However, in instance segmentation, each object belonging to a similar category is distinguished separately and thus this new distinction brings more context to the scene understanding (Bandyopadhyay 22).

There are two main approaches to implementing instance segmentation namely FCN-based and R-CNN-based and we already know that FCNs are the most commonly used

method for semantic segmentation (Bandyopadhyay 22). However, we can modify the semantic segmentation to work on the region level for instance segmentation to work (Bandyopadhyay 22). Despite being translational invariant in nature, convolutional neural networks cannot be used for Instance segmentation (Bandyopadhyay 22). CNNs are unable to detect and segment individual objects in the image and notably, for the same pixel within an image, we will get uniform responses leading to identical classification scores, irrespective of the relative pixel location in the image (Bandyopadhyay 22). Semantic segmentation usually brings the semantic context to the pixels and the same pixels can have different semantic context based on their relative location in the image (Bandyopadhyay 22). However, this cannot be achieved using only a single Fully connected Network (FCN) on a single image (Bandyopadhyay 22). In traditional FCNs, the classifier is trained to predict the pixel likelihood probability of belonging to a particular class (Bandyopadhyay 22). To achieve the translation invariance to this FCN result, we use k2 position-sensitive score maps which tell us the score of the likelihood of pixel belonging to a particular object in the scene at a relative location (Bandyopadhyay 22). Pixel can belong to an object either in the foreground or in the background based on the semantic information about the pixel class and thus we can combine both object detection and semantic segmentation (Bandyopadhyay 22).

R-CNN-based approaches especially Mask R-CNN, one of the state-of-the-art (SOTA) approaches that are commonly being used for implementing instance segmentation (Bandyopadhyay 22). Mask R-CNN has three outputs such as class labels, bounding boxes, and object masks which bring more information related to spatial structures of objects which we lose by the means of the other two outputs (Bandyopadhyay 22). Unlike methods like Faster R-CNN which uses region of interest pooling(ROIPool) operation which might trigger quantization alignment issues between features and regions of interest, Mask R-CNN introduced something called ROIAlign which avoids this mismatch of feature alignment by using exact values from interpolation techniques like bilinear interpolation (Bandyopadhyay 22).

For feature extraction in the backbone of architecture, both R-CNN and FCN use the most prominent ResNet-50, which extracts the ROI features that are fed into a feature pyramid Network (FPN) which builds feature pyramid at different scales using lateral connections (Bandyopadhyay 22). Anchor boxes are assigned to these features at each level of FPN and these anchors are of multiple resolutions at each level of the pyramid (Bandyopadhyay 22). Labels are assigned during training to the respective anchor boxes based on Intersection over union(IoU) and depending on a particular threshold, redundant bounding boxes are removed if they are less than this threshold using non-max suppression, and a higher IoU bounding box is assigned with the label (Bandyopadhyay 22). The overall architecture of Mask R-CNN is depicted below in figure 2.7.

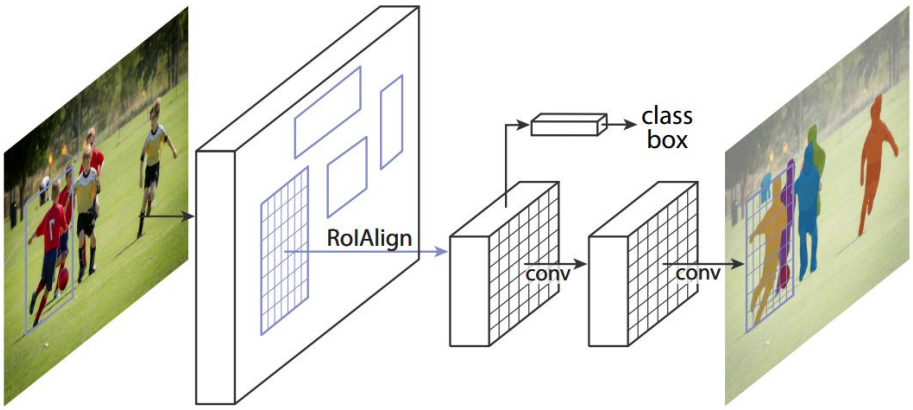


Figure 2.7: Mask R-CNN framework.(He 17)

Panoptic Segmentation is an advanced version of the image segmentation task where it is a combination of semantic segmentation of classification of individual pixels and assigning the class labels as well as instance segmentation of identifying the total number of instances present in the image (Barla 22). This task is explained in detail in section 3.1 with all the network architectures explained and also the metrics used for evaluation.

we will now see the evaluation metrics used in general for Instance segmentation.

2.2.4 Evaluation of Instance Segmentation

For evaluation of our model predictions, for instance, segmentation results, the commonly followed metrics are Average precision, Mean average precision, and Intersection over Union ratios and let us define the precision as simply positive prediction ratio or in other words, of all the predictions made by the model are positive out of them how many are actually positive as per ground truth samples and it is denoted by the equation 2.5 (Liu 23). Recall is another important factor that measures the positive percentages in total predicted positives and is denoted by the equation 2.6 (Liu 23).

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2.5}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2.6}$$

Between these two metrics, there should be always a trade-off as increasing one metric will automatically lead to a decrease in other metric values (Liu 23). When we plot both metrics, the resultant curve is called a PR-curve, and the area under it is called an average precision(AP) (Liu 23). To obtain multiple points to plot this curve, we need multiple precision and Recall values from the model predictions and ground truth (Liu 23).

Selecting a cut-off score for the classification task involves setting the IoU ratios to different values and the corresponding PR values are plotted on the graph (Liu 23). After multiple ratios, we will finally have a PR-curve which should in principle decrease monotonically as shown below in figure 2.8 (Liu 23). Mean Average precision is simply the average of AP values over all the possible classes available in the dataset (Liu 23).

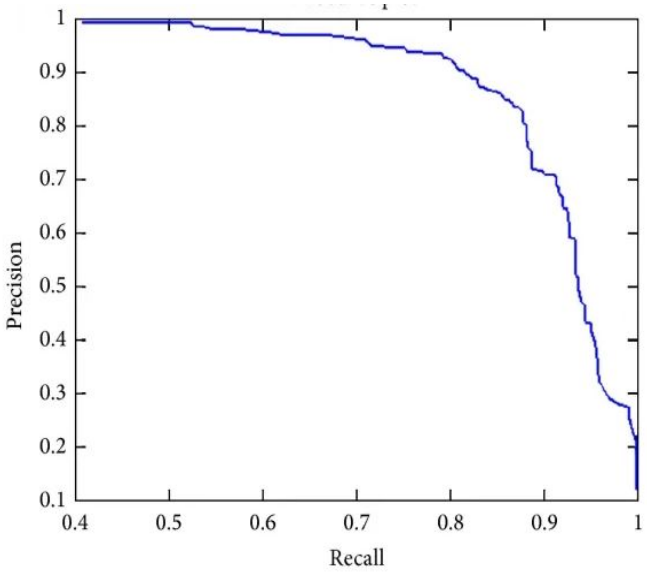


Figure 2.8: Simple PR curve.(Liu 23)

2.3 Sensors for environment perception

For the task of perception in AVs, sensors are the vital elements. They act as the eyes for the vehicle and estimate the surrounding conditions of the vehicle. They help the vehicle to know about the surrounding road conditions, other vehicle movements, traffic signal information, road lane markings present, and all other important surrounding information needed for the vehicle to implement decisions concerning navigation. These sensors are of different kinds as simple visual sensors like cameras and distance estimation sensors like LiDAR, RADAR, and multi-spectral cameras which have found applications these days and we will discuss each type further more in detail below.

2.3.1 Cameras including Stereo cameras

The cost-effective sensor in autonomous vehicles is the camera sensor which can be easily installed in multiple locations of the vehicle and is a primary sensor for major driver assistance systems such as parking, lane departure warning, blind spot detection, and curb detection (Wang 21). cameras can be either a simple monocular or binocular version which are similar to our human eyes and have the added advantage of estimating the depth of field based on disparity effectively (Wang 21). The most commonly used camera systems

in autonomous driving are the Monocular RGB cameras, fish eye cameras with a wide field of view, and time of flight (TOF) cameras for depth estimation, and based on the application each of these types brings forward an added advantage in respective scenarios.

Stereo cameras are vision sensors that use the disparity between the images formed on two single camera modules placed at a distance and based on this disparity, the distance from the object to the camera center is calculated accurately (Zaarane 20). The estimated distance is crucial as it can bring information about surrounding vehicles' distance measured relative to the ego vehicle. This is valuable information for perception, and also depth estimation from the surrounding scene enables the autonomous functionality of the vehicle.

Relying on the camera module alone for navigation decisions is not an ideal choice as the cameras still have limitations regarding the illumination of the scene. The camera is sensitive to external weather such as rain, snow, and dust. The appearance of the scene varies drastically in off-road settings and this is a major concern when using the camera alone for perception. The visual cues vary seasonally and the camera could not adapt automatically to these changes in recognizing the environment.

The processing pipeline from the camera would look like in figure 2.9.

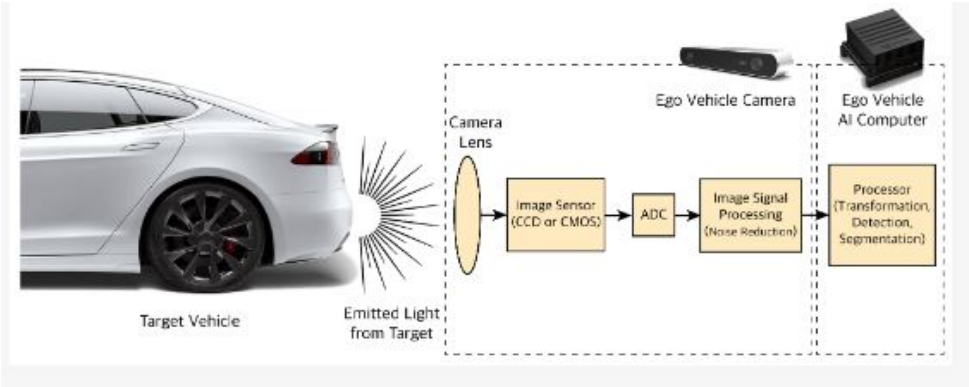


Figure 2.9: Camera Image Processing pipeline.(Shahian Jahromi 19)

2.3.2 Multi-spectral Cameras

For our use case of an off-road environment and to collect the dataset needed to train our panoptic segmentation networks, we used the multi-spectral camera which in addition to recording images, can record other spectra of data like Near Infrared(NIR), Red spectrum band, Green spectrum, Red Edge spectrum (Sequoia 21). Multispectral cameras are regularly used for sensing applications concerning outdoor activities such as farming or mapping an area (Sequoia 21). Figure 2.10 and table 2.1 illustrates the camera itself with its configuration of lenses arranged to capture different spectrum bands and RGB images and also the different samples of images collected using this camera for our dataset (Sequoia 21).

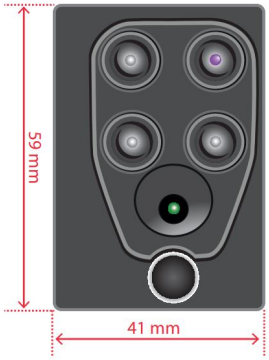


Figure 2.10: MultiSpectral camera.(Sequoia 21)

One important technical specification to be noted is that each lens has a different resolution and RGB image has 16 Megapixel resolution whereas other spectrum images have only 1.2 Megapixel (Sequoia 21). Different modes have different radial distortions in the output they capture (Sequoia 21). we can also linearly combine the output of two lenses and create NDVI-type images which have applications of better visualization of vegetation captured using this camera (Sequoia 21).

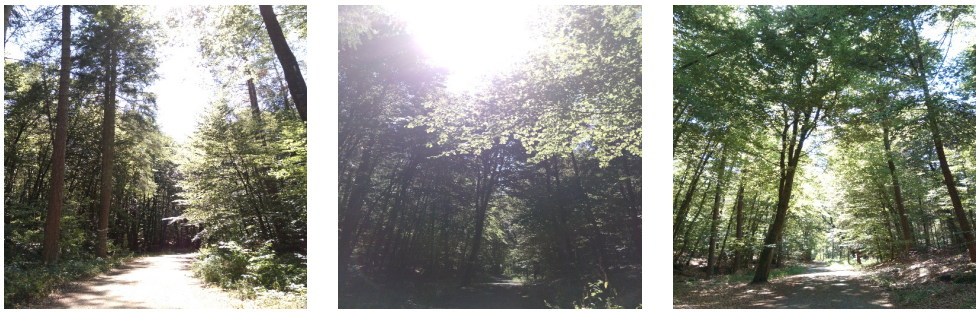


Table 2.1: Sample images of our Forest dataset captured using Sequoia Multispectral camera.

2.3.3 LiDAR

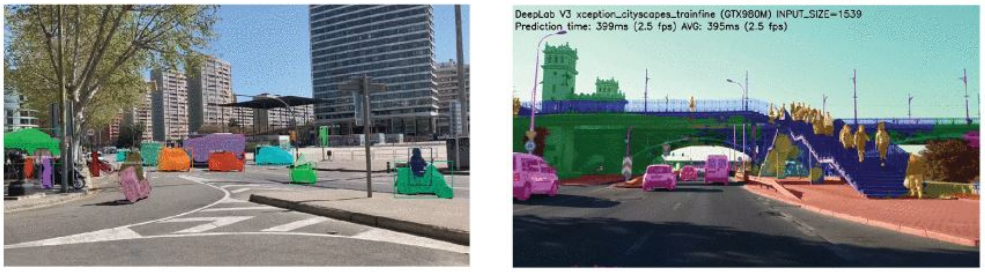
As per (Wang 21), LiDAR is an active sensor that works by transmitting pulses of modulated light and is based on the time of flight concept, in which the time taken by the waves emitted to reflect from an obstacle surface and reach back to the source. These are the most effective solutions so far available in the market to solve the problem of autonomous driving due to fast detection of distances, and longer distance measurements and are the highest accurate among all other sensor solutions but when the weather conditions change, they might be challenging scenario for the LiDAR due to change in point cloud data being modified due to the noises generated by external disturbances and in such cases LiDAR have the application disadvantage and also these are quite expensive to install and use in everyday vehicles.(Wang 21)

2.4 Approaches for environment segmentation

Segmenting the important features from the sensor data and taking actions automatically as per the perceived information is crucial for an autonomous vehicle and this scene parsing can be done either using only vision sensors like cameras or only using point cloud data collected by the LiDAR or sometimes even the combination of multiple sensors for better predictions and we will look into these approaches below in detail.

2.4.1 Segmentation using only images

Using only images, segmentation is carried out by collecting the images from the camera and extracting the features from those captured images such as the distance of detected objects from the camera, the shape of the objects, sizes, and also their relative movement concerning the ego vehicle (Muhammad 22). This information is then used to form a complete surrounding scene of the vehicle (Muhammad 22). This can help us to automate the actions to take such as braking, and steering, and make clear differences between the detected pedestrians, cars, buildings, and traffic conditions, and are a part of the decision-making functionality of an Autonomous Vehicle (AV), illustrated in detail in table 2.2 below (Muhammad 22).



(a) Instance segmentation in AV (b) Semantic segmentation in AV

Table 2.2: Segmentation visualizations in an AV, which can help scene understanding: (a) Instance segmentation where each instance is differentiated with varying color; (b) Semantic segmentation output in which each class is labeled separately.(Muhammad 22)

semantic segmentation and Instance segmentation can be easily implemented using modern Deep learning approaches like FCN, MASK R-CNN, and DEEPLAB architectures (Muhammad 22).

2.4.2 Segmentation using only Point cloud

Image segmentation and object detection can be performed only using the LiDAR point cloud data and one such approach is PointRCNN (Shi 19). The segmentation happens in two stages (Shi 19). In the first stage, semantic segmentation is performed on the point cloud data, in which points are segmented into foreground and background points (Shi 19).

In the second stage, proposals are generated based on these segmentations and further refined into canonical form for object detection tasks as shown in figure 2.11 (Shi 19).

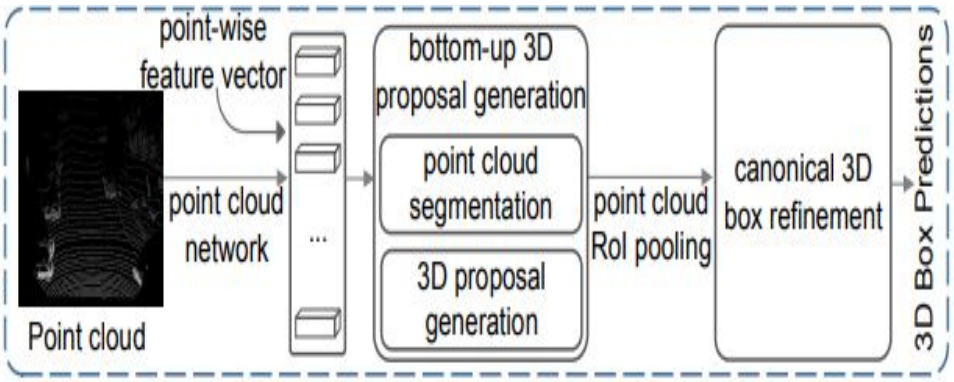


Figure 2.11: PointRCNN network structure.(Shi 19)

LiDAR approaches get confused to differentiate a pedestrian from a tall narrow pole structure (Fei 20). These false positive predictions make this approach more concerning in terms of safety (Fei 20). Thus, semantic segmentation using a camera module brings more context(Fei 20). This camera output is commonly fused with the LiDAR point cloud data for better results (Fei 20).

2.4.3 Sensor fusion approaches

Cameras, Radars, and LiDARs are the most commonly used sensor modalities in an Autonomous vehicle for perception tasks. Cameras being cost-efficient and easier to install in the vehicle, have limitations regarding illumination, dynamic weather changes, and environmental obstacles like dust, rain, snow, etc. LiDAR is also sensitive to external weather conditions and cannot deliver optimal performance in such conditions. Radar is however unaffected by such external factors but its output is limited to a few meters. It is therefore not an efficient choice to rely only on radar measurements to navigate the vehicle. To overcome these individual limitations, there is a solution to fuse the useful results from multiple sensors known as sensor fusion. The Most Commonly visible fusion is where we use cameras for extracting the rich semantic features and fuse the output of point cloud data with this to obtain the real-time capable scene parsing in the autonomous vehicle and we use a combination of camera, LiDAR, and FCN type architecture to perform the image segmentation and is known as early fusion approach which is depicted in the figure 2.12 below (Shahian Jahromi 19).

Late fusion of LiDAR point cloud and radar data is done to perform object detection (Shahian Jahromi 19). This data is further processed and refined to an object level (Shahian Jahromi 19). These outputs are then fed to state estimation filters like the Kalman filter, which then predicts the location of the object inside the input image

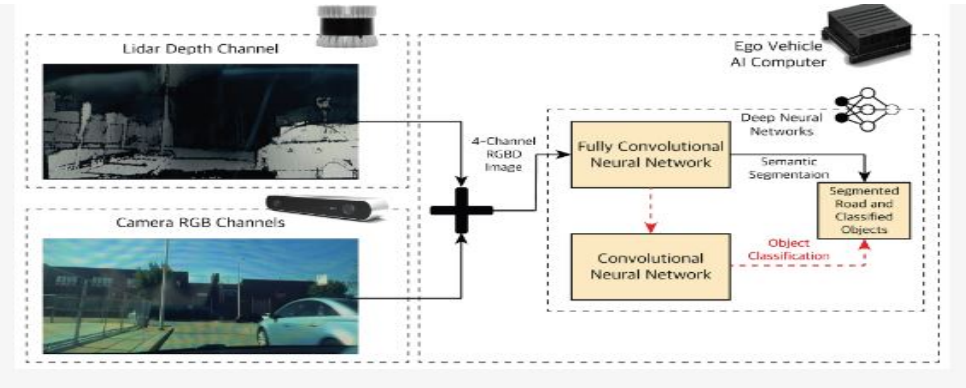


Figure 2.12: Early Fusion approach pipeline.(Shahian Jahromi 19)

(Shahian Jahromi 19). This object detection is further helpful with path planning and navigation modules to take appropriate decisions (Shahian Jahromi 19).

Thus we can fuse the outputs from different sensors to perform the optimal image segmentation for better scene understanding but this in turn leads to increased cost of the vehicle due to a heavy computation processor needed on board for the processing of this sensor data and also sensors such as LiDAR are not cheaper in price point of view.

In the next chapter, we will see the Panoptic Segmentation in detail and also several approaches available for panoptic segmentation, the main network architectures used for panoptic segmentation in urban and off-road environments, and the various challenges involved in particular environments.

3. Related works

The uprising interest for autonomous driving is increasing daily due to efficient processors available in the market which can in real-time perform the perception, decision-making, and path planning algorithms which are state of the art being developed due to advances in rapidly growing technology. Convolutional networks are developed on a large scale for the sole purpose of scene parsing for the vehicle and in contrast to the traditional computer vision approaches like using the bundle of classifiers and detectors used for one task, modern deep learning approaches implement the multiple tasks needed for instance feature extraction of multiple features using a single network which is computationally cost-effective and can be used on onboard computers directly and made camera the invaluable asset to the sensor module of the vehicle.

This chapter is organized into four sections. In section 3.1, a brief explanation of the task Panoptic Segmentation is explained, and section 3.2 gives us the insights into most popular available panoptic segmentation approaches, and section 3.4 reviews the best possible solution for Urban environments and in the end section 3.4 explains the main approach we followed for this thesis and some insights into the network architecture is provided.

3.1 Panoptic Segmentation

The word Panoptic refers to Pan which means all and Optic resembles the vision (Barla 22). In simple terms, it refers to all the visible objects in a scene we are observing and the main goal of this segmentation is to generalize the image segmentation task globally rather than using two separate approaches as semantics and instances (Barla 22). It is the unified approach of combining both of the approaches into a single task in computer vision (Barla 22).

In Panoptic Segmentation, the classification of Objects is done into two categories Things and stuff where things usually refer to the type of objects in the scene that has a proper

physical geometry and we can count the number of instances of these objects in the scene such as pedestrians, cars, animals, etc and stuff refers to the objects that don't possess a proper geometry in the scene but can be heavily recognized due to their texture and type of material like the sky, grass, roads, etc and are usually amorphous regions in the scene (Barla 22).

Dealing with stuff regions in the scene resembles the task of semantic segmentation because as it is uncountable, simply assigning class labels to each pixel is enough without any object detection involved, and the task of dealing with things category is usually related to the task of object detection or instance segmentation where counting the number of objects of same class type is needed and separate them with bounding boxes or object masks, and to achieve these two tasks for stuff and things, semantic segmentation relies on the FCN's with atrous convolutions and instance segmentation uses the concepts of region-based object detections (Kirillov 19b).

For panoptic Segmentation, each pixel is assigned a class label as well as an instance ID and if the pixel has the same label and ID then it belongs to the same object, and in the case of stuff classes, the instance ID is void and the general difference between the semantic, instance and panoptic segmentation can be observed in the figure 3.1 (Kirillov 19b).

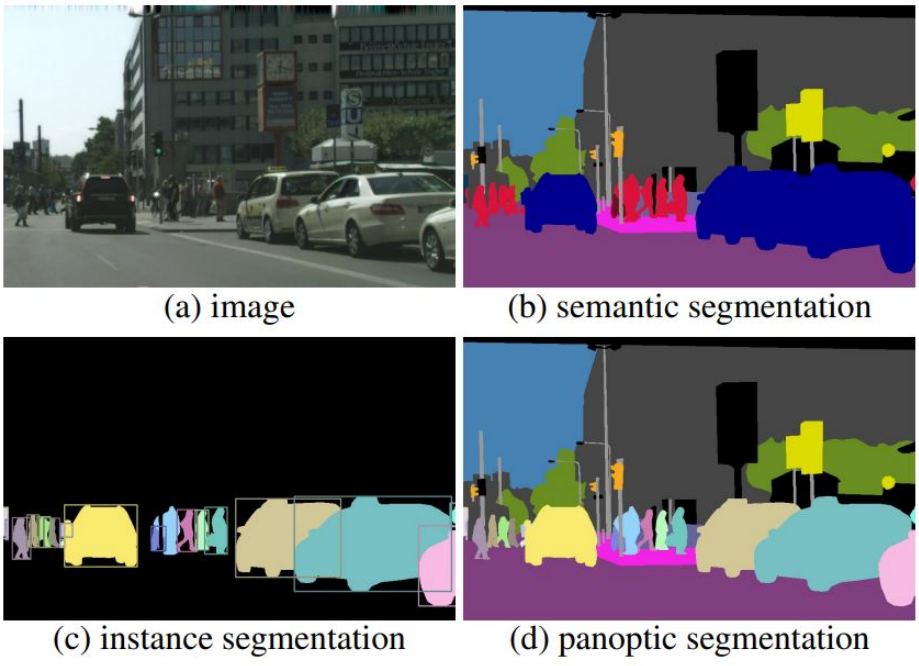


Figure 3.1: Differences between (b) semantic segmentation (class labels per pixel), (c) Instance Segmentation (Class label per Object Mask), and (d) Panoptic Segmentation(Class label per pixel+instance masks per object).(Kirillov 19b)

As per (Kirillov 19b), for the task of panoptic Segmentation, we need to perform the mapping of individual pixels in the image to a pair $(l_i, z_i) \in L \times N$, where L is the set of

L semantic class pixels encoded as $L := \{0, \ldots, L-1\}$. l_i resembles the semantic class pixels of i, and z_i resembles the instance labels which serve as the identifiers and group the same class pixels into different semantic segments in the image, and ground truth annotations need to follow the same scheme as above and in case of ambiguous pixels that are not belonging to a semantic class are assigned as void label category. This clarity we provide for missing semantic labels as void labels provides the Panoptic Segmentation network with more clearer picture for discriminating between semantic classes and instances identification in the image. (Kirillov 19b)

In semantic segmentation, classes are assigned as either stuff L_{St} or things L_{Th} (Kirillov 19b). There exists always the constraints $L = L_{\mathrm{St}} \cup L_{\mathrm{Th}}$ and $L_{\mathrm{St}} \cap L_{\mathrm{Th}} = \emptyset$ (Kirillov 19b). This distinction is needed for instance labels assignment in semantic segmentation, as for the pixels with $l_i \in L_{\mathrm{St}}$ are of stuff category, and for these group of pixels instance label z_i is not important i.e. all the pixels with same stuff class resembles a single instance identifier (e.g., representing the road category) (Kirillov 19b). Conversely, for the pixels belonging to things class ($l_i \in L_{\mathrm{Th}}$), the instance label z_i is very important as the pixels with identical (l_i, z_i) pair resembles the same instance class (e.g., representing a building) (Kirillov 19b). Each pixel within that instance must have at all times the same (l_i, z_i) identifiers and to distinguish which object falls into either things or stuff classes is the dataset decision and is subjective .(Kirillov 19b).

The common association between the tasks of semantic and panoptic segmentation is their requirement for a pixel-level class assignment (Kirillov 19b). when the ground truth data has no things class objects present, then both task objectives align, differing only in the performance metrics used for evaluation (Kirillov 19b). In contrast, the task objectives of instance segmentation need to be adjusted to align with panoptic segmentation (Kirillov 19b). For instance segmentation task, overlapping of object segments is acceptable (Kirillov 19b). But for the panoptic segmentation task, this overlapping should be avoided and each pixel should be identified with a unique class label and an instance label identifier (Kirillov 19b).

3.2 Available approaches for implementing panoptic segmentation

We will now discuss further the recent approaches available for implementing Panoptic segmentation using modern deep learning techniques. Panoptic Segmentation can be achieved using supervised learning techniques, weakly supervised, omni-supervised, and semi-supervised learning methodologies, where in the supervised learning approach, there exist multiple ways such as image-based and video and LiDAR-based panoptic segmentation and we will only focus on Image-based Panoptic segmentation approaches which are relevant to our thesis (Li 22).

Image-based Panoptic segmentation approaches are further divided into four categories: top-down approaches, bottom-up approaches, single-path, and others categories which are illustrated in figure 3.2 (Li 22).

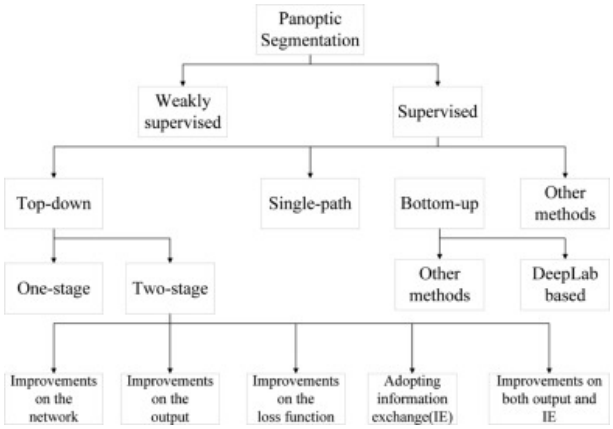


Figure 3.2: Classification of Image Panoptic Segmentation methods.(Li 22)

3.2.1 Top-Down Approaches

The first subsection of the Supervised learning technique of image-based Panoptic segmentation is the top-down methods, which is further classified into two types: two-stage and one-stage approaches, and these top-down methods function by following the sequence of operations detection and then segmentation (Li 22). In the two-stage methods, the first stage involves the generation of object proposals from the image, and in the second stage post-processing is performed to implement the segmentation and one-stage approaches eliminate this part of proposal generation and utilize the concepts of anchor-free or anchor-based object detections (Li 22).

As per (Kirillov 19b), a baseline for Panoptic segmentation is defined in which there exist two individual branches of semantic and instance segmentation, the outputs of which are post-processed and fused to form the final segmentation output, which can be visualized in figure 3.3. (Kirillov 19b)

In this two-staged approach, the instance segmentation is performed using the most popular Mask R-CNN (He 22), whose network pipeline is depicted in figure 3.4, which is an improvement to Faster R-CNN by replacing the Region of Interest (ROI)Pooling operation with ROIAlign layer to minimize the quantization error discussed earlier and this Mask R-CNN uses a smaller FCN in comparison to Faster R-CNN for mask prediction in addition to box regression and output class classification in respect to every pixel in the image (Li 22).

As shown in figure 3.3, during the post-processing stage, the outputs of both branches should be resolved for conflicts that might arise due to differences in class labels from the

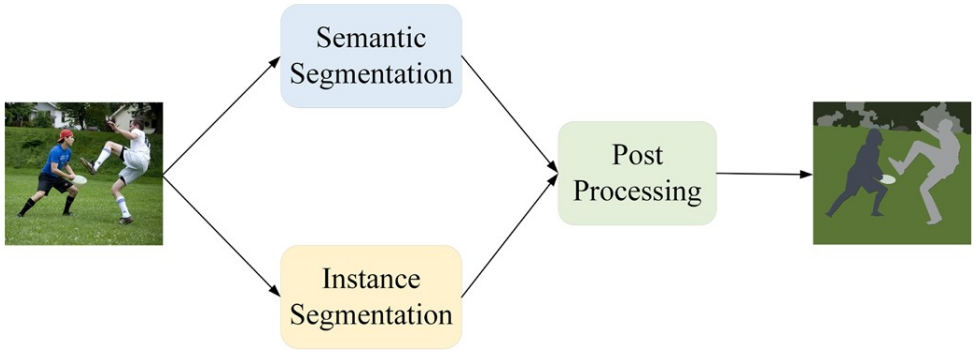


Figure 3.3: Baseline of Panoptic-segmentation in two-staged model.(Li 22)

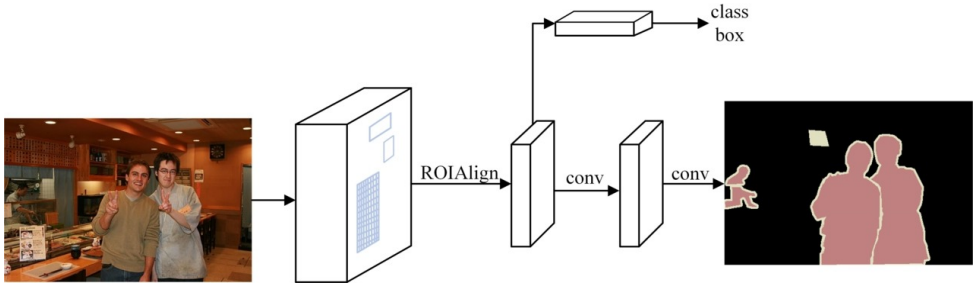


Figure 3.4: Instance segmentation in two-staged approach using Mask R-CNN architecture.(He 22)

predictions and also conflicts inside the individual branches, especially within instance segmentation branches such as overlapping of segments and occlusions (Li 22).

Joint semantic and instance segmentation network(JSIS-Net) (De Geus 18) is another approach to implement panoptic segmentation (Li 22). This network uses a spatial pooling module for feature extraction in the semantic segmentation branch and uses Mask R-CNN for the instance segmentation branch, predicting the output in the form of pixel clusters (De Geus 18). The resulting output is normalized into a mask by the post-processing module (De Geus 18). The conflicts between the branches are resolved by replacing the things class of the stuff category with the things class of the instances category (De Geus 18). The conflicts within individual branches are ignored in this approach and need to be further addressed (De Geus 18). This architecture uses two independent branch outputs and this is not an optimal solution and there is a possibility for optimization (Li 22).

An improvement to JSIS-Net is the Panoptic FPN (Kirillov 19a) that uses ResNet-FPN as the backbone for feature extraction in which ResNet (He 16) acts as an encoder and Feature pyramid network (FPN) (Lin 17) as a decoder that can be used to extract the features at multiscale (Li 22). This is the main approach we are about to follow in this thesis and in the later sections a detailed explanation of this approach is provided.

Another popular approach is the EfficientPS (Mohan 21) which uses the improved EfficientNet (Tan 19) as the backbone for feature extraction and also contains two-way FPN

that enables the bi-directional information flow and results in rich quality panoptic results and this architecture has two output branches for each task of semantic and instance segmentation and in the end a separate fusion block which combines the outputs of these two branches and the whole structure is depicted in the figure 3.5 (Li 22).

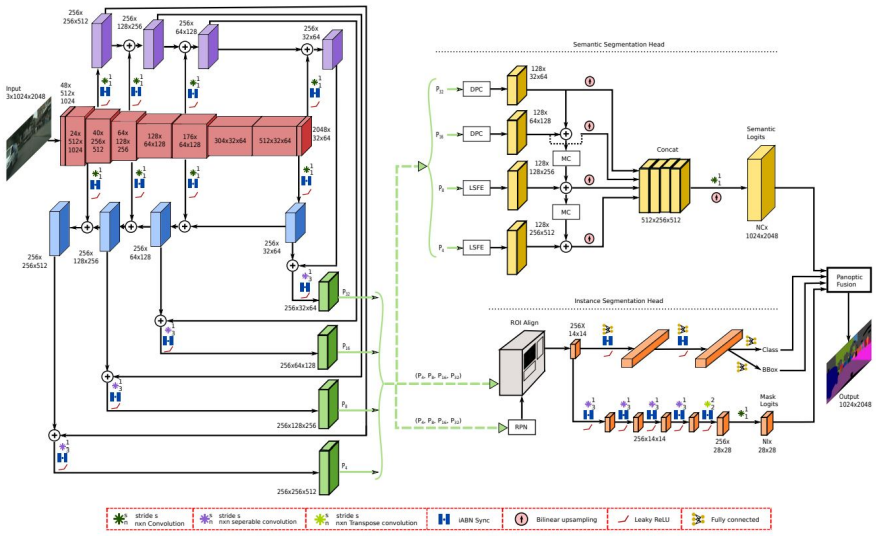


Figure 3.5: EfficientPS network architecture is shown in red and two-way FPN in purple, blue, and green. The semantic segmentation network is in yellow and the instance segmentation network is in orange. The fusion block is shown at the end.(Mohan 21)

EfficientPS operates by feeding the input image to a backbone of EfficientNet which internally contains also a two-way FPN that fuses the output extracted features at multiple scales and gives the result of high-quality rich information embedded features and this output is fed to the two branches of semantic and instance segmentation which have proper mechanisms inside them to extract the information of objects from stuff and things categories and instance segmentation branch is responsible for object detection, classification and masks generation from the extracted features and these outputs are fed into the final fusion block where it fuses the predictions from these both branches to achieve panoptic segmentation (Barla 22).

The Fusion module is a non-parametrized one that doesn't update automatically during the backpropagation and operates in two stages wherein the first stage, it receives the class predictions, bounding box scores, and mask logits and it filters out the confidence scores based on threshold and padding is performed on the rest of the instances and scaling operation is performed on the mask logits predictions to match the input image shape and later in the second stage, evaluation of overlapping logits is done using sigmoid calculations and by setting up the threshold value, we obtain binary masks and if the threshold of overlapping masks is exceeding then those are retained and rest of the masks are removed and the same procedure is followed as well for semantic head predictions and the final

remaining outputs are fused using Hadamard product and the resultant output is the panoptic segmentation (Barla 22).

UPSNet (Xiong 19) uses a parameter-free head for panoptic segmentation which works by using pixel-wise classification; this head fuses the predictions from the instance and semantic branches and refuses to classify unknown class labels and this head is responsible for high panoptic scores but largely dependent on the quality of predictions of instance and semantic segmentation (Li 22). The baseline for UPSNet is as shown in figure 3.6 (Li 22).

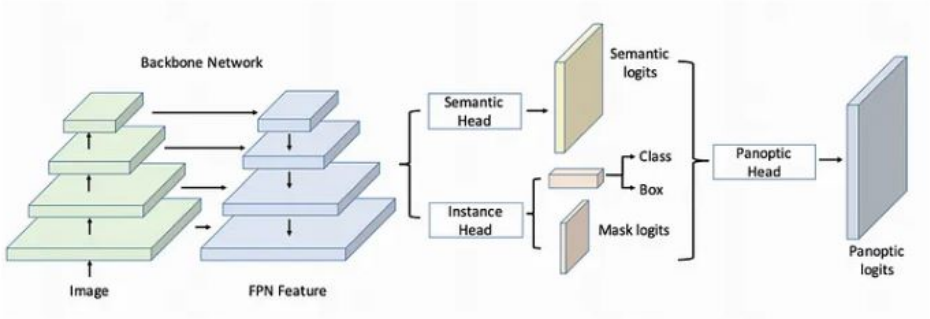


Figure 3.6: UPSNet baseline architecture.(Xiong 19)

3.2.2 Top-down approaches(One-stage)

In Top-down one-stage methods, the proposal generation stage for the object detector is replaced with anchor boxes or anchor free which are essentially reference frames used to detect objects inside any image and these anchor boxes vary in size and can be predetermined using approaches like key-point detection (Li 22).

FPSNet (de Geus 20) is a One-stage approach where proposal generation is replaced and it has a special attention mechanism, for instance, segmentation and it has the task of classification of dense pixels to achieve the panoptic segmentation where attention masks take the place of instance prediction masks and therefore avoiding the need of post-processing for fusing the outputs from semantic and instance branches as in the end we will receive a single feature map from both of the branches to perform the classification task and thus can function in real-time but the performance is far more inferior in comparison to the two-stage methods (Li 22).

Single-shot Panoptic segmentation (Weber 20) and SpatialFlow (Chen 20) are two more approaches that are based on RetinaNet (Lin 17) as object detector and the network architecture has ResNet-FPN combination for encoder-decoder and provides us with rich multi-scale context and the semantic segmentation uses the normal FCN but the outputs from this FCN and RetinaNet object detections are sent to a panoptic branch where it uses the information from instance center, semantic segmentation, and RetinaNet object

detection predictions to perform the final fusion for panoptic segmentation in which it solves the common clashes between the semantic and instance segmentation branch such as overlapping and wrong predictions and mismatches between both branches and also this head deals mostly with object detection rather than instance segmentation and the main advantage being this head simply doesn't merge the outputs from both branches but rather generate instance aware panoptic logits as output and as the SpatialFLow uses RetinaNet as object detector, which can enable it to use the feature interweaving for every pixel and as shown in figure 3.7, the output from the network is of four main categories as box classification, bounding box regression, stuff and things segmentation heads and also there exist improvements in the quality of features from the segmentation tasks for both stuff and things categories as adding of stuff and things subnetworks is done in parallel which facilitates the flow of spatial information that gives us more context information (Li 22).

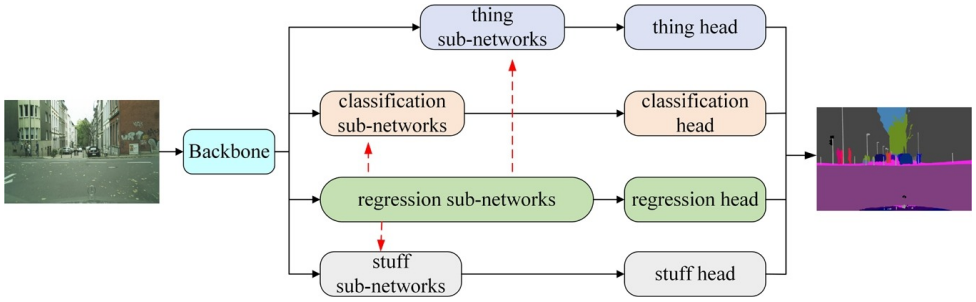


Figure 3.7: SpatialFlow structure.(Chen 20)

3.2.3 Bottom-Up Approaches

The top-down approaches we had discussed before focused on generating instance mask predictions for each instance and therefore very effective in instance detection and bounding boxes are generated to locate the objects in the image effectively and due to this, these approaches need more computational cost, and are therefore not very fast in inference and another approach which doesn't rely on instance segmentation to this granular level but focuses first on semantic segmentation and then adds instance predictions by grouping and clustering is the bottom-up approaches which use schemes like majority voting for merging the output predictions from the instance and semantic segmentations (Li 22).

The first and foremost approach under Bottom-up methods is the DeeperLab (Yang 19) which is based on the famous architecture of DeepLab V3+ (Chen 18) which is the state-of-the-art approach for semantic segmentation being used in the industry (Li 22). As shown in figure 3.8, the network consists of three main parts encoder, decoder, and prediction and in this architecture, an innovative space-depth, depth-space conversion module is introduced to replace the upsampling operations in the decoder module and it saves a lot

of computation resources and thus faster and for the task of instance mask prediction, this architecture employs a method of key point representation in which bounding box coordinates around the object are considered as target key points and the output from the instance segmentation branch predicts four different heatmaps such as keypoint heatmap, long, short, mid-range offset maps as outputs and these all outputs are fused to obtain the final panoptic prediction (Li 22).

Axial-DeepLab (Wang 20b), make modifications to the backbone feature extractor of ResNet by modifying the 3x3 convolution layers with a self-attention module which is sensitive to position and it decomposes two-dimensional self-attention modules into two separate one-dimensional attention modules and forms Axial-ResNet which has proven very good performance on the benchmark of Cityscapes dataset but has limitations with deforming objects in the scene (Li 22).

Panoptic-Deeplab (Cheng 20) is another better-performing architecture that is based on DeeperLab which has special modules introduced for the task of semantic and instance segmentation (Li 22). we will look into this architecture in detail in the later parts of this paper.

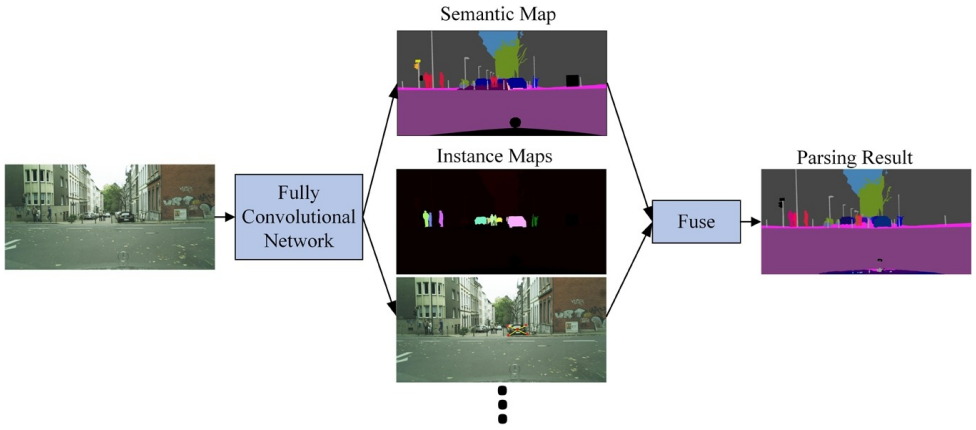


Figure 3.8: DeeperLab structure.(Yang 19)

Another approach that doesn't use Deeplab architectures is the Pixel consensus voting for panoptic segmentation (PCV) (Wang 20a), which uses Hough transformation and unlike other methods that rely merely on densely detecting the object proposals, this method detects instances based on the pixel-wise voting as shown in figure 3.9, an FPN acts like a shared backbone for feature extraction for both semantic segmentation and instance segmentation voting branch and both of these branches are trained pixel-wise with an entropy loss to predict the outputs and this PCV utilizes the scheme of discretizing regions around pixels into cells following voting and generates voting heatmaps using atrous deconvolution and then inversion of this voting filter is convolved with peak regions which are then back-projected to produce instance predictions at that peaks (Chang 23).

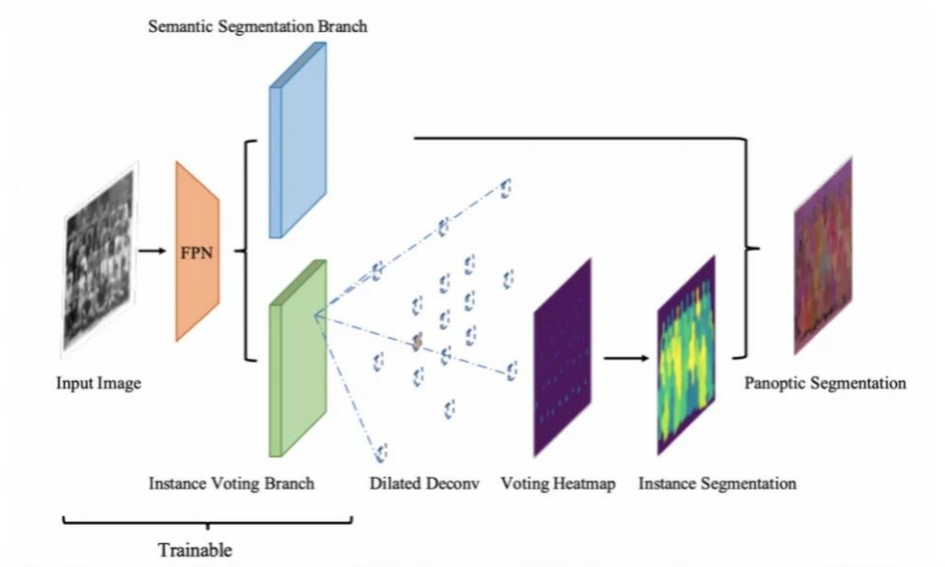


Figure 3.9: Pixel consensus voting for panoptic segmentation(PCV) structure.(Wang 20a)

3.2.4 Single-Path approaches

Until now in top-down and bottom-up approaches we have observed that to be able to implement panoptic segmentation, we need to implement instance segmentation and semantic segmentation but panoptic segmentation has unique characteristics that need for instance predictions such as non-overlapping segments between masks and this single-path approach is a single task approach where we will get the panoptic output directly in the end (Chang 23).

Panoptic FCN (Li 21) is the initial method known in the single-path approach where the object instances and semantics information are embedded into kernel weights with the help of a kernel generator and the final predictions are made by convolving these kernels over features extracted and this approach has semantically and instant aware properties for both stuff and things classes (Chang 23).

As shown in figure 3.10, there exist three major components in this architecture: Kernel generator in which a position head identifies the object center positions, as well as stuff regions, and the kernel head, is responsible for generating weights of this detection into kernel weights and kernel fusion module integrates this kernel weights and feature encoder is responsible for extracting rich features and these weights are convolved onto those features to get the final panoptic segmentation prediction (Chang 23).

K-Net (Zhang 21), proposes a unified panoptic segmentation framework that deals with the overlapping problem between segments by laying a constraint of a single kernel per instance mask and some kernels for semantic classes as well known as semantic kernels and the instance kernels which are designed to detect single segment of instance per kernel needs to have more discriminative capacity and this can be achieved with constant updates

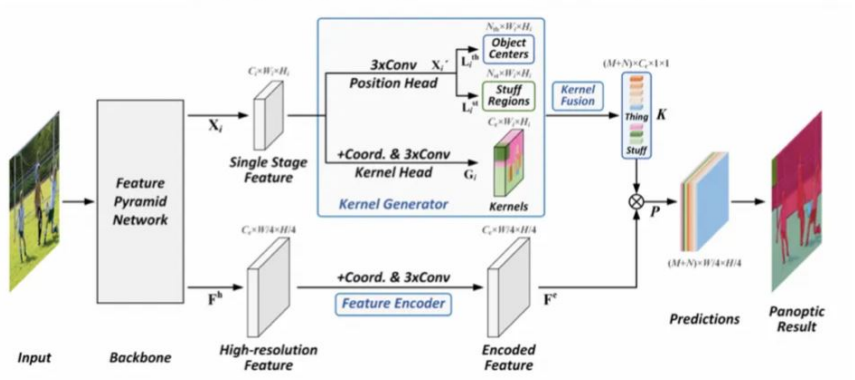


Figure 3.10: Panoptic Fully Connected Network architecture.(Li 21)

of kernel weights with the help of kernel update module which performs group feature assembly and kernel interactions as well as kernel updates and the updated kernel weight is convolved with the image feature map to obtain the panoptic segmentation prediction as shown in figure 3.11 (Li 22).

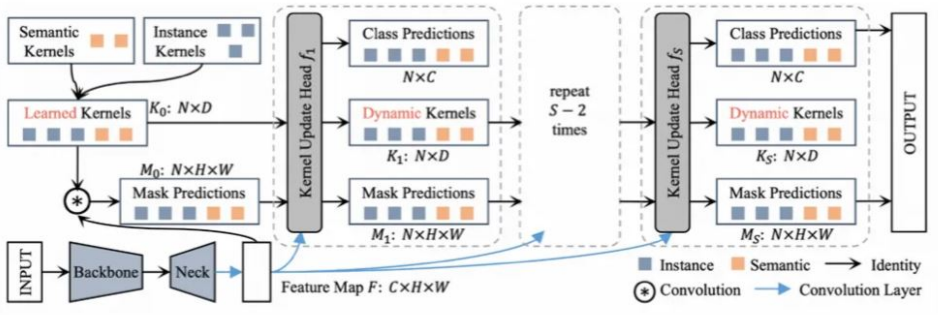


Figure 3.11: Image showing the functioning of Kernel Update procedure in K-Net architecture.(Zhang 21)

There exist other approaches for panoptic segmentation like Video and LiDAR-based panoptic segmentation and panoramic panoptic segmentation approaches and weakly supervised, semi-supervised, and omni-supervised approaches which are not discussed here as they are of least relevance to our work here. Discussed methods above have their advantages and disadvantages in some regards for instance top-down approaches are better for object detection and in the output, we can see the object's precise location using a bounding box with the confidence scores of the model predictions, however, it is slightly computationally costlier in comparison to bottom-up methods as it needs post-processing and fusion modules to resolve the conflicts between different heads and bottom-up methods are highly beneficial for faster computations and lacks in the ability for object detection

to a high level and thus have lower performance compared to top-down approaches and one-stage methods are relatively faster in performance but less accurate (Li 22).

3.3 Works done in Urban environments

Urban environments are often characterized by clearly paved pathways, and different objects in the scene like pedestrians, vehicles, traffic infrastructure elements, buildings, and many more which can provide rich quality information for the sensors to capture and further process. When we have traffic information properly organized in this application area, then the best and state-of-the-art approach to use in this type of scenario is Panoptic-Deeplab for many reasons which are explained below as follows: Availability of Rich information about the Urban environment which is useful to extract the finer features out of Deeplab architecture with the output of fine-grained segmentation by differentiating various objects in the scene including more stuff category regions in urban environments like roads, and sideways. Objects of irregular shapes and sizes can be easily handled with the help of Deeplab architecture. It can identify objects of varying sizes, such as traffic signs, and large-sized buildings. Deeplab is based on semantic segmentation, the context of the scene is highly understood and this helps us better understand the relationship between different objects in the scene. Identification of one object might provide us with some context to detect others in the scene as well and this is the reason Panoptic-DeepLab works excellently on the Cityscapes Dataset (Cordts 16) which is a benchmark dataset involving various traffic-related scenes.

Panoptic-DeepLab belongs to the branch of Bottom-up methods for achieving panoptic segmentation, which usually starts with the semantic segmentation task and then grouping operation for the instance masks generation and then the majority voting rule for merging both of the predictions from these two tasks and this sequential following order is beneficial for bottom-up approaches to achieve faster inference speeds, performance in the evaluation metrics are inferior in comparison to top-down approaches (Cheng 20).

Panoptic-DeepLab is trained using only three loss functions and it is constructed based on the DeepLab architecture with slight modifications and adaptations from its architecture such as dual-ASPP and dual-Decoder modules which are related to semantic and instance segmentation tasks and for the semantic segmentation, it follows the well established DeepLab semantic segmentation approach and for the instance segmentation, the model relies on instance center regression scheme where for every object detected, instance center is estimated and the surrounding pixels will be grouped into instances based on these pixels relative distance to the nearest instance center and in addition a slight computation cost is involved with the existing semantic segmentation network and with the modern days computing power onboard of the vehicles, Panoptic-DeepLab can achieve a real-time performance to predict the scene in a panoptic manner (Cheng 20).

3.3.1 Panoptic-DeepLab architecture

Before we start with the architecture of Panoptic-DeepLab, we will first investigate the DeepLab architecture's main components for a better understanding of semantic segmentation achieved through it. DeepLab is the most famous approach for implementing semantic segmentation of scenes and objects present in it at multiple scales (Chen 17b). It is upgraded into multiple versions like DeepLabv2, DeepLabv3, and DeepLabv3+ with improvements in every upgrade, and starting with v2, the authors introduced the concepts of Atrous convolutions and spatial pyramid pooling modules to achieve semantic segmentation (Chen 17b).

As Per (Chen 17b), Deep Convolutional Neural Networks(DCNN) perform the operations of convolutions followed by pooling operations at each level to extract the features at multiple scales and to do that, they reduce the image size sequentially and this leads to the loss of spatial information needed for semantic segmentation context and to avoid this, in DeepLab they introduced the concept of Atrous convolutions or dilated convolutions where they replace the kernel weights with holes to control the feature resolution which is controlled in DCNNs and using the Atrous convolutions increase the field of view with increasing spaces between kernels to extract the multi-scale context information. with these convolutions, there is no necessity to learn additional parameters. (Chen 17b).

During deconvolution using transpose convolutions, here we use atrous convolutions in DCNNs for retaining spatial information we lose generally with normal convolutions (Chen 17b). The equation for representing atrous convolutions is represented in equation 3.1 (Chen 17b) and it is very similar to the normal convolution equation but the only change is the dilation rate and i is the convolution location in the image, the output is denoted by y and weight is denoted by w and the input is an image vector x and the variable r is the dilation rate or simply spaces between the kernel weights for the atrous convolution and this value is 1 always for the normal convolution and this value can be varied as per the required need to change the field of view we need for semantic segmentation (Chen 17b).

$$y[i] = \sum_{k} x[i + r \cdot k] \cdot w[k] \tag{3.1}$$

Along spatial dimension, defining the distance between two pixels is the same as defining the rate of r-1 or simply inserting holes between filter weights and separating them by this distance or controlling the feature density inside a CNN (Chen 17b) and it is visually represented in the figure 3.12.

DCNNs also have the drawback of fixed size input resolution and to overcome this problem, there exists the technique of spatial pyramid pooling (SPP), which is a pooling strategy that generates a fixed length representation vector that mitigates the need for a specific

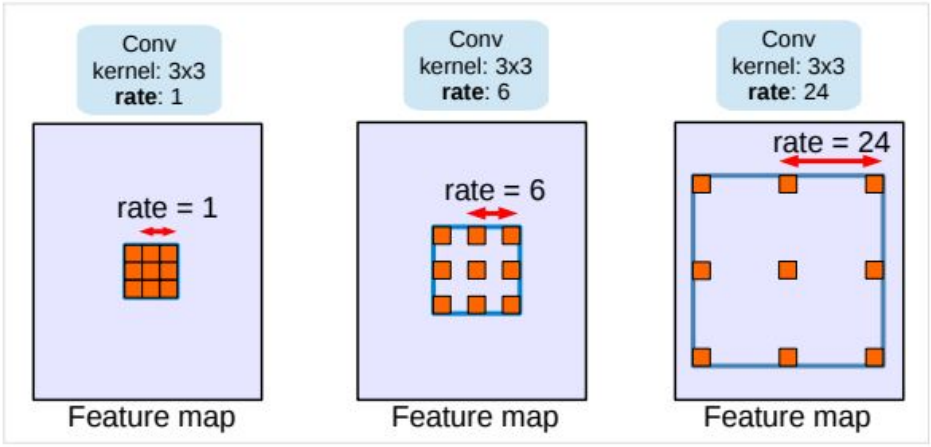


Figure 3.12: Image showing the different dilation rates and their effect on the filters that are used for feature extraction(Chen 17b)

input image size needed for the network to run and this technique can handle the object deformations and as we know that in CNN, there is no constraint of input size due to convolution layers as they can handle any size images and generates features of that input sizes but in the deeper layers for any classification tasks, there are fully connected layers which accept only a fixed size vectors as input and these are the culprits in CNN models and to overcome this constraint we need SPP modules before those fully connected layers to pool the feature sizes to the dimensions that these final layers accept (He 15).

This network avoids the task of repeated feature extraction and based on the features extracted once, pooling is performed at the required layers using this SPP module and models based on ImageNet training, has the softmax activation at the final fully connected layers and this SPP is an improvement to the exisiting Bag of Words (BoW) method and this SPP pools the features extracted at a certain location and due to this, the spatial context information is not lost and the from the features being pooled, the bin size is related to the image size from which features are extracted and as shown in figure 3.14, in DeepLab for the semantic segmentation combination of ResNet and Atrous convolutions and SPP is performed to extract the features at multi-scale which have spatial context information (He 15).

Using the above-mentioned two new techniques, DeepLab can function efficiently for performing semantic segmentation and also the task of Panoptic Segmentation (Cheng 20). Panoptic-DeepLab also uses these components of DeepLab model architecture (Cheng 20). The network of Panoptic-DeepLab is as shown in figure 3.14, where it essentially has four main components: shared encoder network for both semantic and instance segmentation, decoupled ASPP module involving atrous convolutions and spatial pyramid pooling, decoupled decoder blocks for task-specific functions and prediction heads dedicated to tasks alone (Cheng 20).

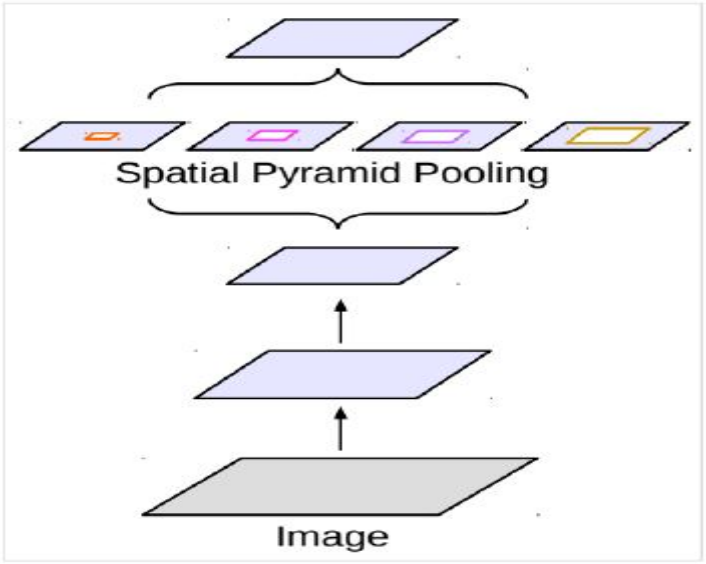


Figure 3.13: Mechanism of Spatial Pyramid Pooling module where features are extracted once and pooled at later stages.(Chen 17b)

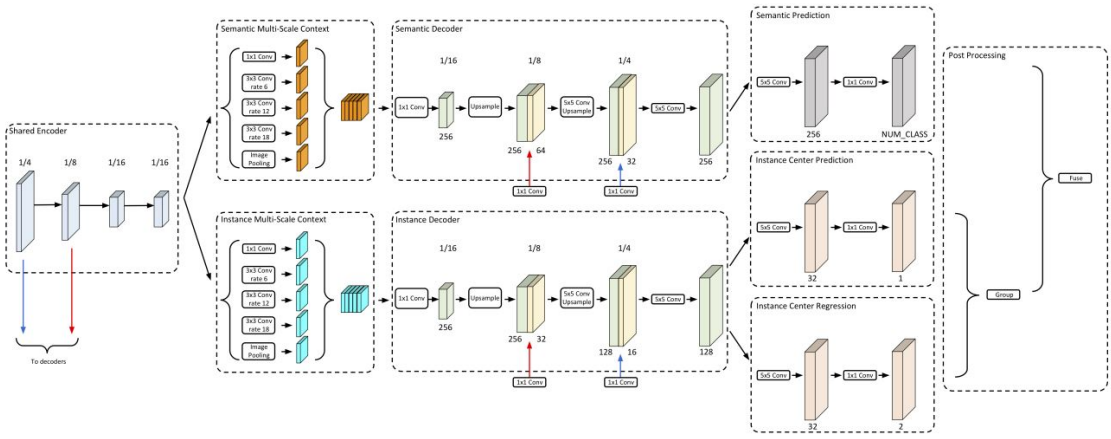


Figure 3.14: Panoptic-DeepLab network architecture along with dual ASPP and dual decoder modules.(Cheng 20)

The encoder module is a shared backbone between the tasks of semantic and instance segmentation, whose responsibility is to generate rich feature maps using the atrous convolutions (Cheng 20). In Panoptic-DeepLab, separate ASPP modules for each task are used and a separate decoder is employed as they both need the different contextual information from the features extracted in the encoder and decoder blocks (Cheng 20). For this panoptic-Deeplab, the decoder used is a lightweight decoder based on the DeepLabv3+ network and modifications are employed such as the addition of a new low-level feature with the stride of 8 to the decoder to preserve the spatial information in the features by a factor of 2 and during the upsampling stage in the decoder, we use 5x5 depthwise separable convolutions (Cheng 20).

For semantic segmentation, the cross-entropy loss is used while training to predict both the stuff and things classes, and for instance segmentation, we use the approach of the voting scheme discussed earlier for regressing the instance center of the object and grouping the pixels accordingly (Cheng 20). The mean squared error loss for reducing the distance between the predicted heatmaps and ground truth 2D Gaussian heatmaps is used, and L1 loss for the instance center regression task applies to only the things category (Cheng 20).

During the inference stage, we utilize the grouping algorithm for generating instance masks and majority voting schemes for the fusion of instance and semantic segmentation predictions (Cheng 20). To represent a single instant object, we use its center of mass denoted by $\{C_n : (i_n, j_n)\}$ and to predict the center points of instances, we use key-point based non-maximum suppression on the instance centers of the predicted heat maps or simply it is same as retaining the points which are constant after applying max pooling on the heat maps and a threshold is used to filter out the low confidence score predictions (Cheng 20). For regressing the instance center to assign an instance id to each pixel within the image for the panoptic segmentation, we predict an offset vector o(i,j) where i,j represents the offset to the pixel value in horizontal and vertical directions and instance id for the pixels is allotted same as the nearest instance center by shifting the pixel location with the predicted offset vector (Cheng 20). This whole regression loss is expressed in the equation 3.2(Cheng 20), where $k_{i,j}$ is the instance id prediction at pixel location (i,j) and semantic predictions are used to eliminate stuff category from being assigned instance ids as their instance id is always set to zero (Cheng 20).

In the end Majority voting scheme is used to fuse the semantic segmentation prediction and class-specific instance prediction as per DeeperLab, from the predicted instance mask, the semantic label is inferred using this majority vote of the semantic labels by accumulating the class labels histograms (Cheng 20).

$$k_{i,j} = \arg\min_{k} \|C_k - ((i,j) + O(i,j))\|_2 \tag{3.2}$$

3.4 Works done in off-road environments

With the availability of huge data and varying types of datasets to train the models for autonomous driving tasks, for Urban scenarios, near-to-perfect implementations are being achieved for example Tesla vehicles, and Google Waymo. This data availability leads to new developments in computer vision tasks such as object detection, Image classification and segmentation, depth, and distance measurements, object pose estimation, and tracking (Lagos 23). Collecting this dataset from the Urban scenes is quite well established due to the controlled environment with clear cues such as road markings, lanes, and traffic signs (Lagos 23). On the contrary, in the case of off-road environments, even this data collection is difficult due to numerous reasons such as uncontrolled environment and varying landscapes, extreme trails filled with different obstacles for the scene capture (Lagos 23). The Appearance of objects in the scene varies with illumination changes, and when we can be able to collect the data from such harsh challenging environments then we can use that to navigate vehicles autonomously in off-road situations like farming, construction, and inside forests (Lagos 23).

Collecting data from the forest environment can help navigate the vehicles in any unstructured environment scenarios like dirt trails, or gravel roads (Lagos 23). Heavy machinery can benefit from learning to drive autonomously (Lagos 23). As the applications involving this type of data are quite limited, other application areas can simply use this data to train by exploiting the nature of scenes being captured which are quite varying and uncontrolled and even models can train well with this type of diversified data (Lagos 23). It is also challenging to collect unique data from forest environments as they change their scene nature as per the different weather conditions and vegetation (Lagos 23). changes that occur seasonally lead to different representations of the scene and hence we need to capture a lot of details than actual datasets (Lagos 23).

An example dataset collected from the forest environment is presented here known as the FinnWoodlands dataset, where the 5170 images are captured using a Stereo camera and LiDAR point clouds data and depth information is recorded for each image in the dataset (Lagos 23). The dataset is annotated in the COCO format for training Instance segmentation, semantic segmentation, and as well as panoptic segmentation tasks as shown in figure 3.15, and it has three stuff categories as Lake, Ground, and Track and a total of five things categories: Obstacle, Spruce, Birch, Tree, and Pine (Lagos 23).

Hence, the collection of a huge dataset for a better generalization of the model and the problems posed by the terrain and the difficulties involved in the setup to stay stable while capturing images is more due to the uneven ground surface. A lot of obstacles in the scene and varying scene visibility due to different weathers and many more factors make the Off-road environment a challenging case for Image segmentation, especially panoptic Segmentation. As the model needs proper annotation ground truth data to make accurate

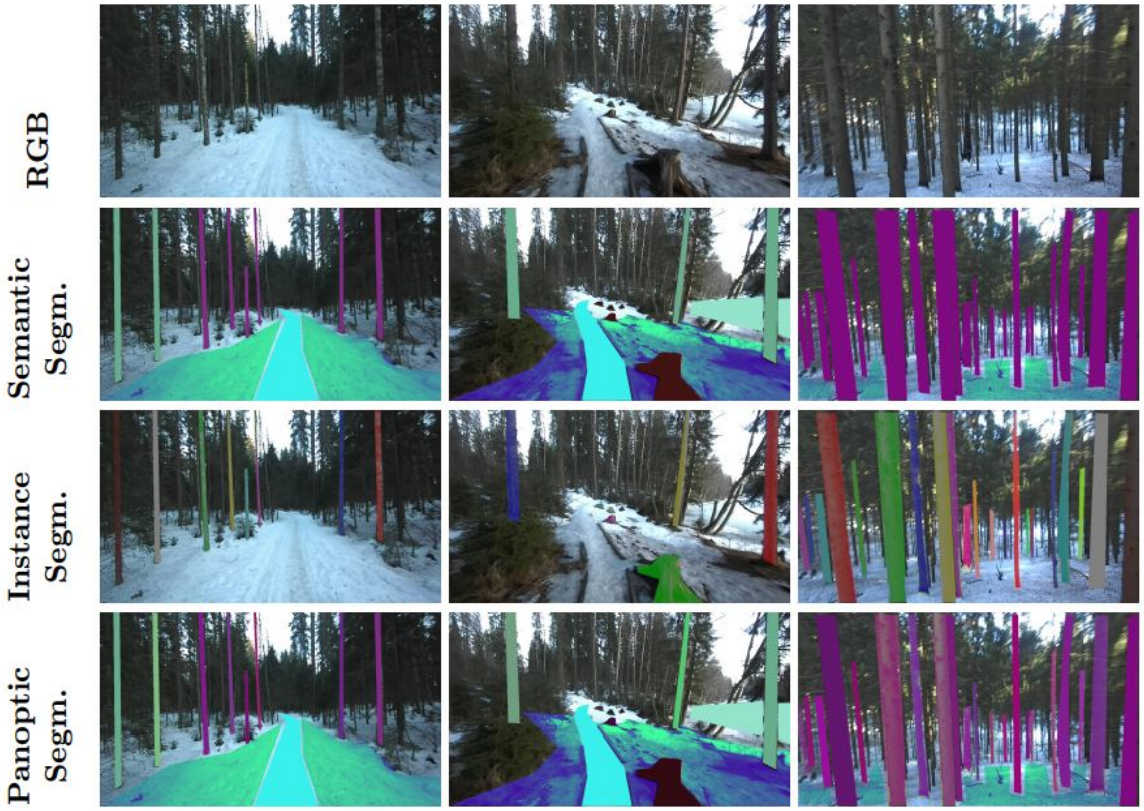


Figure 3.15: Groundtruth Annotations for FinnWoodland Dataset where the first row is the RGB Images captured during dataset creation and Groundtruth annotations for tasks like instance, semantic, and panoptic segmentation.(Lagos 23)

predictions, is also a hurdle when it expects non-overlapping annotations. It is indeed a problematic case and needs more human expertise and thus our work needs to address all the above-mentioned problems to create a baseline for Panoptic Segmentation in Off-road environment especially forest environment for our Unimog to drive autonomously.

In the next chapter, we will discuss the approach we intend to plan to solve the task of Panoptic Segmentation. An overview of our Forest panoptic dataset is presented as well as some implementations and optimizations performed to tackle the drawbacks faced due to this uncontrolled environment and to make the panoptic segmentation perform slightly better than usual.

4. Approach and Implementation

This chapter gives us an overview of the approach being followed for achieving the end goal of implementation of panoptic segmentation with additional optimizations and image processing techniques employed during this thesis. Section 4.1 explains the types of Datasets used for training and testing the Panoptic model, the procedure of collecting it, and reshaping the collected data into model-accepted ground truth annotations COCO format specially designed for the panoptic segmentation task using labelme as the annotation tool.

In section 4.2, the main network architecture is described, and the advantages of it over other available approaches are discussed. In addition to this network, we tried to pre-process the image for shadow removal and exposure correction which are explained in section 4.3. The training procedure is explained in section 4.4, and the difficulties involved are briefly analyzed along with the implementation and testing explained in section **??**.

Section 4.5 introduces the evaluation procedure for the trained model with the dedicated metrics for the task of panoptic segmentation and shortcomings of other evaluation metrics such as Average precision for this task and definitions of the metrics such as panoptic quality(PQ), segmentation quality(SQ) and recognition quality(RQ).

4.1 Dataset

Datasets are the crucial elements needed for any machine-learning algorithm to train and mimic the predictions on a human level. These inputs are not restricted only to computer vision tasks. In the year 1997, using this huge collection of data, a chess-playing computer DeepBlue was able to defeat the world champion of that time, Garry Kasparov using the 700000 grandmaster games data it used to train itself (Campbell 02). Although DeepBlue used a brute force approach, the greatest record of using datasets and creating records

in game history was in the 2016 game of ALPHAGO against Go professional player Lee Sedol (Silver 16).

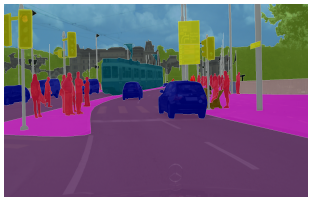
The insufficient availability of annotated ground truth data imposes a major constraint on the applications of Deep Learning techniques. The complexity of real-world problems and the diversified nature of data for each task, make machine learning inaccurate in comparison to human brains. To overcome this problem, the availability of huge varied datasets that can capture the actual representation of the environment for the machine to learn needs to be provided and this is a very difficult task to collect such datasets.

There are a few massive datasets available for the models to learn the initial level low-level features from an image using Imagenet (Deng 09). But when it comes to task-specific training, these low-level features wouldn't help the model to generalize well to the application at hand. In modern days, the tiring manual annotation of images for classification tasks has been avoided to some extent using crowdsourcing, however for the panoptic segmentation task, we need the ground truth annotations without overlapping segments and to achieve it, a little skill and human carefulness is needed which consumes a lot of time.

The camera sensor is the most cost-effective sensor solution for autonomous driving and based on the applications, there exists a few renowned datasets for urban driving scenes such as KITTI (Geiger 12), Waymo Open Dataset (Mei 22), Cityscapes (Cordts 16), etc as shown in table 4.1. The data collection of these sensors involves not only the camera but also other sensor modalities such as RADAR, LiDAR, etc. In addition to providing only image or point cloud data collected, these huge datasets also released ground-truth annotations for Image segmentation applications such as semantic, Instance, and panoptic segmentation. There exist several datasets of Urban driving that focus on providing the training data in Urban scenarios. But when it comes to off-road use cases, there are very few datasets available like Freiburg-forest referenced in (Valada 17), which focuses on the scenes similar to our application usage i.e. forest environments but this dataset is of very limited data and thus if used alone for training our model, it suffers from the generalization and also there is lacking ground-truth annotations and the resolution to train the model is very high and resizing operations leads to drop in the model's prediction accuracy.

Unlike Urban Datasets, off-road datasets are unique in some perspectives such as capturing the object features inside the image and this is quite different in comparison to urban scenes where the features of the object wouldn't vary too much from scene to scene. For example, consider a pedestrian or a car and their features are the same throughout the dataset whereas in the case of off-road environments objects such as trees vary in features due to their changes in appearances from one object to another and due to weather conditions, illuminations, occlusions in the scene like some bush obstructing the camera view and this leads to problems in identifying the features differently than the actual features the object

(a) KITTI Dataset sample



(b) Cityscapes Dataset sample

(c) Waymo Open Dataset sample

Table 4.1: Urban datasets for Image segmentation samples for different datasets along with annotations representation.

holds and while collecting the dataset in an off-road scene, these factors plays a big role. Due to this difficulty, it is rather challenging to collect a detailed dataset to describe such an environment completely.

In off-road applications, the most commonly used datasets for Image segmentation tasks are the Freiburg Forest (Valada 17) and Yamaha-CMU-Off-Road (YCOR) (Maturana 18). YCOR is a custom dataset collected which contains 1076 images in different locations and different seasons that contain the classes as the sky, grass, rough and smooth trails, vegetation, obstacle, and non-traversable vegetation which are annotated using polygon interfaces and is diversified as shown in figure 4.1 (Maturana 18). The scenes presented in the YCOR dataset are almost the same with very little variability as the images have a uniform exposure in them which is not the test scenario in which we want our vehicle to navigate so this dataset is discarded from our training. Another popular dataset is the Freiburg forest dataset (Valada 17), where data is collected using a stereo module of Bumbelbee2 Stereo camera and also a modified dashcam to capture both RGB and Near Infrared(NIR) images and due to capturing the scene with two cameras at the same time pose a problem of synchronizing both views and also to address the varying seasons and illuminations conditions, data was captured on different days and it captured also the Vegetation Index images which are a useful data for of-road navigation applications and this dataset is very little with 229 training and 133 testing images but still serves as a useful starting point to train our original model on our dataset created for this thesis which is known as RPTU-Forest dataset. The sample images of the Freiburg forest dataset are presented below in the table 4.2 (Valada 17).

Table 4.2: Sample Images from Freiburg Forest dataset from (Valada 17).

Figure 4.1: Samples of Yamaha-CMU-Off-Road Dataset. (Maturana 18)

4.1.1 RPTU-Forest Panoptic dataset

For this work, we opted to create and use a custom dataset taken with the multi-spectral camera Sequoia in the forest backside of the RPTU-Kaiserslautern campus. Data is collected on different days to overcome the problem of less variation in the images and the collected images are unique in their way and capture the forest environment to some extent however the total dataset consists of 285 images in total with the combination of RGB, NIR, NDVI images that are directly collected from the camera and the forest environment is quite challenging in these images with half of the image being only vegetation and very little open spaces available for the vehicle navigation.

The common feature among all these off-road environment datasets is the trail on which vehicles are supposed to navigate and the objects they detect in these kinds of environments. We planned to take the same class distribution for our dataset annotation similar to the Freiburg forest dataset but due to differences between both datasets and their variation in scenes, this idea was dropped. Our RPTU-Forest dataset is annotated from scratch by taking similar classes present as in the Freiburg dataset and adding some classes as per the scenes captured in the dataset.

The Freiburg forest dataset class labels are as follows: woods, ground, roads, Sky, and others (for background). For panoptic Segmentation, we decided on our class categories in the RPTU-Forest panoptic dataset as shown in figure 4.2, and a background class label for pixels with no class categories such as sky which is not needed for our application as it is uniform and doesn't bring any useful information.

we used labelme tool for annotating the ground-truth data (in case of panoptic segmentation, it is instance masks, semantic segmentation labels, and panoptic segmentation annotations)

Class	stuff/thing
ground	stuff
bush	stuff
tree_branches	stuff
rock	stuff
road	stuff
tree_trunk	thing
person	thing
vehicle	thing

Figure 4.2: Class labels in RPTU-Forest Panoptic Dataset.

from the images by manually labeling every individual image with their respective classes with proper boundaries drawn using polygons and care is taken to avoid the overlapping of segments for any classes as these might raise a problem during instance segmentation predictions as panoptic segmentation format expects non-overlapping instance segments. Figure 4.3 shows the RGB images and their labelme annotations for visualization on the RPTU-forest dataset.

RGB Images

Ground Truth

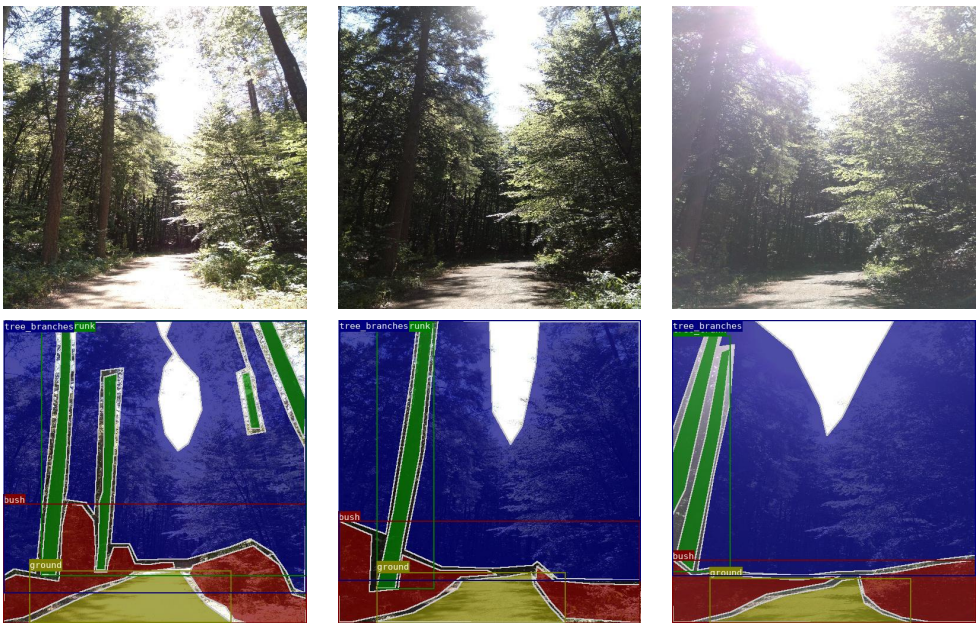


Table 4.3: RPTU-Forest dataset sample RGB images and their ground-truth annotation visualizations using labelme

Panopticapi is a tool used to convert and extract different formats of ground-truth data into desired COCO formats for the tasks of semantic and instance segmentation training

as well as to extract the panoptic segmentation ground truth annotations from these two segmentation annotations using different scripts. First, convert the COCO-detection format annotations to semantic segmentation annotations and also from this COCO-detection format to COCO-Panoptic format annotations and extract the instance masks from this JSON COCO-detection format annotations. Finally, use all the extracted annotations to train the panoptic segmentation network architecture.

The expected format for the panoptic segmentation task for COCO dataset format data is shown in figure 4.3. for each image annotation created in labelme in JSON format, resembled per-image annotation in panoptic segmentation rather than per-object annotation and each annotation consists of a PNG format representation of class-wise segmentation and a JSON annotation format that contains the information about segmentation for each segment of the image. As shown in the figure 4.3, to match the annotation data with the image, we use the image_id field of the annotation category inside the JSON representation, and for each annotation, per-pixel wise segments information is stored in file_name using a single PNG file and each of these segments is assigned a unique identifier and assigned the unlabeled of these pixels with the value of 0. similarly per-segment wise information is stored and classified as either thing class or stuff class.

```
annotation{
  "image_id"          : int,
  "file_name"         : str,
  "segments_info"     : [segment_info],
}

segment_info{
  "id"                : int,
  "category_id"       : int,
  "area"              : int,
  "bbox"              : [x,y,width,height],
  "iscrowd"           : 0 or 1,
}

categories[{
  "id"                : int,
  "name"              : str,
  "supercategory"     : str,
  "isthing"           : 0 or 1,
  "color"             : [R,G,B],
}]
```

Figure 4.3: Figure showing COCO-Panoptic data format structure

Thus the data is prepared for training the Panoptic segmentation network with the information combined concerning instances, semantic context, and class labels and their respective positions in the image we give to the network during training.

4.2 Baseline Network Architecture

We had already seen that top-down approaches hold an upper hand with the performance over the other methods for panoptic segmentation and thus we decided to go with the top-down approach of Panoptic-FPN (Kirillov 19a) for implementation of panoptic segmentation in this thesis. PanopticFPN approach consists of a simplistic baseline network architecture compared to other top-down approaches and it uses the Mask R-CNN network for instance segmentation, which is quite the fastest approach among all others and the most accurate one for the task. we will now look into the details of this network more comprehensively.

PanopticFPN exploits the architectural advantage of shared computations between instance and semantic segment branches and thus avoids the usage of two separate branches for individual tasks (Kirillov 19a). PanopticFPN uses Mask R-CNN for solving instance segmentation tasks and making a few adjustments to the shared feature extractor backbone network using a Feature pyramid network(FPN), it can be able to extract the semantic context from multiple scaled features at different levels and thus be able to do semantic segmentation task in a very light-weight manner (Kirillov 19a).

It is well-established that region-based approaches are predominantly used for instance segmentation tasks (Kirillov 19a). Fully convolutional Networks are commonly used approaches for semantic segmentation tasks (Kirillov 19a). When a combination of both approaches is tried, it leads to redundant computations and slower inference speeds (Kirillov 19a). PanopticFPN overcomes this drawback by not altering the instance segmentation branch in a region-based approach and in parallel adding a distinct branch to perform semantic segmentation (Kirillov 19a). There are no changes performed in the FPN structure while adding the semantic segmentation branch to avoid the changes in instance segmentation predictions and simultaneous training of both branches is needed to perform the unified panoptic segmentation, and to a surprise, this dense semantic prediction branch performed on a similar level as the conventional FCN-based semantic segmentation on benchmark datasets (Kirillov 19a).

Integration of a parallel semantic branch to the existing Mask R-CNN network incurs a slight additional computational cost and the usual computational cost is reduced due to non-usage of atrous convolutions in this approach (Kirillov 19a). This method is thus compatible with large backbone architectures such as ResNeXt, resulting in faster inference and training speeds (Kirillov 19a). Sample images of this network prediction on different datasets are presented below in figure 4.4 (Kirillov 19a).

4.2.1 PanopticFPN Architecture

For the panopticFPN model, the backbone feature extractor is a feature pyramid network(FPN), which is not modified as it is crucial to perform instance segmentation based

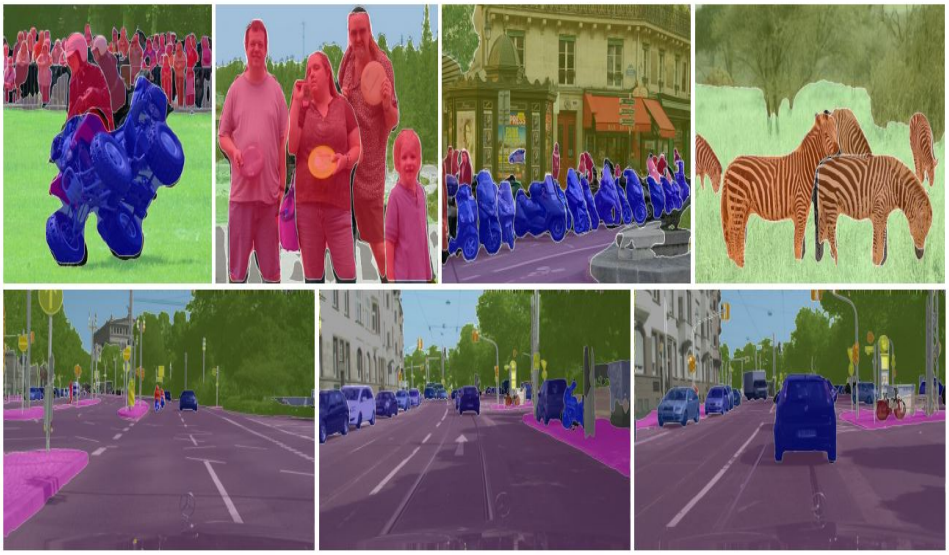


Figure 4.4: Figure showing panoptic predictions of COCO dataset(top) and Cityscapes dataset predictions(bottom) from (Kirillov 19a).

on the Mask R-CNN approach however there is an additional branch based on the decoder part of the pyramid structure which can use the other encoding part to correlate and extract semantic information as shown in figure 4.5 (Kirillov 19a).

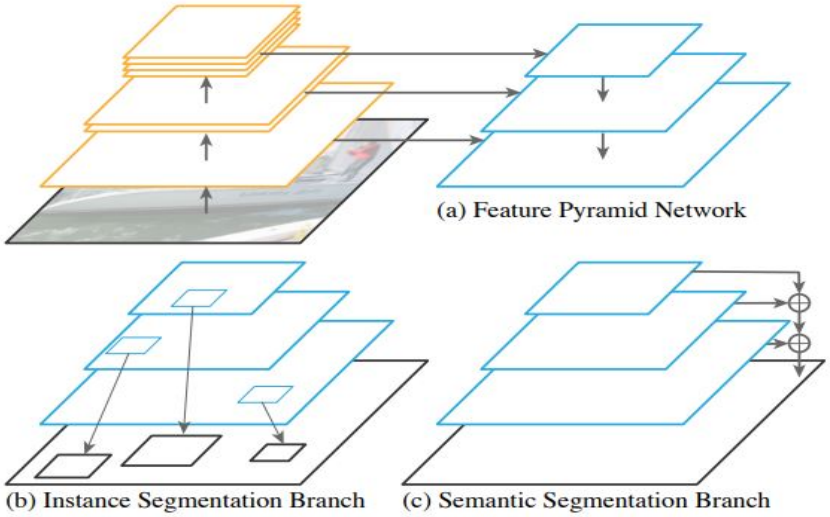


Figure 4.5: Figure showing PanopticFPN architecture blocks (a) FPN backbone for feature extraction at multiple scales (b) Mask R-CNN based on FPN for instance segmentation and (c) additional dense prediction branch for semantic segmentation. (Kirillov 19a).

As shown in figure 4.5a, the FPN extracts the features from the input image at multiple resolutions to extract spatial information and adds lateral connections in the bottom pathway, from the deepest layer of the pyramid i.e. top, the features are upsampled again until it reaches the input shapes and the scales range from 1/32 up to 1/4 resolution

while maintaining the channel dimension as 256 for every layer output (Kirillov 19a). As we maintain the same channel dimension to every pyramid level, it is very convenient to attach an existing object detector like Faster R-CNN to extract region-based proposals and predict the bounding boxes and class labels and just adding an FCN to this network to predict binary masks for each of those regions proposed using Mask R-CNN architecture and complete instance segmentation as shown in figure 4.5b (Kirillov 19a).

For generating the semantic segmentations from the features extracted using FPN, a combination of the information from multiple pyramid levels into a single value is done as shown in figure 4.6. Starting from the deepest layer at 1/32 of input resolution, upsampling is performed to bring the feature map to 1/4 resolution (Kirillov 19a). The operations of 3x3 convolution followed by group normalization, ReLU activation, and 2x upsampling using bilinear interpolation technique at each pyramid level is performed (Kirillov 19a). The resultant output has 1/4 resolution which is maintained constant for element-wise addition and finally, a 1x1 convolution followed by 4x bilinear interpolation upsampling is performed to obtain the result in the same resolution as the input (Kirillov 19a). Softmax is applied to predict the pixel-wise class labels to fulfill the semantic segmentation task and in addition, this branch outputs the 'output' class label for pixels that contain object classes (Kirillov 19a).

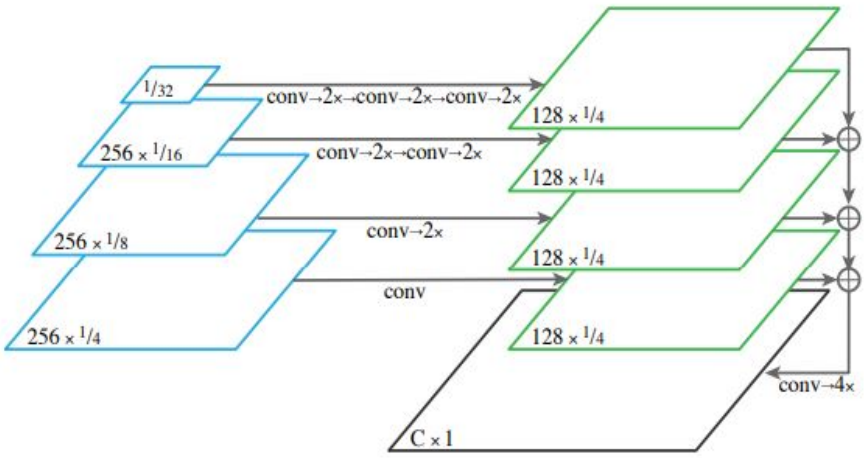


Figure 4.6: Semantic segmentation dense-prediction branch added to FPN (Kirillov 19a).

There is an earlier discussion regarding the drawbacks of top-down approaches mainly with the overlapping of object segments. For panoptic segmentation tasks, this overlapping should be avoided. To overcome this problem, in panopticFPN, a simple post-processing procedure similar to non-max suppression is implemented, and this operation works by resolving the overlaps between instance segments using the confidence scores predicted by the model (Kirillov 19a). Intra-branch conflicts are handled by favoring things over

stuff classes and removing the stuff classes from semantic segmentation that are labeled as 'others' or using a threshold for eliminating such non-important segments from the output predictions (Kirillov 19a).

For training the panopticFPN network, we need to perform training parallel both for instance segmentation and semantic segmentation branches, and these two tasks are generalized by different types of loss functions and normalization procedures, thus for joint training we need to consider loss function with some adjustments (Kirillov 19a). For instance segmentation task, we have three losses such as L_c(classification loss), L_b(Box regression loss), and L_m(Mask prediction loss) and to perform the instance segmentation the network needs to train on reducing the summation of these three losses where classification and box regression losses are normalized by the number of sampled region of interests(ROIs) and mask loss is normalized by the number of foreground Region of interests and for the semantic segmentation task, we have L_s(segmentation loss) which is calculated pixel-wise using cross-entropy loss between predictions and ground truth and is normalized by the total number of labeled pixels and as we have different schemes for normalization of these losses, simply adding them for joint training will result in degradation of model performance and incorrect learning of the model and thus the solution is to re-weighting the losses of semantic and instance segmentation as shown in equation 4.1 (Kirillov 19a). By properly adjusting these parameters of λ_i,λ_s, we can train the single network capable of doing panoptic segmentation in a unified manner with half of the computational cost and a relatively faster inference speeds to implement in real-time applications (Kirillov 19a).

$$L = \lambda_i(L_c + L_b + L_m) + \lambda_s L_s \tag{4.1}$$

4.3 Approaches to improve model predictions

Dynamic weather and lighting conditions of the environment dictate the capabilities of the sensor in an off-road environment, especially for visual perception systems like cameras (Neto 22). As highlighted earlier, the lack of diversified datasets is the main obstacle in training models for autonomous navigation in an off-road environment. The existing datasets for urban driving scenarios have proven to perform poorly in off-road settings (Neto 22).

In this work, planning is done to deal with scenarios of over and under-exposure and illumination problems. The major research has been done to tackle image shadows at first in this thesis and at a later stage realized that exposure correction is also an alternative approach to correct the artifacts in the visual representation of the scenes in our RPTU-Forest Panoptic dataset. Initially, literature research was done on available Deep-learning modern approaches to remove the shadows from the input images, and the first approach studied is called Spa-Former(Zhang 22) which is explained in detail below.

4.3.1 Shadow-Removal and exposure correction

Shadows in images often convey valuable information regarding scene lighting conditions (Zhang 22). Nevertheless, for image processing pipelines, this is a challenging factor as it imposes undesirable features into the image, and removing them requires additional effort for the model (Zhang 22). Shadow removal procedure starts initially with the detection of regions inside the image with varying lighting and replacement of those regions is done with the scene structure that is interpolated from the corresponding locality (Zhang 22). Artifacts raised during this process need to be addressed further (Zhang 22).

Shadow removal involves two main challenges: firstly, lack of relevance in ground occlusion information makes the detection of exact conditions needed for shadow removal harder (Zhang 22). Secondly, preserving the original scene structure in the output image after shadow removal (Zhang 22). In CNNs, the calculations of the operations to establish correspondences increase with the distance between regions and this makes the estimation of neighborhood pixels influence on the region of interest very challenging (Zhang 22). To address these shortcomings, a semi-supervised transformer model named SpaFormer is investigated for shadow removal task (Zhang 22).

Shadow removal networks often focus on the global information about the image while decoding the information embedded into the decoder neglecting the local details, and thus in this method, a two-wheel RNN combined with a spatial attention mechanism is used and this attention mechanism brings us the spatial information from the image local regions (Zhang 22). The conventional feature extractor ResNet blocks cannot influence low-frequency information and only concentrate on high-frequency domain information and to mitigate this, the Fourier transform Residual module is employed to observe the long and short-term dependencies and concatenate the information of high and low-frequency domains to understand the relationships between them to remove shadows (Zhang 22).

The network structure is presented for Spa-Former in figure 4.7, consists of a Transformer network and a CNN module in which features are extracted from the transformer block with a 3x3 convolution and then a bottle net followed by Two-wheel RNN spatial attention block which focuses on the specific region of the image to remove shadows (Zhang 22).

The transformer layer shown in figure 4.7, can capture both global and local correspondences at a time and works in parallel with the Two-way Spatial attention block which enables us to learn jointly features from the shadow and shadow-free images in a supervised manner to extract more textual features out of them and this module has no sequence length restrictions as RNN blocks and can easily access long-term dependencies and the process starts with taking the high-resolution input and encoder block gradually reduces the spatial dimensions with an increase in channel dimensions and the transformer block sums up the low-level encoder features with the high-level decoder features to extract rich structural

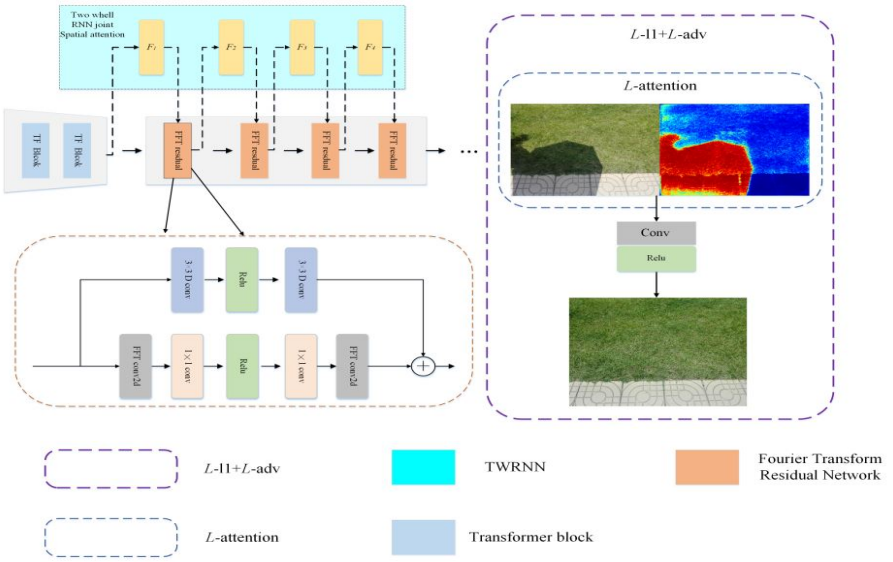


Figure 4.7: SPA-Former network architecture.(Zhang 22).

and spatial details in the image and the decoder output is further enriched by the Deep Features method introduced to work on high-resolution image inputs (Zhang 22).

The computational cost of transformers' self-attention layer is solved by using linear complexity by using channel dimensions information rather than spatial information in the images, thus generating encoding global context in a graph manner and Deep convolution uses local region information in the global attention map before the step of calculating feature covariance and using the 1x1 convolution by the transformer block for pixel-wise aggregation of channel dimensions context information and 3x3 convolutions for channel dimensions space information encoding (Zhang 22). Combining convolutions to extract the context locally is done using the concepts of key(k), and query(q). and value(v) pairs predictions and the transformations of these predictions produce an attention map with local context information needed for shadow removal (Zhang 22).

Training needed for this Spa-Former involves the paired images of Shadow and Shadow-free to learn the relationships between them and in practice for our dataset it is nearly impossible to collect the same scene with two different scenarios. It requires a lot of time as well to collect a sufficient sized dataset to train the model with such settings so we opted to neglect this approach for shadow removal.

One more shadow removal supervised approach we investigated is the ShadowFormer (Guo 23). This is based on Retinex theory for modeling the degradation of shadows and applied the physics method of image restoration from the information available from the non-shadow images to reconstruct the shadow images with shadow removal and the structure is shown in figure 4.8 (Guo 23). As per Retinex model (Porter 84), the shadow region of the image I_s is decomposed into both shadow and non-shadow regions as per

equation 4.2, where I_{ns} and I_m denote the non-shadow and mask representations and $'\odot'$ represents element-wise multiplication and the task is to interpret the shadow-free image I_{sf} from shadow image I_s after controlling the illumination and color of the shadow image (Guo 23). This is closely related to the low-light Image enhancement task in which the Image is decomposed into both illumination 'L' and Reflectance 'R' components and the model defines shadow-free image as $I_{sf} = L_{sf} \odot R$ and the Retinex based shadow removal model is presented in the equation 4.3, where the components of 'L' represents the illumination parts of the image (Guo 23).

$$I_s = I_m \odot I_s + (1 - I_m) \odot I_{ns} \tag{4.2}$$

$$I_s = I_m \odot L_s \odot R + (1 - I_m) \odot L_{ns} \odot R \tag{4.3}$$

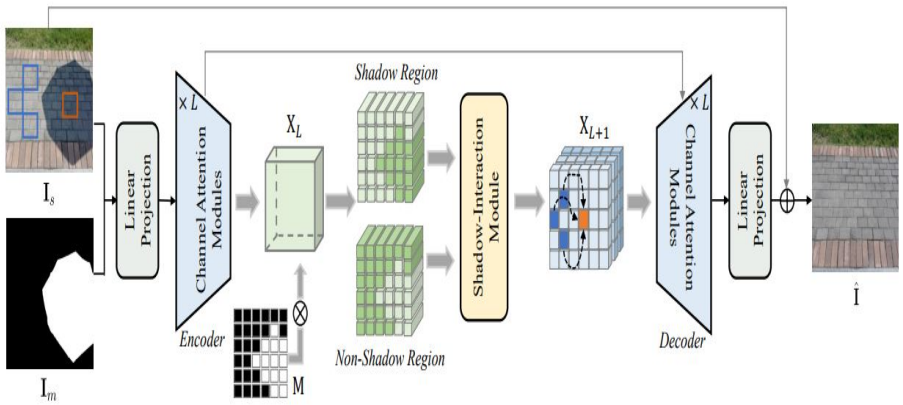


Figure 4.8: ShadowFormer network architecture.(Guo 23).

Using this model, the illumination variation in the reconstructed shadow-free image is avoided by considering the illumination component in the shadow image as well, and the reflectance 'R' components in both images are captured and thus in comparison to other approaches, the global context is captured in this method and thus effective in shadow removal process (Guo 23).

Again in this approach, we came across the conclusion that it also needs a paired dataset of images for its training, and as discussed earlier, this is highly impractical to collect such a dataset from the forest environment and thus we need to go with another approach to tackle the problem of shadow removal or image enhancement as the off-road environment is quite challenging with varying illuminations and for the predictions to be precise in such environments, image pre-processing is needed to avoid the failures of the vehicle that drives in these environments autonomously. We thus decided to investigate the unsupervised

approaches available for shadow removal as they wouldn't expect the input data to be in the paired format and we will discuss the approach MaskShadowGAN (Hu 19) now.

To avoid the tedious work of collecting and preparing the paired dataset for shadow removal, there are unsupervised approaches that translate the images from shadow to shadow-free domain and MasKShadowGAN (Hu 19) is such an approach. we can simply use the unpaired data for this approach in which the main idea is to find the hidden relationship between a set of images with shadows in shadow Domain D_s with the set of images with no shadows (shadow-free) in the shadow-free domain D_f where there is no particular similarities between both domains (Hu 19).

As it is the General Adversarial Network, there are generators and discriminators involved. The task of shadow removal is achieved from the unpaired data by training a shadow-free generator G_f that takes in the input of shadow images and tries to generate an output image which is hard for the discriminator to find out whether it is the real shadow-free image that belongs to the shadow-free domain D_f or it is a generated fake image through adversarial learning and if we only train the network in this manner, there are very little constraints on it and it might not function and therefore we should also train another generator which uses mask information and tries to generate shadow images from the previously generated shadow-free image and the end output should be same as the initial input shadow image, thus maintaining the cycle consistency on both shadow and shadow-free set of images i.e. $G_s(G_f(I_s))$ is similar to I_s and also $G_f(G_s(I_f))$ is same as I_f(shadow-free image) (Hu 19).

The problem with the above constraints is that with the same background of different shadow images, the generator of shadow-free images G_f will generate the same shadow-free image, and shadow image generator G_s will fail to generate different output images which are supposed to be same as input images as shown in figure 4.9, the cycle consistency wouldn't work and training G_f,G_s is difficult and therefore, Mask-Guided constraint is laid in which a mask is generated from the input image which can be used by the generator G_s to generate shadow images to maintain still the cycle-consistency constraint during training (Hu 19). If we have a shadow-free image, using different shadow masks we can use the generator G_s to generate different shadow images, and in the case of shadow images, as discussed in cycle constraint the process is the same and thus we can train the network again for shadow removal (Hu 19).

The overall learning outputs of this approach are presented in figure 4.10, which consists of a learning approach from both shadow and shadow-free images in which G_f,G_s are the generators of shadow-free and shadow images and D_f,D_s are discriminators to distinguish whether the generated images are real or fake and I_s,I_f are the real shadow and shadow-free images and $\widetilde{I}_s$ and $\widetilde{I}_s^i$ denotes the shadow images generated and $\widetilde{I}_f$ and $\widetilde{I}_f^i$ denotes the shadow-free generated images and M_n,M_l and M_r are shadow masks in the figure 4.10 (Hu 19).

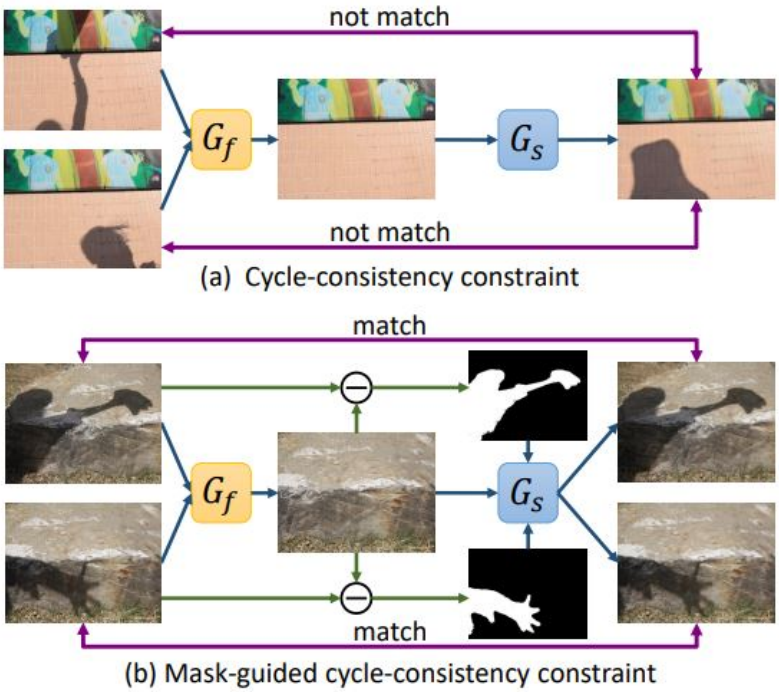


Figure 4.9: Figure showing the cycle-consistency and Mask-guided Cycle consistency constraints from (Hu 19)

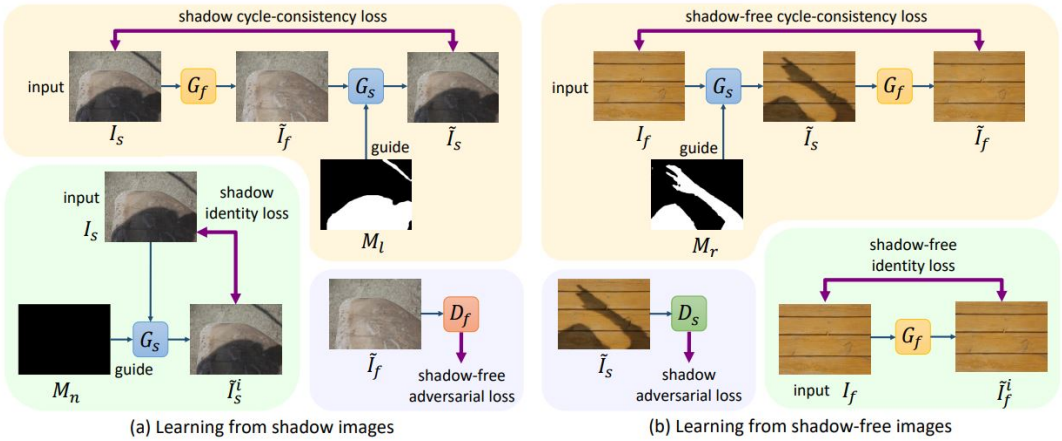


Figure 4.10: Figure showing the architecture of Mask-ShadowGAN with all the losses used for learning to remove shadows from shadow images. (Hu 19)

Shadow Masks are generated using the concept of thresholding i.e. if the pixel value is greater than the threshold value, then it is set to the value of 1, or else it is made 0 and thus we obtain the Mask by measuring the difference between a generated shadow-free and a real image from shadow domain I_s and then binarizing the output values using the thresholding the resultant mask is denoted as $M = B(\tilde{I}_f - I_s, t)$ where t is the threshold value (Hu 19).

for this work, we trained this network on different types of datasets and a combination of them to make this network work on our forest dataset and the detailed results are presented in the next chapter along with visual outputs. Training this network also has some limitations as in general, we know that training GANs is a challenging task and time-consuming due to multiple iterations taken by the discriminator to function properly. The outputs we received through this training generated decent outputs but were not sufficient for our environment as they imposed artifacts on the output images due to illumination variations being huge in our data and thus we planned to finalize the exposure correction rather than this shadow removal using the GANs approach.

4.3.2 Exposure correction

Extreme lighting conditions affect the machine learning or computer-vision-based models with degraded performance. The input images should be well processed before passing to the model with the natural light conditions. Progressive Self-Enhancement Network (PSENet) is an unsupervised approach that generates the pseudo-ground-truth images from a set of sequence of images to avoid the necessity of providing well-exposed images as ground-truth data and thus the easiest one to gather the input dataset for its enhancement network training (Nguyen 23a).

Input images should be of high-quality features such as color, contrast, and texture details but are often harder to collect due to harsh weather conditions or backlighting and this leads to over or under-exposure conditions in the images and these exposure differences will affect in turn the tasks such as instance segmentation or object detection in particular (Nguyen 23a). Many recent approaches are available to handle low-light conditions, especially under-exposed image enhancement but PSENet is theoretically designed to handle the over-exposure (Nguyen 23a), which is the most common exposure scenario in our RPTU-Forest dataset due to the daylight conditions in which the dataset is recorded.

Upon training with well-exposed Ground truth(GT) images, supervised learning approaches have proven to yield superior exposure correction performance (Nguyen 23a). However, obtaining such well-exposed images is a formidable task (Nguyen 23a). Consequently, the unsupervised approach of PSENet is opted, eliminating the necessity of such GT images during training (Nguyen 23a). This approach utilizes the dynamically generated pseudo-GT images from the set of input ill-exposed images and uses these pseudo-GT images as training input (Nguyen 23a). These pseudo-GT images progressively evolve by selecting the visually pleasing images of enhancement networks' previous epoch outputs. Darker and brighter images are generated for the initial input by adjusting gamma values in the prior epoch (Nguyen 23a). Thus the output from the current epoch is always equal to or better than the previous epoch's output (Nguyen 23a).

The goal of this network is to take a sample RGB image I, which is captured in a harsh lighting condition that has low contrast and faded-out details, then enhance it to a more visually nice looking image Y, with good contrast and color details in an unsupervised manner without any need for additional GT data and this pipeline has multiple modules involved as shown in figure 4.12 (Nguyen 23a).

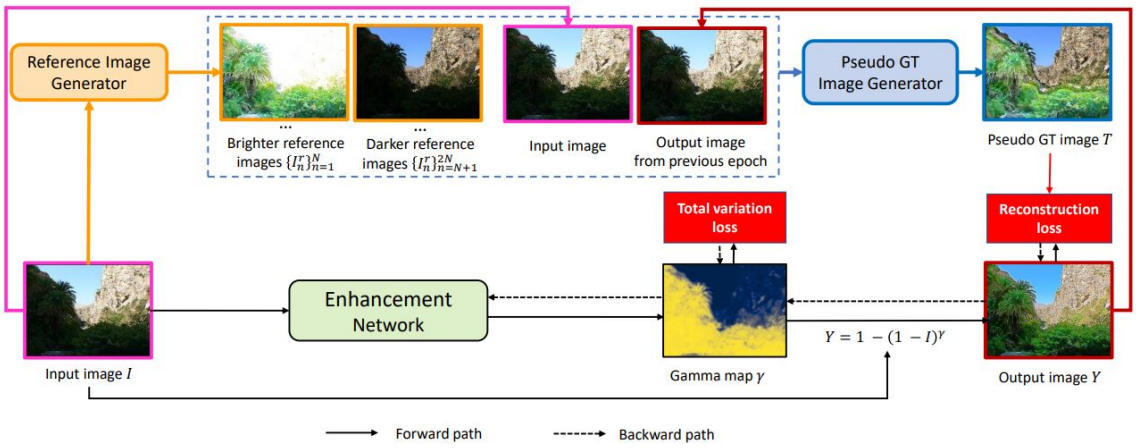


Figure 4.11: PSENet structure with main modules as reference image generator, pseudo-GT generator, and enhancement network. (Nguyen 23a)

The training of enhancement network works in a self-supervised manner in which a synthesized input is used which is generated from a set of images from which the model synthesizes the GT image based on the visual appeal and this network only uses pseudo-GT data for training that is synthesized from a large group of ill-exposed images rather than generating the pseudo-GT images from GT well-exposure images and the GT is progressively improved with more epochs as the best image output is chosen as a reference image to generate new synthetic GT for the current epoch and the module reference Image generator takes in an ill-exposed image and generates a couple of 2N images in which N images are darker and the N number of brighter images than the input image and also the output from the previous epoch to generate the pseudo-GT image (Nguyen 23a).

Inside the reference Image generator, to mimic the exposure problems out of sample inputs, the gamma mapping function is applied on the inverted image of input as inverted images share the same properties of low dynamic range and noise levels as poor lightning images which we are dealing with and as gamma function is widely used in image contrast enhancement networks due to its relationship with the human visual system and it is applied on the input inverted image to generate 2N set of reference images Y_n as shown in equation 4.4, where γ_n is just a random number and $X_n = log(\gamma_n)$ is sampled differently for under and over exposed reference images (Nguyen 23a).

$$Y_n = 1 - (1 - I)^{\gamma_n} \tag{4.4}$$

The Pseudo GT image generator module is responsible for taking in reference Images 2N generated from the reference image generator module, the actual sample input image, and also the output image of the previous epoch of the enhancement network and generating a single pseudo-GT image after comparing them and combine them into a single image and this combining concept is based on measuring the perceptual quality of every pixel on the set of a reference image sequence as they carry useful enhancement features like brightness, contrast, and saturation and this information can be used to generate a rich high dynamic range image as an output (Nguyen 23a). The combination procedure of all the sets of images is based on the best regions from those images and weightage is given as per the enhancement features like well-exposedness, which describes how much closer our pixel is to the well-exposed zone, Local Contrast, and Color Saturation for PSENet (Nguyen 23a).

The losses involved in training this network are reconstruction loss(L_{rec}) and total variation loss (L_{tv}) where reconstruction loss is the mean squared error computed between the prediction from the network and the pseudo-GT module output and variation loss is the smoothness we apply to image restoration on the predicted gamma map and the total loss for training PSENet is $L_{rec} + \alpha L_{tv}$ where $'\alpha'$ is the balancing weighing term between two losses (Nguyen 23a).

we had trained this PSENet on different types of data available to us for it to work effectively on the RPTU-Forest dataset images. We implemented it for this thesis for the image enhancement task which surprisingly has made the panoptic segmentation results predictions better and also the input images are visually appealing due to darker regions being reduced in the image. The overall image is enhanced which is beneficial, especially for our application area. The results of the training and inference are presented in the next chapter where we will visually observe how the image enhancement is better for the panoptic segmentation task.

4.4 Training

For training the main model of panoptic segmentation, we have chosen to follow the detectron2 (Wu 19) approach for training the PanopticFPN module as discussed earlier as our chosen top-down method for this work. Detectron2(an object detection library developed by the Facebook AI research group) is built upon the Mask R-CNN benchmark and is implemented in PyTorch with a modular build that helps us with the object detection and image segmentation tasks (AI 19). It is also an easier way to implement training even on a single GPU server with fewer memory constraints and high performance (AI 19).

Detectron2 uses PyTorch models of DeepLearning by default and this benefits our work (Wu 19). It is modular in design, we can add custom data to the existing framework of

models with minimal scripts, and as our training involves a custom RPTU-Forest panoptic dataset which needs to be processed into the default format of COCO-Panoptic to work, Detectron2 is a very useful approach to make our work simpler. Also, Detectron2 has a large database of pre-trained models available which can be useful to us in training our panopticFPN model on a large dataset before training on our data to make its performance better.(Wu 19).

For Detectron2 to load our processed ground-truth annotated data, it needs to be converted into separate modules to be fed directly to the network and we will look into the actual way to perform it now. For any object detection model inside the Detectron2, it takes the basic configuration of Faster-RCNN (BASE-RCNN-FPN), from which we can modify our configuration based on our application and here in the base configuration, ground truth data such as images and labeled annotations need to be prepared and they are used by the Region Proposal Network(RPN) and the box head as shown in the figure 4.12 (Honda 20).

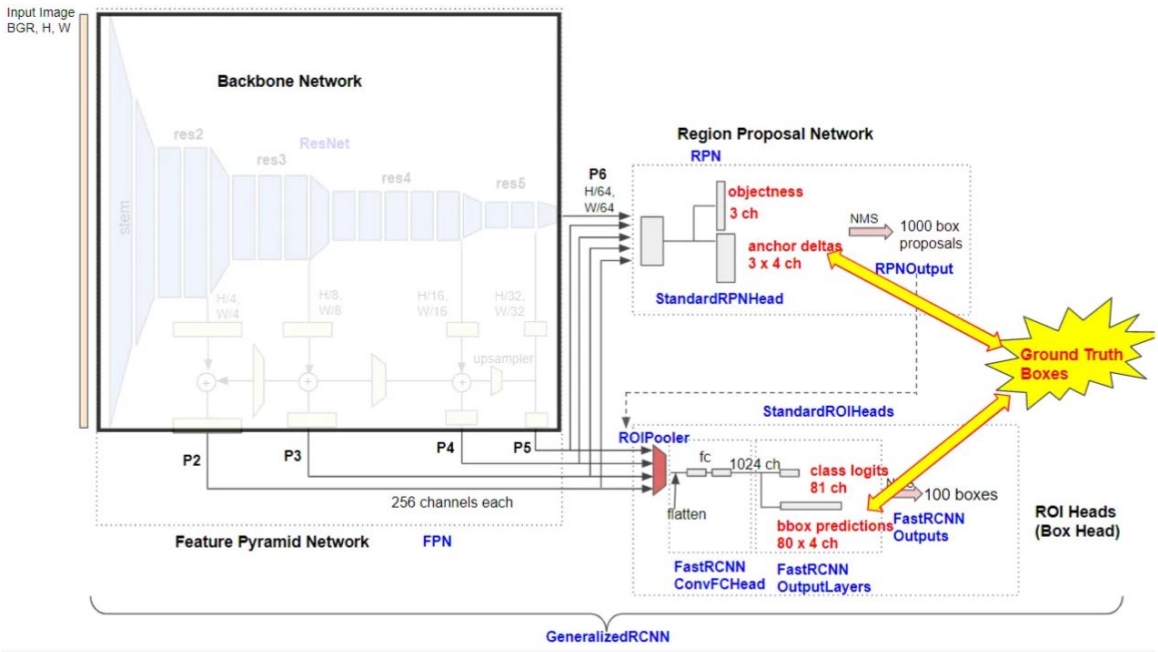


Figure 4.12: Groundtruth data used by the RPN and box head modules. (Honda 20)

Annotated data in any object detection task consists essentially of Bounding box labels such as coordinates of the box locations surrounding the object in the image and a category label such as instance-id details (Honda 20). Region proposal network (RPN) usually doesn't learn the task of classification and this annotated category data is only used by the Region of Interest Heads (RoI Heads) (Honda 20). This data is directly loaded from the annotation file we provide to the model (Honda 20).

The data loader of Detectron2 is a multi-layered module that is built on Pytorch using a build module by following steps: Use the dataset names defined in the config file to query

class: DatasetCatalog, and return a list of dictionaries as the output and use parallel computing of workers to work on the dicts. Each worker will map each metadata dict into the model-accepted format and generate the batch-wise data by making the dicts into lists. The batches of that list of dictionaries are the end result of the data loader. The datasets_dicts is a list of the annotation ground truth we give as a dataset and this dict is converted into a torch format dataset structure with DatasetFromList function in the script and DatasetMapper is another utility function using which mapping of each annotation to the object instances for the model to recognize and the whole data loader structure looks like in the figure 4.13 (Honda 20).

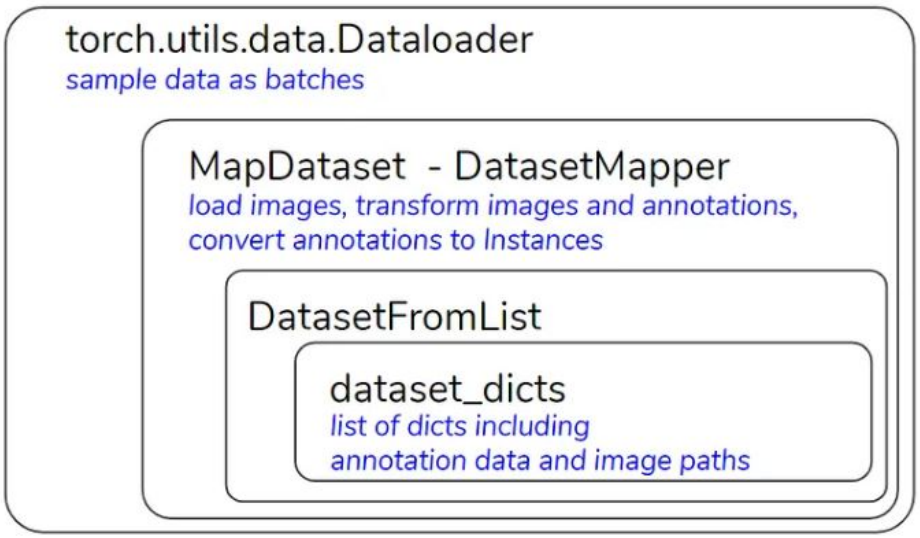


Figure 4.13: Data Loader from Detectron2. (Honda 20)

During the process of training, annotation data is to be provided for the model in JSON format and the corresponding images as the input and the DatasetMapper function maps these inputs together for the model to learn by loading and augmenting the images and the resultant dict structure is represented in the figure 4.14 (Honda 20).

With the provided inputs, the model can now learn the associations between the image and the annotations provided. Training is performed to reduce the pre-defined loss functions for this task. For our work, there is an additional task of mapping stuff and things classes to instances that have to be performed. This is performed in the training script for the model to learn the associations perfectly.

For our model training, we need to define a categories.json file which defines the list of categories defined while annotating our dataset and describes whether the segment belongs to a stuff or things class and a sample category file looks like in the figure 4.15.

we need to define our dataset with a name before training and load the ground truth data in corresponding formats as RGB image data in a folder and the converted formats

```
{'file_name': 'imagedata_1.jpg',
'height': 640, 'width': 640, 'image_id': 0,
'image': tensor([[[255., 255., 255.,  ...,  29.,  34.,  36.],...
[169., 163., 162.,  ...,  44.,  44.,  45.]]]),
'instances': {
'gt_boxes': Boxes(tensor([[100.55, 180.24, 114.63, 103.01],
[180.58, 162.66, 204.78, 180.95]])),
'gt_classes': tensor([9, 9]),
}
```

Figure 4.14: example of a dataset_dict returned through a data loader in Detectron2. (Honda 20)


```
[{"supercategory": "", "color": [57, 226, 241],"isthing": 0, "id": 0, "name": "_background_"},
 {"supercategory": "", "color": [247, 60, 123], "isthing": 0, "id": 1, "name": "ground"},
 {"supercategory": "", "color": [102, 245, 189], "isthing": 0, "id": 2, "name": "bush"},
 {"supercategory": "",  "color": [79, 141, 229], "isthing": 0, "id": 3, "name": "tree_branches"},
 {"supercategory": "",  "color": [180, 22, 44],"isthing": 1, "id": 4, "name": "tree_trunk"}
]
```


Figure 4.15: Sample categories file defined for training PanopticFPN using Detectron2.

of coco-panoptic data of our dataset. Using Panopticapi converters, we converted our annotation JSON files into panoptic JSON which contains the mask data in coco-panoptic format as shown in figure 4.16 and also semantic masks for semantic head module and instances location data in a JSON file. These ground truth masks are then saved into a metadata dictionary inside the Detectron2 datasets library. The next step is to segregate the stuff and things class objects from the category file, assign an identifier for all these classes, and update this information in metadata. Additional training input parameters are provided which are tailored to our custom dataset such as the number of classes present.

Thus the whole pipeline for training of PanopticFPN model is created using Detectron2 utilities and a custom trainer is used to train with the above specifications on our dataset using Multiple GPU servers for some 1000 iterations and tested again on our dataset sample images and the results are detailed in the next chapter with the loss curves we got during training and the visualizations of the dataset stored inside the detectron2 metadata format.

As discussed earlier, optimizations are performed to improve the performance of this panoptic segmentation model in the form of PSENet-based image exposure correction and this trained model is integrated with the panoptic model in the inference stage where the trained models of both tasks are combined in a parallel fashion. First, the input image is fed to the model of PSENet, the output of which is used to draw the panoptic predictions as shown in figure 4.17.

Figure 4.16: sample RGB image and its panoptic segmentation mask given as input during training of PanopticFPN model

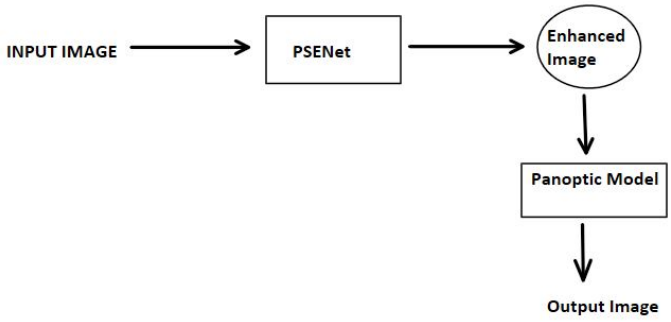


Figure 4.17: Inferene pipeline of this panoptic segmentation.

4.5 Evaluation Metrics for Panoptic Segmentation

As we know already panoptic segmentation is a joint task of semantic and instance segmentation and individual task metrics for the evaluation of combined tasks may not

be effective as the task-specific metrics are designed in such a way only to measure the performance of the model in predicting either stuff or things category identifications (Kirillov 19b). But the metric for Panoptic Segmentation should fulfill the criteria such as it should treat both stuff and things classes in a unified manner, should be easily interpretable in conveying its meaning, and should be simple to implement (Kirillov 19b). The semantic segmentation task is evaluated using the IOU score and Per-pixel accuracy of classification and the instance segmentation is evaluated usually using Average precision (AP) over different IOU threshold scores, mean of these AP as mAP scores, and combining both task metrics is not meaningful.

We need to define a unique metric for our task of Panoptic Segmentation and that is defined in (Kirillov 19b) as Panopitc Quality metric(PQ) which measures the predicted panoptic segment quality to that of our ground truth annotation and the calculation of this metric is done in two stages as segment matching and PQ computation with the matches which are discussed in detail below.

4.5.1 Segment matching

To understand this segment matching, we first need to look into the Intersection over union (IoU) score classification metric. To measure the quality of the output of the object detection task, we define the IoU as in the equation 4.5 (Mechea 19). By dividing the intersection area with the union area of prediction and ground truth, a result in the range of 0-1 is achieved (Mechea 19). The value 1 represents the perfect alignment of GT and prediction and in contrast, 0 represents the perfect misalignment (Mechea 19).

$$\text{IoU} = \frac{\text{prediction} \cap \text{groundtruth}}{\text{prediction} \cup \text{groundtruth}} \tag{4.5}$$

This IoU calculation is for the usual bounding box involving segmentation tasks and the same concept can be used for segments as well in the task of Panoptic Segmentation. In general, a true positive is a sample that has an IoU threshold of 0.5 or greater. Accordingly, if we combine the Panoptic Segmentation condition of non-overlapping segments, we will obtain a unique matching criterion which poses a condition that there should be at most one matching segment predicted per each GT segment (Kirillov 19b). The diagrammatic representation of these segment-wise classifications of objects in the image is shown in the figure 4.18. In figure 4.18, we can see how the person class is categorized into True Positives(TP) if the matching is positive and False Negatives(FN), and False positives(FP)if there are mismatches between GT and predictions of Panoptic Segmentation network.

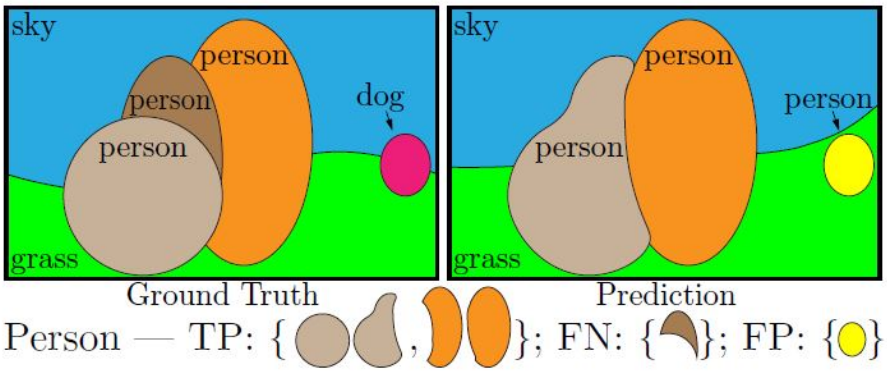


Figure 4.18: Figure showing predicted and GT Panoptic segments of an image and IoU score of greater than 0.5 is considered Positive prediction and therefore matched.(Kirillov 19b)

4.5.2 PQ Computation given matches

There are normally class imbalances in any large dataset and it is a bigger problem of bias in modern Deep Learning algorithms. Here to make our evaluation metric insensitive to such imbalances, we calculate our PQ metric for each class separately and calculate the mean of all classes in the end (Kirillov 19b). As described in the above section, a unique matching strategy classifies the image segments into TP, FP, and FN which denotes the matched segments, unmatched predictions, and unmatched GT segments as shown in figure **??**, and with these segments, PQ is denoted as in the equation 4.6 (Kirillov 19b).

$$\mathrm{PQ} = \frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \tag{4.6}$$

As seen from the equation 4.6, PQ is another simple representation of average IoU scores of matched segments as $\frac{1}{|TP|}\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)$, while $\frac{1}{2}|FP| + \frac{1}{2}|FN|$ is an addition term in the denominator for dealing of unmatched GT and unmatched panoptic segments from the model and here every prediction of the model is treated with equal importance irrespective of how much area they occupy and if we multiply and divide the above equation with TP then the resultant PQ equation can be represented as a multiplication of two important segment quality metrics terms as Segmentation Quality (SQ) and recognition Quality (RQ) as in the equation 4.7 (Kirillov 19b).

$$\mathrm{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}}. \tag{4.7}$$

We can observe from the above equation 4.7, that RQ is simply the F1 score which is a harmonic mean of precision(P) and Recall(R) denoted as $F1 = 2PR/P + R$ which is a popular quality estimate in object detection tasks (Kirillov 19b). SQ is the mean of IoU of

matched segments and the overall PQ is decomposed into SQ and RQ for better analysis (Kirillov 19b). Thus we define a unique metric for Panoptic Segmentation which is easily interpretable, conveys meaning, and is simple to implement(Kirillov 19b). Furthermore, we need to consider the void labels in our dataset and their effect on this metric (Kirillov 19b). As void labels may arise due to out-of-class context or invalid pixels in the image and for simplicity, we usually ignore such labels in the calculations of IoU scores and also in the unmatched segments to avoid the wrong calculations due to these unambiguous labels (Kirillov 19b).

To avoid confusion between object boundaries when they are not recognizable easily, annotate them as group labels, this labeling technique may also lead to wrong evaluation often, and before matching, these labels are also neglected for predictions and if still in the predictions, these exist then using an IoU threshold, these labels are removed from our predictions (Kirillov 19b). Thus PQ metric takes into account all stuff and things classes independently and thus a suitable metric for measuring the quality of a panoptic segmentation task as required.

In the next chapter, we will look into the detailed results that came out during training multiple approaches and also the model predictions of the Panoptic Segmentation network as well as PSENet after rigorous training on multiple datasets and inference on our own RPTU-Forest Panoptic Dataset.

5. Experiments and Results

In this chapter, we will look into the detailed outcomes of the approaches we have discussed in the previous chapter along with the different experiments conducted as a part of training and testing those approaches which resulted in achieving our main goal. The results of segmentation and the optimizations done are analyzed both qualitatively and quantitatively which are presented below. Several experiments on different dataset usage and their results are presented in detail in later parts of this chapter. The visual results of ground truth annotations are also presented for a better understanding.

5.1 Qualitative Performance

we will look into the results in three categories: Shadow Removal, Image exposure correction, and panoptic segmentation, and further dive into individual task outcomes in detail. we will start our explanation with the task of shadow removal from the input images as a pre-processing attempt we have tried to better the performance of the main segmentation network.

5.1.1 Shadow-Removal

As discussed in section 3.4, there are very few Off-Road datasets available to train our Panoptic Segmentation model on them as most of the modern research is concentrated on the Urban environments with traffic scenes as the focus area for e.g. KITTI, CityScapes, NyScenes, Waymo Open Dataset. Due to the availability of less data, we planned to create our own dataset for training and testing purposes along with other datasets that resemble our use case. But we intend to tackle the abnormal environmental effects on the dataset we collected such as often these forest environments images are darker with a major portion of the image covered in shadows or visually darker in appearance due

to the natural self, casting of shadows from trees, bushes which will hinder our model to recognize the driveable area available and thus cause our vehicle to face difficulties in navigation during autonomous operation.

Thus, we investigated the modern approaches that are being used in modern research to tackle this problem of shadows and low-exposure that are common in Off-Road environments. As discussed in the previous chapter, we started with supervised approaches and our first network is Spa-Former, the training of which needs the ground truth data in the form of paired images of shadows and shadow-free. There are some datasets available in pairs that are deliberately collected for this task of shadow removal and one such prominent dataset is the Image Shadow Triplets dataset (ISTD) with paired images that contain shadows, shadow-free and a binary mask of the shadow portion of the image pairs and some sample images of this dataset looks like in the figure 5.1. It was later observed that the usage of this dataset to train our Spa-Former network for shadow removal and using this trained checkpoint, testing the performance of the trained model on some sample forest images from the internet, and the predictions of those are presented in the table 5.1, where we can observe that the dark regions of the image that represents shadows are being replaced with brighter patches.

Later we tested the model performance on the most related dataset(Freiburg-Forest) that mimics our own RPTU-Forest panoptic dataset in the similarity of nature of the scenes and the results have proven to make the image regions brighter and make them visually more pleasing and as shown in the table 5.2. It can be observed that several artifacts are appearing on the shadow removed image prediction on the Freiburg forest test environments as the data is of a completely different domain image than the ISTD dataset images and thus the model cannot generalize well on the new environment scenes and is unable to learn the features precisely and thus we decided to proceed to experiment with another popular approach of Shadow-Former to remove the shadows from the input images before passing them onto the panoptic segmentation network for predictions.

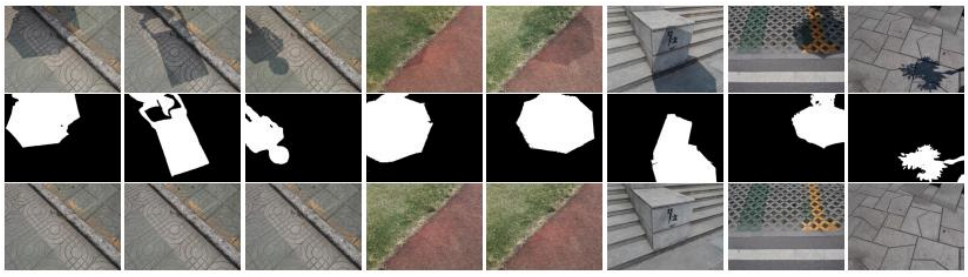


Figure 5.1: Sample ISTD dataset images

Similarly, testing of the ShadowFormer network is done and the results are quite similar to that of SPAFormer, which can be observed below in the table 5.2, which displays the inference results on the Freiburg Forest dataset. Both the networks are trained on the

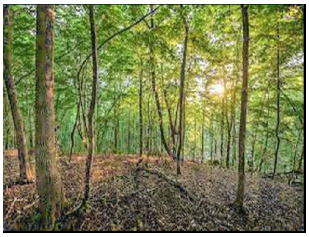
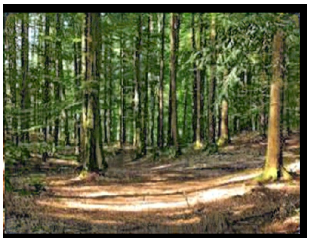
Original Images			
SpaFormer Images			

Table 5.1: Inference results of SPA-Former on random test images of the forest environment from the ISTD dataset. The top row shows original shadow images, and the bottom row shows the shadows removed image predictions by the model.

Original Images			
ShadowFormer Images			

Table 5.2: Freiburg forest test images inference results of Shadow-Former model trained on ISTD dataset. The top row shows original shadow images and the bottom row of images shows the shadows removed image predictions from the model.

paired dataset of ISTD and therefore could not generalize well enough to the new scenes of the Freiburg forest dataset. There are visual artifacts on most parts of the predictions that appear to be unnatural. Thus further research is done on unsupervised approaches that do not require this paired dataset to train the network. Thus experimentation is done on the GAN approach of MaskShadowGAN, which was assumed to be well-suited for our application. As it is an unsupervised approach, there is no need to provide the model with paired data from which the shadow mask acts as supervision and thus this approach is a step better than the previous approaches.

The MaskShadowGAN network requires input in the form of a set of images of shadows and shadow-free for training. The network then can automatically estimate the underlying relation between these two sets and predicts the shadow-free from the shadowed images at the inference stage. An analysis is done on the feature maps learned by the generator in the main blocks of its network architecture and the feature maps visualizations helped us in exactly visualizing where the network is concentrating to learn the features. One such visualization for main blocks looks like in the figure 5.2 and it can be observed that the last layers of the generator block are where the high-level features are concentrated and the same procedure was implemented on the discriminator block as well.

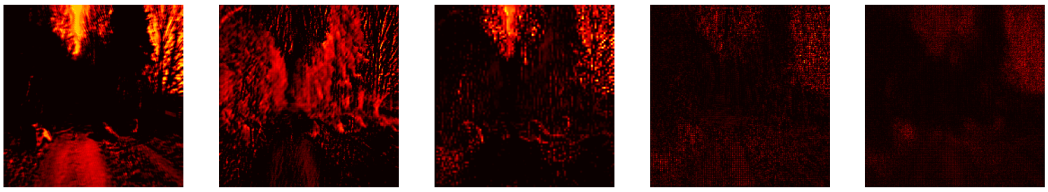


Figure 5.2: Feature Maps visualization of MaskShadowGAN Generator module performed after convolution block, downsampling, residual block, upsampling block, output layer.

In the initial training, we opted for the ISTD dataset for training the MaskShadowGAN network as it has already grouped its data into shadow and shadow-free pairs. It is a well-established paired dataset designed for shadow removal tasks. Training is performed on the ISTD dataset for 100 epochs and the loss trend curves are represented in figure 5.3. It is observed that the loss trend in the generator module is gradually reducing with an increasing number of epochs and for the discriminator modules, the trend is increasing. This is a normal trend of the loss curves for a General adversarial network (GAN) which resembles that the network is learning from the training data as generator loss is reduced which means it can be able to generate more realistic images. The discriminator loss curve also supports the learning trend by increasing the loss gradually resembling it can be able

to discriminate between real and fake samples of images. Results of inference of ISTD test images and also on our dataset as shown in the figure 5.6.

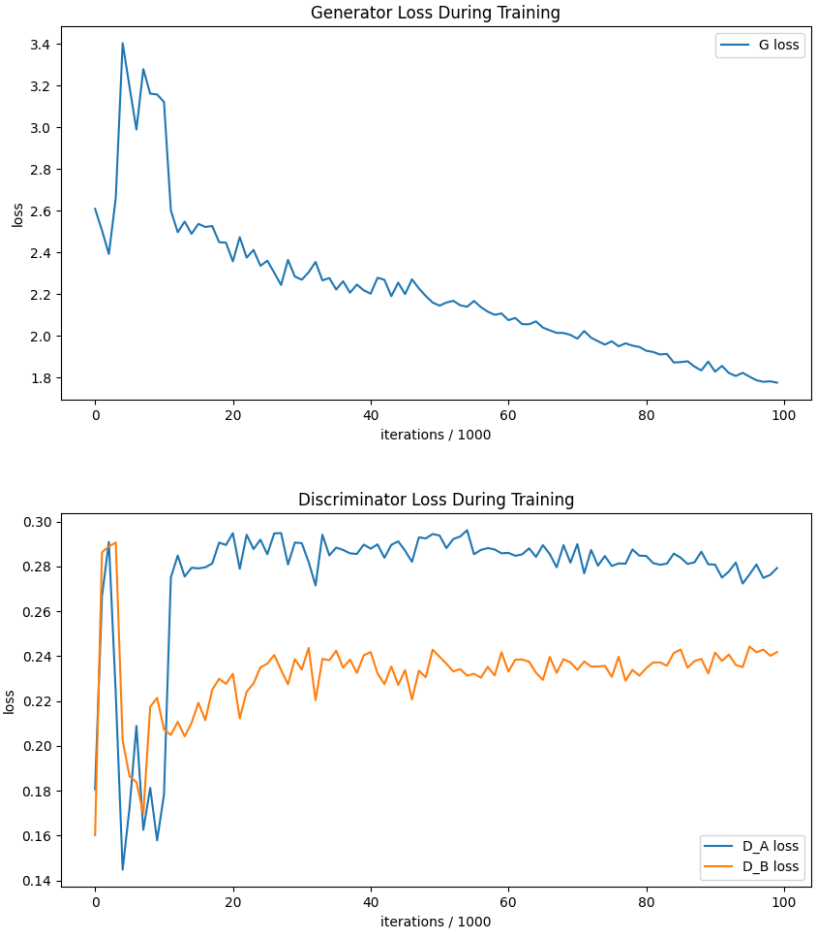


Figure 5.3: Training loss curves of ISTD Dataset on MaskShadowGANs Generator and discriminator

It is observable from the figure 5.4, that the GAN network couldn't remove shadows properly even on the test set of the ISTD dataset as the network could not learn the associations between the images until that point of training. Therefore we decided to add more data to the existing single paired dataset with other paired datasets like the Shadow Removal Dataset(SRD), and Unpaired Shadow Removal(USR) dataset into a single dataset of two folders with shadows and shadow-free image sets as input. Training is again performed on MaskShadowGAN for another 100 epochs on this large dataset of existing paired images and the training loss trends are satisfactory. However, the results of these predictions are still not visibly clearer and samples of these figures are shown in the table 5.3. It can be observed that the results are getting better on the relevant dataset type of environment images we used for training but not on the forest environment scenes and thus we need to add samples of forest environments for training as our end application concerns the forest off-road environments.

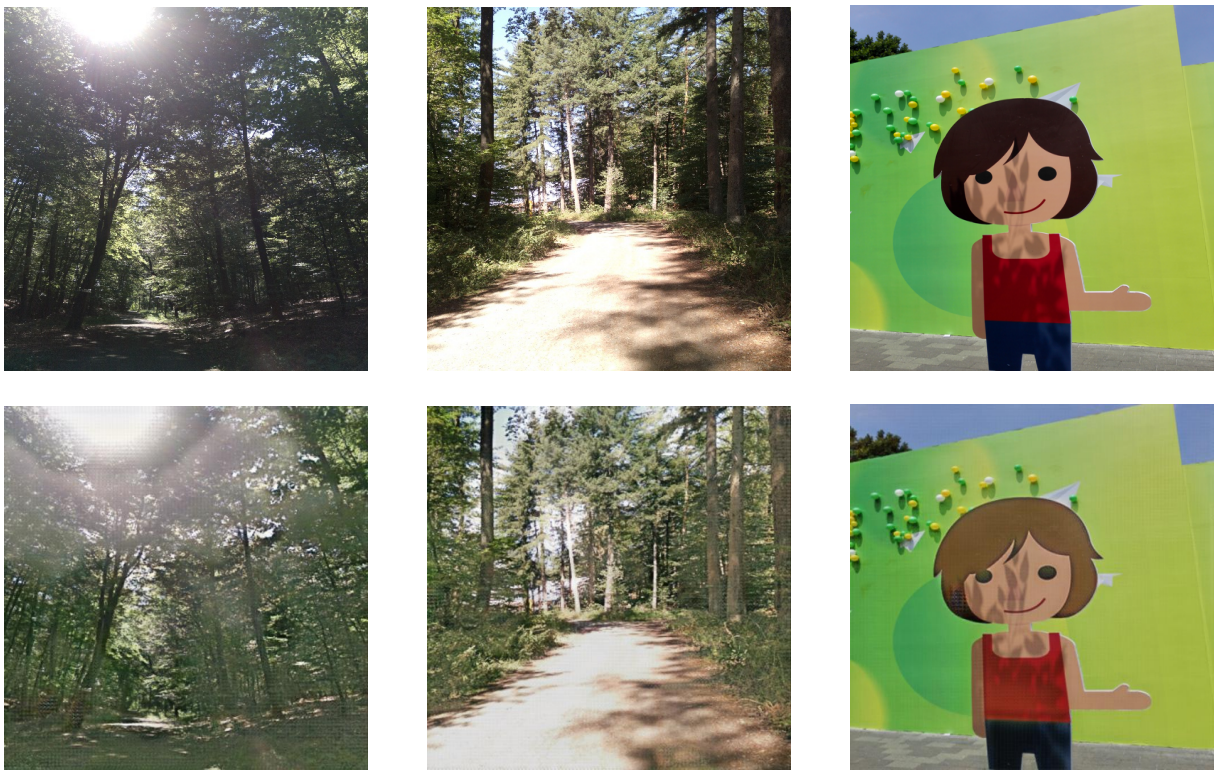


Figure 5.4: Test inference results of MaskShadowGAN model trained on ISTD dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model.

Further, the training of MaskShadowGAN is performed on the Freiburg Forest dataset from scratch for 200 epochs. The training dataset consists of 250 shadow ground truth images along with 175 shadow-free image samples for training the network. Resizing operation is performed on these images to a resolution of 512*512 for simplicity and to maintain a common aspect ratio that has been maintained while testing other datasets on MaskShadowGAN.

After training, we performed inference on our RPTU-forest dataset samples and the results are better in comparison to other training approaches and are shown in table 5.4. we can observe that there is some contrast difference in the output as the model is learning from the Freiburg forest dataset features in which this type of contrast is clearly visible and we don't want to impose this type of contrast in our final predictions and thus we decided to create a smaller dataset of our own RPTU-forest dataset for training the MaskShadowGAN from the beginning again. we need to separate the images in our dataset into sets of images with shadows and shadow-free in them and then train the model for 100 epochs again.

The training times are very high for this network and the training speeds are very low as it was designed to run on batch size 1 and could not be altered as the parameters it learns are very high and require huge computational resources as well. For every epoch, on a

single GPU, it took almost nearly 20 minutes to complete one iteration, and depending on the dataset chosen for training, this time went longer than the usual time. The final results after training on the RPTU-forest dataset are quite satisfactory however, there is a chance that due to the smaller dataset size of 106 shadow images and 68 shadow-free images, there is a high chance of over-fitting. In the case of our dataset, it is challenging to visually access the images as shadowed and shadow-free images and thus in turn affected the results.

Although the training is performed, the model still could not generalize well to the trained environment due to the absence of variability in the training data, and also the size of the dataset is very less. These factors made a significant impact on the results of the inference and are unable to tackle the over-exposure of the sky region in the images and shadows sections after removing some visual artifacts such as blur or patches of high-intensity lightning on them which are seen in table 5.5. It can be observed that in the sky region in the image samples presented, there are glares and artifacts observable and it cannot handle over-exposure of the images properly. There are only slight changes in the darker regions of the image even with this approach. Thus it is not well suited to handle the exposure variations of the application area.

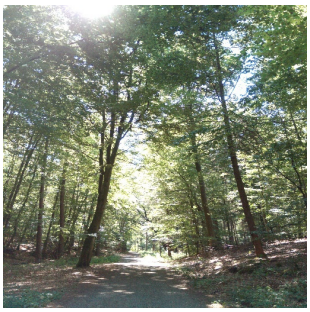
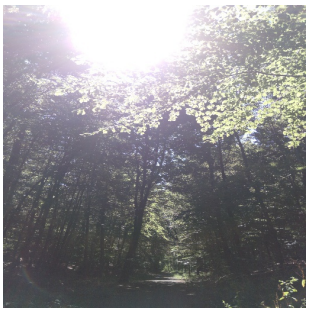
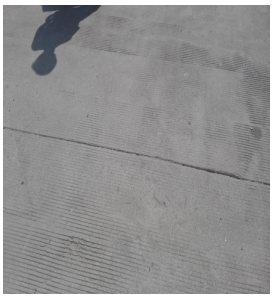
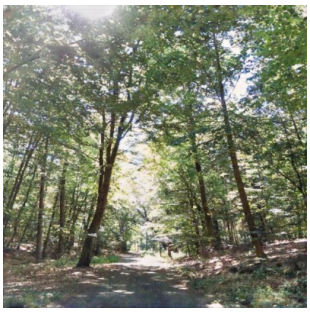
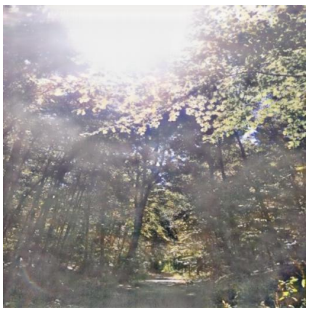
Original Images			
predicted output			

Table 5.3: Test inference results of MaskShadowGAN model trained on Multi dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model..

The only possible solution is to train it with more data relevant to our usage application area, present the network with more variability in the dataset, and train it for the high number of epochs. GANs are harder to train and there is always a possibility of fooling the discriminators. This network benefits our work to a certain extent but does not completely

fulfill our purposes. Thus we chose to proceed with the exposure correction approaches rather than shadow removal processes.

The forest environment also makes it difficult for the GAN network to learn the hidden features as the images in the dataset are of varying illumination and exposures from scene to scene and there is a high variability in the scenes. To test this network performance in extremely low light conditions, we also conducted training and inference on a special dataset called EXclusive Dark Image Dataset (Loh 19). However, the results are not satisfactory even on the trained images due to the requirement of MaskShadowGAN with the set of shadow and shadow-free images, and with this dataset, it is very difficult to distinguish between them due to extremely dark conditions in the images and thus does not provide valid results as required.

Original Images			
predicted output			

Table 5.4: Test inference results of MaskShadowGAN model trained on Freiburg-forest dataset for 200 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model.

As the MaskShadowGAN approach also failed to deliver the expected outcomes in shadow removal and we already tried the supervised approaches, we have decided to try other image processing methods to overcome the darker regions in the images which are a major hindrance for the network to recognize the features with accuracy. Thus, we further researched methods to remove image noises, especially in the challenging environment of Off-Road conditions. We tried even traditional Image exposure correction methods like CLAHE(Contrast Limited Adaptive histogram equalization) with no improvement being achieved on our forest image dataset.

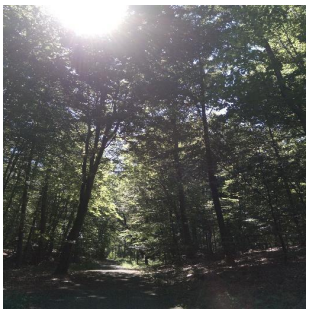
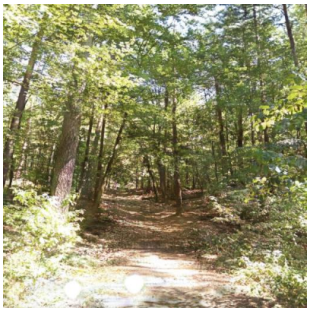
Original Images			
predicted output			

Table 5.5: Test inference results of MaskShadowGAN model trained on RPTU-forest dataset for 100 epochs. The top row shows the original input images and the bottom row of images shows the shadows removed image predictions from the model.

5.1.2 Image exposure correction

As discussed in the section 4.3.2, a model architecture called PSENet was implemented for exposure correction on input images and the results are more satisfactory than the MaskShadowGAN predictions. The training started with the default dataset of SICE (Single Image Contrast Enhancer) (Cai 18) as described in the official paper (Nguyen 23b). SICE is a multi-exposure dataset that contains scenes with different exposure conditions for the PSENet to generate pseudo-Grountruth for training as discussed in section 4.3.2.

Using this training checkpoint, as a first trial, we inferred on our RPTU-Forest images and the results appeared to be satisfactory as shown in figure 5.5. The over-exposed region of the sky is handled correctly and the darker regions are handled to some extent as shown in figure 5.5. The losses are already explained in detail in the section 4.3.2, whose trend during training is displayed in the figure 5.6.

Attempts were made to optimize this training of PSENet such as fine-tuning our data using the SICE-trained checkpoint and making the changes to the LearningRateScheduler in the trainer from ReducedonPlateau to constant Learning rate throughout the training. The observation from this training was that there was no improvement in the predictions and they remained constant as shown in figure 5.10. The next experiment attempted to change the Learning rate scheduler(Lrscheduler) to StepLr(step learning rate) and infer using a previously trained checkpoint file and the output was the same as the initial output. This visual comparison is not a good metric for comparing the Images and thus we tested with the SSIM(Structural Similarity Index metric) which tells us how similar the two images

Figure 5.5: Image showing the PSENet inference on SICE dataset trained checkpoint. the left image is our input, the middle is the gamma mapping applied input image, and the rightmost is the inverted image on which gamma mapping is applied which is our output as described in (Nguyen 23b).

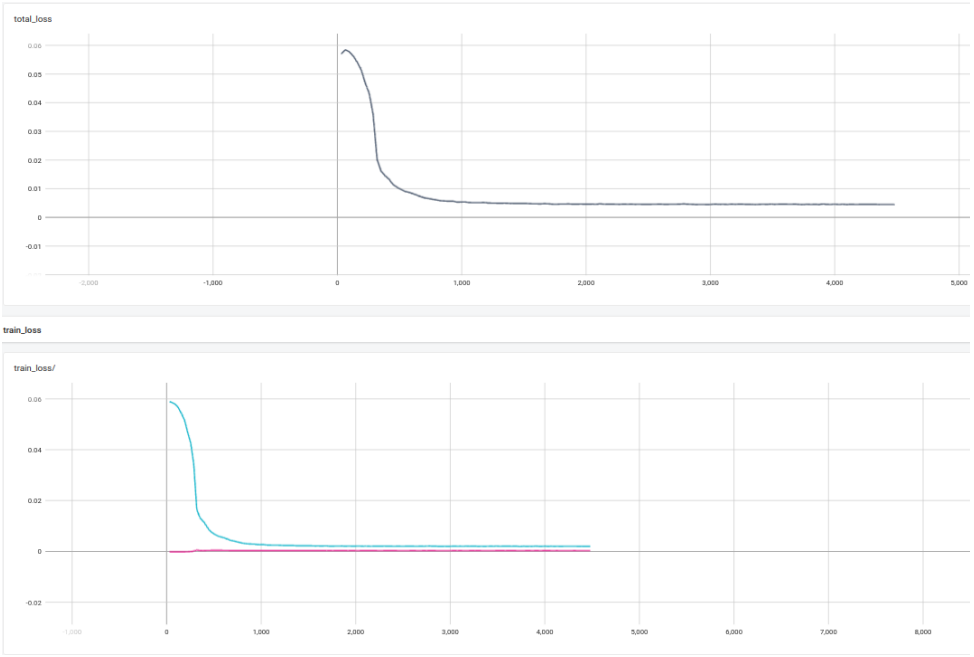


Figure 5.6: Image showing the PSENet training loss curves trend on SICE Dataset. The X-axis represents the iterations and the Y-axis represents the loss values. The top curve is the total loss curve. The bottom blue loss curve is the reconstruction loss and the purple color curve represents the variational loss trend as described in the section 4.3.2.

are, and the result came out to be 0.99 meaning 99 percent of both images are similar and this denotes that there is no significant change in the output with this LRscheduler change.

we also intended for the betterment of training of this network, as we decided to use it as the final optimization method that complements our main Panoptic task. Thus, we planned to increase the training data available for this model and also diversify the training samples present for the model to learn more features and adapt itself to other new environments more dynamically. It should be able to generalize well to the unseen

test data. we created a new large dataset by combining all the available real and synthetic data available to us in RRLab. This attempt gave us a total of 15284 images of different scenes varying from indoors to forest environments. we trained the network from the point after SICE training using it as a checkpoint and trained the PSENet for about 600 epochs.

Unfortunately, this attempt does not fetch us any improvements regarding the previously trained SICE dataset results. However, the model is performing with decent performance even on the SICE dataset training as it is also a large dataset and the inference results on our environment also is quite visually pleasing. So we decided to stop further training and proceed with the integration of this network with the main panoptic model for combination.

5.1.3 Panoptic Segmentation

Panoptic-FPN was our chosen top-down approach for implementing the panoptic segmentation task in this work. As discussed in 4.4, the training of this network is carried out using Detectron2. we already discussed that ground truth annotation for images has to be carried out manually using the labelme polygons enclosed. The class labels we intend to use are pre-determined and labeled accordingly on the RPTU-Forest dataset as shown in figure 5.7. Each class is labeled in the image without any overlaps between the enclosing polygons as that is the strict condition for the panoptic segmentation task.

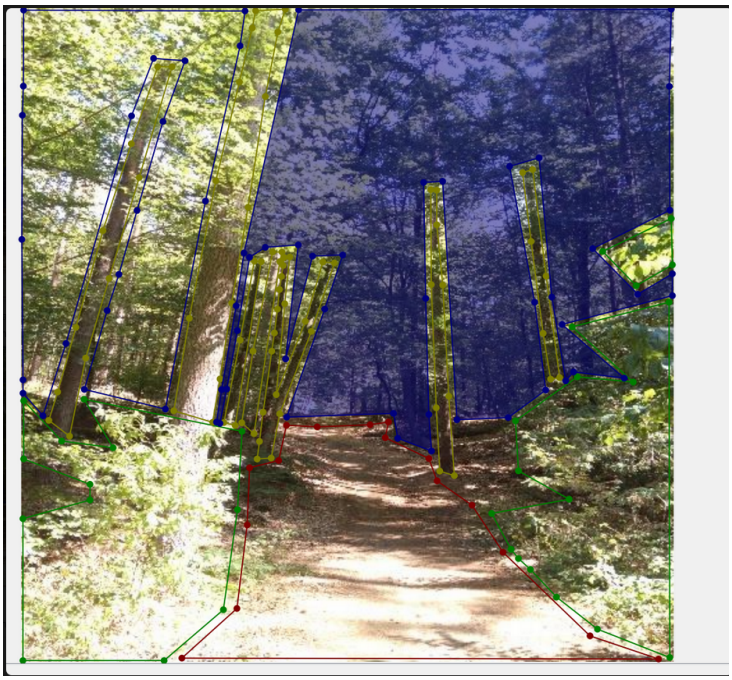


Figure 5.7: Image showing the Labelme polygon-based annotation of ground truth for panoptic segmentation on the RPTU-Forest dataset.

After creating such annotations, as discussed previously we need to convert them into the coco-panoptic format to train the network and there is a convenience before training in

detectron2 to cross-check whether the metadata registered dataset is loaded properly i.e. whether the image we give the model for training is taken with proper annotations or not. If we visualize the data accepted by the model for the training, the sample image is shown in figure 5.8. The annotations are properly in place with the image along with bounding box representations as shown. This step resembles that our training data is correctly processed for training.

Figure 5.8: Sample input image visualization along with annotations used by the model for training.

we trained our network for about 1000 iterations on our RPTU-Forest dataset. Then we performed inference on a sample image to check the model predictions which is shown in figure 5.9. As we can see the result is pretty decent even on the unseen data with all the classes being recognized even with the bounding box predictions of tree trunk object classes. The gaps in the predictions are due to the coarse annotations in our ground truth and this can be avoided to improve the model performance by annotating to a fine pixel level which is a time-consuming process. Hence we can say that the model prediction is on par the best with our limited training dataset size.

The training loss curve trend is very smooth and the losses are reducing concerning the task of panoptic segmentation as shown in figure 5.10. This curve is a weighted combination of losses concerning the individual tasks of semantic and instance segmentation. In comparison, this is a good trend pattern despite the task at hand.

We compared the panoptic predictions on the images with shadows removed and without shadows removed to study the effect of shadow removal on the final prediction and the

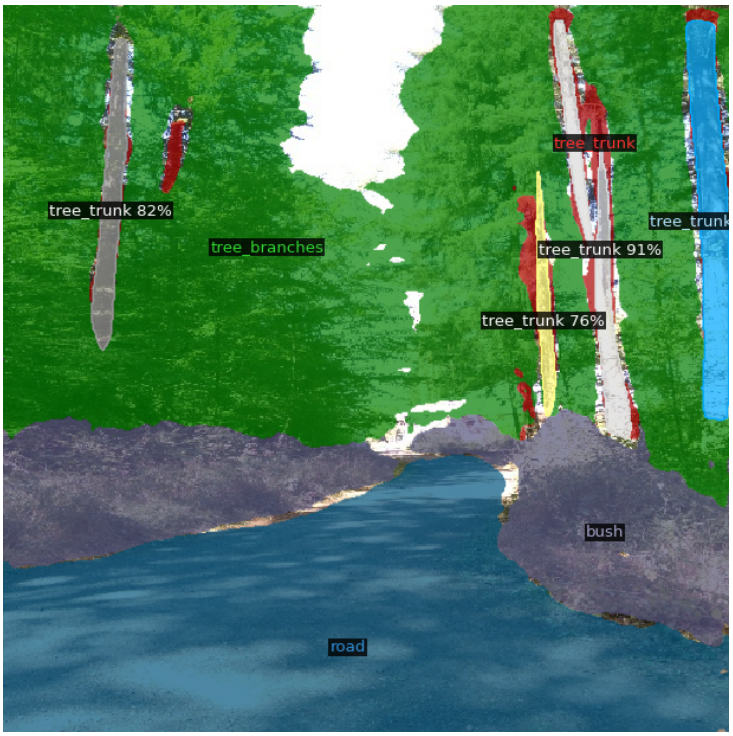


Figure 5.9: Inference on the test set of RPTU-Forest dataset sample.

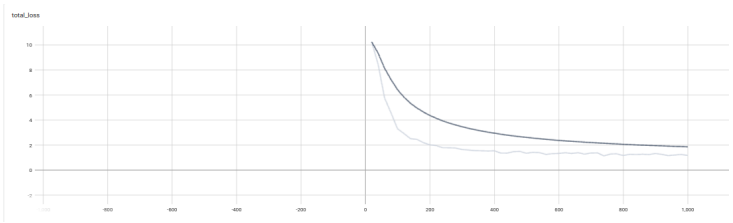


Figure 5.10: Training loss curve trend for Panoptic segmentation task.

results are displayed in the figure 5.11. Before removing shadows, the road was not properly predicted with panoptic segmentation but after removing shadows the road part of the image on which shadows are in major portion are the improving regions for the network to recognize this class. But other classes such as bushes are predicted before shadow removal and strangely this class was not predicted after removing shadows. This is also not our goal of this work to omit the other classes present in the scene and focus only on the path. So, considering the shadow removal task using the MaskShadowGAN approach is proven to be an ideal decision for this work.

In the end, we finally inferred the images with exposure correction using PSENet and fed this input to the panoptic model. comparison is done against the normal image directly fed to the Panoptic network to observe the performance gap between the improvements that image exposure correction brought to the task of Panoptic Segmentation. The visual representation is displayed in figure 5.12, whereupon careful inspection we can see that there is a slight improvement in the model's ability to segment object classes as well as

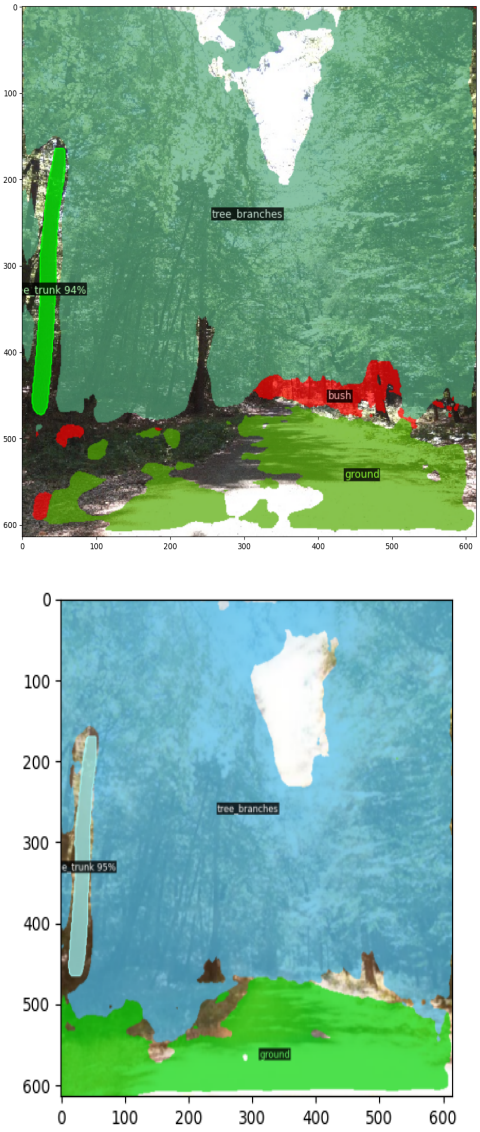


Figure 5.11: Images showing the panoptic predictions on the images with and without shadows.

stuff classes with more precision. This is achieved only with a very small dataset of around 300 training samples and can be improved further if the training dataset size is large. The model can now learn more features independently in the absence of image noises and thus our segmentation performance can be improved further.

The classification of additional instances by the model that are not detected without exposure correction makes this approach more reliable and better performing than the normal Panoptic Segmentation applied directly to the input images. The overall panoptic evaluation metrics are getting better upon more training epochs as can be observed from the figures of these metrics trend curves shown in figure 5.13. we can observe that with increasing iterations, the trend goes up with thing and stuff classes, which resembles the model learning the recognition of both classes with increasing training and updating

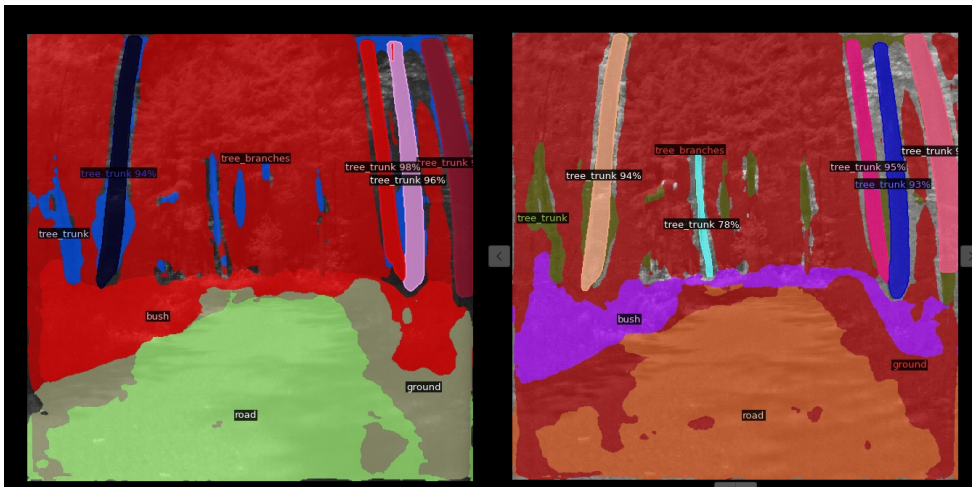


Figure 5.12: Figure showing Panoptic predictions without exposure correction on left and with exposure correction in right.

weights. we were able to achieve a decent level of these evaluation metrics which stands on the same level as the Urban datasets metrics of evaluation.

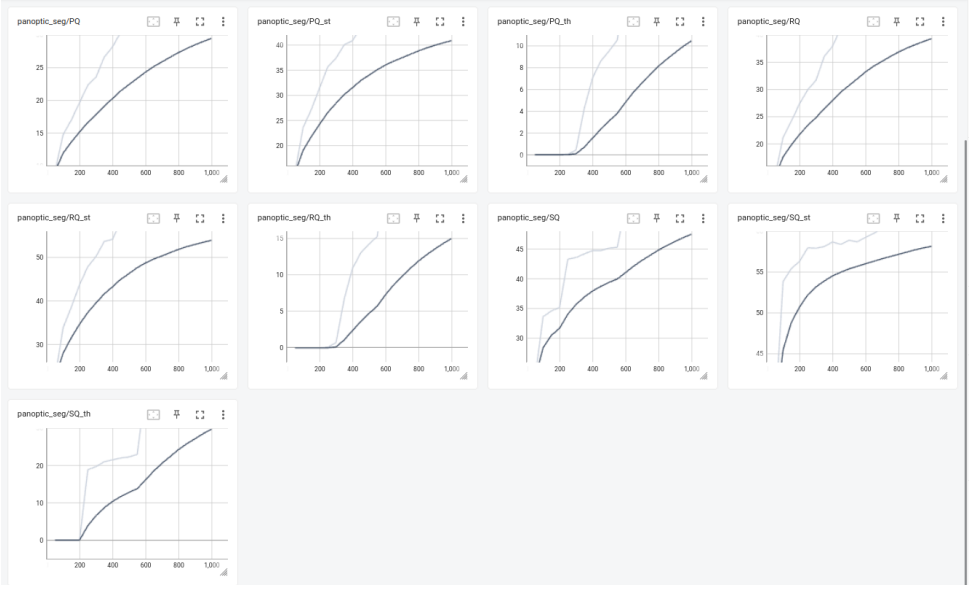


Figure 5.13: Figure showing Panoptic evaluation metrics trend curves. th represents things, st represents stuff classes.

Now we will look into the Quantitative metrics achieved in training this network for Panoptic Segmentation and the evaluation metrics achieved in detail.

5.2 Quantitative Peformance

To evaluate the task in terms of quantitative performance, we usually need a test dataset of Our RPTU-Forest dataset. we need to have ground truth annotations of these images

as well and this added the extra workload for this work as the manual annotations require human effort and accuracy, especially for Panoptic segmentation tasks. Due to the non-overlapping property between segments in the image, this took a significant time for this thesis. with varying the size of these test datasets, the performance of the model varies and we planned to follow the training dataset of 315 images and a validation dataset of 91 images. For this task, the combined performance of qualitative and quantitative measures needs to be considered.

For training MaskShadowGAN on the RPTU-Forest dataset, we used a training dataset of 106 shadow images and 68 shadow-free images which are hand-picked based on visual cues in the images and are subjective. This approach is the drawback of the model performing poorly on the dataset as this selection of training data is not reliable. we trained on this dataset for about 200 epochs on a Nvidia GeForce GTX 1080i single GPU processor. The training took for about more than 24 hours as the batch size is constricted to 1 as discussed earlier and the final losses are updated as shown in figure 5.14. Generator loss is increased as expected and the discriminator loss is reduced and the identity loss is reduced slightly. Due to the limited dataset size, the metrics are not getting better even after 200 epochs of training.

Epoch	Loss_generator	Loss_identity	Loss_Discriminator
0	2.03	0.41	0.60
200	2.18	0.10	0.10

Figure 5.14: Table showing the final losses modified during training of MaskShadowGAN network on RPTU-Forest panoptic dataset

For training PSENet, we combined many datasets available only for panoptic segmentation alone and also all other datasets available to experiment with the model performance enhancement. Initial training with the SICE dataset alone consists of 2002 images of constant resolution of 512*512. we observed that each epoch on the same single GPU processor took about 20 minutes to complete. with the combined dataset of 15000 images, it took almost a full day to finish the training. The loss became stagnant after some 10 epochs alone when pre-trained on the SICE dataset. This model reached its maximum learning capability with the SICE dataset alone and generally assumed not to train further.

For training Panoptic Segmentation, we used the RPTU-Forest dataset with 315 training images along with Panoptic segmentation ground truth annotation masks, semantic segmentation binary masks, Instance segmentation object masks for training along with

the validation set of Images and annotations. we trained the Panoptic FPN network architecture using detectron2 for about 1000 iterations and the final evaluation metrics are as shown in figure 5.15.

PQ	PQ_st	PQ_th	RQ	RQ_st	RQ_th	SQ	SQ_st	SQ_th
37.57	48.46	19.41	48.67	61.91	26.60	57.24	61.44	50.22

Figure 5.15: Table showing the evaluation metrics of Panoptic Segmentation trained using PanopticFPN network architecture. St denotes the stuff class and th denotes the things classes in the above table.

For comparison of our final obtained metrics, we compared against the results of EfficientPs on the Cityscapes dataset which is a well-renowned dataset for Urban scenes, and the results are shown in figure 5.16. Even on the very large dataset, the metrics are still at 64 PQ and in comparison with our dataset size, reaching 38 PQ is satisfactory at this point.

	PQ	SQ	RQ	N
All	64.4	81.8	77.7	19
Things	59.9	80.6	74.0	8
Stuff	67.7	82.6	80.5	11

Figure 5.16: Table showing the evaluation metrics of Panoptic Segmentation trained using EfficientPS network architecture on Cityscapes Dataset. St denotes the stuff class and th denotes the things classes in the above table.

we then combined the PSENet with the Panoptic model for inference in which the input image is first fed into the PSENet checkpoint, which corrects the exposure and then the Panoptic model predicts the segments in it, and on a single GPU we could achieve a Frames per Second(FPS) rate of 2.5 with PanopticFPN architecture. Additionally, in this work, our trained Panoptic model is converted into ONNX format for further testing purposes in real time.

6. Conclusion

In this chapter, our whole work approach and results achieved are summarized and an overview is provided. The future improvements that can be made to this model are presented as ideas.

6.1 Summary and Discussion

In this thesis, a model that is capable of predicting panoptic segments of the image is trained and tested with specific application areas to Off-Road environments was developed. The first and foremost difficulty for this task is the availability of public datasets designed for off-road environments. This implies that this is an area of research that needs further attention. As this is a Deep-Learning approach, data is the most important element for the model to train and this lack of data for this work makes it a challenging task. A special Dataset is collected from the forest behind the Technical University of Kaiserslautern campus which is named as RPTU-Forest dataset for this work. Images are collected using a multi-spectral camera for collecting additional information along with RGB information for using this dataset for other uses.

The manual annotations of the whole dataset with around 400 images are performed in this work. These annotations are further converted into the desired format as per model requirements. The uses of the system that can perform Panoptic segmentation in real-time require more than a lightweight model and also a high-performance computation system onboard. Therefore a balance between inference times and the accuracy of the model has to be figured out. The PanopticFPN architecture is relatively simple in this regard and is proven to be remarkably fast enough in real-time applications. The architecture is explained in detail and training is performed on it using our dataset, approaches to improve its performance are explained and implemented in this work. Several data augmentation techniques are also performed to enhance the training of the model.

The drawbacks of the performance of Deep Learning approaches that involve the feature extraction stage are complemented in this work by providing additional support to the model by attempting to remove the image noises such as shadows and over-exposure. Training experiments with several available datasets in addition to our dataset is performed to improve the model generalization capability and to avoid overfitting problem due to the lack of a large dataset. The evaluation metric of Panoptic Quality is used rather than task-specific metrics to unify the joint task of semantic and instance segmentation and this metric has proven to be of good value achieved in this work. During the annotation of ground truth, the clear distinction of objects and their boundaries is a challenge that needs to be overcome to achieve a finer annotation than the coarse annotation followed in this work. Many subjective decisions are made during the training and testing of this model due to uncertainty in the dataset and the environment we are dealing with and this is a major factor to improve the model performance further.

Results on the RPTU-Forest dataset images are better for the images with exposure corrected than those of images without image exposure correction and PQ obtained in this work is quite reasonable value in terms of dataset size. Shadow removal was unable to play a significant role in improving the model performance rather it misclassified several classes after shadow regions were removed. Overall the trained network provided us with a baseline to implement Panoptic Segmentation with reasonable performance in off-road environments.

6.2 Further work and Improvements

There is ample room available for improving this panoptic model starting with collecting a more diversified and huge dataset. A larger dataset can help the model to generalize well and also learn the patterns hidden better and can perform better in predictions. Instead of collecting data on a single day, it is better to collect dataset over a span of seasons which brings the variability in the dataset. It is even beneficial to annotate and capture rare occurrence elements inside a forest environment such as cyclists, mud pits, swaps, etc. to capture the whole representation of the environment.

Generating high-quality ground truth is another area of improvement that can be carried out for this task. Manual annotations are very labor-intensive tasks that require time and human efforts and for panoptic segmentation, this cannot be automated as of now. The constraint of non-overlapping segments makes it hard for the existing annotation tools to automate this task. Here improvements can be made to automate this task alone which makes the training process simpler. Subjective decisions such as to which class should we assign a segment while annotation should be made more complementary as relying on one single opinion is often the limiting factor. A better infrastructure can be deployed while collecting the dataset to capture the details with more clarity.

Deep learning methods are updated daily and choosing the recent advancements for achieving this panoptic segmentation task can improve the performance and results. Using sophisticated hardware can bring down the inference times and thus our model can be deployed in real-time vehicles and image cropping techniques can be supplemented to achieve this task. The choice of more efficient networks and faster computation onboard enables the implementation of this panoptic segmentation task to navigate it in Off-Road environments such as forests autonomously.

Bibliography

[AI 19] M. AI, "Detectron2: A PyTorch-based Modular Object Detection Library", 2019.

[Bandyopadhyay 21] H. Bandyopadhyay, "Image Segmentation: A Comprehensive Guide", 2021.

[Bandyopadhyay 22] H. Bandyopadhyay, "Instance Segmentation Guide", 2022.

[Barla 21] N. Barla, "Semantic Segmentation: A Comprehensive Guide", 2021.

[Barla 22] N. Barla, "Panoptic Segmentation: A Comprehensive Guide", 2022.

[Berns 11] K. Berns, K.-D. Kuhnert, C. Armbrust, "Off-road robotics—an overview", *KI-Künstliche Intelligenz*, vol. 25, pp. 109–116, 2011.

[Cai 18] J. Cai, S. Gu, L. Zhang, "Learning a deep single image contrast enhancer from multi-exposure images", *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 2049–2062, 2018.

[Campbell 02] M. Campbell, A. J. Hoane Jr, F.-h. Hsu, "Deep blue", *Artificial intelligence*, vol. 134, no. 1–2, pp. 57–83, 2002.

[Campbell 10] M. Campbell, M. Egerstedt, J. P. How, R. M. Murray, "Autonomous driving in urban environments: approaches, lessons and challenges", *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010.

[Campbell 18] S. Campbell, N. O'Mahony, L. Krpalcova, D. Riordan, J. Walsh, A. Murphy, C. Ryan, "Sensor technology in autonomous vehicles: A review", in *2018 29th Irish Signals and Systems Conference (ISSC)*, IEEE. 2018, pp. 1–4.

[Chang 23] M. Chang, "The Introduction of Panoptic Segmentation: Applications and Enable Technologies - Part 3", 2023.

[Chen 17a] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and

fully connected crfs", *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[Chen 17b] L.-C. Chen, G. Papandreou, F. Schroff, H. Adam, "Rethinking atrous convolution for semantic image segmentation", *arXiv preprint arXiv:1706.05587*, 2017.

[Chen 18] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation", in *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 801–818.

[Chen 20] Q. Chen, A. Cheng, X. He, P. Wang, J. Cheng, "Spatialflow: Bridging all tasks for panoptic segmentation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 6, pp. 2288–2300, 2020.

[Chen 23] J. Chen, "What Is a Neural Network?", `https://www.investopedia.com/terms/n/neuralnetwork.asp`, 2023. Accessed on November 30, 2023.

[Cheng 20] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, L.-C. Chen, "Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 12475–12485.

[Cordts 16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, "The cityscapes dataset for semantic urban scene understanding", in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.

[De Geus 18] D. De Geus, P. Meletis, G. Dubbelman, "Panoptic segmentation with a joint semantic and instance segmentation network", *arXiv preprint arXiv:1809.02110*, 2018.

[de Geus 20] D. de Geus, P. Meletis, G. Dubbelman, "Fast panoptic segmentation network", *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1742–1749, 2020.

[Deng 09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", in *2009 IEEE conference on computer vision and pattern recognition*, Ieee. 2009, pp. 248–255.

[Fei 20] J. Fei, W. Chen, P. Heidenreich, S. Wirges, C. Stiller, "SemanticVoxels: Sequential fusion for 3D pedestrian detection using LiDAR point cloud and semantic segmentation", in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, IEEE. 2020, pp. 185–190.

[Geiger 12] A. Geiger, P. Lenz, R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite", in *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[Geitgey 14] A. Geitgey, "Machine Learning is Fun!", 2014. Accessed on November 30, 2023.

[GoogleAI 23] "What is Artificial Intelligence?", `https://cloud.google.com/learn/what-is-artificial-intelligence`, 2023. Accessed on November 15, 2023.

[Guo 23] L. Guo, S. Huang, D. Liu, H. Cheng, B. Wen, "Shadowformer: Global context helps image shadow removal", *arXiv preprint arXiv:2302.01650*, 2023.

[He 15] K. He, X. Zhang, S. Ren, J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition", *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[He 16] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[He 17] K. He, G. Gkioxari, P. Dollar, R. Girshick, "Mask R-CNN", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct 2017.

[He 22] H. He, H. Xu, Y. Zhang, K. Gao, H. Li, L. Ma, J. Li, "Mask R-CNN based automated identification and extraction of oil well sites", *International Journal of Applied Earth Observation and Geoinformation*, vol. 112, p. 102875, 2022.

[Honda 20] H. Honda, "Digging into Detectron 2 - Part 3", 2020.

[Hu 19] X. Hu, Y. Jiang, C.-W. Fu, P.-A. Heng, "Mask-shadowgan: Learning to remove shadows from unpaired data", in *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 2472–2481.

[Karpathy 18] A. Karpathy, J. Johnson, F.-F. Li, "CS231n: Convolutional Neural Networks", 2018.

[Kirillov 19a] A. Kirillov, R. Girshick, K. He, P. Dollár, "Panoptic feature pyramid networks", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 6399–6408.

[Kirillov 19b] A. Kirillov, K. He, R. Girshick, C. Rother, P. Dollár, "Panoptic segmentation", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 9404–9413.

[Lagos 23] J. Lagos, U. Lempiö, E. Rahtu, "FinnWoodlands Dataset", in *Scandinavian Conference on Image Analysis*, Springer. 2023, pp. 95–110.

[Li 21] Y. Li, H. Zhao, X. Qi, L. Wang, Z. Li, J. Sun, J. Jia, "Fully convolutional networks for panoptic segmentation", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 214–223.

[Li 22] X. Li, D. Chen, "A survey on deep learning-based panoptic segmentation", *Digital Signal Processing*, vol. 120, p. 103283, 2022.

[Lin 17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, "Feature pyramid networks for object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[Liu 23] Y. Liu, "The Confusing Metrics of AP and mAP for Object Detection", *Medium*, 2023.

[Loh 19] Y. P. Loh, C. S. Chan, "Getting to Know Low-light Images with The Exclusively Dark Dataset", *Computer Vision and Image Understanding*, vol. 178, pp. 30–42, 2019.

[Maturana 18] D. Maturana, P.-W. Chou, M. Uenoyama, S. Scherer, "Real-time semantic mapping for autonomous off-road navigation", in *Field and Service Robotics: Results of the 11th International Conference*, Springer. 2018, pp. 335–350.

[Mechea 19] D. Mechea, "Panoptic Segmentation: The Panoptic Quality Metric", 2019.

[Mei 22] J. Mei, A. Z. Zhu, X. Yan, H. Yan, S. Qiao, L.-C. Chen, H. Kretzschmar, "Waymo open dataset: Panoramic video panoptic segmentation", in *European Conference on Computer Vision*, Springer. 2022, pp. 53–72.

[Mohan 21] R. Mohan, A. Valada, "Efficientps: Efficient panoptic segmentation", *International Journal of Computer Vision*, vol. 129, no. 5, pp. 1551–1579, 2021.

[Muhammad 22] K. Muhammad, T. Hussain, H. Ullah, J. Del Ser, M. Rezaei, N. Kumar, M. Hijji, P. Bellavista, V. H. C. de Albuquerque, "Vision-based semantic segmentation in scene understanding for autonomous driving: Recent achievements, challenges, and outlooks", *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[Neto 22] N. A. F. Neto, M. Ruiz, M. Reis, T. Cajahyba, D. Oliveira, A. C. Barreto, E. F. Simas Filho, W. L. de Oliveira, L. Schnitman, R. L. Monteiro, "Low-latency perception in off-road dynamical low visibility environments", *Expert Systems with Applications*, vol. 201, p. 117010, 2022.

[Nguyen 23a] H. Nguyen, D. Tran, K. Nguyen, R. Nguyen, "PSENet: Progressive Self-Enhancement Network for Unsupervised Extreme-Light Image Enhancement", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 1756–1765.

[Nguyen 23b] H. Nguyen, D. Tran, K. Nguyen, R. Nguyen, "PSENet: Progressive Self-Enhancement Network for Unsupervised Extreme-Light Image Enhancement", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2023.

[NVIDIA 23] NVIDIA, "What's the Difference Between Artificial Intelligence, Machine Learning, and Deep Learning?", 2023. Accessed on November 30, 2023.

[Pathmind 23] Pathmind, "Neural Network", 2023. Accessed on November 30, 2023.

[Pomerleau 88] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network", *Advances in neural information processing systems*, vol. 1, 1988.

[Porter 84] T. Porter, T. Duff, "Compositing digital images", in *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. 1984, pp. 253–259.

[Raicea 17] R. Raicea, "Want to Know How Deep Learning Works? Here's a Quick Guide for Everyone", 2017. Accessed on November 30, 2023.

[SAE-International 14] SAE-International, "Automated driving levels of drivings are defined in new SAE international standard J3016", *AS: Warrendale*, 2014.

[Sequoia 21] P. Sequoia, "Sequoia User Guide", `https://www.parrot.com/assets/s3fs-public/2021-09/sequoia-userguide-en-fr-es-de-it-pt-ar-zn-zh-jp-ko_0.pdf`, 2021. Accessed on December 05, 2023.

[Shahian Jahromi 19] B. Shahian Jahromi, T. Tulabandhula, S. Cetin, "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles", *Sensors*, vol. 19, no. 20, p. 4357, 2019.

[Shi 19] S. Shi, X. Wang, H. Li, "PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[Silver 16] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot et al, "Mastering the game of Go with deep neural networks and tree search", *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[Stavens 12] D. Stavens, S. Thrun, "A self-supervised terrain roughness estimator for off-road autonomous driving", *arXiv preprint arXiv:1206.6872*, 2012.

[Synopsys 23] "What Is an Autonomous Car?", `https://www.synopsys.com/automotive/what-is-autonomous-car.html`, 2023. Accessed on November 4, 2023.

[Tan 19] M. Tan, Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks", in *International conference on machine learning*, PMLR. 2019, pp. 6105–6114.

[Terra 23] J. Terra, "Regression vs Classification in Machine Learning", 2023. Accessed on November 30, 2023.

[Tesla 17] Tesla, "Tesla's Autopilot", `https://www.tesla.com/autopilot`, 2017. Last accessed on November 2023.

[Valada 17] A. Valada, G. L. Oliveira, T. Brox, W. Burgard, "Deep multispectral semantic scene understanding of forested environments using multimodal fusion", in *2016 International Symposium on Experimental Robotics*, Springer. 2017, pp. 465–477.

[Van Brummelen 18] J. Van Brummelen, M. O'Brien, D. Gruyer, H. Najjaran, "Autonomous vehicle perception: The technology of today and tomorrow", *Transportation research part C: emerging technologies*, vol. 89, pp. 384–406, 2018.

[Wang 20a] H. Wang, R. Luo, M. Maire, G. Shakhnarovich, "Pixel consensus voting for panoptic segmentation", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9464–9473.

[Wang 20b] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, L.-C. Chen, "Axial-deeplab: Stand-alone axial-attention for panoptic segmentation", in *European conference on computer vision*, Springer. 2020, pp. 108–126.

[Wang 21] P. Wang, "Research on comparison of LiDAR and camera in autonomous driving", in *Journal of Physics: Conference Series*, vol. 2093, no. 1, IOP Publishing. 2021, p. 012032.

[Weber 20] M. Weber, J. Luiten, B. Leibe, "Single-shot panoptic segmentation", in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE. 2020, pp. 8476–8483.

[Wendt 18] Z. Wendt, J. Cook, "Saved by the Sensor: Vehicle Awareness in the Self-Driving Age", *Machine Design. https://www. machinedesign. com/mechanical-motion-systems/article/21836344/saved-by-the-sensor-vehicle-awareness-in-theselfdriving-age [last accessed: May 20, 2020]*, 2018.

[Wienrich 23] J. Wienrich, "6 Levels of Automated Driving", 2023. Accessed on November 4, 2023.

[Wu 19] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, "Detectron2", `https://github.com/facebookresearch/detectron2`, 2019.

[Xiong 19] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, R. Urtasun, "Upsnet: A unified panoptic segmentation network", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8818–8826.

[Yang 19] T.-J. Yang, M. D. Collins, Y. Zhu, J.-J. Hwang, T. Liu, X. Zhang, V. Sze, G. Papandreou, L.-C. Chen, "Deeperlab: Single-shot image parser", *arXiv preprint arXiv:1902.05093*, 2019.

[Yeong 21] D. J. Yeong, G. Velasco-Hernandez, J. Barry, J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review", *Sensors*, vol. 21, no. 6, p. 2140, 2021.

[Zaarane 20] A. Zaarane, I. Slimani, W. Al Okaishi, I. Atouf, A. Hamdoun, "Distance measurement system for autonomous vehicles using stereo camera", *Array*, vol. 5, p. 100016, 2020.

[Zhang 21] W. Zhang, J. Pang, K. Chen, C. C. Loy, "K-net: Towards unified image segmentation", *Advances in Neural Information Processing Systems*, vol. 34, pp. 10326–10338, 2021.

[Zhang 22] X. F. Zhang, C. C. Gu, S. Y. Zhu, "SpA-Former: Transformer image shadow detection and removal via spatial attention", *arXiv preprint arXiv:2206.10910*, 2022.

[Zhu 17] H. Zhu, K.-V. Yuen, L. Mihaylova, H. Leung, "Overview of environment perception for intelligent vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2584–2601, 2017.