Ground detection: Road Detection

A. Introduction

Road detection is an important task, together with obstacle detection perhaps one of the main tasks in autonomous navigation. In the past, road and obstacle recognition have been heavily dependent on information gained from LIDAR and other active surface recognition methods. However use of LIDAR may pose a health hazard in certain situations. Hence a passive detection method based solely on visual feedback is essential.

We have explored and implemented two methods for road detection. The first method uses optical flow technique, and second one is based on semantic segmentation using Deep Learning. Semantic segmentation is the task to categorize each pixel into a class. Many deep learning networks developed for semantic segmentation can be fine-tuned for road segmentation. We have studied different models and trained two of them to develop a road detection system. We used existing datasets like Cityscapes for training our models. We have used the images clicked around the UMD campus for further testing of the models.The output of this project is a comparison between the performance of selected Deep Learning models/other methods.

Although a driveable path for an autonomous vehicle may not be limited to roads, (for e.g. an autonomous scooter might be able to drive on sidewalks), our project looks at detection of the roads as the main goal.

B. Methods explored

We implemented two methods for road detection. Our initial plan was to implement semantic segmentation using deep learning. But before going ahead with that , we also explored optical flow as a method for detecting.

- Optical Flow
  - Sparse Optical Flow
  - Dense Optical Flow
- Semantic segmentation using deep learning We trained and compared the two models:
  - Unet with Resnet18 backbone
  - DeepLabV3 with Resnet18 backbone

C. Optical Flow

The idea behind using optical flow was that for a given set of images a vehicle moving on a road, the road is assumed to have the least number of features. Hence it will have the lowest velocity (close to zero). Thus we will by using velocity as our metric, we will be able to classify the environment into different objects and will be able to identify the road out of them.

We tried out two methods int Optical FLow.

C.1. Sparse Optical Flow

Sparse optical flow refers to a technique that estimates the motion of selected key points or features between consecutive frames of an image or video sequence.Sparse optical flow focuses on a subset of interest points, and not all pixel points.

We take the first frame, detect some Shi-Tomasi corner points in it, then we iteratively track those points using Lucas-Kanade optical flow. The results obtained are not satisfactory, since for a case of a car moving on the road, a sparse method yields features and can be more useful to estimate the path of movement or velocity, but inefficient for road detection.

Also, it did not work for the case where the car is moving too fast, the trajectory came out to be not very useful.

C.2. Dense Optical Flow

Unlike sparse optical flow, which focuses on tracking selected key points, dense optical flow calculates motion vectors for every pixel in the image. The objective of dense optical flow is to capture the detailed motion information within an image or video. It provides a dense field of motion vectors that represent the displacement of each pixel between frames. These motion vectors describe the magnitude and direction of pixel movement, allowing for a comprehensive understanding of the dynamics in the scene

One commonly used dense optical flow algorithm is the Gunnar Farneback method, which is used in our case.

For the same set of videos in the previous section :

- Dense optical flow is found using the above algorithm.
- We get a 2-channel array with optical flow vectors (u,v).
- We find their magnitude and direction. We color code the result for better visualization. →Direction corresponds to the Hue value of the image. Magnitude corresponds to the value plane.

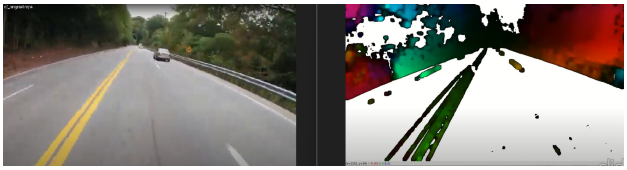

Figure 1. Dense optical flow output (Gunnar - Farneback)

- Additionally, we changed the dark colors in the color-coded frames to white, for better understanding.
- The algorithm computes the relative velocity of pixels. In the case of road planes, since the road color pretty much remains the same, their velocity vectors are mostly constant, which is how we detect roads.
- There are a few limitations that come with using such an approach for road detection. This method is bound to fail in the case of the absence of a significant change of pixels from one frame to the other, typically in the case of a road with barren land (deserted) surroundings. The magnitude thresholding, in this case, will be challenging as the algorithm might not be able to detect the pixels corresponding to the surroundings to be moving, which makes it difficult to tell the road apart.
- Another challenge is high computational complexity. Due to the high computational complexities of this algorithm, it might not be very feasible to deploy this in real-time situations, for road detection-related applications.

D. Semantic segmentation using Deep Learning

As all the team members were new to the field of deep learning, the first step was to understand what semantic segmentation and deep learning is. Some part of it was taught during the class. We also followed a few tutorials on pytorch website to learn the practical implementation.

To get started, we tried to use some pre-trained models available on PyTorch hub and tested with some of the outdoor images captured at the UMD campus.

We explored and tested the following models:

- DeepLabV3 with ResNet101 CNN architecture
- FCN(Fully Convolutional Networks) with ResNet101 backbone
- Unet with Resnet18 and Resnet34 backbone

  We didn't get expected results from the above pre-trained models.
- HybridNets HybridNets is a state-of-art perception network, specially for the drivable area segmentation and lane detection. It was working well for the images we tested.
- A benchmark semantic segmentation network from the ADE20K MIT Scene Parsing Benchchmark: This model uses resnet50dilated and ppm deepsup architectures for encoder and decoder respectively. It was trained by authors on ADE20K Dataset which has 150 semantic categories including road, grass etc. We used it as a pretrained model. The results of segmentation are shown below.

Figure 2. HybridNet Results

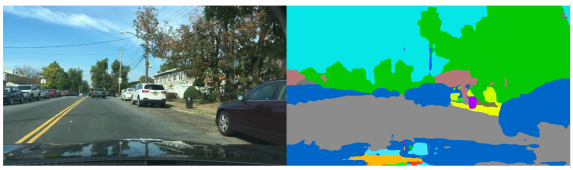

Figure 3. Results1

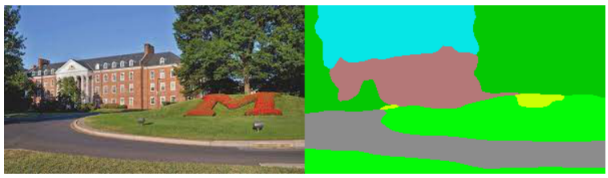

Figure 4. Results2

Figure 5. Results3

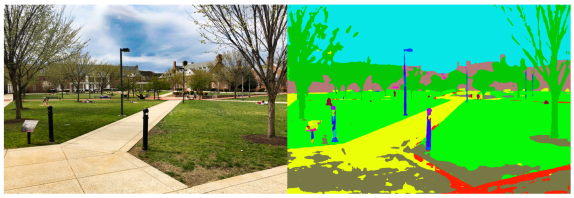

Figure 6. Results4

E. Information about architectures used

E.1. Resnet

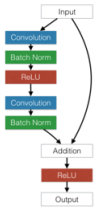

Figure 7. ResNet

ResNet is a short form for Residual Network. The key feature about this architecture is the introduction of residual blocks.It solves the problem of vanishing gradients.The ResNet implements skip connections by creating shortcut that bypass one or more layers in a network, allowing the gradient to flow directly across multiple layers.This property allows resnet to train extremely deep neural networks.

There are several variations of the ResNet architecture, such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152, which differ in terms of depth. The number after "ResNet" indicates the number of layers in the network. The deeper variants (e.g., ResNet-101, ResNet-152) have been shown to achieve better performance on certain tasks but require more computational resources.

ResNet has been widely used and has achieved state-of-the-art results in various computer vision tasks, including image classification, object detection, and semantic segmentation. It has also been a foundation for many subsequent architectures and has greatly influenced the field of deep learning.

E.2. Unet architecture

The U-Net architecture is a popular deep learning architecture for semantic segmentation tasks[3].It gets its name from its U-shape structure, which is shown below. It is particularly effective for tasks where precise localization is required, such as segmenting objects in medical images.

U-Net has encoder-decoder type architecture.The encoder takes images and transforms it into a compact and abstract representation.It consists of a series of convolutional and pooling layers, where each convolutional layer is typically followed by a rectified linear unit (ReLU) activation function. The pooling layers reduce the spatial dimensions of the feature maps while increasing the number of channels.

The decoder performs upsampling and reconstructs the segmented output based on the features learned in the encode. It implements transpose convolutions for this. This skip connection allows the decoder to access fine-grained

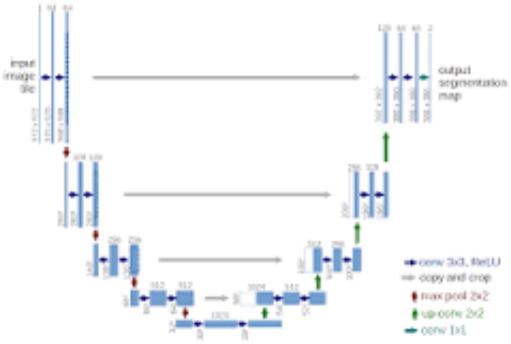

Figure 8. UNet Architecture

information from earlier layers, aiding in precise localization.The final layer of the U-Net architecture is a 1x1 convolutional layer that maps the high-dimensional feature maps to the desired number of output channels, representing the segmented output.

Overall, the U-Net architecture combines the benefits of both convolutional neural networks for feature extraction and fully convolutional networks for dense prediction. The skip connections enable the network to leverage multi-scale features, which is especially useful for accurate segmentation of objects with varying sizes.

E.3. DeepLabV3

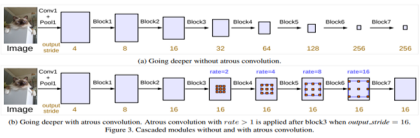

Figure 3. Cascaded modules without and with atrous convolution.

Figure 9. DeepLabV3 Architecture

DeepLabv3 is a popular semantic segmentation architecture.It is a fully Convolutional Neural Network (CNN) model to tackle the problem of semantic segmentation. To handle the problem of segmenting objects at multiple scales, modules are designed which employ dilated convolution. It is an end-to-end trainable deep learning system.

Deeplabv3 cemented its status as one of the go-to models for segmentation because of its high speed,better accuracy, architectural simplicity and generalizability to custom tasks.

From an architectural standpoint, DeepLabv3 uses the ResNet models (trained on Imagenet) as the backbone along with the widely popular Atrous(dilated) Convolution and Atrous Spatial Pyramid Pooling (ASPP) module.

F. Dataset

We used the Cityscape dataset for training our model. The Cityscapes dataset is a widely used benchmark dataset

for urban scene understanding and computer vision tasks, particularly in the context of autonomous driving. It provides a large-scale collection of high-quality images with pixel-level annotations for various semantic segmentation tasks.

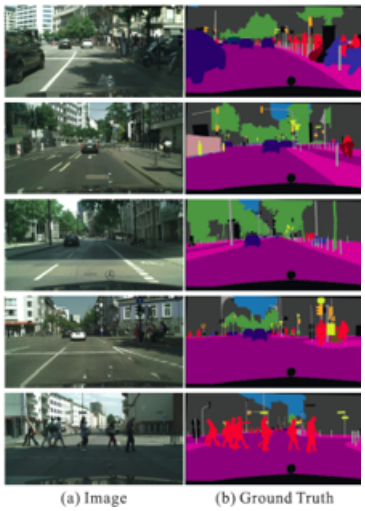

Figure 10. Cityscape Dataset

Below are the key features of the Cityscapes dataset:

1. **Data Collection:** The dataset consists of images captured in different cities across Germany. The images were taken from a car-mounted stereo camera setup, resulting in high-resolution and high-quality RGB images.

2. **Semantic Segmentation:** It provides pixel-level annotations for 30 different classes, including road, sidewalk, buildings, vehicles, pedestrians, traffic signs, and vegetation. The annotations enable the training and evaluation of models for semantic segmentation tasks in urban environments.

3. **Fine-annotated Subset:** The dataset also includes a ”fine” subset, which has more detailed annotations for selected object classes. This subset contains more accurate and dense annotations for object boundaries and instance-level information.

4. **Coarse-annotated Subset:** In addition to the fine-annotated subset, the dataset includes a ”coarse” subset, which provides annotations with lower spatial resolution. The coarse annotations are intended for faster processing and can be useful for certain applications where fine-grained details are not necessary.

5. **Additional Data:** Cityscapes also provides complementary data, including disparity maps, which are useful for tasks such as depth estimation, and camera calibration files, which facilitate accurate geometric alignment of the data.

G. Approach

Our approach to implement and train deep learning model for the task of semantic segmentation can be described in the following steps.

1. **Data Preparation:** As mentioned in previous section, We are using Cityscapes dataset from torchvision.datasets for training our models.We are using Fine-annotated subset in particular.It has images and annotations split in train, val and test sub-directories to use it for training,validation and testing purpose. It has 30 classes such as road, sidewalk, vegetation, buildings etc. However, for our application, we just need to detect the road. We want to know whether the pixel is road or not.Therefore, we removed unwanted classes and rectified the labels of wanted classes.We also Performed data augmentation using albumentation library.In data augmentation technique, various transformations can be applied on original data to increase the size of training data.It improves the performance of model, and make it robust.

2. **Build and Train Models:** We used PyTorch Lightning for this task.It is a lightweight version of PyTorch that simplifies the training and development of neural networks. It provides a high-level interface and abstractions for common deep learning tasks, allowing users to focus more on the model architecture and experimentation rather than the low level code.

   Our model class has been inherited our from LightningModule in PyTorch Lightning. We selected Unet and DeeplabV3 architectures.We selected resnet18 encoder initialized with ‘imagenet‘ pre-trained weights for both the models.We have used AdamW Optimiser, which is a variant of the Adam optimizer that introduces weight decay, along with combining adaptive learning rates to accelerate convergence during training. The training loop is handled by Trainer class object. We provide maximum number of epochs and checkpoint callbacks for retrieving training and validation losses and IoUs for each epoch. The next section discuss more about these metrics.

H. Results

The training loss, validation loss, training IOU and validation IOU(Intersection over union) curves have been plotted for both the models.We also tested the trained model on the images from CityScapes Test dataset and images that we have collected. The results are shown below.

H.1. UNet and DeepLabV3

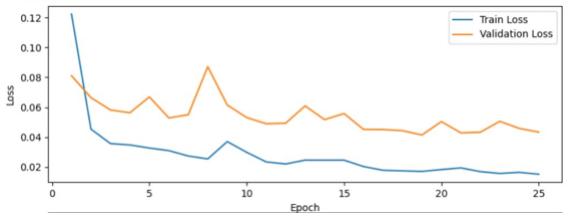

Figure 11. UNet: Training and Validation Losses

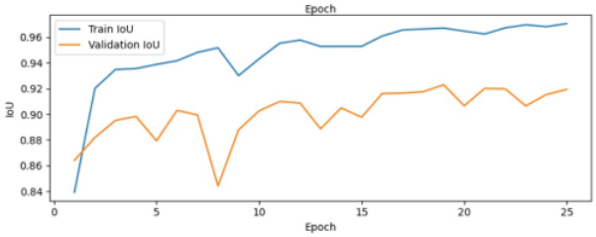

Figure 12. UNet: Training and Validation IOU

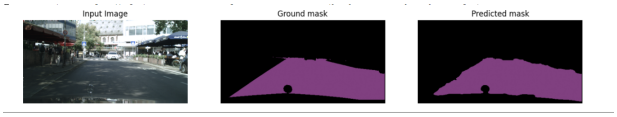

Figure 13. UNet: CityScape Test image1

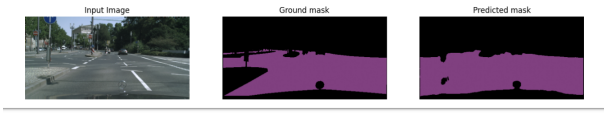

Figure 14. UNet: CityScape Test image2

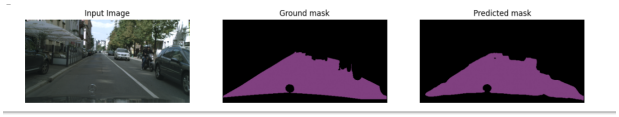

Figure 15. UNet: CityScape Test image3

H.2. Comparison

As can be seen below in the training and validation loss curves plotted against epochs, the training loss curves turned out to be relatively lesser than that of validation loss, in both models. The curves recovered until the 23rd epoch and started getting higher after that. This is how we fixed the epoch value to 23, to avoid over-fitting. The training loss for DeepLabV3 is observed to be relatively lesser (Figures 11 and 19). We also observed that the segmentation outputs from the DeepLabV3 model turned out more efficient for our test data. IOU is calculated by measuring the overlap between the predicted bounding box or segmenta-

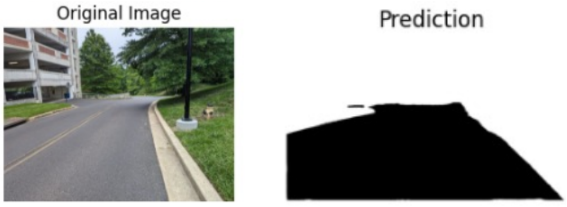

Figure 16. UNet: Our Test image1

Figure 17. UNet: Our Test image2

Figure 18. UNet: Our Test image3

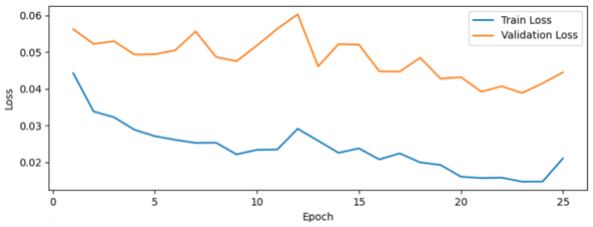

Figure 19. DeepLabV3: Training and Validation Losses

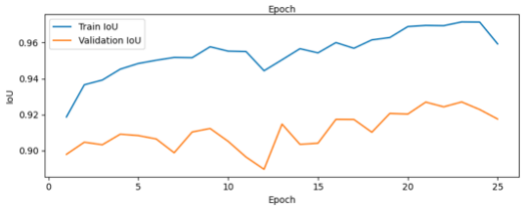

Figure 20. DeepLabV3: Training and Validation IOU

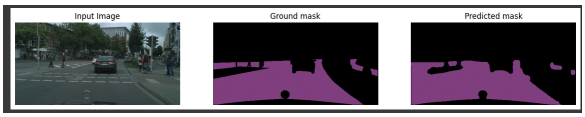

Figure 21. DeepLabV3: Cityscape Test image1

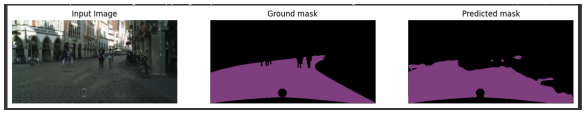

Figure 22. DeepLabV3: Cityscape Test image2

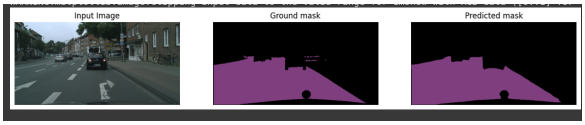

Figure 23. DeepLabV3: Cityscape Test image3

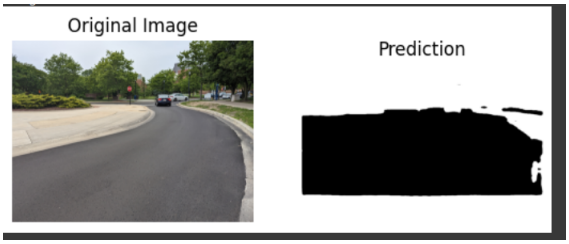

Figure 24. DeepLabV3: Our Test image1

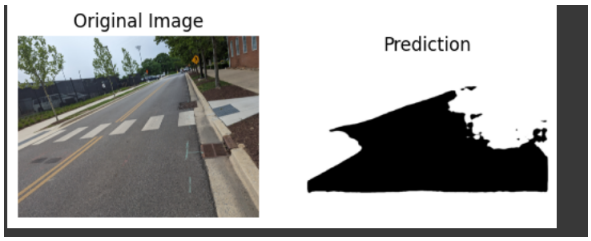

Figure 25. DeepLabV3: Our Test image2

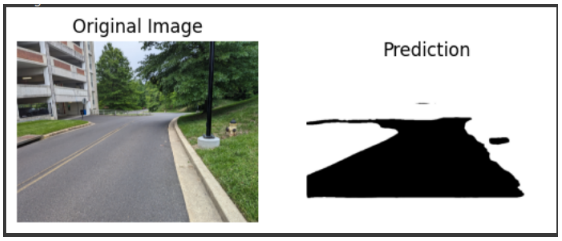

Figure 26. DeepLabV3: Our Test image3

tion mask and the ground truth bounding box or mask. It is computed as the ratio of the area of intersection between the predicted and ground truth regions to the area of their union. The most ideal case is when the IOU is 1. As we can see in Figure 12 and Figure 20, both the models have their IOU curves almost reaching 1 near the epoch 25. For some of our test cases, both the models are detecting sidewalk as a road which is incorrect. There is a scope of improvement to train the models on custom dataset.

References

[1] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[2] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[3] Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham. https://doi.org/10.1007/978-3-319-24574-4_28

[4] Torchvision Semantic Segmentation – PyTorch for Beginners https://learnopencv.com/pytorch-for-beginners-semantic-segmentation-using-torchvision/

[5] HybridNets: End-to-End Perception Network - https://arxiv.org/abs/2203.09035

[6] MIT Scene Parsing Benchmark http://sceneparsing.csail.mit.edu/