Vishal Murugavel

Jingdao Chen

CSE 4000

December 8, 2023

Off-Road Autonomous Driving Analysis Report

Mississippi State University's Center for Advanced Vehicular Systems (CAVS) offers a world-class technology development center comprised of engineering, research, development, and technology transfer teams. CAVS is committed to exploring solutions to complex problems, in areas such as autonomous vehicles, materials science, high-performance computing, advanced controls, and human-machine interaction. Off -road, heavy duty vehicle automation and industrial are the last frontiers of autonomous mobility. It is on the forefront of autonomous mobility research focusing on developing solutions for non-urban environments. This world class facility and learning opportunity it offers in artificial intelligence and autonomous driving cars created the motivation in me to do the research project. With top rated vehicle proving grounds of 50 acre for testing off road autonomous vehicles and a full suite capability for autonomous system development, with sensor research, artificial intelligence, and vehicle robotization.

With this interest during the Fall 2023 Semester, I met my AI professor Jingdao Chen and requested a research project opportunity. My professor gladly accepted the request and offered me four different topics to choose from namely Machine learning, Deep learning, 3-D Reconstruction and Robotics. I was more interested in doing 3-D reconstruction and got the approval to proceed to do the project on Semantic Navigation on Off-Road Autonomous Driving with the process of 3-D reconstruction in Image segmentation.

Image segmentation is a technique of digital image processing and analysis of converting an image into multiple parts or regions that are represented by a mask and this is based on the characteristics of the pixels in the image. By dividing an image into segments, we can process the required segments of the image in lieu of processing the whole image.

Neural Network:

It is a specific architecture inspired by the human brain's neural structure and a subset of machine learning focused on deep learning. It is designed with interconnected layers of artificial neurons for complex pattern recognition, especially when dealing with large and high dimensional data. It can learn intricate representations from the data. For real time applications this can be used especially with advancements in hardware and optimized model architectures, allowing for fast and efficient processing of data in real time scenarios. Another name for Neural Networks is Artificial Neural network which is comprised of three node layers, such as input layer, one or more hidden layers, and an output layer. I learned that the neural network is a method in artificial intelligence that can teach computers to process data and it is a type of machine learning process called deep learning, which uses neurons in a layered structure that resembles the human brain.

In Artificial intelligence, using Bayes classifier I used the dataset to predict the classification of different images with the accuracy of 0.75%. For my research project I had to use a neural network which is also known as artificial neural networks, this is a subset of machine learning. I

learned Pytorch tutorials and used those concepts for better performance of classification of images.

In this project the neural network uses a U-Net Architecture and divides the images and mask to determine whether the path is traversable or not for the vehicle to go through. The first step in my research was designing the dataset by downloading the images from the CAVs website. This CAV database is large and has several hundred images. I chose a hundred images from CAVS database for my research. Using these images created two subsets of dataset namely Training data set and Test data set.

I learned in my Artificial Intelligence class of using probabilistic methods to perform modeling, prediction, and generation for the domain of fashion images which I used in my research project with the CAVs Dataset. Under the Google Colab Implementation, I developed the machine learning model in training sets and evaluated the model in test sets.

The reason why I used the CAVs dataset is because it has la large collection of unstructured, off-road autonomous driving images of grass, hills, obstacles, dense vegetation, similar to forest environment, stones, trees, green bushes, uphill images that can be used to test and train the machine learning model, or train neural networks on off-road autonomous driving traversability conditions.

I have selected hundered images from CAV dataset and created the training data set for this project. As I need to load these images in small subsets to the dataset I used a Data Loader which divides up all of the images into batches and passes it to the neural network. Through multiple iterations, the data loader has loaded the data into the training dataset to get trained.

In the below sections illlustrate the Pytorch notebook with code and sample images to explain how the images are divided by using image segementation, making preditions on whether it can detect if a particular image is traversable or not traversable.

The Pytorch code mentioned below has the header files to import the dataset, dataloader, and random split to split the images .

```
import torch
from torch.utils.data import Dataset, DataLoader, random_split
from torchvision import transforms
from PIL import Image
import matplotlib.pyplot as plt
import torchvision
import numpy as np
from google.colab import drive
import glob
import os
import sys
```

After this header file is processed , I mount the google Drive folder where the dataset can be accessable.The steps are provided below in the code block:

```
[2]  # Mounting the Google Drive Folder so that data set file are accessed
     drive.mount('/content/drive', force_remount=True)
     %cd drive/MyDrive/CAVS Dataset/MAVS_Simulated_Data
     sys.path.insert(0,'content/drive/MyDrive/CAVS Dataset/MAVS_Simulated_Data/')

     # Define the folder path in Google Drive
     folder_data = '/content/drive/MyDrive/CAVS Dataset/MAVS_Simulated_Data/imgs'
     folder_masks = '/content/drive/MyDrive/CAVS Dataset/MAVS_Simulated_Data/masks'
```

```
Mounted at /content/drive
/content/drive/MyDrive/CAVS Dataset/MAVS_Simulated_Data
```

Then we define the custom CAVs dataset in the U-Net Architecture to evaluate those images and make predictions whether those images are traversable or non-traversable.

Defining the CAVsDataset and the U-Net Architecture is below in the following Pytorch Code:

```
# CAVsDataset class definition
class CAVsDataset(Dataset):
    def __init__(self, image_paths, target_paths, transform=None):
        self.image_paths = image_paths
        self.target_paths = target_paths
        self.transform = transform

    def __getitem__(self, index):
        image = Image.open(self.image_paths[index]).convert("RGB")
        mask = Image.open(self.target_paths[index]).convert("RGB")

        if self.transform is not None:
            image = self.transform(image)
            mask = self.transform(mask)

        return image, mask

    def __len__(self):
        return len(self.image_paths)
```

```
# U-Net architecture
class UNet(nn.Module):
    def __init__(self):
        super(UNet, self).__init__()

        # Define U-Net architecture
        self.encoder = nn.Sequential(
            nn.Conv2d(3, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.middle = nn.Sequential(
            nn.Conv2d(64, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(128, 128, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.decoder = nn.Sequential(
            nn.Conv2d(128, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.ConvTranspose2d(64, 3, kernel_size=2, stride=2)
        )
```

The total images which need to be loaded into the training dataset are divided into batches. Those batches are called epochs. An epoch is when the training data is used at once and is defined as the total number of iterations of all the training data in one cycle for training the machine learning model.

```
# Training loop
num_epochs = 10
for epoch in range(num_epochs):
    for images, masks in train_loader:
        optimizer.zero_grad()
        outputs = model(images)
        masks = nn.functional.interpolate(masks, size=outputs.size()[2:], mode='bilinear', align_corners=False)
        loss = criterion(outputs, masks)
        loss.backward()
        optimizer.step()

# Example code to use the trained model for prediction
with torch.no_grad():
    # Display one image and its corresponding ground truth mask
    index_to_display = 0
    image, mask = custom_dataset[index_to_display]
    image = image.unsqueeze(0)  # Add batch dimension
    mask = nn.functional.interpolate(mask.unsqueeze(0), size=outputs.size()[2:], mode='bilinear', align_corners=False)

    # Display input image and ground truth mask
    plt.subplot(1, 2, 1)
    imshow(torchvision.utils.make_grid(image), title='Input Image')

    plt.subplot(1, 2, 2)
    imshow(torchvision.utils.make_grid(mask), title='Ground Truth Mask')

    plt.show()
```

After that is done, we have to create a custom Dataset for a neural network to test if the image will work by using a U-Net architecture and it will convert the image and transform it for data augmentation or otherwise normalization which is a process that will transform the images to the correct size. Once that is done it will show an example image called input image, ground Truth mask, and Predicted mask.

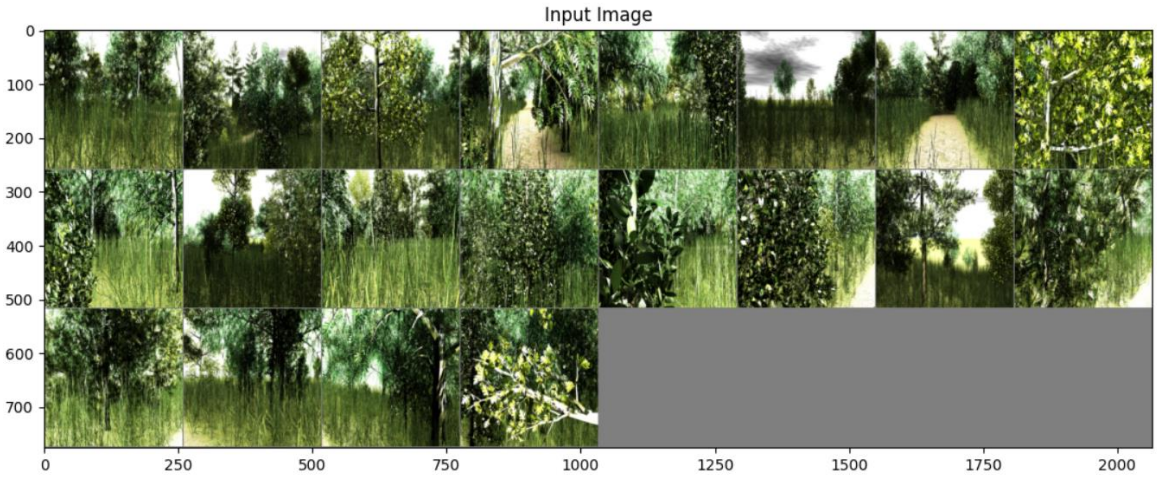
A U-Net Architecture is a type of neural network that consists of four encoder blocks and four decoder blocks that are connected through one bridge. So, in the following pictures we can see the original input images, the ground truth mask images and predicted mask images.

Input Images are the images which we are sending to machine learning model.

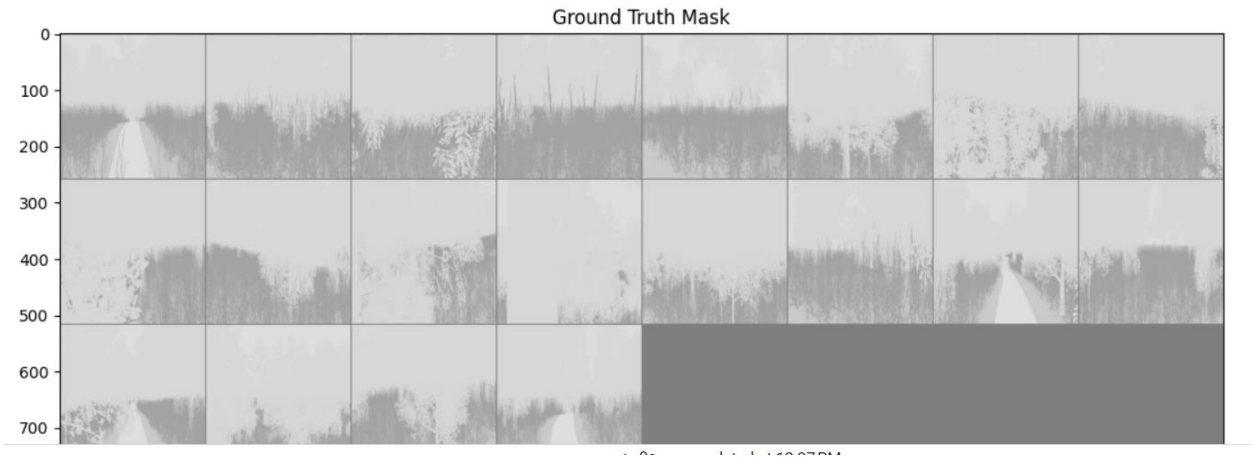
Ground Truth Mask images are the images predicted by Machine Learning Model

Predicted Mask images are the images humans are predicting, which are correct.

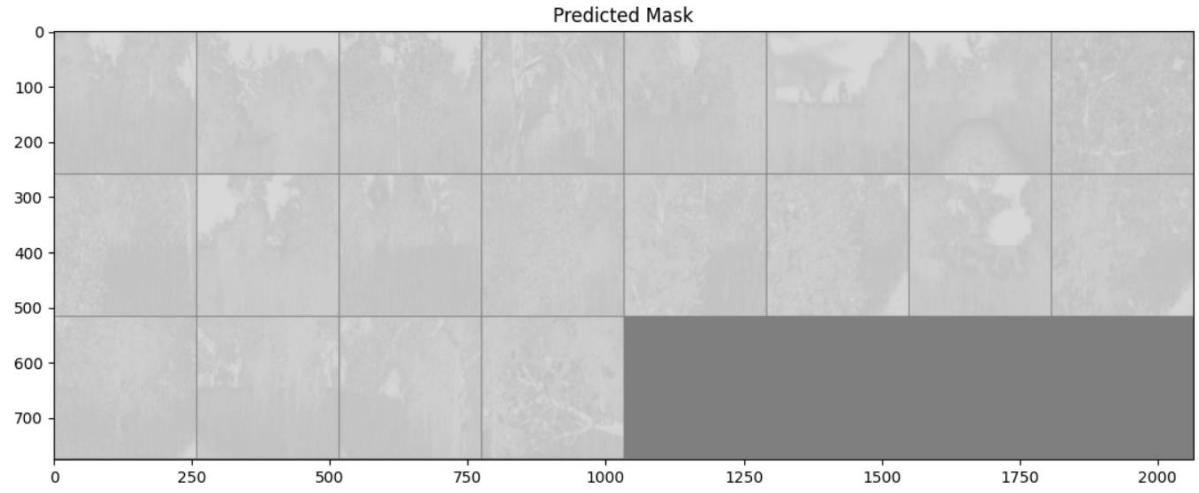
The below are the input image samples sent to Machine Learning Model



The images shown below are Ground Truth Mask what the Machine Learning Model is prediting for those input images. For example if it is grass in the input image, the Machine Learning Model will predict it as a road which is not right.



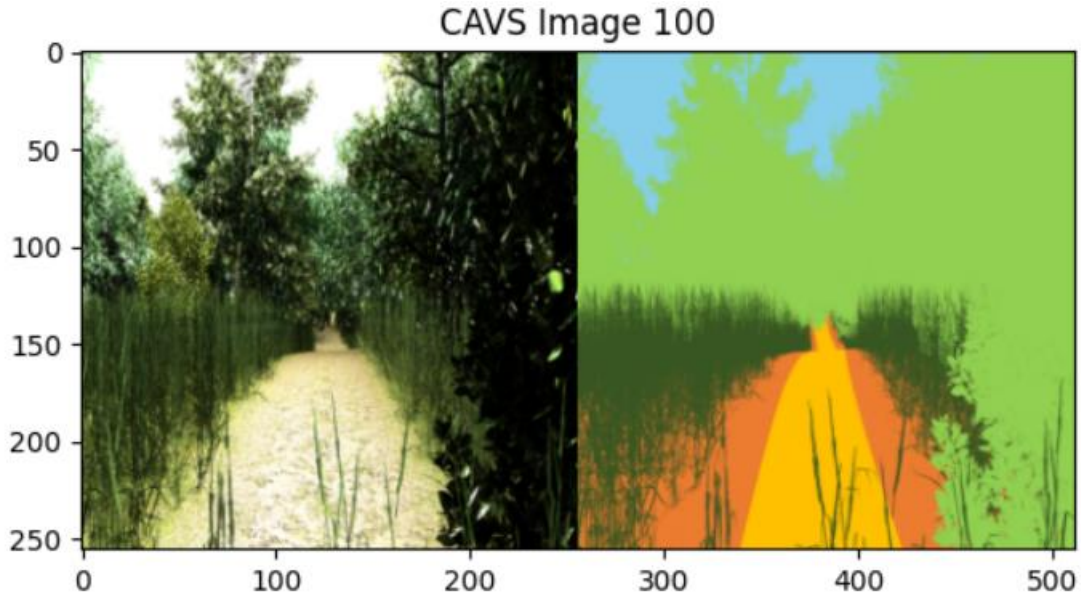
Below are the Prediction Mask images what humans would expect or what should be the right image from the Machine Learning Model.



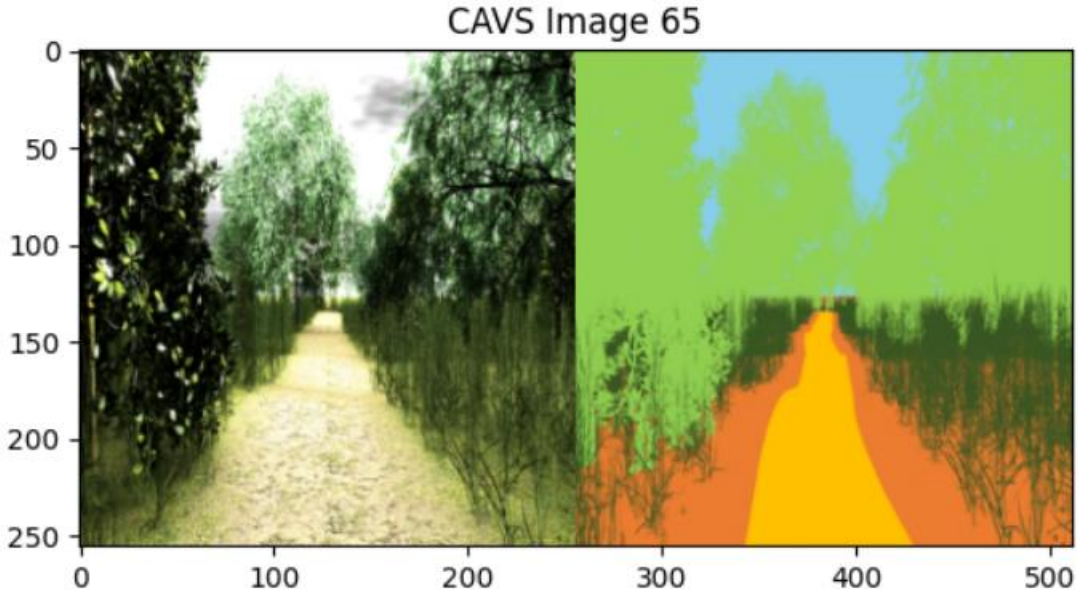
The below Pytorch code will display all the images assigned to an input image on the left and corrected Predicted Mask image to the right of what humans would expect from this machine learning model

```
# Define the imshow function
def imshow(img, title=None):
    img = img / 2 + 0.5  # unnormalize
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))

    if title is not None:
        plt.title(title)

    plt.show()

# Create the custom dataset
image_paths = sorted(glob.glob(os.path.join(folder_data, 'image_*.bmp')))
target_paths = sorted(glob.glob(os.path.join(folder_masks, 'annotated_*.bmp')))
custom_dataset = CustomDataset(image_paths, target_paths)

# Example code to display images and masks side by side in a horizontal layout
for i in range(10):  # Display the first 10 samples
    image, mask = custom_dataset[i]

    # Display image and mask side by side
    combined_image = torch.cat([image, mask], dim=2)  # Concatenate images and masks horizontally
    imshow(torchvision.utils.make_grid(combined_image.unsqueeze(0)), title=f'CAVS Image {i+1}')
```

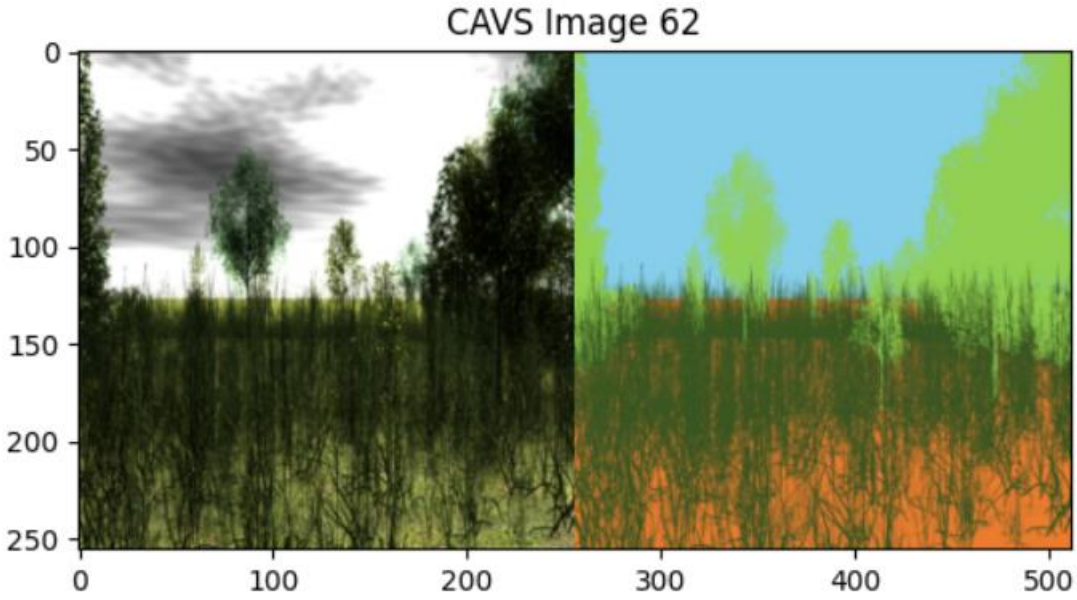
Below are some Results of the CAVs images along with the mask that is traversable or not. The First image below shows it is traversable because there is a path for the Machine learning vehicle to go through.



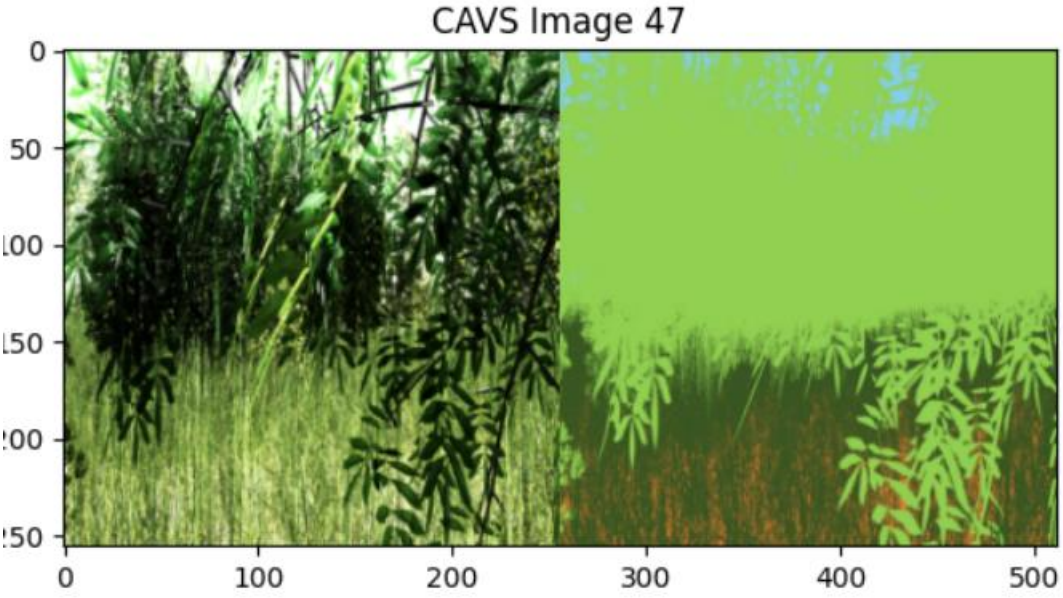
The Second image below shows it is traversable because there is path for Machine learning vehicle to go through.



The third image below shows it is non-traversable because in the previous images there was grass and bush which will block the path for Machine Learning Vehicle to go through.



This image below shows it is a non-traversable because there is no path which will block the Machine Learning Vehicle to go through.



I had the challenge of downloading the images from the Mississippi State University CAVS off-road autonomous Driving Segmentation as the size is 38 GB with several hundreds of images. Initially I was trying to use Google Cloud storage, but I did not have the Google drive space hence I downloaded it to a local laptop hard drive. When I was working on this CAVS dataset, I realized that the data set has more than 1700 images which is a lot. Initially I thought of taking 500 images, but I realized it will take lot of time to train it and evaluate it with the machine learning model hence I decided to use only 100 images for my research project.

In my future, I want to expand a more extensive dataset under different sets of off-road scenarios with an improved version of the model and better performance with exploring some advanced techniques in image segmentation to increase the size of my dataset like scaling and image transformations that can enhance the model's ability to manage different variations when the vehicle is off road.

This research project made me dig deep into the realm of off-road autonomous driving, focusing on semantic navigation with the help of 3-D reconstruction and machine learning techniques. Being part of this great autonomous vehicle search project was a great fulfilling learning experience where I had the opportunity to gain experience for the greater impact of AI and Machine learning in real life scenarios. The exploration of machine learning, deep learning, and robotics, with a particular focus on the U-Net Architecture in Pytorch, has provided valuable insights into the challenges and possibilities within this dynamic field. This research gave me a hold on practical implication for the development of autonomous vehicles capable of navigating challenging off-road environments. As technology advances, the safety of efficient off-road

autonomous navigation becomes more tangible, marking it a more promising future with the help of artificial intelligence and vehicular automations.

Works Cited

https://www.cavs.msstate.edu/resources/autonomous_dataset.php

https://pytorch.org/tutorials/beginner/basics/quickstart_tutorial.html

https://www.ibm.com/topics/machine-learning

https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-epoch-in-machine-learning#:~:text=An%20epoch%20is%20when%20all,dataset%20takes%20around%20an%20algorithm.

https://www.analyticsvidhya.com/blog/2023/08/unet-architecture-mastering-image-segmentation/#:~:text=A.,abstract%20and%20high%2Dlevel%20features.