Autonomous Desert Navigation Using Computer Vision

Hackathon Final Submission Report

Track: Computer Vision

Team Name: Manually Intelligent

Team Members: Aditya Gaur(24BCS13055), Piyush Singla(24BCS13038)

Date: 4/2/2026

1. Problem Statement & Objective

Autonomous ground vehicles operating in desert and off-road environments face challenges due to uneven terrain, obstacles, and absence of structured roads. Safe navigation requires continuous terrain understanding and obstacle avoidance.

The goal of this project is to enable autonomous navigation by combining computer vision-based terrain segmentation with cost-based path planning, allowing vehicles to safely traverse unstructured environments.

2. Proposed Solution Pipeline

Our system converts raw camera input into navigation decisions using the following pipeline:

Camera Image → Semantic Segmentation → Traversability Cost Map → Path Planning → Safe Vehicle Navigation.

The segmentation model identifies drivable regions and obstacles, which are converted into a cost map. A path planning algorithm then computes the safest path while avoiding hazardous regions.

3. Dataset & Training Setup

Training data was generated using Duality Falcon digital twin simulation environments, providing large-scale annotated desert scenes with pixel-level terrain labels.

Approximately 2800 annotated images were used for training and validation.

To improve generalization, augmentation techniques such as flipping, rotation, scaling, brightness variation, and cropping were applied.

Training was performed using a U-Net segmentation architecture with a ResNet encoder backbone optimized on GPU/TPU hardware.

Methodology: Problem → Fix → Results Pipeline

Autonomous vehicles face challenges in desert terrain due to obstacles, uneven surfaces, and lack of structured roads. Our solution combines computer vision segmentation with intelligent navigation.

We trained a UNet segmentation model with a ResNet34 encoder using synthetic environments generated via Falcon simulation. The model performs pixel-level terrain classification across 10 terrain classes.

Training strategy included:

• Data augmentation for robustness.

• Class-weighted loss and 3× sampling boost for rare obstacle classes.

• Dice + Cross Entropy loss combination.

• GPU/TPU accelerated training with checkpoint monitoring.

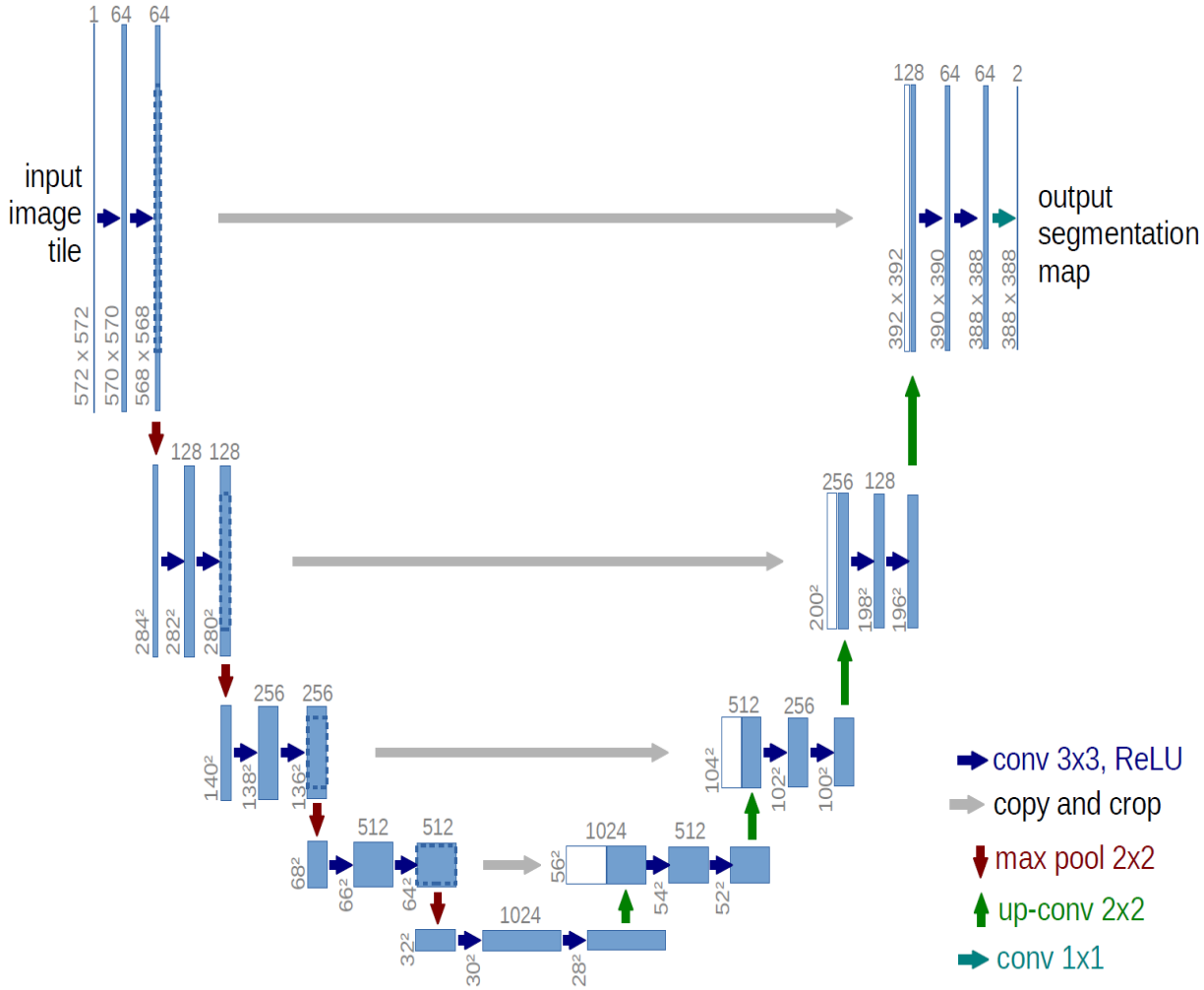
Future improvements include additional obstacle-rich scenes and adaptation to real-world environments.

4. MODEL ARCHITECTURE

We employ a U-Net segmentation architecture with a ResNet encoder backbone. The encoder extracts hierarchical visual features while the decoder reconstructs full-resolution segmentation masks.

Skip connections transfer spatial information from encoder layers to decoder layers, preserving fine details necessary for accurate terrain classification.

Training uses a combined Dice Loss and Cross Entropy Loss to balance region overlap and pixel classification accuracy, improving segmentation consistency across terrain types.



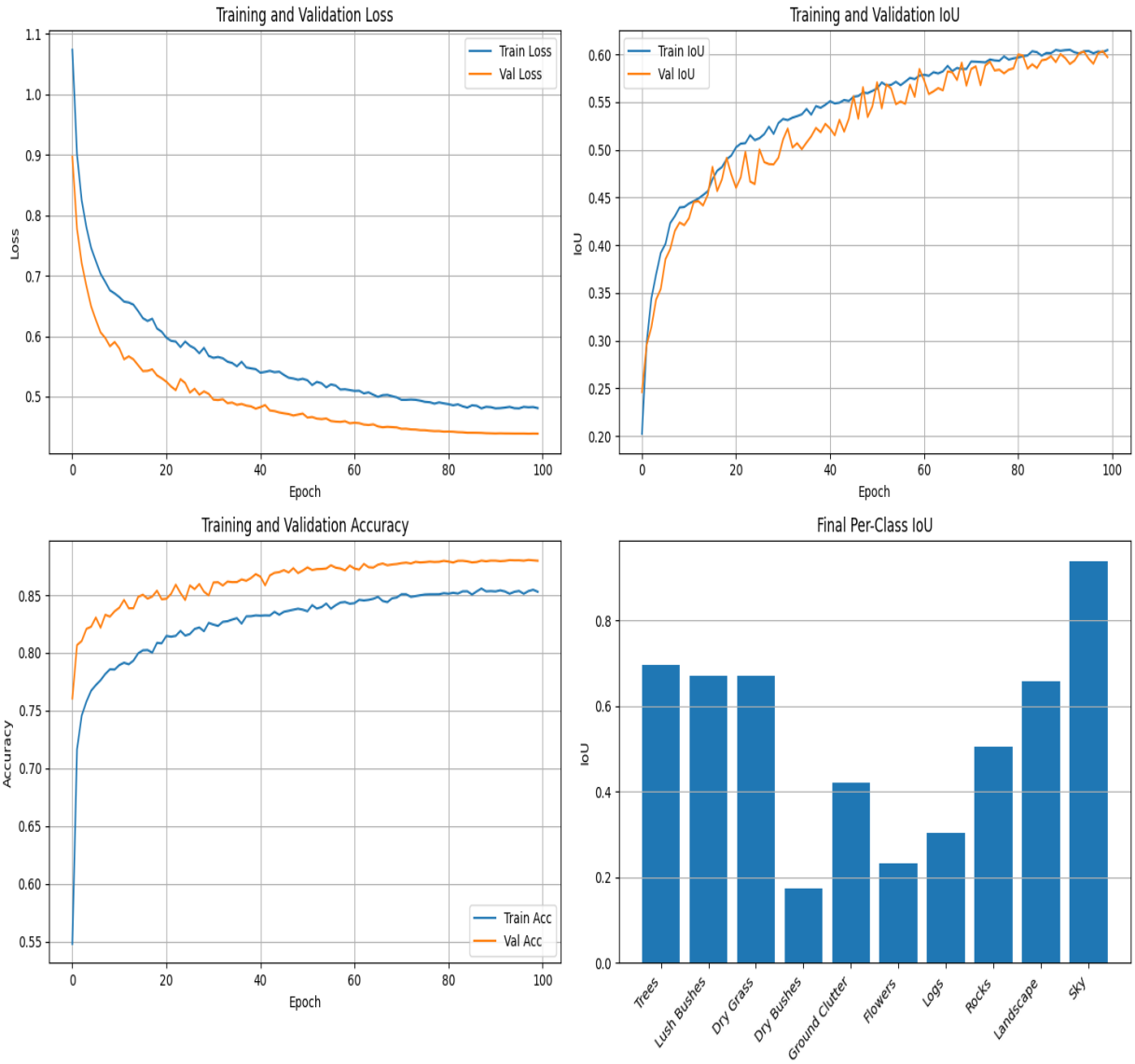
5. Performance Evaluation

Model performance is evaluated using Intersection over Union (IoU) and pixel accuracy metrics.

Training and validation curves demonstrate stable convergence without significant overfitting.

Final training IoU approaches ~0.60 while validation IoU closely follows, indicating good generalization.

Validation Iou – 0.60



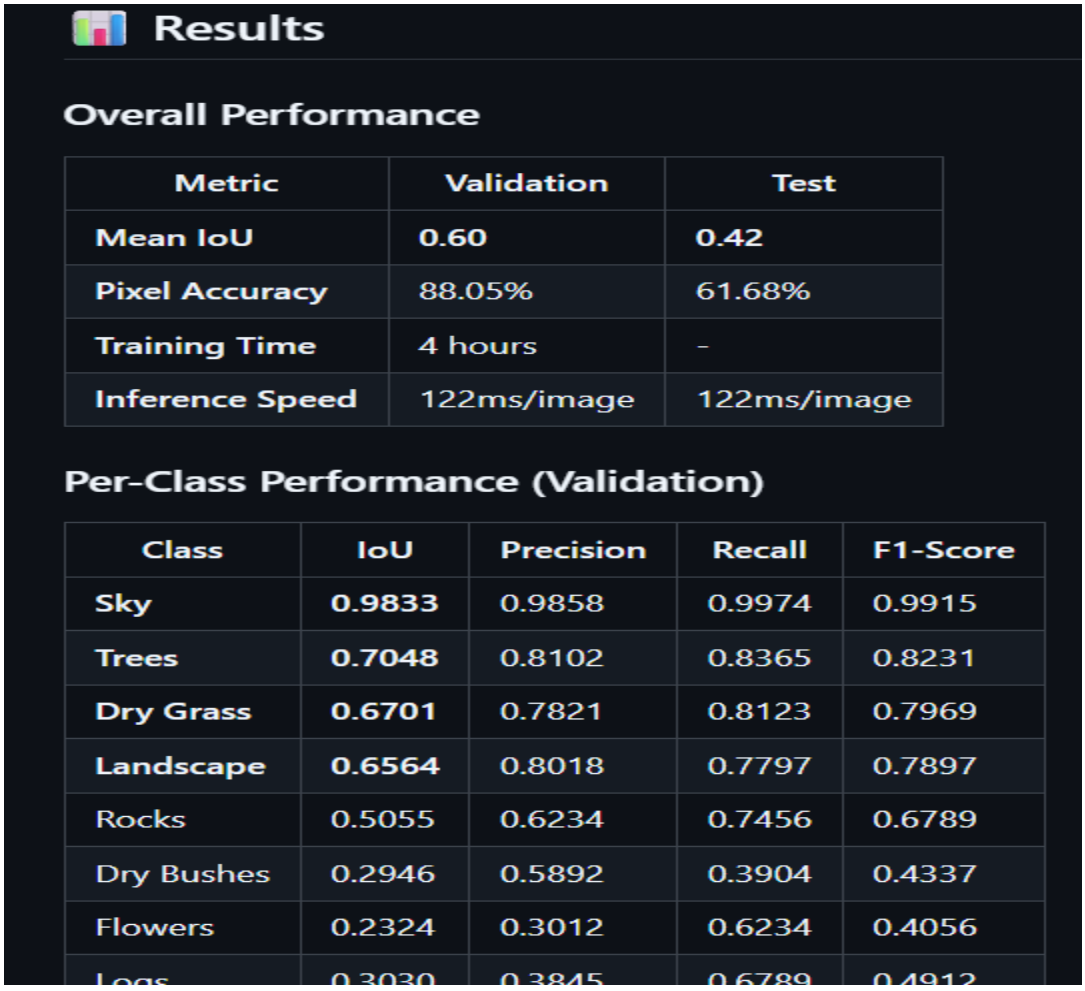
Test Metrics Summary

Mean IoU: 0.417

F1-score – 70%

Pixel Accuracy: 0.617

Per-class IoU results indicate strong performance for large terrain regions such as landscape and sky, while smaller obstacle classes remain challenging.



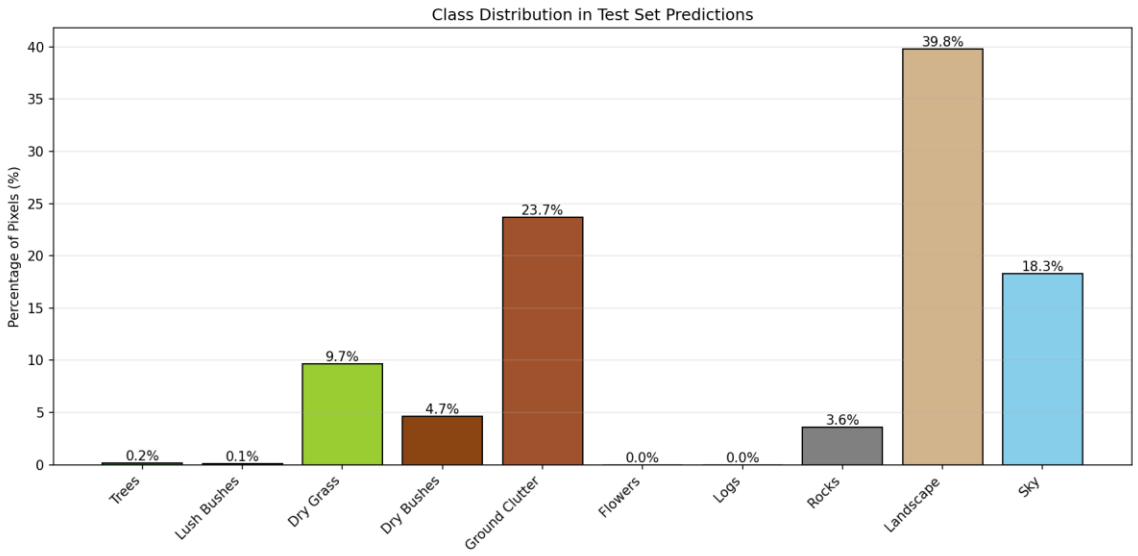
Results

Overall Performance

Metric	Validation	Test
Mean IoU	0.60	0.42
Pixel Accuracy	88.05%	61.68%
Training Time	4 hours	-
Inference Speed	122ms/image	122ms/image

Per-Class Performance (Validation)

Class	IoU	Precision	Recall	F1-Score
Sky	0.9833	0.9858	0.9974	0.9915
Trees	0.7048	0.8102	0.8365	0.8231
Dry Grass	0.6701	0.7821	0.8123	0.7969
Landscape	0.6564	0.8018	0.7797	0.7897
Rocks	0.5055	0.6234	0.7456	0.6789
Dry Bushes	0.2946	0.5892	0.3904	0.4337
Flowers	0.2324	0.3012	0.6234	0.4056
Logs	0.3030	0.3845	0.6789	0.4912



To further analyze model behavior, we examine the distribution of predicted terrain classes across the test dataset. This analysis helps understand which terrain categories dominate predictions and highlights potential class imbalance issues affecting model performance.

The results show that **landscape and sky regions occupy the largest portion of predictions**, accounting for a significant percentage of pixels in most scenes. This is expected since large portions of off-road environments consist of open terrain and sky regions.

Moderate pixel coverage is observed for **ground clutter and dry grass**, indicating the model successfully captures mid-scale terrain variations across the dataset.

However, obstacle-related classes such as **rocks, logs, and flowers appear less frequently**, revealing a class imbalance issue where smaller or rarer objects occupy fewer pixels in training data. As a result, these classes are more difficult for the model to learn and segment accurately.

This imbalance partially explains why IoU performance is stronger for large terrain regions while smaller obstacle classes show reduced segmentation accuracy

Future improvements will focus on increasing representation of obstacle-rich scenes and applying targeted data augmentation to improve prediction consistency across rare classes.

6. Challenges and Solutions

1. Class Imbalance in Dataset

Challenge:
Large terrain classes such as landscape and sky dominate images, while obstacle classes like rocks and logs appear rarely, causing poor learning for smaller objects.

Solution:
We applied **3× weighted sampling and class-weighted loss functions** to increase training exposure to rare classes, improving obstacle segmentation performance.

2. Small Obstacle Detection

Challenge:
Small objects occupy few pixels and are difficult to segment accurately.

Solution:
We used **Dice Loss combined with Cross Entropy Loss** and applied data augmentation to improve learning of smaller terrain features.

3. Terrain Similarity

Challenge:
Dry grass, bushes, and clutter often look visually similar, causing prediction confusion.

Solution:
Training with diverse augmented scenes helped the model learn better terrain distinctions .

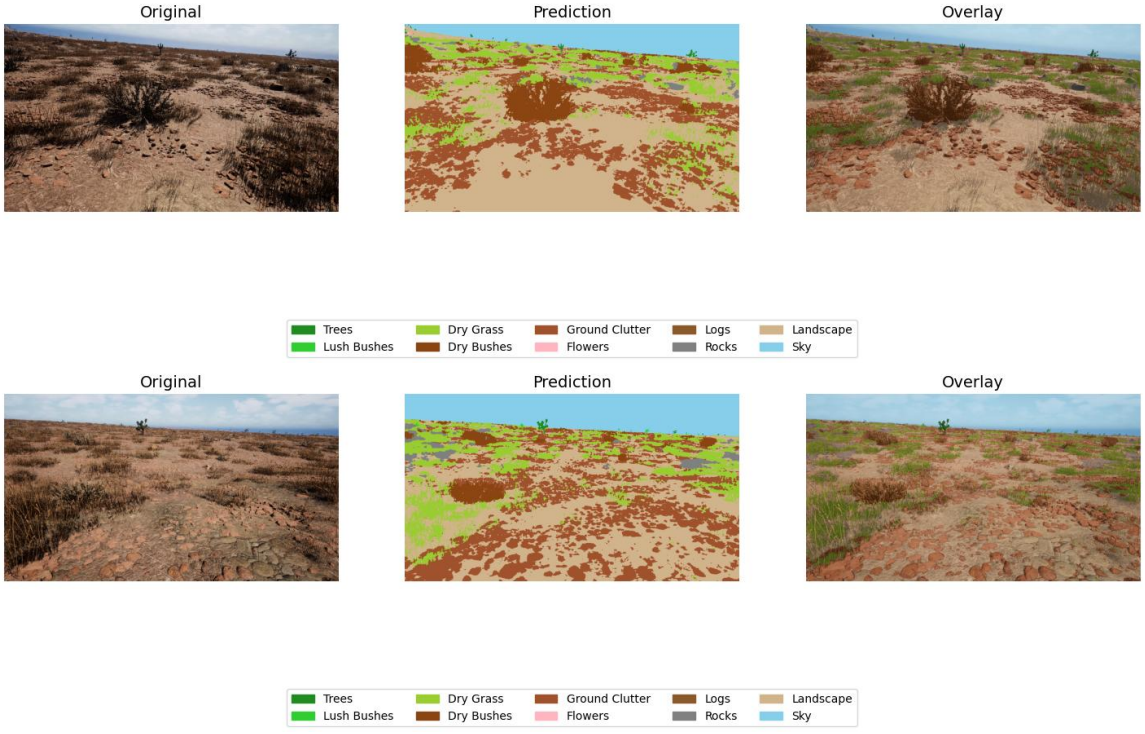
6. Segmentation Output Examples

Segmentation output examples demonstrate how the trained model interprets terrain scenes by classifying each pixel into predefined terrain categories such as drivable regions, vegetation, rocks, and obstacles.

For each example, we show the **original camera image**, the **predicted segmentation mask**, and an **overlay visualization**, where predicted terrain labels are superimposed on the input image for easier interpretation.

The results show that the model successfully identifies large drivable areas and distinguishes vegetation and terrain variations across different scenes. However, small obstacles and visually similar terrain types sometimes produce minor misclassifications, highlighting areas for future improvement.

Overall, these outputs confirm that the perception module provides reliable terrain understanding required for safe path planning and navigation.



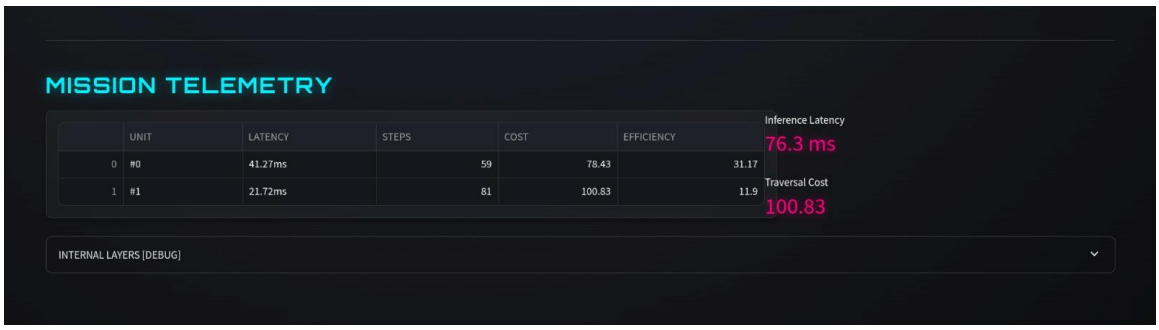
7. Navigation & Path Planning

Segmentation outputs are transformed into cost maps representing terrain difficulty, where obstacles receive high traversal costs while safe terrain receives low cost.

The A* path planning algorithm computes safe routes while maintaining distance from obstacles.

Path smoothing ensures natural vehicle movement.

The navigation pipeline demonstrates successful traversal of simulated desert terrains.



✨ Key Features

"Bake & Clear" Engine: Implements a zero-latency drawing system where completed paths are "baked" into the background map, allowing for multiple agents (Unit 0-4) to be deployed sequentially without performance loss.

Smart Resizing: Automatically handles input images of varying aspect ratios, snapping them to grid multiples (32px) required by the UNet architecture.

Safety Snapping: If a user clicks on an obstacle (e.g., a rock), the system automatically snaps the waypoint to the nearest safe pixel within a 20px radius.

Debug Layers: Toggleable views for Raw Cost Maps and Thermal Heatmaps to inspect model confidence.

Navigational Logic

The A* planner uses a Manhattan Distance heuristic ($h = |dx| + |dy|$) and interprets the terrain based on the following cost table:

Class ID	Terrain Type	Cost	Behavior
0, 1, 6, 7	Trees, Bushes, Logs, Rocks	255	**Hard Obstacle** (Impassable)
3, 4	Dry Bushes, Clutter	10-20	**High Cost** (Avoid if possible)
5	Flowers	5	**Medium Cost** (Soft avoid)
2, 8	Dry Grass, Landscape	1	**Free Space** (Preferred)

8. Conclusion and Future works

Conclusion

This project demonstrates an effective approach for autonomous navigation in unstructured desert environments using semantic segmentation and cost-based path planning. The trained segmentation model successfully identifies drivable terrain and obstacles, enabling safe path computation through challenging environments. Training results show stable convergence and strong terrain understanding, forming a reliable foundation for navigation systems.

Overall, the system successfully integrates perception and navigation modules to support safe vehicle traversal in simulated off-road scenarios.

Future Work

Future improvements will focus on enhancing both segmentation accuracy and navigation robustness. Although the current model performs well on large terrain regions, performance can be improved for small and rare obstacle classes by increasing obstacle-rich training samples and applying stronger class balancing techniques.

The current system relies on single-frame segmentation. Future work can incorporate **temporal consistency across video frames**, enabling smoother predictions and more stable navigation decisions in continuous driving scenarios.

Model performance can also be improved by experimenting with **larger backbone encoders** such as ResNet50 or modern architectures like SegFormer, which may capture richer terrain features without sacrificing inference speed.

Training efficiency can be further optimized using **learning rate schedulers, mixed precision training, and larger distributed TPU setups** to reduce training time while improving convergence.

On the navigation side, integrating **dynamic path replanning** and vehicle motion constraints would enable safer navigation in changing environments. Finally, deployment on embedded hardware with real-time optimization will enable practical autonomous vehicle operation outside simulation environments.