

GA-Nav: Efficient Terrain Segmentation for Robot Navigation in Unstructured Outdoor Environments

Tianrui Guan, Divya Kothandaraman, Rohan Chandra, Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, and Dinesh Manocha

(Code and Videos at <https://gamma.umd.edu/offroad>)

Abstract—We present an efficient learning-based method for identifying safe and navigable regions in off-road terrains and unstructured environments from RGB images. Our approach classifies terrains based on their navigability levels using coarse-grained semantic segmentation. We propose GA-Nav, a novel group-wise attention mechanism to distinguish between navigability levels of different terrains, which can improve different backbone designs. Our group-wise attention loss enables the network to explicitly focus on the different groups’ features with low spatial resolution for efficient inference while maintaining a high level of accuracy compared to other SOTA methods. We show through extensive evaluations on the RUGD and RELLIS-3D datasets that our learning algorithm improves visual perception accuracy in off-road terrains for navigation. We compare our approach with prior work on these datasets and achieve an improvement over the state-of-the-art mIoU by 2.25–39.05% on RUGD and 5.17–19.06% on RELLIS-3D. In addition, we deploy GA-Nav on a Clearpath Jackal and a Husky robot for real-world navigation demonstrations. Our approach improves the performance of the navigation algorithm in terms of success rate by 10% and results in smoother trajectories while maintaining the best surface selection capabilities above 80%. Further, our method reduces the false positive rate of forbidden regions by 37.79%. Code, videos, and a full technical report are available at gamma.umd.edu/offroad.

I. INTRODUCTION

Recent developments in autonomous driving and mobile robots have significantly increased the interest in outdoor navigation [1], [2]. In many applications such as mining, disaster relief [3], agricultural robotics [4], or environmental surveying, the robot must navigate in uneven terrains or scenarios that lack a clear structure or well-identified navigation features.

A key issue in developing autonomous navigation capabilities in off-road environments is finding safe and navigable regions that can be used by a mobile robot. For instance, terrains like concrete or asphalt are smooth and highly navigable, while rocks or gravel are typically bumpy and may not be navigable. Our goal in this work is to learn to visually differentiate various terrains in unstructured outdoor environments and classify them based on their navigability.

An important aspect of classifying navigable regions in unstructured environments is using visual perception capabilities such as semantic segmentation [5], [6], [7], [8]. Semantic segmentation is a pixel-level task that assigns a label for every pixel in the image. Prior works in segmentation [5], [6] have been limited to structured environments [1], [2] and may not be well-suited, in general, for aiding robot navigation in off-road terrains for several reasons.

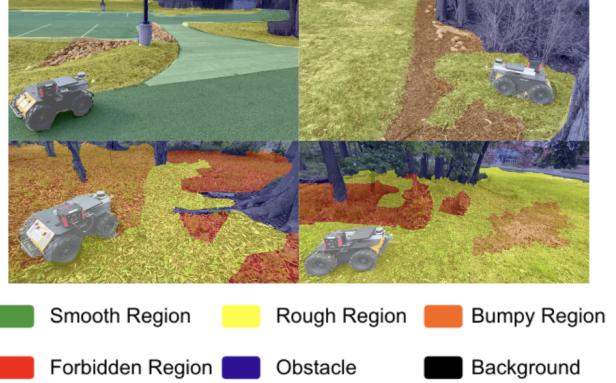


Fig. 1: We highlight the performance of our algorithm on unstructured outdoor terrains. We classify the navigable regions (shown in different colors) using our segmentation algorithm that uses RGB images as the input in real time. The top images correspond to different outdoor terrains where the robots are navigating in an autonomous manner. The bottom images show the terrain segmentation results. Our model achieves an improvement of 2.25 – 39.05% in terms of mIoU over prior segmentation algorithms on complex outdoor terrains and improves navigation in the real world by 10% in terms of success rate.

Firstly, many off-road scenes require differentiating terrain classes that have highly similar appearances (*e.g.*, water and a puddle) with overlapping boundaries. In some cases, certain terrain classes could be occupying a small portion of the image, which require highly accurate segmentation. In terms of existing methods [5], [6], [7], [8], such scenarios lead to similar feature embeddings for multiple classes that can ultimately result in wrong classifications. Furthermore, for navigation, misclassifications of dangerous or forbidden regions for the robot could lead to disastrous consequences.

Secondly, perception methods that aid off-road robot navigation must manage their computational efficiency without compromising on their segmentation accuracy. Thirdly, it has been shown that in unstructured environments, navigation systems may not require fine-grained semantic segmentation [9], [10]. For example, it is sufficient to recognize trees and poles as obstacles for collision avoidance rather than segmenting them individually as different types of objects. A coarser approach that classifies various types of objects based on their navigation characteristics is more appropriate for aiding navigation, which is the focus of our method.

Main Results: We present GA-Nav, a novel and efficient learning-based approach for identifying the navigability of different terrains from RGB images or videos. Our approach

is designed to identify different *terrain groups* in off-road and unstructured outdoor terrains. We present a novel architecture that finds a good balance between segmentation accuracy and computational overhead and outperforms existing transformer-based segmentation methods. Unlike previous methods [11], [12], [7], [8], which only focus on fine-grained classification, our method implicitly learns how to classify and cluster different classes simultaneously according to annotations, which leads to improved accuracy among different terrains. We show that our method has an advantage over existing methods on coarse-grained segmentation task used for navigation. To demonstrate GA-Nav’s benefits for real-time robot navigation, we integrate it with a learning-based navigation approach and evaluate the performance on Clearpath Jackal and Husky robot in unstructured outdoor environments.

The key contributions of our work include:

- 1) We propose a novel architecture which fuses a multi-scale feature extractor with a transformer architecture. Our group-wise segmentation head can fuse visual features from different scales and explicitly focus on different terrain types, which leads to better accuracy on different surfaces of varying areas. Our approach is compatible with different feature extractors and can improve performance on various SOTA multi-scale backbone designs. We outperform prior methods for semantic segmentation and navigable region classifications in terms of mIoU by $2.25 - 39.05\%$ on RUGD and $5.17 - 19.06\%$ on RELLIS-3D.
- 2) We introduce a group-wise attention loss for fast and accurate inference. We show that with our group-wise attention (GA) loss, we can improve the inference complexity without much performance degradation. Our proposed method results in better performance than existing transformer-based and other segmentation methods with one of the lowest run-times and complexity requirements.
- 3) We integrate GA-Nav with TERP[13], an outdoor navigation planner to highlight the benefits of our segmentation approach for robot navigation in complex outdoor environments. We show that our modified navigation method outperforms other navigation methods by 10% in terms of success rate and, 2-47% in terms of selecting the surface with the best navigability and a decrease of 4.6-13.9% in trajectory roughness. Further, GA-Nav reduces the false positive rate of forbidden regions by 37.79%.

II. RELATED WORK

A. Semantic Segmentation

Semantic segmentation is an important task in computer vision that involves assigning a label to each pixel in an image. The problem has been widely studied in the literature [14]. Many deep learning architectures have been proposed, including OCRNet [6], PSPNet [5], etc. Most recently, there have been many transformer-based methods [15], [11] that demonstrate better performance than

other CNN-based methods but are more computationally expensive. Therefore, there has been increased discussion of more efficient design based on transformers for fast inference time and lower computational cost, including [7], [8]. While most of these architectures work well on structured datasets like CityScapes, they do not work well for off-road datasets due to ill-defined boundaries and confusing class features. Segmentation methods like Global Convolution Networks [16] are designed to deal with complex boundaries and do not scale well in terms of performance due to convoluted and overlapping boundaries that are generally not found in structured datasets.

B. Navigation in Outdoor Scenes

Prior work in uneven terrain navigation includes techniques for everything from mobile robots [9], [17], [18], [19] to large vehicles [10], [20], [21]. In these algorithms, learning the navigation characteristics of the terrain and its traversability is a crucial step. At a broad level, there are three types [22] of approaches that are used to determine the terrain features: 1) proprioceptive-based methods, 2) geometric methods, and 3) appearance-based methods. The proprioceptive-based methods [23], [24] use frequency domain vibration information gathered by the robot sensors to classify the terrains using machine learning techniques. However, these methods require the robot to navigate through the region to collect data and assume that a mobile robot can navigate over the entire terrain. The geometric-based methods [21], [20], [25] generally use Lidar and stereo cameras to gather 3D point cloud and depth information of the environment. This information can be used to detect the elevation, slope, and roughness of the environment as well as obstacles in the surrounding area. Nevertheless, the accuracy of the detection outcome is usually governed by the range of the sensors. Appearance-based methods [26] usually extract road features with SIFT or SURF features and use MLP to classify a set of terrain classes. Our goal is to focus on the visual perceptive, leveraging the performance and efficiency for robot navigation. Our method computes the navigable regions from RGB images in unstructured environments and outputs a safe trajectory in this condition.

C. Off-road Datasets

Recent developments in semantic segmentation have achieved high accuracy on object datasets like PASCAL VOC [27], COCO [28], and ADE20K [29], as well as driving datasets like Cityscape [1] and KITTI [2]. However, there has not been much work on recognition or segmentation in unstructured off-road scenes, which is important for navigation. Perception in an unstructured environment is more challenging since many object classes (e.g., puddles or asphalt) lack clear boundaries. RUGD [30] and RELLIS-3D [31] are two recent datasets available for off-road semantic segmentation. The RUGD dataset consists of various scenes like trails, creeks, parks, and villages with fine-grained semantic segmentation annotations. The RELLIS-3D dataset is derived from RUGD and includes unique terrains

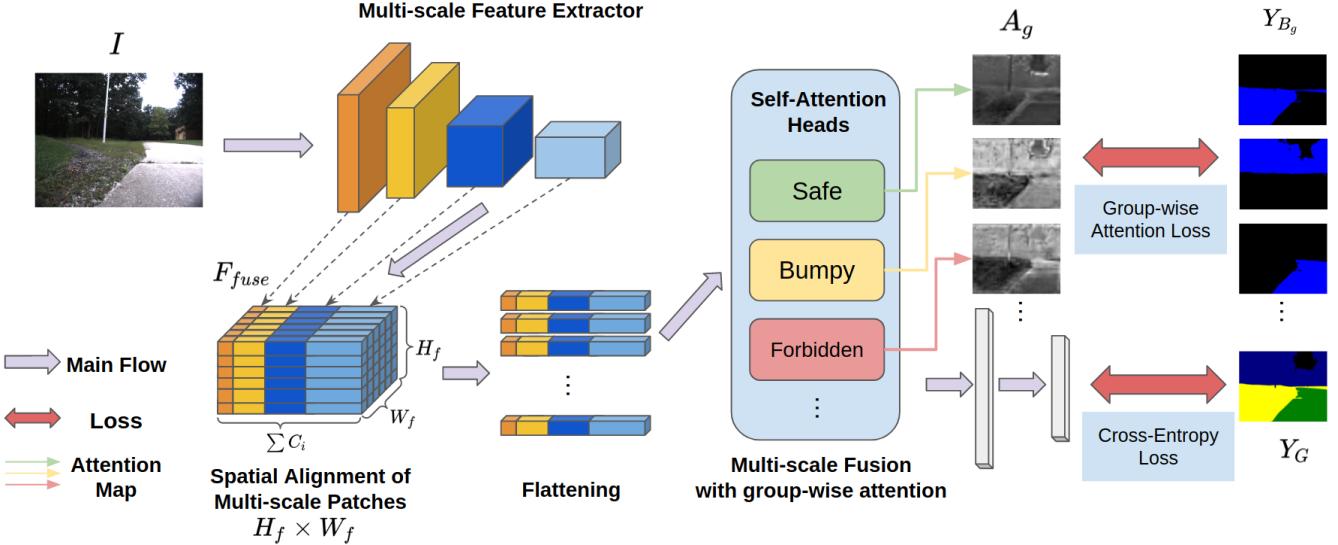


Fig. 2: **Architecture of our proposed network GA-Nav:** We introduce a novel group-wise attention head for multi-scale fusion. After we convert multi-scale features into several groups of features, the group-wise attention loss acts between the multi-head attention outputs A_g corresponding to group g and the corresponding group-wise binary ground truth Y_{B_g} to explicitly guide the network towards accurate predictions of different groups.

like puddles. In addition, RELLIS-3D includes Lidar data and 3D Lidar annotations. We use these datasets to evaluate the performance of our algorithm.

III. GA-NAV FOR TERRAIN CLASSIFICATION

We present GA-Nav, an approach for classifying the navigability of different terrains in off-road environments via coarse-grained segmentation. Most of the existing segmentation methods fail in unstructured environments due to highly similar visual features and unclear boundary of the objects. Use of multi-head self-attention is popular in neural network design due to its superior performance over prior state-of-the-art methods in segmentation [15], [11], image classification [32], and object detection [28], [33], [34], but the quadratic nature of its computational complexity has been an issue for real-time use like navigation.

To solve performance degradation in unstructured environment and computation limitation, we design a simple and lightweight architecture, which uses self-attention at the multi-scale fusion step and generated attention maps to calculate loss for better performance and low level of computational overhead.

The rest of this section is structured as follows. We begin by stating our problem in Section III-A. We then discuss the various components of our approach, namely the backbone architecture (Section III-B), GA-Nav segmentation head (Section III-C), and an attention-based loss function (Section III-D). Finally, we clarify the difference between GA-Nav and other attention-based methods and briefly explain its integration with planning and navigation modules for navigation demonstration.

A. Problem Definition

The input consists of an RGB image $I \in \mathbb{R}^{3 \times H \times W}$ and the corresponding ground-truth semantic segmentation labels

$Y \in \mathbb{Z}^{H \times W}$ denoting the category to which each pixel belongs among G different groups. We use new coarse-grain labels $Y_G \in \mathbb{Z}^{H \times W}$ based on semantic labels provided by the dataset. For each group, we also compute the binary mask $Y_{B_g} \in \{0, 1\}^{H \times W}$ for $g = 1, \dots, G$. Our goal is to perform coarse-grain semantic segmentation on terrain images I and compute probability maps $P \in \mathbb{R}^{G \times H \times W}$. Each entry in P is a probability distribution of a pixel in that entry location belonging to one of the G groups.

B. Backbone Design

Vision Transformers (ViTs) [33], [35] are a type of deep neural network (DNN) for feature extraction and are designed as an alternative to the widely used ResNet [36] architecture via Multi-Head Self-Attention (MHSA). MHSA, or self-attention applied on image patches, is the main component of ViTs, contributing to much of their success over ResNets in computer vision-based tasks. Our approach also uses a transformer-based backbone architecture and leverages the MHSA module to extract and fuse multi-scale features. Our architecture can adapt to any backbone design with multi-scale features; in particular, we use Mixed Transformer (MiT) [7] with some modifications as our backbone. In addition, to show the advantages of our design, we also show two alternative backbones with our design, ResNet50 [36] and Bottleneck Transformer (BoT) [33].

Given an RGB image $I \in \mathbb{R}^{3 \times H \times W}$, we first divide it into several patches as input to the transformer encoders. Unlike ViT and MiT, which take non-overlapping patches of shape 16 by 16 and 4 by 4, respectively, we choose patches of size 7 by 7 with a stride of 4 in our design. After passing the patches to the first transformer encoder blocks, we continue to take 2 by 2 patches on the output features. Therefore, after the initial encoders, each following block of transformer encoders reduces the spatial resolution

further by a factor of 2 on both height and width. After each transformer encoder block, we obtain multi-scale features F_i with spatial resolution $H_i \times W_i = \frac{H}{2^{i+1}} \times \frac{W}{2^{i+1}}$, where the feature channel $C_i = \{32, 64, 160, 256\}$ and $i = \{1, 2, 3, 4\}$. We utilize all those multi-scale features in our segmentation head.

C. GANav Segmentation Head

The RGB image I is passed through the feature extractor to obtain multi-scale feature maps $F_i \in \mathbb{R}^{C_i \times H_i \times W_i}$ after i -th encoder block.

Spatial Alignment and Resolution Reduction: To prepare for our fusion, we need to combine features at the same spatial locality. We use bi-linear interpolation to resize F_i into shape $(C_i \times H_f \times W_f)$, where H_f and W_f are equal to H_i and W_i with any chosen i for all scales. The final feature before fusion, F_{fuse} , is concatenated over channel dimension with shape $(\sum C_i \times H_f \times W_f)$.

To reduce the computational complexity, our efficient design prefers to align patch features from different scales to one scale with a smaller spatial resolution as the network bottleneck (i.e., choose a smaller H_f and W_f). As we expect, as we reduce H_f and W_f , the spatial resolution of F_{fuse} reduces so the complexity goes down with some performance degradation. In Section III-D, we also introduce the group-wise attention loss to not only improve the overall performance but also mitigate the performance drop from choosing a smaller spatial resolution.

Multi-scale Fusion with Group-wise Attention: Given an input feature vector $A_{in} \in \mathbb{R}^{N \times C_{in}}$, the output self-attention map $A_{out} \in \mathbb{R}^{N \times C_{out}}$ is computed as follows:

$$A_{out} = \text{softmax}(k(A_{in})^T \cdot q(A_{in}))^T \cdot v(A_{in}), \quad (1)$$

where $k(A_{in})$, $q(A_{in})$, $v(A_{in})$ represent the key, query, and value feature maps, respectively, and k , q , v are linear projections, as in the self-attention [37] literature.

This feature map F_{fuse} is reshaped (and transposed) into $F_{flat} \in \mathbb{R}^{H_f W_f \times \sum C_i}$, which is passed through the MHSA block with G attention heads. The MHSA component fuses flatten multi-scale feature F_{flat} to produce the new feature maps $F_{out} \in \mathbb{R}^{C_{out} \times H_f \times W_f}$ and generates G attention maps (one for each group), $A_1, A_2, \dots, A_G \in [0, 1]^{H_f W_f \times H_f W_f}$. The final output $P \in \mathbb{R}^{G \times H \times W}$ is obtained through a standard procedure of a segmentation network by a series of 1×1 convolutions and up-sampling from F_{out} .

As demonstrated in Fig. 2, in our detection head, we use MHSA to fuse multi-scale features and generate different attention maps for each group. We use those attention maps as an additional branch in the detection head and train the detection head to resemble the group distribution by explicitly guiding each attention map towards a corresponding category using a binary cross-entropy loss. Our network skips the attention map processing branch during inference time for efficiency.

Attention maps can also capture the relevancy between two pixel locations. Intuitively, $A_{g,[x,y]}$ represents the amount of attention that the pixel at the x^{th} position must pay to

the pixel at the y^{th} position. Given an attention map A_g , its main diagonal represents the relevance of each location with respect to the attention head h_g . We train the attention network to learn feature maps that are most relevant to its corresponding group through group-wise attention loss.

We use group-wise attention as a fusion method for multi-scale features, which has been shown to be quite powerful in combining different modalities [38], different scales and representations [34], and temporal information [39] for various tasks. We will show that our novel design of group-wise attention for multi-scale fusion can outperform most existing transformer-based methods with outstanding efficiency on coarse-grain segmentation tasks in challenging off-road environments.

D. Group-wise Attention Loss

Whenever we choose to reduce the resolution during the spatial alignment step, performance degradation is unavoidable. Based on the structure of our group-wise attention head, we propose group-wise attention loss to boost the performance and mitigate such effects.

For each attention head h_i , we have its corresponding attention map $A_i \in [0, 1]^{L \times L}$, where $L = H_f \times W_f$. We take the main diagonal of A_i and reshape it to $B_i \in [0, 1]^{H \times W}$ using bi-linear image resizing. Each pixel in B_i represents the self-attention score with respect to h_i .

To guide each attention-map in the multi-head self-attention module, we apply a binary cross-entropy loss function:

$$\mathcal{L}_{GA}^g = - \sum_{h,w} y_G \log(B_g), \quad (2)$$

where $y_G \in Y_G$. This equation calculates loss between the predictions of the self-attention output and the corresponding group's binary ground truth with respect to the group.

The purpose of group-wise attention loss is not to use attention maps to accurately predict the group's distribution; instead, as an intermediate layer, it aims to provide guidance regarding which region an attention head should focus on for further classification.

As in traditional segmentation models, we optimize our network with a multi-class cross-entropy loss and an auxiliary loss:

Cross-Entropy Loss: This is the standard semantic segmentation cross-entropy loss, defined as follows:

$$\mathcal{L}_{CE} = - \sum_{h,w} \sum_{g \in G} y_{GT} \log(P_g), \quad (3)$$

where h, w represent the dimensions of the image, G represents the set of groups, P_g denotes the output probability map corresponding to group g , and y_{GT} corresponds to the ground-truth annotations.

Auxiliary Loss via Deep Supervision: Deep supervision was first proposed in [40] and has been widely used in the task of segmentation [5], [6], [41]. Adding an existing good-performing segmentation decoder head like FCN [42] in parallel to our GA head during training not only provides strong regularization but also reduces training time.

E. GA-Nav v.s. Prior Methods

In this section we highlight the difference between some transformer-based methods and the proposed method, GA-Nav.

SETR [15], DPT [11]: The designs are significantly different, particularly regarding the computational overhead of those methods. In addition, their method uses positional embedding in the encoding stage while our method removes any positional embedding in the backbone.

Segformer [7], Segmenter [8]: Those methods, as well as GA-Nav, also focus on efficient training and inference. Segformer does not have any parameters dedicated to different classes except for the final classification layer. In addition, the decoder consists of several MLP layers for simple and efficient design. Segmenter and GA-Nav both have trainable parameters dedicated to specific classes/groups: Segmenter uses G class embeddings that correspond to each semantic class and appends those tokens to the end of the entire image patch sequence for the decoder, while GA-Nav splits feature channels into G portions of equal size for different attention heads that belong to different groups. Finally, GA-Nav uses a smaller spatial resolution in the decoder bottleneck than the other two methods to improve the run-time and compensated for the complexity brought by MHSAs.

F. GA-Nav-based Segmentation for Robot Navigation

We combine GA-Nav with TERP [13], an outdoor navigation planner to highlight the benefits of our segmentation algorithm for robot navigation. TERP utilizes the robot’s pose, goal information and a point cloud based elevation map to generate a cost map from an attention based perception module. Before interfacing with TERP, we convert GA-Nav’s segmentation results into a segmentation cost map. This is performed based on the following steps: 1. Projecting GA-Nav’s segmented output onto the ground plane in front of the robot using homography transformations, and 2. Assigning discrete cost values for the different groups. We assign zero cost for smooth regions and increase the non-zero costs for rough, bumpy, forbidden and obstacle regions.

The computed segmentation cost map is element-wise added with the elevation cost map generated from TERP’s perception module to obtain the final navigation cost map. This resulting cost map is fed into TERP’s navigation module, which computes least-cost trajectories towards the robot’s goal. The overall system architecture is highlighted in Fig. 3.

IV. RESULTS AND ANALYSIS

A. Implementation Details

We use two off-road terrain datasets, RUGD [30] and RELLIS-3D [31], as benchmarks for all experiments and comparisons. Optimization is done using a stochastic gradient descent optimizer with a learning rate of 0.01 and a decay of 0.0005. We adopt the polynomial learning rate policy with a power of 0.9. We augment the data with horizontal random flip and random crop. In the ablations and comparisons, for fairness, we benchmark all our models at 240K iterations.

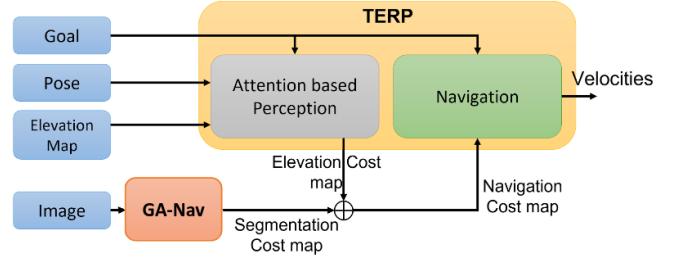


Fig. 3: GA-Nav-Based Outdoor Navigation: We integrate GA-Nav with an elevation based planner TERP [13] to highlight benefits of our segmentation algorithm. Our architecture decouples semantic segmentation and elevation data for better perception and planning. The advantages of this formulation are highlighted in Section IV-F.

For the RUGD dataset, we use a batch size of 8 and a crop size of 300×375 (the resolution of the original image is 688×550). On the other hand, for the RELLIS-3D dataset, due to the high resolution of the image, we train models with a batch size of 2 and a crop size 375×600 (the resolution of the original image is 1920×1600). Our implementation is based on MMSEG [43] and will be released.

We benchmark our models on the standard segmentation metrics: Intersection over Union (IoU), mean IoU (mIoU), mean pixel accuracy (mAcc), and average pixel accuracy (aAcc). For an image I , let $P(x, y)$ be the predicted label at pixel location (x, y) , $G(x, y)$ be the ground truth label at (x, y) , $\mathbb{1}(X)$ be the indicator function, and B be a set of the class labels:

Intersection over Union (IoU) for class i :

$$IoU_i = \frac{\sum_I \sum_{x,y} \mathbb{1}(P(x, y) = i \text{ and } G(x, y) = i)}{\sum_I \sum_{x,y} \mathbb{1}(P(x, y) = i \text{ or } G(x, y) = i)}$$

Mean IoU (mIoU):

$$mIoU = \frac{\sum_i mIoU_i}{\sum_B 1}$$

Mean Pixel Accuracy (mAcc):

$$mAcc = \frac{\sum_{i \in B} (\sum_I \sum_{x,y, G(x,y)=i} \mathbb{1}(P(x, y) = G(x, y)))}{\sum_B 1}$$

Averaged Pixel Accuracy (aAcc):

$$aAcc = \frac{\sum_I \sum_{x,y} \mathbb{1}(P(x, y) = G(x, y))}{\sum_I \sum_{x,y} 1}$$

We evaluate our methods with 6 coarse-grain semantic classes: smooth (concrete, asphalt), rough (gravel, grass, dirt), bumpy (small rocks, rock-bed), forbidden (water, bushes), obstacle, and background. Some visualizations for method comparisons are provided in Fig. 4.

B. Ablation Studies

Effect of Multi-scale Fusion Design: In Table I, we show the effect of our multi-scale fusion design with group-wise attention. We show an improvement of 14.76% and 11.43% in terms of mIoU on RUGD and RELLIS-3D, respectively.

Group-wise Attention (GA) Design: In Table II, we show the effect of our segmentation head design with different

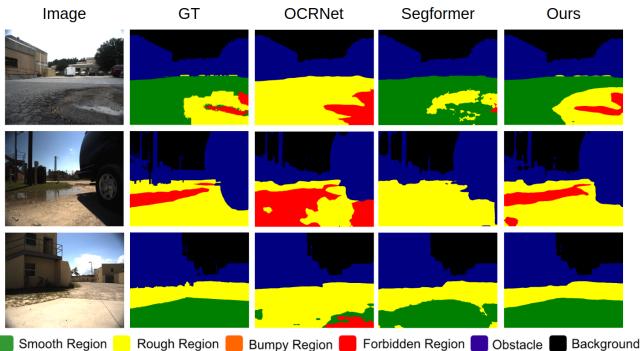


Fig. 4: Qualitative Results: Each row highlights the original image, the ground truth (GT) label, the predictions of two SOTA methods, and the prediction of our method. We observe that our method can detect different navigable regions and is robust with respect to objects with similar colors.

Dataset	Multi-scale	Fusion method	mIoU \uparrow	mAcc \uparrow
RUGD	\times	-	74.32	80.41
	\checkmark	Linear	77.64	86.78
	\checkmark	GW-Attn.	89.08	93.55
RELLIS-3D	\times	-	63.01	68.7
	\checkmark	Linear	68.91	81.5
	\checkmark	GW-Attn.	74.44	86.49

TABLE I: Ablation studies on multi-scale fusion on RUGD and REELLIS-3D val sets: The “Multi-scale” column shows whether we use a single or all scale features from the feature extractor, and the “Fusion method” column shows either we use a simple linear layer or the proposed segmentation head for the fusion.

backbones on RUGD and REELLIS-3D datasets. We observe that, regardless of the backbone choice, group-wise attention improves performance (in terms of mIoU) by 6.67 – 10.85% and 4.18 – 10.31% on RUGD and REELLIS-3D, respectively. **Ablation on Spatial Reduction with GA Head:** In Table III, we show the relations between performance and resolution change during spatial alignment. We can see that without GA heads, as we change the resolution to 1/16 and 1/32, there are performance drops of 1.3% and 4.0% in terms of mIoU on REELLIS-3D. The performance drops do not appear for RUGD when we change the spatial reduction from 1/8 to 1/16; the main reason may be that the overall performance is low for RUGD before using GA heads, or the original resolution for RUGD is lower than REELLIS-3D, so the performance drop due to spatial information loss has not started to appear. We can still see the performance drop when reducing it to 1/32. After we add the GA component, the mIoU performance improves by 7.46 – 10.85% and 2.85 – 5.85% on RUGD and REELLIS-3D, respectively. In addition, we no longer see the performance drop on REELLIS-3D with the proposed GA heads. Use of GA heads allows us to reduce spatial resolution for efficient inference without compromising the final performance.

C. Run-time Analysis and Computational Cost

In Fig. 5, we highlight the model efficiency of our method while maintaining SOTA performance. We can see that, even if we compromise the performance for inference rate in GA-Nav-r16 and GA-Nav-r32, our method still outperforms other

Dataset	Backbone	GW-Attn.	mIoU \uparrow	mAcc \uparrow	aAcc \uparrow
RUGD	ResNet50[36]	\times	71.55	87.67	91.63
		\checkmark	78.22	85.09	92.91
	BoT [33]	\times	74.32	80.41	93.53
RELLIS-3D	MiT [7]	\times	77.64	86.78	93.47
		\checkmark	89.08	93.55	95.66
	ResNet50[36]	\times	61.82	71.55	81.72
RELLIS-3D	BoT [33]	\times	63.01	68.7	92.2
		\checkmark	70.63	81.32	90.23
	MiT [7]	\times	68.91	81.5	85.57
		\checkmark	74.44	86.49	91.69

TABLE II: Ablation studies on GA heads with different backbones on RUGD and REELLIS-3D val sets: We observe that the group-wise attention mechanism is successful in improving mIoU by 7.26 – 11.44% and 4.18 – 10.31% on the RUGD and REELLIS datasets, respectively, regardless of the backbone choice.

Dataset	Spatial Reduction	GW-Attn.	mIoU \uparrow	mAcc \uparrow	aAcc \uparrow
RUGD	$\frac{1}{8} \times \frac{1}{8}$	\times	77.64	86.78	93.47
		\checkmark	89.08	93.55	95.66
	$\frac{1}{16} \times \frac{1}{16}$	\times	79.61	87.76	92.85
RELLIS-3D	$\frac{1}{32} \times \frac{1}{32}$	\times	76.81	85.14	92.24
		\checkmark	84.9	90.36	94.24
	$\frac{1}{8} \times \frac{1}{8}$	\times	68.91	81.5	85.57
RELLIS-3D		\checkmark	71.76	81.62	90.31
	$\frac{1}{16} \times \frac{1}{16}$	\times	67.61	79.95	84.03
		\checkmark	71.62	84.51	90.26
	$\frac{1}{32} \times \frac{1}{32}$	\times	64.91	75.56	83.51
		\checkmark	70.76	84.4	90.44

TABLE III: Ablation studies on GA heads with different spatial reduction on RUGD and REELLIS-3D val sets: We show that higher spatial reduction can decrease the perception accuracy and the group-wise attention can mitigate the performance drop due to resolution reduction. “ $\frac{1}{N} \times \frac{1}{N}$ ” in the second column means the bottleneck spatial resolution is reduced by a factor of N on height and width compared to the input resolution.

methods in terms of mIoU.

In Table IV, we give more details on model parameter size, GFLOPS, cost during inference, and run-time. While our method doesn’t always have the smallest model size and GLOPS, our method has a very low memory requirement. Our method also has the best inference time among listed transformer-based methods and most of the CNN-based methods, given that our method achieves state-of-the-art accuracy, as shown in Section IV-D.

D. Results and State-of-the-art Comparisons

The results on different coarse-grain classes are presented in Table V. Our architecture improves the state-of-the-art mIoU by 1.66 – 44.35% on RUGD. We can see that our method has the best overall mIoU scores. While GA-Nav does not always give the best results for each separate class and the performance is sometimes close to the second best IoU score, our method has very stable performance for each class. GA-Nav performs well on detecting different terrains, while other methods only have good performance on one of the terrain classes. Similarly, on the REELLIS dataset, GA-Nav demonstrates an improvement of 5.17 – 19.06% in terms of mIoU.

Method	Params ↓ (M)	GFLOPS ↓	Inf Mem ↓ (MiB)	Run-time ↑ (img/s)
PSPNet [5]	48.97	258.9	1635	21.97
DeepLabv3+ [44]	43.58	256.08	1443	43.48
DANet [37]	49.82	288.81	1425	29.97
OCRNet [6]	36.51	221.47	1407	20.76
PSANet [41]	59.13	289.52	1629	28.26
BiseNetv2 [45]	14.78	17.88	1137	90.33
CGNet [46]	<u>0.493</u>	5.05	1087	62.35
FastSCNN [47]	1.45	<u>1.35</u>	<u>1081</u>	<u>110.63</u>
FastFCN [48]	68.7	189.61	1707	47.83
*SETR [15]	309.17	312.21	2295	22.96
*DPT [11]	109.67	255.49	1633	37.03
*Segformer [7]	<u>3.72</u>	9.29	1143	<u>73.49</u>
*Segmenter [8]	6.69	<u>6.57</u>	<u>1109</u>	52.72
*GA-Nav-r8	6.94	18.69	1415	65.53
*GA-Nav-r16	6.57	7.7	<u>1121</u>	76.20
*GA-Nav-r32	<u>6.21</u>	<u>4.54</u>	1131	<u>76.74</u>

TABLE IV: **Model Complexity:** We show the model parameter size, GLOPS, GPU memory cost during inference, and run-time. For each measurement, we underline one best number among CNN-based methods, one among transformer-based methods, and one among variations of GA-Nav.

E. Outdoor Navigation

We evaluate the benefits of our GA-Nav segmentation algorithm for robot navigation in real-world outdoor scenarios using a Clearpath Jackal and a Husky robot mounted with an Intel RealSense camera. The camera has a field of view (FOV) of 70 degrees. We use an Alienware laptop with an Intel i9 CPU (2.90GHz x 12) and an Nvidia GeForce RTX 2080 GPU mounted on the robot to run GA-Nav. We also utilize a velodyne 3D Lidar for robot localization and elevation map generation. We test the navigation performance of our segmentation and elevation-based planner on a variety of terrains around a university campus, including concrete walkways, asphalt trails, dirt roads, and off-road terrains.

The following metrics are used to compare the navigation performance of our method with the Dynamic Window Approach (DWA) [49], TERP [13], Segformer [7], and OCRNet [6] based planners.

- Success Rate:** The number of successful attempts for the robot to reach its goal while avoiding relatively non-traversable regions and collisions over the total number of trials.
- Trajectory Roughness** - The cumulative vertical motion gradient experienced by the robot along a trajectory.
- Trajectory Selection:** The percentage of the robot’s trajectory length along the most navigable surface out of the total trajectory length (i.e., the selection of smoother regions over others).
- False Positive of the Forbidden Region** - The percentage of the frames that include a misclassified forbidden region out of all the frames during navigation.

F. Navigation Comparisons

The navigation results for GA-Nav, DWA [49], TERP [13], Segformer [7], and OCRNet [6] based planners are shown in Table VI. We observe that all three segmentation-based planners perform well in terms of success rate under Walkway and Trail scenarios due to the clear margins between

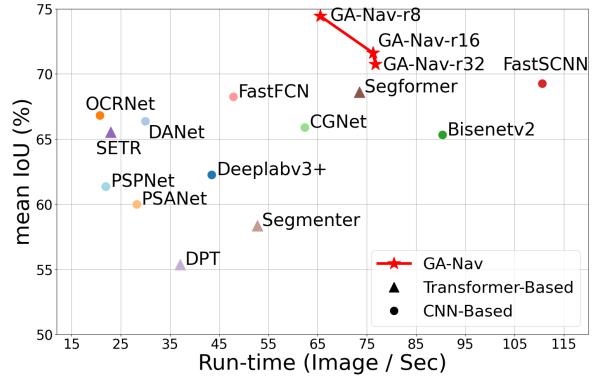


Fig. 5: **Performance vs. Inference Time on RELLIS-3D:** The run-times are calculated based on the average time of 200 images.

the different surfaces. Even though Segformer [7] and OCRNet [6] display reasonable performances under scenarios similar to trained datasets, we observe significant performance degradation in terms of trajectory selection during off-road conditions. However, GA-Nav displays significant improvement in terms of all three metrics under cluttered and unseen scenarios in Fig. 6 (c) and 6 (d). Specifically, accurate segmentation from GA-Nav increases the utilization of the most navigable surface during navigation. This behavior leads to better trajectory selection percentages than the other methods, as shown in the Table VI.

Our baseline planner and TERP incorporate terrain elevation data to perform the navigation task. Hence, such methods cannot differentiate between the surface properties in multi-terrain scenarios with similar elevation conditions. This leads to trajectory generation along less navigable surfaces, in contrast to segmentation-based methods. Therefore, we observe a noticeable degradation in the trajectory smoothness. However, the usage of segmentation data for the baseline planner significantly enhances the navigable trajectory selection capabilities of the algorithm.

V. CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

We present a learning-based method for classifying various terrain types in off-road environments. We propose a novel and efficient segmentation network that can provide robust predictions and distinguish regions with different semantic meanings using group-wise attention. We demonstrate improvement in both accuracy and efficiency over SOTA methods on complex, unstructured terrain datasets and show that our method can be used in real-world scenarios and result in higher success rate for navigation.

Our approach has some limitations. For example, our current work focuses on perception and integrates with a learning-based planner implementation for real-world scenarios. As a result, our current performance analysis is mostly based on perception metrics and not takes into account many navigation metrics. In addition, we need to make sure that the data is labeled correctly and consistently. For example, trees can either be obstacles or background objects. An annotated dataset might not label them separately.

Dataset	Methods (IoU)	Smooth Region	Rough Region	Bumpy Region	Forbidden Region	Obstacle	Background	mIoU↑	aAcc↑
RUGD	PSPNet [5]	48.62	88.92	69.45	29.07	87.98	78.29	67.06	92.85
	DeepLabv3+ [44]	5.86	84.99	50.40	25.04	87.50	81.47	55.88	91.51
	DANet [37]	2.26	81.47	8.69	15.00	82.54	74.86	44.14	88.81
	OCRNet [6]	66.29	89.47	76.15	59.14	88.77	79.17	76.50	93.46
	PSANet [41]	34.92	87.70	35.64	8.66	86.95	78.97	55.47	92.13
	BiseNetv2 [45]	24.27	89.99	89.99	83.31	90.93	75.29	75.1	93.4
	CGNet [46]	40.84	90.39	85.67	76.21	89.75	74.48	76.22	93.29
	FastSCNN [47]	83.03	92.82	87.69	81.05	90.94	75.11	85.11	94.77
	FastFCN [48]	26.27	89.85	85.95	<u>84.13</u>	<u>91.23</u>	75.63	75.51	93.46
	*SETR [15]	89.77	92.46	84.58	70.33	89.55	70.47	82.86	94.09
	*DPT [11]	1.04	81.23	22.98	25.84	89.18	74.5	49.13	88.77
	*Segformer [7]	<u>93.26</u>	<u>93.16</u>	87.56	77.31	91.2	78.5	<u>86.83</u>	<u>95.17</u>
	*Segmenter [8]	90.39	91.17	83.96	65.43	87.8	68.17	81.15	93.22
	*GA-Nav-r8	95.15	94.45	<u>89.83</u>	86.25	91.95	<u>76.86</u>	89.08	95.66
	*GA-Nav-r16	93.68	93.95	88.73	83.86	90.86	71.36	87.07	94.99
	*GA-Nav-r32	92.76	93.28	87.44	79.9	89.55	66.46	84.9	94.24
RELLIS-3D	PSPNet [5]	69.21	80.99	8.89	53.7	60.7	94.67	61.36	86.01
	DeepLabv3+ [44]	65.76	79.84	19.72	47.52	64.88	95.92	62.27	85.84
	DANet [37]	72.93	<u>85.18</u>	13.10	<u>60.60</u>	70.53	<u>95.95</u>	66.38	<u>89.11</u>
	OCRNet [6]	<u>74.67</u>	83.04	27.76	60.44	62.35	92.58	66.81	86.95
	PSANet [41]	64.06	75.29	17.08	47.45	61.74	94.31	59.99	83.17
	BiseNetv2 [45]	65.56	73.24	39.35	48.17	<u>71.91</u>	93.78	65.33	83.03
	CGNet [46]	62.84	74.17	49.57	45.41	68.88	94.53	65.9	82.7
	FastSCNN [47]	67.06	77.6	56.49	49.76	70.31	94.43	<u>69.27</u>	84.51
	FastFCN [48]	70.51	79.15	49.72	51.37	63.9	94.82	68.24	84.1
	*SETR [15]	65.37	78.64	40.89	52.59	63.8	91.87	65.53	83.59
	*DPT [11]	5.42	76.65	47.13	54.87	62.74	85.5	55.38	81.61
	*Segformer [7]	60.28	79.78	53.35	53.78	70.15	94.37	68.62	85.37
	*Segmenter [8]	51.67	78.4	19.38	42.61	66.04	92.05	58.36	82.16
	*GA-Nav-r8	78.5	88.25	<u>37.28</u>	72.34	74.75	96.07	74.44	91.69
	*GA-Nav-r16	78.37	85.58	28.63	67.55	73.82	95.73	71.62	90.26
	*GA-Nav-r32	76.73	86.86	23.49	71.58	71.24	94.65	70.76	90.44

TABLE V: **Comparison with state-of-the-art methods on RUGD and REELLIS-3D:** We compare the performance of our method with transformer-based methods (with *) as well as other methods in terms of IoU and aAcc. GA-Nav improves the state-of-the-art mIoU by 2.25 – 44.94% on RUGD and 5.17 – 19.06% on REELLIS-3D. Further, we show that our grouping method can improve the accuracy of classes like smooth region, bumpy region, and forbidden regions by large margins, thus improving safe navigation. We bold the best number and underline one best result from our method and one from other methods. The suffix of our method shows different designs of our network. For example, “r8” means we reduce height and width of the spatial resolution by a factor of 8 at the spatial alignment step.



Fig. 6: **Navigation Results:** Robot trajectories when navigating with Ours (red), Segformer (orange), OCRNet (yellow), TERP (blue), and DWA (green). (a) Walkway; (b) Trail; (c) Dirt road; (d) Off-road. We observe that the trajectories generated by GA-Nav maximize the navigation along the smoothest surface of the available terrains while maintaining the highest success rate.

As part of our future work, we need to consider how to evaluate the accuracy of our method with more navigation metrics. We want to explore how this learning method can be applied and customized to other planning and navigation schemes and extend our approach to multiple sensor inputs for designing a more robust navigation scheme.

Acknowledgement. This research was supported by Army Cooperative Agreement No. W911NF2120076 and ARO grant W911NF2110026.

APPENDIX

VI. UNSTRUCTURED ENVIRONMENTS: LEVELS OF NAVIGABILITY

There are certain characteristics, *e.g.*, texture, color, temperature, etc., that highlight the differences between various terrains. For robot navigation, it is important to identify different terrains based on the navigability of that terrain, which is indicated by its texture [9], [50], [20]. Furthermore,

Metrics	Method	Walkway	Trail	Dirt road	Off-road
Success Rate (%) ↑	DWA [49]	67	62	64	46
	TERP [13]	79	73	79	62
	OCRNet [6]	75	69	70	62
	Segformer [7]	77	73	80	58
Traj. Roughness ↓	GA-Nav (ours)	86	77	79	71
	DWA [49]	0.186	0.254	0.318	0.952
	TERP [13]	0.115	0.172	0.272	0.843
	OCRNet [6]	0.109	0.483	0.234	1.182
	Segformer [7]	0.124	0.233	0.346	1.038
Traj. Selection (%) ↑	GA-Nav (ours)	0.098	0.164	0.211	0.725
	DWA [49]	05	48	23	14
	TERP [13]	03	94	96	32
	OCRNet [6]	51	58	17	08
False Positive of Forbidden Region (%) ↓	Segformer [7]	46	62	57	72
	GA-Nav (ours)	98	96	94	81
	OCRNet [6]	39.13	42.38	52.14	56.14
False Positive of Forbidden Region (%) ↓	Segformer [7]	30.84	35.64	46.57	51.26
	GA-Nav (ours)	0.88	1.23	3.17	7.85

TABLE VI: **Navigation Comparisons:** We compare the performance of our method with state-of-the-art algorithms on four different scenarios, each with at least ten trials. GA-Nav consistently outperforms other methods in terms of success rate, trajectory roughness, and false positive rate of forbidden regions even in challenging unstructured environments. We observe an average increase of 10% in terms of success rate, 43.5% in terms of the best navigable trajectory selection and an average decrease of 10.82% in trajectory roughness. Further, GA-Nav reduces the false positive rate of forbidden regions by 37.79%.

there may be terrains with similar textures and navigability that similarly affect navigation capabilities but may pose challenges to visual perception systems such as semantic segmentation (due to the long-tailed distribution problem). In such cases, it is advantageous to group terrains with similar navigability into a single category. Given a dataset with C different semantic classes, we regroup them into G classes using the following criteria.

Terrain navigability varies for different robot systems. For instance, large autonomous vehicles may be better able to navigate on dirt roads than smaller mobile robots that would find such dirt roads difficult to traverse. Therefore, our approach is generally designed to identify different groupings of terrain categories and is not restricted to a fixed set of groupings. In Table VII, we show an example of a grouping based on terrain texture and other characteristics. Below, we describe in detail the characteristics of different terrains based on their texture and navigability.

- **Concrete, Asphalt (Smooth):** Concrete and asphalt are smooth terrains commonly found in urban roads. These terrains correspond to the “flat” category in the grouping provided by the CityScapes [1] autonomous driving dataset. These terrains are navigable in outdoor envi-

TABLE VII: **Texture-based Terrain Classification:** We show an example of a grouping of terrain classes based on their texture and navigability. Our approach is general and designed for dynamic groupings in which different terrains may be re-classified into different hierarchy levels depending on the type of robot.

Hierarchy level	Classes
Navigable	Smooth
	Rough
	Bumpy
Forbidden	Water, Bushes, Tall Vegetation
Obstacles	Trees, Poles, Logs, etc.
Background	Void, Sky, Sign

ronments [9], and most mobile robots should be able to navigate in these terrains.

- **Gravel, Grass, Dirt, Sand (Rough):** These terrains have been termed rough on account of the increased friction encountered while traversing them [50], [51]. Most existing perception modules in off-road environments [20], [21] either do not consider these factors or may conservatively avoid such regions [9], [19] resulting in a sub-optimal solution, particularly in off-road navigation environments. To handle a large class of navigation methods, we want to be able to explicitly distinguish between smooth surfaces and rough surfaces. For example, in the presence of a smooth navigable region, a planning algorithm could prioritize it over a rough navigable region to reduce energy loss.
- **Small rocks, Rock-bed (Bumpy):** Autonomous vehicles or large robots may be able to navigate through rocks, while smaller robots with weaker off-road capabilities may face issues in such terrains [9], [20], [21], [52]. We provide a safe and flexible option where the planning scheme can be customized according to different scenes and different hardware characteristics of the robot. Specifically, this level of grouping can be ignored for off-road robots or vehicles by re-allocating these terrains to a different navigability group such as rough (for larger robots) or forbidden (for smaller robots). Our approach is general and handles dynamic groupings.
- **Water, Bushes, etc. (Forbidden Regions):** These are regions that the agent must avoid to prevent damage to the robot hardware.
- **Obstacles:** Detecting obstacles such as trees, poles, etc. is critical for safe navigation. There has been a lot of work [52], [21], [20] on obstacle and hazardous terrain detection.
- **Background:** We use this buffer group to include background and non-navigable classes, including void, sky, and sign, that are not commensurate with any of the earlier definitions.

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] H. Alhaija, S. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, “Augmented reality meets computer vision: Efficient data generation for urban driving scenes,” *International Journal of Computer Vision (IJCV)*, 2018.
- [3] R. R. Murphy, *Disaster robotics*. MIT press, 2014.
- [4] R. R. Shamshiri, C. Weltzien, I. A. Hameed, I. J. Yule, T. E Grift, S. K. Balasundram, L. Pitonakova, D. Ahmad, and G. Chowdhary, “Research and development in agricultural robotics: A perspective of digital farming,” 2018.
- [5] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.
- [6] Y. Yuan, X. Chen, and J. Wang, “Object-contextual representations for semantic segmentation,” in *16th European Conference Computer Vision (ECCV 2020)*, August 2020.
- [7] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, “Segformer: Simple and efficient design for semantic segmentation

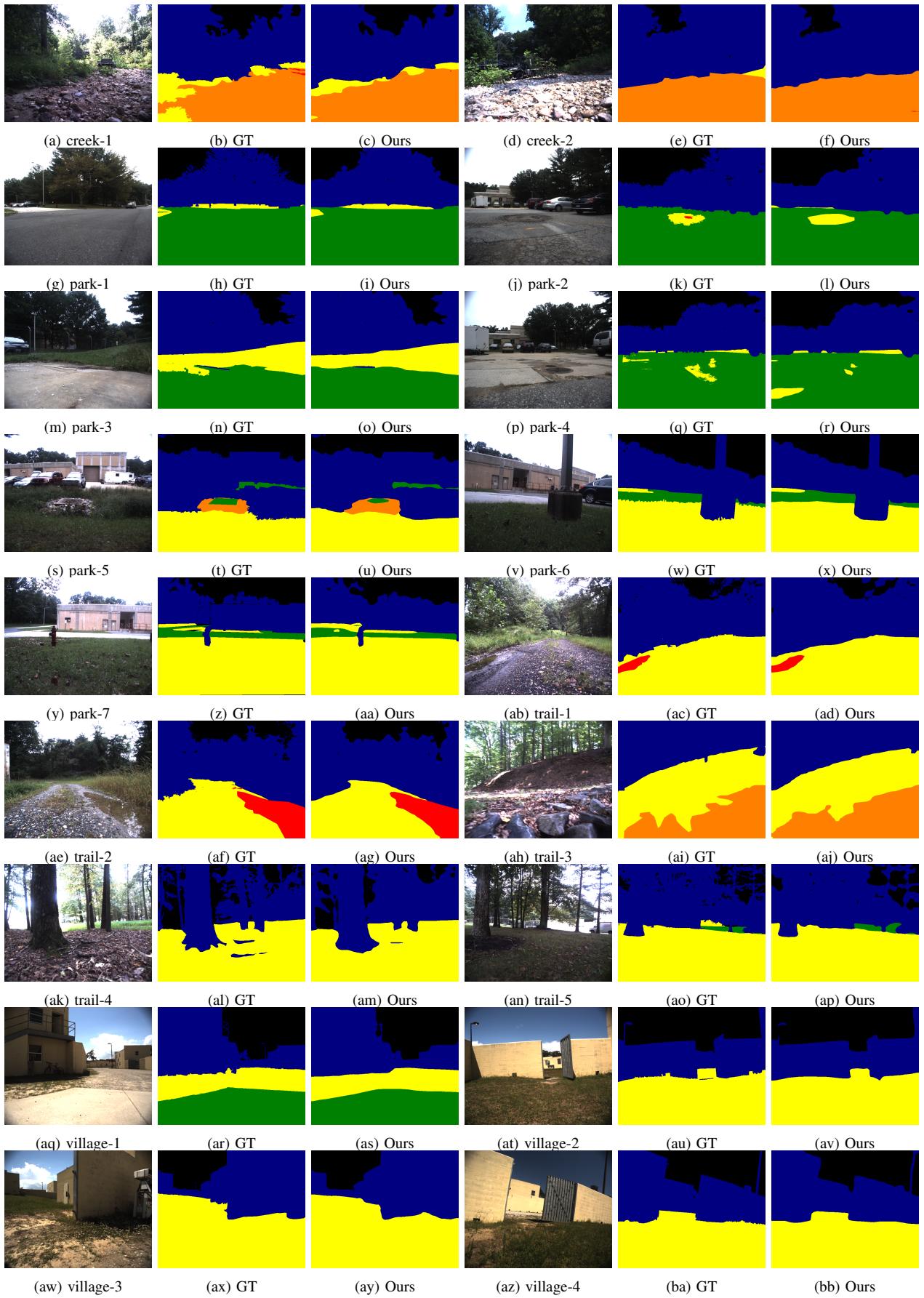


Fig. 7: More Qualitative Results on RUGD

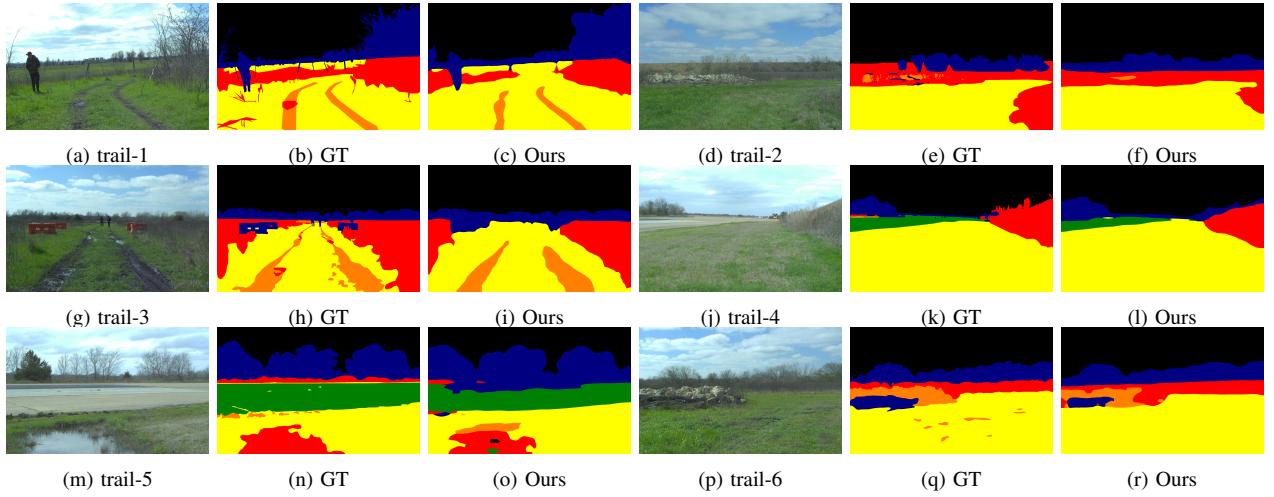


Fig. 8: More Qualitative Results on RELLIS-3D

- with transformers,” in *Advances in Neural Information Processing Systems*, 2021.
- [8] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, “Segmenter: Transformer for semantic segmentation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [9] S. Matsuzaki, K. Yamazaki, Y. Hara, and T. Tsubouchi, “Traversable region estimation for mobile robots in an outdoor image,” *J. Intell. Robotics Syst.*, vol. 92, no. 3-4, p. 453–463, Dec. 2018.
- [10] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, “Obstacle detection and terrain classification for autonomous off-road navigation,” *Autonomous Robots*, vol. 18, no. 1, pp. 81–102, jan 2005.
- [11] R. Ranftl, A. Bochkovskiy, and V. Koltun, “Vision transformers for dense prediction,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.
- [12] B. Cheng, A. Schwing, and A. Kirillov, “Per-pixel classification is not all you need for semantic segmentation,” in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.
- [13] K. Weerakoon, A. J. Sathyamoorthy, U. Patel, and D. Manocha, “Terp: Reliable planning in uneven outdoor environments using deep reinforcement learning,” in *2022 IEEE International Conference on Robotics and Automation (ICRA)*, 2022.
- [14] S. Hao, Y. Zhou, and Y. Guo, “A brief survey on semantic segmentation with deep learning,” *Neurocomputing*, 2020.
- [15] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr, and L. Zhang, “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 6881–6890.
- [16] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, “Large kernel matters—improve semantic segmentation by global convolutional network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4353–4361.
- [17] M. J. Procopio, J. Mulligan, and G. Grudic, “Learning terrain segmentation with classifier ensembles for autonomous robot navigation in unstructured environments,” *Journal of Field Robotics*, 2009.
- [18] G. Kahn, P. Abbeel, and S. Levine, “Badgr: An autonomous self-supervised learning-based navigation system,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1312–1319, 2021.
- [19] G. Pereira, L. Pimenta, L. Chaimowicz, A. Fonseca, D. Almeida, L. Corrêa, R. Mesquita, and M. Campos, “Robot navigation in multi-terrain outdoor environments,” vol. 28, 01 2006, pp. 331–342.
- [20] G. Reina, A. Millella, and R. Rouveure, “Traversability analysis for off-road vehicles using stereo and radar data,” in *2015 IEEE International Conference on Industrial Technology (ICIT)*, 2015, pp. 540–546.
- [21] J. Larson, M. Trivedi, and M. Bruch, “Off-road terrain traversability analysis and hazard avoidance for ugvs,” 2011.
- [22] G. Wilson, “Towards speed selection for high-speed operation of autonomous ground vehicles on rough off-road terrains,” 2018.
- [23] A. Vicente, Jindong Liu, and Guang-Zhong Yang, “Surface classification based on vibration on omni-wheel mobile base,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2015.
- [24] E. DuPont, C. Moore, E. Collins, and E. Coyle, “Frequency response method for terrain classification in autonomous ground vehicles,” *Autonomous Robots*, vol. 24, pp. 337–347, 05 2008.
- [25] G. Wilson, “Terrain roughness identification for high-speed ugvs,” *Journal of Automation and Control Research*, 10 2014.
- [26] S. Matsuzaki, K. Yamazaki, Y. Hara, and T. Tsubouchi, “Traversable region estimation for mobile robots in an outdoor image,” *J. Intell. Robotic Syst.*, vol. 92, no. 3-4, pp. 453–463, 2018.
- [27] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *Int. J. Comput. Vision*, vol. 88, no. 2, p. 303–338, June 2010.
- [28] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, 2014.
- [29] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [30] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, “A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2019.
- [31] P. Jiang, P. Osteen, M. Wigness, and S. Saripalli, “Rellis-3d dataset: Data, benchmarks and analysis,” *arXiv preprint:2011.12954*, 2020.
- [32] H. Pham, Z. Dai, Q. Xie, M.-T. Luong, and Q. V. Le, “Meta pseudo labels,” 2021.
- [33] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, “Bottleneck transformers for visual recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 16 519–16 529.
- [34] T. Guan, J. Wang, S. Lan, R. Chandra, Z. Wu, L. Davis, and D. Manocha, “M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2022, pp. 772–782.
- [35] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” in *International Conference on Learning Representations*, 2021.
- [36] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.
- [37] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, “Dual attention network for scene segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [38] A. Prakash, K. Chitta, and A. Geiger, “Multi-modal fusion transformer for end-to-end autonomous driving,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

- [39] “Temporal fusion transformers for interpretable multi-horizon time series forecasting,” *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021.
- [40] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, “Deeply-Supervised Nets,” in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, 2015.
- [41] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia, “Psanet: Point-wise spatial attention network for scene parsing,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- [42] E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional networks for semantic segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.
- [43] M. Contributors, “Mmsegmentation, an open source semantic segmentation toolbox,” 2020.
- [44] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.
- [45] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, “Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation,” *International Journal of Computer Vision*, 11 2021.
- [46] T. Wu, S. Tang, R. Zhang, and Y. Zhang, “Cgnet: A light-weight context guided network for semantic segmentation,” *IEEE Transactions on Image Processing*, vol. 30, pp. 1169–1179, 2021.
- [47] R. P. K. Poudel, S. Liwicki, and R. Cipolla, “Fast-scnn: Fast semantic segmentation network,” in *BMVC*, 2019.
- [48] H. Wu, J. Zhang, K. Huang, K. Liang, and Y. Yizhou, “Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation,” 2019.
- [49] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robotics Autom. Mag.*, 1997.
- [50] K. Zhao, M. Dong, and L. Gu, “A new terrain classification framework using proprioceptive sensors for mobile robots,” *Mathematical Problems in Engineering*, vol. 2017, pp. 1–14, 09 2017.
- [51] C. Bai, J. Guo, G. Linli, and J. Song, “Deep multi-layer perception based terrain classification for planetary exploration rovers,” *Sensors*, vol. 19, p. 18, 07 2019.
- [52] R. A. Hewitt, A. Ellery, and A. de Ruiter, “Training a terrain traversability classifier for a planetary rover through simulation,” *International Journal of Advanced Robotic Systems*, vol. 14, no. 5, p. 1729881417735401, 2017.