VINet: Visual and Inertial-based Terrain Classification and Adaptive Navigation over Unknown Terrain

Tianrui Guan¹∗, Ruitao Song²†, Zhixian Ye², and Liangjun Zhang²

Abstract—We present a visual and inertial-based terrain classification network (VINet) for robotic navigation over different traversable surfaces. We use a novel navigation-based labeling scheme for terrain classification and generalization on unknown surfaces. Our proposed perception method and adaptive scheduling control framework can make predictions according to terrain navigation properties and lead to better performance on both terrain classification and navigation control on known and unknown surfaces. Our VINet can achieve 98.37% in terms of accuracy under supervised setting on known terrains and improve the accuracy by 8.51% on unknown terrains compared to previous methods. We deploy VINet on a mobile tracked robot for trajectory following and navigation on different terrains, and we demonstrate an improvement of 10.3% compared to a baseline controller in terms of RMSE.

I. INTRODUCTION

The problems of perception and navigation have been extensively explored individually in the past. The perception tasks, including detection [1], segmentation [2], [3], classifications [4], [5], etc., can obtain very good overall accuracy running on challenging offline data sets [6], [7], which are clean and well-annotated. For mobile robot navigation in the outdoor environment, efficient perception methods such as terrain traversability analysis [8] have been proposed. Meanwhile, many control policies have been developed for mobile wheel or tracked robots to navigate on different terrains [9], [10].

Perception and navigation problems, however, are rarely considered as a whole. Many sophisticated perception models confine the task and problem settings for standard benchmarking and fair comparisons. To truly benefit some downstream tasks like navigation and control, we need to reconsider the definition and context of the perception algorithms that are the same as those in standard benchmarks. For example, categories defined in a traditional classification or semantic segmentation task might not be used directly in navigation settings, since we need a mapping between the original categories and traversability of the terrain. There are some works that focus on robotic perception for terrain understanding [11], [12] and traversability analysis [13], [8], but most of those definitions are hand-crafted and heuristic-based. Instead, metrics based on navigation could provide insight on how to define such mapping or labeling scheme.

Perception for outdoor navigation is challenging, especially in an environment that is unknown and constantly changing with uncertainty. The perception model needs to generalize to unseen terrain well since it’s not possible to collect information and label all surfaces. In the context of navigation, unlike a traditional perception task, the perception algorithms should be trained in a way that maximizes the navigation performance and evaluated according to some navigation-related metrics, which is rarely explored in the past. The perception needs to be coupled with navigation and output the most useful result that can lead to stable and safe decision for the navigation and control modules.

In this paper, we propose VINet, a novel terrain classification network that combines both visual and inertial sensing through camera and Inertial Measurement Unit (IMU). Our method can work well on known terrains as well as completely unseen surfaces. There are two streams in the network: 1) an image stream that provides robust predictions on the training set and seen surfaces, and 2) an IMU temporal stream that can extract features of unseen terrains through inertial sensing and generalize to unknown classes. As shown in Figure 1, we used a navigation-based perception metric to measure the performance of our network for unseen terrains, which maximizes the performance of navigation. In addition, we propose a novel adaptive scheduling control framework and show that this pipeline with the proposed VINet leads to better navigation performance on both known and unknown terrain.

Fig. 1: Semantic labels v.s. Navigation-based labels: The traditional classification (left) has semantic-based labels, which are ambiguous but not generalizable to unknown terrain because the prediction would be meaningless. Our proposed navigation-based classification (right) recasts the terrain labels to known classes based on navigation properties distinguished by different inertial feedback as part of the training data to train our proposed VINet, which can outperform other existing methods on both semantic-based and navigation-based labeling.

Authorized licensed use limited to: IEEE Xplore. Downloaded on July 15, 2023 at 20:08:52 UTC from IEEE Xplore. Restrictions apply.
The key contributions of our work include:
1) We propose a novel terrain classification network which fuses visual and inertial features of the terrain. Our method can achieve 98.37% in terms of accuracy under the supervised setting and lead to an increase of 8.51% under the unsupervised setting compared to previous methods.
2) We propose a novel fusion scheme, which closes the latent embeddings between image and inertial data, and leverages those two modalities channel-wise. We also introduce an IMU denoising strategy to process noisy IMU data and further boost the classification performance.
3) We deploy VInet on a tracked robot and propose a novel adaptive scheduling control framework enabled by VInet with navigation-based label generalization. Our navigation performance outperforms a baseline single terrain controller by 10.3% in terms of RMSE.

II. RELATED WORK

A. Terrain Classification

In [14], Xue et al. proposed an orderless-texture-based and spatial information-targeted neural network. [15] used both 3D LiDAR and camera to classify terrain by computing a 3D-scan feature that overcomes the issue in different lighting conditions. Similarly, Nguyen [16] introduced a structure-based method as well as estimating the surface smoothness to classify different roughness surface objects. [17] proposed an angular imaging method to distinguish different material types.

In addition, some methods try to solve the classification problem with inertial sensing. The earlier method like [18] utilizes a dynamic model to estimate the terrain profile and further label the terrain based on the spatial frequency part of the profile predicted. Likewise, [19] adopts the same frequency analysis to classify the terrain by filtering out the terrain impulses from the identified terrain profile. [20] tries to reveal the terrain types using the inertial data collected from the human walk. Recent work [21] proposes several deep learning methods for classifying the time-series IMU data on top of the engineered features.

B. Navigation on different Terrain

Indoor navigation has been well developed in the past decade [22], [23], [24], but it still remains a challenge in the outdoor environment [25], [26], [27], especially on different terrains [28], [29]. Kumar et al. [28] investigate adaptive control strategies to enable a quadruped robot to walk on different material terrains, such as sand, mud, grass, and dirt. The wheeled robot or tracked robot [30] is another type of platform that is used in terrain navigation. [31], [32] and [33] cover the topics of unstructured environment modeling and controller design from the angle of the tracked robot. As mentioned in [28], the adaptive control policy enables the legged robot to quickly recover stability from falling, which poses a key factor towards the success of navigating in dynamics-changing environments of different terrain. Such related learning-based adaptive controller methods have been discussed in [34], [35], [36].

III. METHOD

In this section, we present VInet, a novel terrain classification method for adaptive navigation and control. Our method combines visual and inertial information to make a classification prediction on different types of terrains and can classify known and unknown terrains categorized by navigation metrics. We later show that our network can directly benefit decision-making for control and navigation, even on terrains that the network has never seen before.

The rest of the sections are structured as follows: We begin by defining our problem settings and each component of our network, including backbones, IMU denoising, classification head, and loss functions. We move on to our implementation of adaptive scheduling control and the entire framework for navigation. Finally, we give some insights into the designs, including the navigation-based labeling and using IMU data as input for the classification task.

A. Problem Definition and Terminology

The inputs of the network consist of an image patch $I_{patch} \in \mathbb{R}^{3 \times l \times l}$ taken along the intended trajectory of the mobile robot from the RGB camera, a sequence of 6 dimensional IMU data $I_{imu}(t_s) = [a_x, a_y, a_z, w_x, w_y, w_z](t_s)$ $\in \mathbb{R}^6$, $t_s = \{t_0, ..., t_0 + D\}$ with three linear accelerations and three angular velocities for each axis, and a pulse-width modulation (PWM) signal $S(t_s) \in [0, 1]^m$, where $m$ is the number of motors relating to navigation and each dimension represents the command sent to a specific motor. The task of our method is to utilize visual and inertial perception sensors on a mobile robot for terrain classification. Our model will output a terrain category $t \in T = \{t| t = 0, ..., n - 1\}$, where $n$ is the number of terrain types. In our setting, we only focus on surfaces that are navigable with different textures and physical properties and try to reduce the trajectory-following errors during navigation. Avoiding non-traversable surfaces like water or large rocks is not the point of the paper.

let $T$ be the set of known terrains, and the set of unknown terrains $T_u$ be the complement of $T$. Given several pre-defined trajectories on surface $i$, we define $MSE_i(p)$ as the mean squared error of the controller $p$ trying to follow those trajectories on surface $i$. From the set of known surfaces $T$, we can obtain a set of trained controller $P = \{p_i|i = 0, ..., k - 1\}$, and we have

$$\arg\min_x MSE_i(p_x) = i,$$

where $p_i$ is the controller pre-trained on the known surface $i \in T$. The ground truth category $c_i$ of surface $i$ is defined as follows:

$$c_i = \begin{cases} i & \text{if } i \in T \\ \arg\min_x MSE_i(p_x) & \text{if } i \in T_u \end{cases}$$

In our setting, known terrains are surfaces on which our model has training data, so the perception model and
nonlinear model predictive controller (NMPC) can be trained or fine-tuned on those surfaces; unknown terrains are ones that our model has never encountered during training. The ground truth categories of the terrain are defined by the characteristics of the surfaces during navigation, or specifically, by the type of controllers that has the best performance on such terrain. For known terrain, we collected image, IMU, state of the tracked robot, planned trajectories, and actual trajectories to train the perception model and controller; for unknown terrain, we collected perception data as well as the performance of each controller pre-trained on the known surfaces according to definition 1, for the purpose of evaluation.

**B. Image and IMU Backbone**

The proposed VINet consists of one image stream and one IMU stream, as shown in Figure 2. For the image stream, we adopted EfficientNet-B0 [37] as our image feature extractor. Given an image patch \( I_{\text{patch}} \), the image branch takes a series of convolution operations with kernel size of 3 and 5, and eventually down-sampled to 1280 channels with spatial resolution of \( 7 \times 7 \). After flattening and two linear layers with batch normalization and drop-out, we can obtain an intermediate feature \( F_{\text{img}} \in \mathbb{R}^d \), where \( d \) is the shared dimension between \( F_{\text{img}} \) and IMU feature \( F_{\text{imu}} \).

Considering the effect of actions on the IMU pattern, our IMU stream takes an IMU sequence \( I_{\text{imu}} \) and the corresponding PWM signal \( S \) for some time duration \( D \). Given the raw input sequence \( H = [I_{\text{imu}}; S] \in \mathbb{R}^{D \times (n+m)} \), the output feature \( F_{\text{atten}} \) is computed as:

\[
F_{\text{atten}} = \text{softmax}(k(H)^T \cdot q(H))^T \cdot v(H),
\]

where \( k(H), q(H), v(H) \) represent the key, query, and value feature, respectively, as in the self-attention [38] literature. Before the features are passed into the classification head, we process the feature \( F_{\text{atten}} \) with the IMU denoising module.

**C. IMU Denoising**

Due to the noisy nature of IMU data, we introduce an attention-based denoising method to sample useful features in the time dimension. The attention map \( A_{\text{map}} \in \mathbb{R}^{D \times D} \) is defined as \( \text{softmax}(k(H)^T \cdot q(H)) \), and we can calculate the denoised IMU feature \( F_{\text{d}} \) as:

\[
F_{\text{d}} = F_{\text{atten}}[\text{argsort}(-A_{\text{map}}[i][i])|0 : f_s \times D|],
\]

where \( f_s \) is the sampling rate, and \( \text{argsort} \) sort the value in increasing order and returns the corresponding indices. This step removes the features at some specific timestamp with a low attention value for denoising purposes. After passing through a few linear layers with batch normalization and flattening, we can obtain an IMU feature \( F_{\text{imu}} \in \mathbb{R}^d \).

**D. Classification Head in Shared Embedding Space**

After going through the image and IMU stream, we can obtain \( F_{\text{img}} \) and \( F_{\text{imu}} \) with dimension \( d \). We used a learnable weight for each channel of the features to fuse \( F_{\text{img}} \) and \( F_{\text{imu}} \). After several fully connected layers and activation, the network will output the vector with a classification score.

In the classification head, we want to train the network to learn a shared embedding space for \( F_{\text{img}} \) and \( F_{\text{imu}} \), which is achieved by adding a similarity constraint in the network during training. The purpose of creating a shared embedding space is to alleviate the effect of one feature taking over the other. In addition, we want to add some robustness to the network: if the data from one branch is corrupted and produced a terrible feature, it would not significantly affect and corrupt the classification head because the other branch could have some effect of averaging the corrupted features back to the shared embedding space.

**E. Loss Function**

**Mean-Squared-Error Loss**: To create a shared embedding space for \( F_{\text{img}} \) and \( F_{\text{imu}} \), we penalize the network for ex-
tracting significantly different feature vectors from different modalities of the same data point. The MSE loss is defined as follows:

$$\mathcal{L}_{\text{MSE}} = \sum (F_{\text{img}} - F_{\text{imu}})^2,$$  \hspace{1cm} \text{(4)}

**Cross-Entropy Loss:** This is a standard loss for the classification task, defined as follows:

$$\mathcal{L}_{\text{CE}} = - \sum_{t \in T} c_t \log (P_t),$$ \hspace{1cm} \text{(5)}

where $T$ denotes the set of known terrain, $P_t$ denotes the output probability map corresponding to terrain $t$, and $c_t$ corresponds to the ground-truth label. We use this loss on each branch of the network as well as the final classification head.

**F. Adaptive Scheduling Control Framework**

In this subsection, we give more details of our controller and present our adaptive scheduling control framework with VINet on known and unknown terrain.

**Nonlinear Model Predictive Controller:** The overall goal of the controller is to closely follow the planned trajectory, which is significantly affected by the terrain type. For known terrains, we collected some navigation data and use those to fit a nonlinear model predictive controller (NMPC) and a Gaussian Process (GP) model to learn the interaction between the tracks and the terrain. The NMPC trajectory tracking control problem at time instance $k$ can be formulated as follows:

$$\min_u J(x(k), U) = \sum_{i=0}^{N-1} (||Q x(i|k)||_2 + ||R u(i|k)||_2 + ||R_d \Delta u(i|k)||_2)$$

s.t. $x(i+1|k) = x(i|k) + f(x(i|k), u(i|k)) T_s$, $x(0|0) = x_0$, $x(i|k) \in \mathbf{X}$,

$u(0|0) = u_0$, $u(i|k) \in \mathbf{U}$,

$U = \text{col} \{u(0|k), \ldots, u(N-1|k)\}$, \hspace{1cm} \text{(6)}

where $N$ is the prediction horizon. $x(i|k)$ and $u(i|k)$ denote the predicted values of the model state and input, respectively, at time $k + i$ based on the information that is available at time $k$. $\Delta u(i|k) = u(i|k) - u(i-1|k)$ is included in the cost function to smooth the control output. The state vector is $[x_c, y_c, \theta]^T$, where $x_c$ and $y_c$ are the 2D Cartesian coordinates of the robot, and $\theta$ is the heading. The control output vector $u = [v, \omega]^T$ is composed of the linear and angular speed commands of the robot. The nonlinear prediction model $f(x, u)$ can be described by the following equations:

$$\begin{cases}
\dot{x} = v \cdot \cos (\theta) \\
\dot{y} = v \cdot \sin (\theta) \\
\dot{\theta} = \omega
\end{cases}$$ \hspace{1cm} \text{(7)}

The output of the NMPC, $u$, is then mapped to the PWM signals $S$ controlling the two motors driving each track of the robot. The terrain track interaction, caused by terrain resistance and deformation, is captured by the GP model: $S = gp(u)$. The GP models provide the required $S$ for each terrain, so that the actual linear and angular speeds match with the desired $u$ calculated by the NMPC. Since $S$ is two-dimensional, corresponding to the left and right track motors, two one-dimensional GP models are fitted for each known terrain. The Radial Basis Function (RBF) kernel is used in the GP models:

$$\kappa(u, u') = \sigma_f^2 \exp \left( -\frac{1}{2} (u-u')^T L^{-2} (u-u') \right) + \sigma_n^2$$ \hspace{1cm} \text{(8)}

where $L$ is the diagonal length scale matrix and $\sigma_f, \sigma_n$ represent the data and prior noise variance, and $u_i, u_j$ represent data features.

**Adaptive Scheduling Control:** As shown in Figure 3, we deploy VINet on a tracked robot, and our model can use visual and inertial feedback during navigation to predict a terrain class, which corresponds to the pre-trained controller and would have the best performance on this terrain if the prediction is correct. Once the robot receives the prediction, it will adjust to the suitable controller on this terrain.

For unknown terrain, our classifier would still choose one of the categories in the original label set since our classifier is only trained on known surfaces. Because of the introduction of IMU as inertial feedback and the terrain label assignment measured by the navigation feedback from the controller, VINet can make a prediction based more on generalized navigation features instead of purely semantic labels, leading to a better choice of the controller on an unknown surface. For evaluation of perception, we find a few unknown terrains and test our pre-trained controller to assign those terrains to one of our pre-trained controllers, as defined in Equation 1.

**G. Motivation for Using IMU data and Navigation-based Labeling Scheme**

While the image-based classifier can achieve a good result on standardized classification benchmarks, it does not directly translate to good performance in navigation. First, the categories in standard data sets are usually semantic-based; on the other hand, it’s less intuitive to give a fair traversability or material label that is useful for navigation, and most existing practices are usually heuristic-based [12], [13] for benchmarking purposes.
Second, a purely vision-based method could easily overfit the known data and makes it harder to generalize to more unknown terrains. For example, grass and artificial grass might be visually similar, but they might have different hardness and slipperiness. Because of such difference, the controller fitted with data navigating over grass might lead to worse navigation on the artificial grass because the classifier could not distinguish their navigation features by only relying on the visual difference.

To solve those issues, we eliminate the heuristics in the labeling by using some pre-trained controllers as labels for the terrain, on which the controller has the best performance. The labeling becomes navigation-based instead of semantic-based: once our model is trained according to this labeling scheme and the model can predict correctly, the output would choose the best controller to use while navigating on the corresponding surface, which could directly benefit the navigation. Through navigation-based labeling, the unknown terrain will be predicted as one of the known terrains and the interpretation is that those two types of terrains could have similar navigation features due to some visual textures. The output could still be random and meaningless sometimes unless we have some feedback during the navigation. To improve the Interpretability and generalizability of the output, we add an IMU sequence as input during navigation, in addition to the visual input.

By introducing the IMU, we want to improve the model’s ability to generalize on unseen terrains. The model can build connections between the extracted features from inertial sensing of IMU and the navigation features of the terrain, so the network can have less bias due to over-fitting on the images. Because all of the terrain labels are based on navigation data from known terrains, the prediction output is optimized with respect to the IMU and Image input. In Section IV, we validate our statement by showing comparisons with and without IMU as input to the network.

IV. IMPLEMENTATION DETAILS AND ANALYSIS

A. Data Collection and Controller Pre-training

We collected 4923 images of size $1280 \times 720$ among 11 different terrain types and the corresponding inertial sequences at 30 Hz. The 11 terrain types include black carpet, green carpet, smooth tiles, concrete, grass, soil, mulch, wood rocks, rocky surface, asphalt, and small rocks.

Gaussian Process models used in the tracking controller are fitted by the collected data on each known terrain, including the PWM signal $S$ and the corresponding robot’s linear and angular speeds $u = [v, \omega]^T$ calculated based on the robot localization output. For each terrain, about 1500 data pairs $\{S, u\}$ were collected and used to train the GP model. Then, the NMPC controller is tuned manually on each terrain type to further optimize the trajectory tracking performance on each known type. For the purpose of evaluation, we find some other surfaces as unknown terrains and evaluate those pre-trained controllers on such terrains to get the best controller as its navigation-based label. Due to limited space,

Second, a purely vision-based method could easily overfit the known data and makes it harder to generalize to more unknown terrains. For example, grass and artificial grass might be visually similar, but they might have different hardness and slipperiness. Because of such difference, the controller fitted with data navigating over grass might lead to worse navigation on the artificial grass because the classifier could not distinguish their navigation features by only relying on the visual difference.

To solve those issues, we eliminate the heuristics in the labeling by using some pre-trained controllers as labels for the terrain, on which the controller has the best performance. The labeling becomes navigation-based instead of semantic-based: once our model is trained according to this labeling scheme and the model can predict correctly, the output would choose the best controller to use while navigating on the corresponding surface, which could directly benefit the navigation. Through navigation-based labeling, the unknown terrain will be predicted as one of the known terrains and the interpretation is that those two types of terrains could have similar navigation features due to some visual textures. The output could still be random and meaningless sometimes unless we have some feedback during the navigation. To improve the Interpretability and generalizability of the output, we add an IMU sequence as input during navigation, in addition to the visual input.

By introducing the IMU, we want to improve the model’s ability to generalize on unseen terrains. The model can build connections between the extracted features from inertial sensing of IMU and the navigation features of the terrain, so the network can have less bias due to over-fitting on the images. Because all of the terrain labels are based on navigation data from known terrains, the prediction output is optimized with respect to the IMU and Image input. In Section IV, we validate our statement by showing comparisons with and without IMU as input to the network.

IV. IMPLEMENTATION DETAILS AND ANALYSIS

A. Data Collection and Controller Pre-training

We collected 4923 images of size $1280 \times 720$ among 11 different terrain types and the corresponding inertial sequences at 30 Hz. The 11 terrain types include black carpet, green carpet, smooth tiles, concrete, grass, soil, mulch, wood rocks, rocky surface, asphalt, and small rocks.

Gaussian Process models used in the tracking controller are fitted by the collected data on each known terrain, including the PWM signal $S$ and the corresponding robot’s linear and angular speeds $u = [v, \omega]^T$ calculated based on the robot localization output. For each terrain, about 1500 data pairs $\{S, u\}$ were collected and used to train the GP model. Then, the NMPC controller is tuned manually on each terrain type to further optimize the trajectory tracking performance on each known type. For the purpose of evaluation, we find some other surfaces as unknown terrains and evaluate those pre-trained controllers on such terrains to get the best controller as its navigation-based label. Due to limited space,

Second, a purely vision-based method could easily overfit the known data and makes it harder to generalize to more unknown terrains. For example, grass and artificial grass might be visually similar, but they might have different hardness and slipperiness. Because of such difference, the controller fitted with data navigating over grass might lead to worse navigation on the artificial grass because the classifier could not distinguish their navigation features by only relying on the visual difference.

To solve those issues, we eliminate the heuristics in the labeling by using some pre-trained controllers as labels for the terrain, on which the controller has the best performance. The labeling becomes navigation-based instead of semantic-based: once our model is trained according to this labeling scheme and the model can predict correctly, the output would choose the best controller to use while navigating on the corresponding surface, which could directly benefit the navigation. Through navigation-based labeling, the unknown terrain will be predicted as one of the known terrains and the interpretation is that those two types of terrains could have similar navigation features due to some visual textures. The output could still be random and meaningless sometimes unless we have some feedback during the navigation. To improve the Interpretability and generalizability of the output, we add an IMU sequence as input during navigation, in addition to the visual input.

By introducing the IMU, we want to improve the model’s ability to generalize on unseen terrains. The model can build connections between the extracted features from inertial sensing of IMU and the navigation features of the terrain, so the network can have less bias due to over-fitting on the images. Because all of the terrain labels are based on navigation data from known terrains, the prediction output is optimized with respect to the IMU and Image input. In Section IV, we validate our statement by showing comparisons with and without IMU as input to the network.

IV. IMPLEMENTATION DETAILS AND ANALYSIS

A. Data Collection and Controller Pre-training

We collected 4923 images of size $1280 \times 720$ among 11 different terrain types and the corresponding inertial sequences at 30 Hz. The 11 terrain types include black carpet, green carpet, smooth tiles, concrete, grass, soil, mulch, wood rocks, rocky surface, asphalt, and small rocks.

Gaussian Process models used in the tracking controller are fitted by the collected data on each known terrain, including the PWM signal $S$ and the corresponding robot’s linear and angular speeds $u = [v, \omega]^T$ calculated based on the robot localization output. For each terrain, about 1500 data pairs $\{S, u\}$ were collected and used to train the GP model. Then, the NMPC controller is tuned manually on each terrain type to further optimize the trajectory tracking performance on each known type. For the purpose of evaluation, we find some other surfaces as unknown terrains and evaluate those pre-trained controllers on such terrains to get the best controller as its navigation-based label. Due to limited space,
TABLE II: Model comparisons under 11 known terrains setting:
All models are trained and tested on 11 terrains. We only use 6-axis of the IMU when IMU dimension is 6.

| Modality         | IMG Backbone | Pre-training | Shared Dims | Accuracy ↑ |
|------------------|--------------|--------------|-------------|------------|
| SVM-based [40]   | -            | -            | -           | 0.5943     |
| IMU Branch       | -            | -            | 512         | 0.7109     |
| EfficientNet-b0  | ✓            | ✓            | 512         | 0.5811     |
| EfficientNet-b0  | ✓            | ✓            | 256         | 0.6143     |
| EfficientNet-b0  | ✓            | ✓            | 512         | 0.9026     |
| EfficientNet-b0  | ✓            | ✓            | 256         | 0.9079     |

TABLE III: Ablation study on 5 known terrains setting: The inter. CE loss is the cross-entropy loss used on the intermediate output of each branch. For shared embedding, we remove the MSE loss in the network for ablation. For ablation of IMU denosing and learned weighting, we directly use the original attention module and concatenation of the features from image and IMU.

| Inter. CE Loss | Shared Embedding | IMU Denoising | Learned Weighting | Shared Dims | Accuracy ↑ |
|---------------|------------------|---------------|-------------------|-------------|------------|
| ✓             | ✓                | ✓             | ✓                 | 512         | 0.6803     |
| ✓             | ✓                | ✓             | ✓                 | 256         | 0.6711     |
| ✓             | ✓                | ✓             | ✓                 | 512         | 0.6578     |
| ✓             | ✓                | ✓             | ✓                 | 256         | 0.6728     |
| ✓             | ✓                | ✓             | ✓                 | 512         | 0.6515     |
| ✓             | ✓                | ✓             | ✓                 | 256         | 0.6561     |
| ✓             | ✓                | ✓             | ✓                 | 512         | 0.6619     |
| ✓             | ✓                | ✓             | ✓                 | 256         | 0.6912     |
| ✓             | ✓                | ✓             | ✓                 | 512         | 0.6786     |
| ✓             | ✓                | ✓             | ✓                 | 256         | 0.7648     |

TABLE IV: Tracking Error Comparison

| Lateral RMSE (m) | Adaptive Carpet baseline | Mulch baseline |
|------------------|--------------------------|----------------|
| 0.026            | 0.050                    | 0.029          |

Fig. 4: Trajectory tracking control result of the VINet-based adaptive controller: Note that our classifier will classify artificial grass as carpet instead of grass, which is based on navigation properties and inertial sensing, instead of purely visual features. We tested the grass controller on artificial grass and could not record a meaningful trajectory because the tracking error is too large to reach the goal location. We see chatting classifications during transition between surfaces due to the lag between visual and IMU input.

V. CONCLUSIONS, LIMITATIONS, AND FUTURE WORKS

In this paper, we present a novel terrain classification method VINet for terrain classification and adaptive navigation control. We propose a novel IMU denoising module and a shared latent embedding space for image and IMU sequence. Our proposed navigation-based labeling and generalization can enable safe and stable navigation on unknown terrain, which is more intuitive and beneficial for navigation tasks compared to traditional semantic labels.

There are some limitations. For example, we did not extensively test our adaptive framework on more unknown terrains in a single run, since it’s difficult to find multiple terrains on a single site. In the future, we want to fully exploit the controller design to better utilize the navigation-based classification scheme, and also combine it with other functionalities like traversability analysis and obstacle avoidance.
