

Appendix for Velociraptor: Leveraging Visual Foundation Models for Label-Free, Risk-Aware Off-Road Navigation

Anonymous Author(s)

Affiliation

Address

email

1 A Implementation Details

2 A.1 Platform

3 We test our method using a full-scale Yamaha Viking VI modified for autonomous driving by Mai
4 et al. [1] and Sivaprakasam et al. [2]. This platform contains a front-facing Multisense stereo
5 camera and Velodyne lidar, and is capable of autonomous speeds of up to roughly $10m/s$. In order
6 to register scans and maps together, we use Super Odometry [3]. Our local maps are centered on the
7 robot in the odometry frame and represent an area of $120m \times 120m$ (60m forward) at $0.5m$ per cell.

8 A.2 Geometric Feature Extraction

9 A list of geometric features extracted is presented in Table 1. Our terrain estimation algorithm
10 works by computing the minimum elevation for each cell containing points. This elevation is then
11 interpolated and smoothed using a Markov Random Field.

12 A.3 Visual Feature Extraction

13 For visual feature extraction, we leverage several state-of-the-art neural network backbones. For se-
14 mantics, we leverage the open-source implementation of GA-Nav [4], a state-of-the-art FPV model
15 for off-road semantic segmentation. Images for GA-Nav are resized to $[688 \times 550]$. For Dinov2 [5],
16 we leverage values (as opposed to query/key) from the tenth layer of the ViT-b backbone. This was
17 chosen via qualitative analysis of the features and the guidance from AnyLoc [6], as well as run-time
18 constraints. For Segment Anything, we use the image features provided by the open-source API [7].
19 For both VFM-based methods, images are resized to $[686 \times 364]$. Similar to prior work Emernerf
20 [8], we perform a Principal Component Analysis (PCA) on the visual feature embeddings in FPV-
21 space for which we had lidar returns. The number of features was reduced to sixteen to fit memory
22 constraints for on-board local mapping, and roughly match the number of semantic channels from
23 GA-Nav. Image features to be included in the PCA analysis are randomly selected from pixels in
24 the train set which have a corresponding LiDAR return. From this collection of features, we can
25 then perform a PCA decomposition (Equation 1). At test time, the PCA reduction can be computed
26 efficiently per-pixel via Equation 2.

27 Instantaneous visual maps \mathbf{M}_t are computed by first projecting the point cloud into the image frame
28 using camera intrinsics K and extrinsics $[R|t]$ (Equation 3). All visual features with a corresponding
29 point are then projected into BEV with the spatial locations of their corresponding points, given local
30 map parameters. Cells with multiple points have their embeddings averaged (Equation 4). Maps are
31 aggregated over time by first transforming the previous map $\mathbf{M}_{1:t-1}$ to the current pose x_t and then
32 combining with \mathbf{M}_t using an exponential moving average (Equation 5) to generate an aggregated
33 visual map $\mathbf{M}_{1:t}$ centered at x_t .

$$\mu_{PCA} = \frac{1}{n} \sum_i [f_i], \quad U, S, V^T = SVD(F - \mu_{PCA}) \quad (1)$$

$$f_{PCA} = (f - \mu_{PCA})V^T \quad (2)$$

$$[u, v, w]^T = \mathbf{K}[\mathbf{R}|\mathbf{t}][x, y, z, 1]^T, \quad u = \frac{u}{w}, \quad v = \frac{v}{w} \quad (3)$$

$$\mathbf{P}^{i,j} = \{p \in \mathbf{P} | p_x \in [ri, r(i+1)], p_y \in [rj, r(j+1)]\}, \quad \mathbf{M}^{(i,j)} = \frac{1}{|\mathbf{P}^{i,j}|} \sum_{p \in \mathbf{P}^{i,j}} p_f \quad (4)$$

$$M_t^{i,j} = \begin{cases} M_{t-1}^{i,j} & \#M_t^{i,j} \\ M_t^{i,j} & \#M_{t-1}^{i,j} \\ \alpha M_t^{i,j} + (1 - \alpha)M_t^{i,j} & \text{otherwise} \end{cases} \quad (5)$$

Table 1: Qualitative Description of the Geometric Terrain Features

| Feature | Description |
|---------------|---|
| terrain | The output of our terrain estimation algorithm. |
| terrain slope | The magnitude of the slope of the terrain estimate |
| diff | The difference between the maximum elevation of a cell and its terrain estimate |
| SVD_{1-3} | The principal components of the points in each grid cell |
| roughness | The variance in elevation of the points in each cell |
| unknown | Binary indicator of whether there are measurements for each cell |
| VF_{1-N} | The features from the visual mapping (DINO, SAM, GA-NAV) |

Our mapping pipeline is able to run at 10hz on a laptop with a 13th gen Intel i9 CPU and Nvidia 3080 Laptop GPU.

A.4 Model-Predictive Controller

We use MPPI [9] to operate on our learned representation. We use a modified version of MPPI that leverages Gaussian Random Walks and an action library, and a kinematic bicycle model with additional steering and throttle dynamics (Equation 6, $a(x, u), d(x, u)$ were nonlinear functions that we fit on a small sysid dataset, following the equation forms from Mai et al. [1]). Our implementation of MPPI attempts to minimize a weighted sum of distance to goal, and vehicle footprint projection onto the costmap (Equation 7). The speedmap is treated as a log barrier function, and speeds that exceed the value in the speedmap are assigned infinite cost (Equation 8). Values in the uncertainty map are treated as obstacles and also assigned infinite cost. In practice, we also include a footprint (treated as a fixed set of poses calculable from robot state) for the map terms to account for the fact that the vehicle will reside in multiple cells. Costs are averaged over the footprint, which is applied for both training and deployment.

$$X = \begin{bmatrix} x \\ y \\ \psi \\ v \\ \delta \end{bmatrix}, U = \begin{bmatrix} t \\ \delta_{des} \end{bmatrix}, \dot{X} = \begin{bmatrix} v \cos(\psi) \\ v \sin(\psi) \\ v \tan(\delta)/L \\ a(x, u) \\ d(x, u) \end{bmatrix} \quad (6)$$

$$J(\tau, C, S) = K \|p(x_T) - g\|_2 + \sum_t \left[\frac{1}{|F(x_t)|} \sum_{p_t \in F(x_t)} [C(p_t) + \hat{S}(p_t, v(x_t)) + U(p_t)] \right] \quad (7)$$

$$\hat{S}(p_t, v_t) = \begin{cases} \exp(\|v_t\|_2 - S(p_t)), & \text{if } \|v_t\|_2 < S(p_t) \\ \infty, & \text{otherwise} \end{cases} \quad (8)$$

Table 2: MPPI parameters

| Parameter | Value | Description |
|-------------------------|----------------|----------------------------------|
| H | 50 | number of timesteps |
| dt | 0.1s | timestep discretization |
| K | 0.1 | goal weight |
| num actlib samples | 100 | number of action library samples |
| num random walk samples | 512 | number of random walk samples |
| Footprint | $4m \times 2m$ | vehicle footprint |

Table 3: Dataset Description

| Dataset | Num samples | Num unique runs | H | K |
|-----------|-------------|-----------------|-----------|---------|
| Train | 1209 | 9 | 100 (10s) | 20 (2s) |
| Dataset 1 | 429 | 5 | 100 (10s) | 20 (2s) |
| Dataset 2 | 634 | 8 | 100 (10s) | 20 (2s) |

48 A.5 Dataset Generation

49 We generate samples for our dataset \mathcal{D} via the following procedure, given a set of states $x_{1:T}$, maps
50 $\mathbf{M}_{1:T}$, planning horizon H , and spacing k (in our case, $k = 20$, or $2s$):

- 51 1. For every t st. $t \% k = 0$,
- 52 2. Let $\tau_E = x_{t:t+H}$
- 53 3. Let $x_0 = x_t$
- 54 4. Let $x_g = x_{t+H}$
- 55 5. Let $M = \mathbf{M}_t$
- 56 6. Let $\mathcal{D} = \mathcal{D} \cup (x_0, x_g, \tau_E, M)$

57 That is, every k frames, we add to our dataset the current map representation \mathbf{M}_t and next H steps
58 of expert trajectory. The learner will start at the current state x_t and must reach the state H steps in
59 the future x_{t+H} . This process is repeated for each unique run in the dataset. Our dataset statistics are
60 presented in Table 3. Additional qualitative visualizations of the terrain in the datasets is presented
61 in Figure 1.



Figure 1: A qualitative comparison of the test environments. (Top row) Dataset 1 contains open scenarios, slopes, trails and seasonal variation. (Bottom row) Compared to Dataset 1, Dataset 2 has more grassland terrain, vegetation and open slopes.

Table 4: Training parameters

| Parameter | Value | Description |
|-----------------|-----------|---|
| Epochs | 5 | number of epochs |
| Batch Size | 4 | batch size |
| Speed coeff | 1.0 | weight of speed learning and cost learning objectives |
| MPPI iterations | 10 | number of mppi iterations to converge |
| Optimizer | Adam [10] | optimizer |

Table 5: Network Parameters

| Parameter | Value | Description |
|-------------------|------------|--------------------------------|
| E | 16 | ensemble size |
| s_{max} | 15m/s | maximum speed |
| s_n | 15 | number of speed bins |
| hidden sizes | [128, 128] | intermediate feature map sizes |
| hidden activation | tanh | activation type |

62 B Training Details

63 Parameters for our training procedure, network architecture and uncertainty estimation are presented
 64 in Tables 4, 5 and 6, respectively.

65 C Additional Qualitative Results

66 Provided in this section are additional qualitative results that highlight the differences between each
 67 method (Figs. 2 and 3). Figure 2 demonstrates a scenario in which the trail is obscured by tall
 68 vegetation. This vegetation results in both the geometric and semantic analysis underperforming.
 69 However, our continuous-valued features are able to disambiguate the trail, resulting in better cost-
 70 ing. In Figure 3, we are in a scenario in which there is multiple grass types. The grayer grass is
 71 marshy and should be avoided if possible. The features from Dino are best able to capture the intra-
 72 class variation, resulting in better costing for IRL-Dino, as compared to geometry-only and semantic
 73 features.

74 D Additional Future Work

75 While we present results in a previously unseen environment, due to logistical limitations, the en-
 76 vironment is fairly similar to the environment in which the training data was collected (e.g. highly
 77 vegetated, hilly terrain). Further experimentation is required to assess Velociraptor’s capability to
 78 generalize to more varied terrain (e.g. deserts, mountainous terrain, etc.). Such experiments can
 79 include evaluating the importance of adapting the Dino PCA, assessing the generalization of the
 80 network trained in one biome to another, and the amount of expert data required to achieve suffi-
 81 cient performance in a novel environment. Additionally, we would like to demonstrate this system’s
 82 capability on additional platforms.

Table 6: Uncertainty Parameters

| Parameter | Value | Description |
|-----------------------|--------|--|
| num clusters | 8 | number of visual feature clusters |
| distance function | cosine | distance metric for clustering |
| uncertainty threshold | .9 | min value of residual to be considered uncertain |

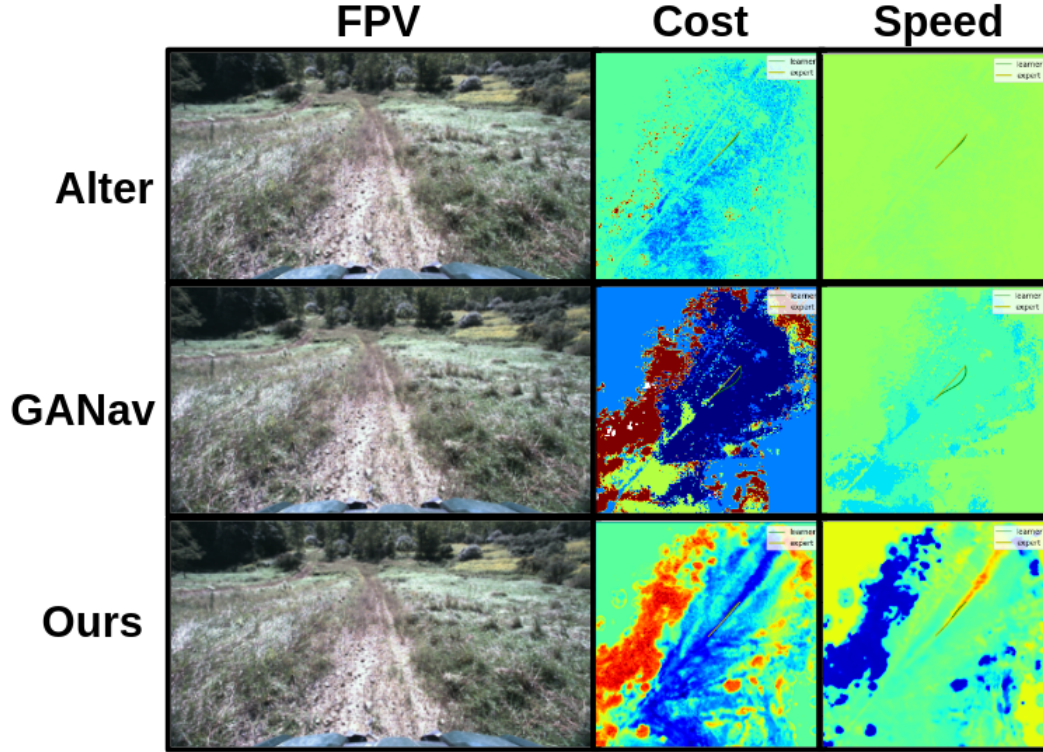


Figure 2: An example from our test set. The trail in front is obscured by grass, negatively impacting the performance of the geometric and semantic baselines. However, our method is able to differentiate between the trail and surrounding vegetation.

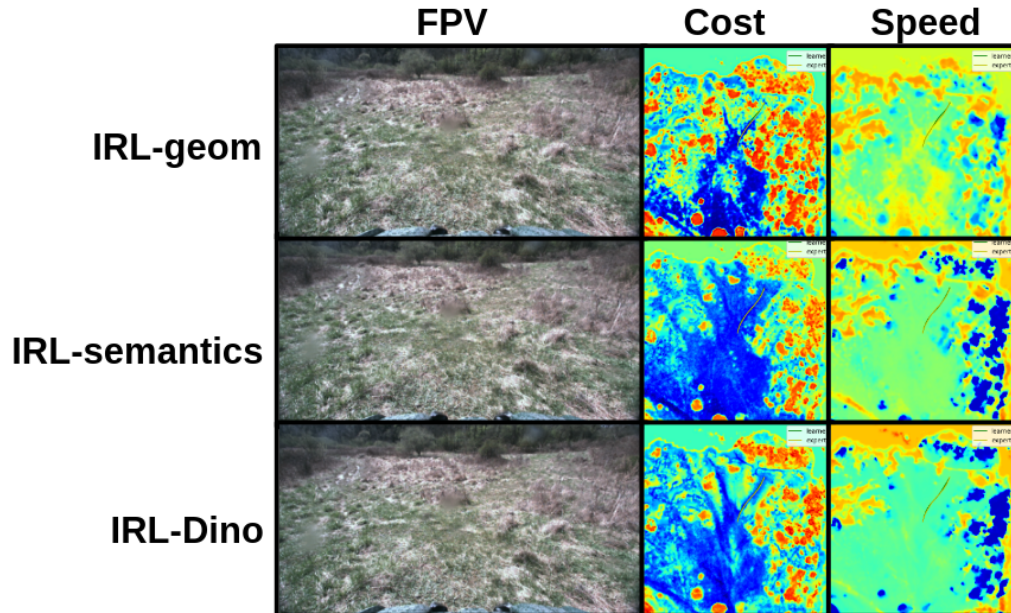


Figure 3: Another example from our test dataset. The brown grass contains marshier soil and was avoided when possible in the train set. IRL trained with only geometric features is not able to fully assign low cost to the grass. IRL trained with semantic features is not able to differentiate between the grass types. IRL trained with Dino features can differentiate between the grass types.

References

- [1] J. Mai. System design, modelling, and control for an off-road autonomous ground vehicle. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, July 2020.
- [2] M. Sivaprakasam, P. Maheshwari, M. G. Castro, S. Triest, M. Nye, S. Willits, A. Saba, W. Wang, and S. Scherer. Tartandrive 2.0: More modalities and better infrastructure to further self-supervised learning research in off-road driving tasks. *arXiv preprint arXiv:2402.01913*, 2024.
- [3] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736. IEEE, 2021.
- [4] T. Guan, D. Kothandaraman, R. Chandra, A. J. Sathiamoorthy, K. Weerakoon, and D. Manocha. Ga-nav: Efficient terrain segmentation for robot navigation in unstructured outdoor environments. *IEEE Robotics and Automation Letters*, 7(3):8138–8145, 2022.
- [5] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- [6] N. Keetha, A. Mishra, J. Karhade, K. M. Jatavallabhula, S. Scherer, M. Krishna, and S. Garg. Anyloc: Towards universal visual place recognition. *IEEE Robotics and Automation Letters*, 2023.
- [7] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.
- [8] J. Yang, B. Ivanovic, O. Litany, X. Weng, S. W. Kim, B. Li, T. Che, D. Xu, S. Fidler, M. Pavone, et al. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *arXiv preprint arXiv:2311.02077*, 2023.
- [9] G. Williams, A. Aldrich, and E. A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017.
- [10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.