Neural Elevation Models for Terrain Mapping and Path Planning

Adam Dai[1], Shubh Gupta[2] and Grace Gao[2]

***Abstract*— This work introduces Neural Elevations Models (NEMos), which adapt Neural Radiance Fields to a 2.5D continuous and differentiable terrain model. In contrast to traditional terrain representations such as digital elevation models, NEMos can be readily generated from imagery, a low-cost data source, and provide a lightweight representation of terrain through an implicit continuous and differentiable height field. We propose a novel method for jointly training a height field and radiance field within a NeRF framework, leveraging quantile regression. Additionally, we introduce a path planning algorithm that performs gradient-based optimization of a continuous cost function for minimizing distance, slope changes, and control effort, enabled by differentiability of the height field. We perform experiments on simulated and real-world terrain imagery, demonstrating NEMos ability to generate high-quality reconstructions and produce smoother paths compared to discrete path planning methods. Future work will explore the incorporation of features and semantics into the height field, creating a generalized terrain model.**

I. Introduction

Autonomous ground robots operating in challenging terrain play a vital role in various applications, such as search and rescue operations in disaster zones, infrastructure inspection in remote areas, and planetary exploration. These environments are highly diverse and complex, with uneven surfaces, loose materials, and difficult-to-model terramechanics posing significant risks to the robot's functionality and mission success if their intricacies are not factored into navigation. Therefore, there is a need for navigation methods that can both effectively model these environments and leverage those models for safe and efficient navigation.

Traditionally, terrain details have been captured through expensive and time-consuming high-fidelity surveys using 3D sensors like LiDAR and Radar. Ranging data is then used to generate a Digital Elevation Model (DEM) of the terrain. DEMs represent elevation data using either a discrete grid of squares (raster) or a network of interconnected triangles (mesh). Path planning typically consists of generating a costmap from the DEM based on slope and roughness and using global search algorithms such as A* or Field D* [1], [2]. While high-fidelity DEMs provide detailed representation of terrain, their reliance on expensive sensors makes them prohibitive to use at scale and on changing terrain.

To address the limitations of expensive 3D sensors, methods like multi-view stereo (MVS) [3] and photogrammetry rely on inexpensive cameras to reconstruct 3D environments. These techniques leverage multiple images of the terrain captured from different viewpoints. However, these methods rely on fixed, discrete representations (e.g., point clouds or meshes), limiting their resolution and potentially leading to artifacts when capturing intricate details. Additionally, they are highly sensitive to image quality (e.g., lighting, occlusion, texture) and can be computationally demanding, especially for large-scale scenes.

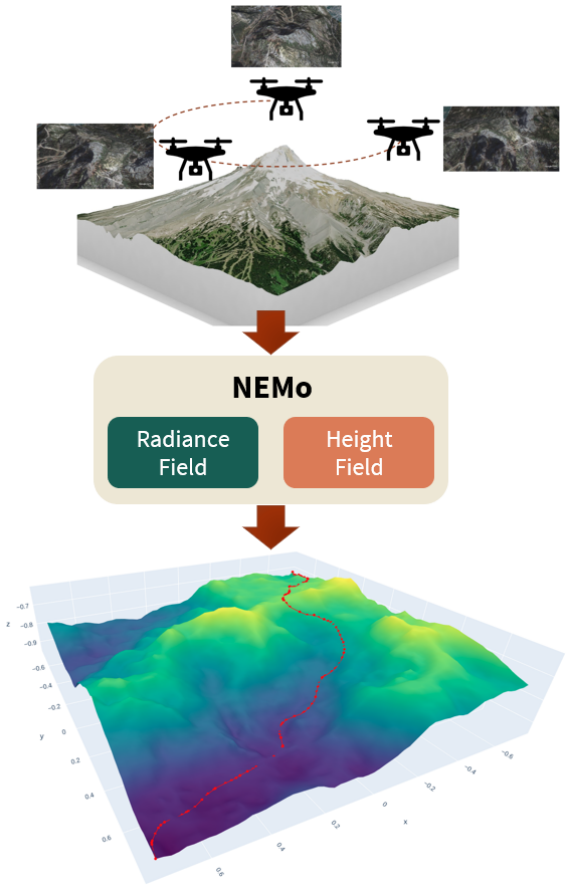


Fig. 1: Neural Elevation Model (NEMo) framework. Aerial imagery of terrain is captured and used to simultaneously train a radiance field and height field of the scene. The height field is then used as a continuous and differentiable map representation for gradient-based path planning.

Recently, Neural Radiance Fields (NeRFs) [4] have emerged as powerful tools for environment representation using camera images. NeRFs can be directly generated from posed images and have demonstrated remarkable performance in capturing visual detail in challenging conditions [5]. Compared to DEMs, NeRFs offer several advan-

[1]Department of Electrical Engineering, Stanford, CA 94305, USA, addai@stanford.edu

[2]Department of Aeronautics and Astronautics, Stanford, CA 94305, USA {shubhgup, gracegao}@stanford.edu

tages: faster generation, denser reconstruction, and a compact yet continuous and differentiable representation. However, their usability for terrain navigation suffers from two key limitations:

- **High Processing Overhead:** Unlike DEMs, which provide interpretable height information associated with a ground coordinate, NeRFs capture a full scene with implicit density and color. This richer representation, while powerful, requires additional processing steps to extract the relevant terrain data for informing navigation.
- **Focus on Complex Scenes:** NeRFs were initially designed to represent a variety of complex 3D scenes, including visual properties of buildings and several non-terrain elements [6], [7]. This can introduce unnecessary complexity for ground robots solely focused on navigating the terrain.

These limitations highlight the potential for combining both approaches. By leveraging the strengths of NeRFs in capturing complex terrain detail and the suitability of DEMs for path planning, we can create more robust and versatile navigation solutions.

In this work, we introduce Neural Elevation Models (NEMos), which integrate a Neural Radiance Field alongside a height field. We propose a novel approach for jointly training the NeRF and the height field, in which height is learned from the NeRF through quantile regression, and, in turn, the height network supervises the NeRF to eliminate spurious density above the surface. We then demonstrate a method for path planning that leverages the continuous and differentiable nature of the height network to achieve smoother paths than those obtained via discrete planning over equivalent DEMs. The field of Neural Elevation Models is still nascent, so we demonstrate initial results in this paper and discuss directions for future research and exploration.

II. Terrain Mapping

Our objective is to construct a map of terrain from aerial images $\{I_1, \ldots, I_N\}$ with associated camera poses $\{T_1, \ldots, T_N\}$. We assume that the camera poses are provided in an East-North-Up (ENU) frame. This ensures that the scene's ground plane aligns with the *XY* plane in the NeRF coordinate system, which is crucial for accurate elevation estimation during terrain reconstruction.

A. Neural Elevation Model (NEMo)

NEMo offers a novel approach to continuous terrain elevation representation, allowing direct generation from 2D camera images without additional depth data.

NEMo simultaneously trains two models:

- A *NeRF model* captures the overall 3D scene with density and color information. We use $\rho : \mathbb{R}^3 \to \mathbb{R}^+$ to refer to the NeRF density.
- A *height field*, parametrized using a neural network, specifically focuses on predicting terrain elevation. The height field is denoted as $\mathcal{H} : \mathbb{R}^2 \to \mathbb{R}$, which maps any ground coordinate $(x, y) \in \mathbb{R}^2$ to a corresponding height value $z \in \mathbb{R}$ in the scene.

After the training, the NeRF component is discarded and only the height field is retained to maintain a compact representation.

B. Training

1) NeRF Training: NEMo employs a standard training procedure for the NeRF component, minimizing a volumetric rendering loss function [8]. Additionally, the estimated height from the height network is used to mask out regions that exceed the predicted elevation. This effectively removes artifacts or "floaters" above the actual terrain, allowing the NeRF to focus on the surface for accurate scene reconstruction. We observed that this masking process during the NeRF training leads to a cleaner NeRF representation with fewer artifacts.

2) Height Network Training: The height network leverages quantile regression [9] to learn the height information from images alongside the NeRF density distribution. During training, for each ray cast onto the scene, we sample n 3D points $\{(x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)\}$ along its path. We then compute a quantile loss $\mathcal{L}_q$ based on the error $e_i = z_i - \hat{z}_i$ between height z_i of the i-th point and predicted height $\hat{z}_i$ from the network. To prioritize the information from occupied space, we weight the loss using a factor $w_i = T_i \alpha_i$, where T_i is transmittance and α_i is opacity, which are computed along the ray based on the learned density ρ. The total loss $\mathcal{L}$ as a function of the predicted heights $\hat{\mathbf{z}}$ for training the height network is obtained as follows:

$$\mathcal{L}(\hat{\mathbf{z}}) = \sum_{i=1}^{n} w_i \mathcal{L}_q(z_i, \hat{z}_i) \tag{1}$$

$$\mathcal{L}_q(z_i, \hat{z}_i) = \begin{cases} -(1-q) \cdot e_i & \text{if } e_i < 0 \\ q \cdot e_i & \text{if } e_i \geq 0 \end{cases} \tag{2}$$

Intuitively, when the error is negative and $\hat{z}_i$ is greater than z_i, the loss pulls the estimate $\hat{z}_i$ down with strength proportional to $(1-q)$. Conversely, when the error is positive and $\hat{z}_i$ is less than z_i, the loss pulls the estimate up with strength proportional to q. This loss, also known as the "pinball loss," has been shown to converge to the quantile q [10]. The quantile regression process is illustrated in Fig. 2.

III. Path Planning

Now, given trained height field $\mathcal{H} : (x, y) \to z$, a start location (x_0, y_0), and goal location (x_f, y_f), our objective is to plan a path $\mathcal{P}$ from start to goal which minimizes (1) distance traveled, (2) slope along the path, and (3) necessary control effort. The implicit nature of $\mathcal{H}$ allows us to formulate a cost function and optimize for this path in continous space. Our planning approach is most similar to that of [11], in which a path is initialized with A*, then optimized with gradient descent for control cost and collision avoidance—however, in our case, we penalize path slope from our height field as opposed to collision avoidance from the NeRF density field.

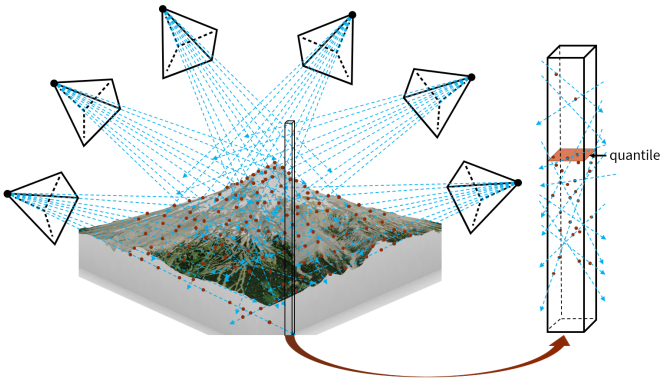


Fig. 2: Height field training via quantile regression. Training images cast rays (depicted in light blue) into the scene, along which points are sampled (dark red). By minimizing the weighted quantile loss over the multitude of sampled points, we effectively regress the desired quantile of vertical NeRF density as height.

A. Initialization

In practice the cost manifold possesses many local minima, and thus we first initialize with A* over a discretized version of $\mathcal{H}$ to obtain an initial solution in the neighborhood of the global optimum. We use change in height between cells for cost, and thus A* attempts to optimize for objectives (1) and (2). Note that the worst-case runtime of A* scales roughly exponentially with respect to grid size, and thus it is prohibitively expensive to run on larger grids.

B. Continuous Path Optimization

Next, we leverage the fact that $\mathcal{H}$ allows us to evaluate the height and gradients (through automatic differentiation) of an arbitrary point in continuous space to formulate the path optimization over a continuous path. For curve $\mathcal{P}: [0,T] \rightarrow \mathbb{R}^2$ which represents a 2D path, we parameterize $\mathcal{P}$ with a series of differentially flat output waypoints $W = \{\sigma_1, \ldots, \sigma_N\}$ under dynamics $\dot{x} = f(x,u)$. The dynamics can be used to integrate between waypoints and evaluate $\mathcal{P}$ at any $\tau \in [0,T]$, thus providing a continuous representation of the path, $\mathcal{P}(\tau)$.

Now, in terms of $\mathcal{P}(\tau)$, we formulate three cost terms J_1, J_2, J_3 for minimizing the three desired path objectives mentioned above:

(1) *Path distance*: This is computed as arc length by integrating path speed $\left\|\frac{\partial \mathcal{P}}{\partial t}\right\|$ over the path. Thus,

$$J_1(\tau) = \left\| \left. \frac{\partial \mathcal{P}}{\partial t} \right|_\tau \right\|. \tag{3}$$

(2) *Slope along path*: The slope of the terrain along the path at $\mathcal{P}(\tau)$ is the directional derivative of $\mathcal{H}$ in the path direction $\frac{\partial \mathcal{P}}{\partial t}$. Thus,

$$J_2(\tau) = \left. \frac{\partial \mathcal{P}}{\partial t} \right|_\tau \cdot \nabla \mathcal{H}(\mathcal{P}(\tau)). \tag{4}$$

(3) *Control effort*: The control $u(\tau)$ at τ is determined from the differential flatness mapping [12]. We then write

$$J_3(\tau) = u(\tau)^T R u(\tau), \tag{5}$$

where R is a weighting matrix.

The total path cost is expressed as the integral over the path of the weighted sum of these cost terms:

$$J_{total} = \int_0^T \beta_1 J_1(\tau) + \beta_2 J_2(\tau) + J_3(\tau)\ dt \tag{6}$$

where β_1 and β_2 are weighting scalars.

This cost function is optimized over the waypoints W which parameterize $\mathcal{P}$ to obtain optimal waypoints W^* that parameterize the refined path $\mathcal{P}^*$

$$W^* = \min_W\ J_{total}(W). \tag{7}$$

In practice, to perform optimization we still need to sample points on the path, but the continuous formulation allows us to do so at a much higher resolution for cost evaluation, while still optimizing only over the sparse waypoints W. For future work, we plan to include additional objectives to optimize for traversability based on semantics and features extracted from the NeRF.

IV. Results

For our experiments[1], we consider two different terrain scenes, one from simulated imagery and one from real-world imagery:

1) "**KT-22**": Imagery and camera poses collected from Google Earth Studio [13] of the KT-22 peak in Olympic Valley, CA.
2) "**Red Rocks**": Drone imagery collected by Falcon fixed-wing UAV over Red Rocks, CO [14]. COLMAP [3] is used to estimate camera poses.

For each scene, we train a NEMo and analyze the quality of the NeRF and height field. We then use the height field to plan paths over the terrain and evaluate them with metrics for distance, average slope, and smoothness.

A. NEMo Training

The NeRF component of NEMo is based on the Nerfacto model from Nerfstudio [8]. The height network is implemented as a hashgrid encoding [15] followed by a single-layer MLP with ReLU activation. Fig. 3 shows RGB and depth renders from the trained NeRF component of NEMos for each scene, demonstrating detailed reconstruction with underlying geometry. Anecdotally, we also observe that NEMo NeRFs possess less haze above the scene than those trained with standard Nerfacto, due to the height field masking out spurious density above the ground in free space.

We note that our approach for elevation estimation and representation differs from training a standard NeRF then rendering its top down density as a height map, in that:

[1]Our code is available at: `https://github.com/Stanford-NavLab/nerfstudio/tree/adam/terrain` (NEMo model and training) and `https://github.com/adamdai/neural_elevation_models` (analysis and path planning).

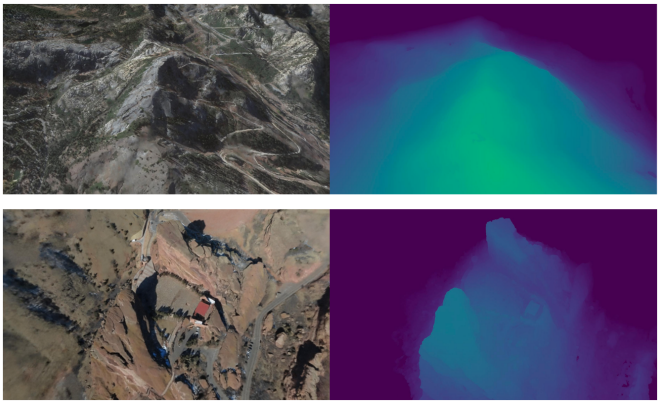


Fig. 3: RGB (left) and depth (right) renders from trained NEMos of the scenes used in our experiments. KT-22 is shown on the top row, and Red Rocks on the bottow row.

(1) a render must sample points and thus is an explicit and discrete representation whereas the height network is an implicit function, and (2) the NEMo jointly trains the NeRF and height network, allowing the NeRF to benefit from height supervision.

B. Path Planning

For path optimization, we use a Dubin's car dynamics model for differential flatness, and employ the Adam [16] optimizer with learning rate 1×10^{-3}. Fig. 4 shows path planning results for both scenes. We evaluate the planned paths based on three metrics: 2D path distance, average slope, and smoothness. Smoothness is computed as estimated jerk from triple differencing the positions (lower is smoother). We report a comparison of these metrics in Table I for the select paths shown in Fig. 4.

The path optimization produces much smoother and dynamically feasible paths. The average slope is similar to or slightly higher than that of A*—this is due to A* utilizing switchbacks to minimize vertical transition while sacrificing smoothness, while the path optimization is currently unable to route smooth switchbacks. This behavior should be possible with further improvement and tuning of the planning algorithm.

TABLE I: Path planning metrics.

	Path distance (2D)	Average slope	Smoothness
A* path	4.892	0.347	0.0258
Refined path	3.278	0.365	0.0174

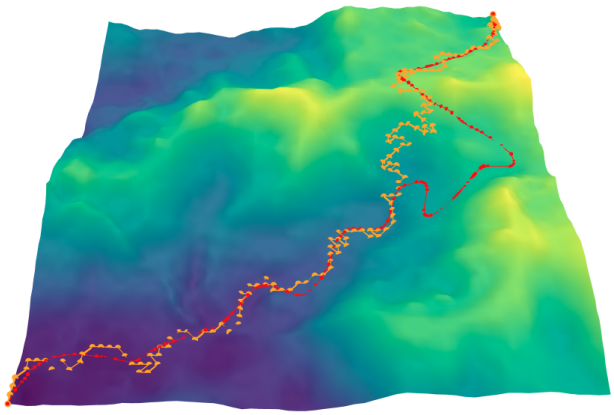
(a) KT-22

	Path distance (2D)	Average slope	Smoothness
A* path	2.932	0.398	0.0195
Refined path	2.172	0.412	0.0138

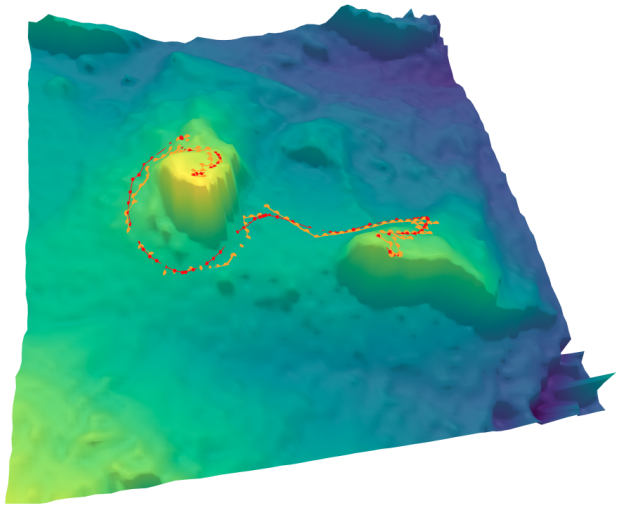
(b) Red Rocks

V. Conclusion

In this work, we propose Neural Elevation Models, a novel representation for terrain adapted from Neural Radiance Fields. This representation leverages the advantages of NeRFs in fast generation solely from imagery and rich visual reconstruction, while also capturing compact terrain geometry in the form of a continuous and differentiable height field. As next steps we plan to distill feature and semantic information from the NeRF into the height network, which can then be readily leveraged for path planning. In addition, we aim to eventually perform validation of planned paths with rovers in simulated and real-world environments.



(a) Planned paths for KT-22 height field.



(b) Planned paths for Red Rocks height field.

Fig. 4: Path planning on the NEMo height field. The initial A* path is shown in orange, while the refined path is shown in red. The refined paths from method designed for the implicit NEMo height field are smoother and more efficient.

Acknowledgment

The authors would like to thank Professor Mac Schwager for very insightful discussion, and Daniel Neamati for valuable feedback and for reviewing this paper.

References

[1] M. Ono, B. Rothrock, E. Almeida, A. Ansar, R. Otero, A. Huertas, and M. Heverly, "Data-Driven Surface Traversability Analysis for Mars 2020 Landing Site Selection," in *2016 IEEE Aerospace Conference*. IEEE, 2016, pp. 1–12.

[2] J. Carsten, A. Rankin, D. Ferguson, and A. Stentz, "Global Path Planning on Board the Mars Exploration Rovers," in *2007 IEEE Aerospace Conference*. IEEE, 2007, pp. 1–11.
[3] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
[5] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li, "NeRF: Neural Radiance Field in 3D Vision, A Comprehensive Review," *arXiv preprint arXiv:2210.00379*, 2022.
[6] R. Marí, G. Facciolo, and T. Ehret, "Sat-NeRF: Learning Multi-View Satellite Photogrammetry with Transient Objects and Shadow Modeling using RPC Cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1311–1321.
[7] Y. Xiangli, L. Xu, X. Pan, N. Zhao, A. Rao, C. Theobalt, B. Dai, and D. Lin, "BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-Scale Scene Rendering," in *European conference on computer vision*. Springer, 2022, pp. 106–122.
[8] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, *et al.*, "Nerfstudio: A Modular Framework for Neural Radiance Field Development," in *ACM SIGGRAPH 2023 Conference Proceedings*, 2023, pp. 1–12.
[9] R. Koenker and K. F. Hallock, "Quantile Regression," *Journal of economic perspectives*, vol. 15, no. 4, pp. 143–156, 2001.
[10] Y. Yang, H. J. Wang, and X. He, "Posterior Inference in Bayesian Quantile Regression with Asymmetric Laplace Likelihood," *International Statistical Review*, vol. 84, no. 3, pp. 327–344, 2016.
[11] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-Only Robot Navigation in a Neural Radiance World," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022.
[12] J. Lévine, "On Necessary and Sufficient Conditions for Differential Flatness," *Applicable Algebra in Engineering, Communication and Computing*, vol. 22, no. 1, pp. 47–90, 2011.
[13] "Google Earth Studio." [Online]. Available: https://earth.google.com/studio/
[14] "DroneMapper Sample Data," https://dronemapper.com/sample_data/, accessed: March 19, 2024.
[15] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding," *ACM transactions on graphics (TOG)*, vol. 41, no. 4, pp. 1–15, 2022.
[16] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.