2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Zero-Shot Metric Depth Estimation via Monocular Visual-Inertial Rescaling for Autonomous Aerial Navigation

Steven Yang*, Xiaoyu Tian*, Kshitij Goel, and Wennie Tabib

***Abstract*— This paper presents a methodology to predict metric depth from monocular RGB images and an inertial measurement unit (IMU). To enable collision avoidance during autonomous flight, prior works either leverage heavy sensors (e.g., LiDARs or stereo cameras) or data-intensive and domain-specific fine-tuning of monocular metric depth estimation methods. In contrast, we propose several lightweight zero-shot rescaling strategies to obtain metric depth from relative depth estimates via the sparse 3D feature map created using a visual-inertial navigation system. These strategies are compared for their accuracy in diverse simulation environments. The best performing approach, which leverages monotonic spline fitting, is deployed in the real-world on a compute-constrained quadrotor. We obtain on-board metric depth estimates at 15 Hz and demonstrate successful collision avoidance after integrating the proposed method with a motion primitives-based planner.**

I. INTRODUCTION

First Person View (FPV) drone pilots leverage a single forward-facing camera video stream transmitted over a radio feed and sensors embedded in the flight controller (e.g., IMU) to aggressively maneuver through dense clutter (e.g., through tree branches, under bridges, etc.). In contrast, autonomous aerial systems leverage stereo cameras [1] or heavy onboard LiDARs [2] for perception and collision avoidance. The addition of sensors increases the system's size and mass, which reduces flight time. The objective of this paper is to demonstrate collision avoidance in flight with the same set of minimal sensors (a single camera and IMU) used by FPV drone pilots to navigate in cluttered environments.

Prior works leverage monocular depth prediction for aerial navigation via interpolating methods such as the plane sweeping algorithm [3] to enable motion planning with a single RGB camera and an IMU [4]. While successful in large open environments, the interpolation does not provide sufficient accuracy to avoid thin obstacles in cluttered environments. Learning-based metrically-accurate monocular depth estimation (MDE) is a promising alternative [5, 6]. However, these zero-shot approaches require retraining with domain-specific images for metrically-accurate results. Robots operating in *a priori* unknown environments (e.g., search and rescue robots) do not have access to data beforehand, so retraining is challenging for these applications.

Alternatively, recent works correct or complete the depth data reported by an RGB-D sensor (e.g., RealSense[1]) using monocular depth estimation (MDE) neural networks and rescaling operations for metric accuracy [7, 8]. However, we are interested in using one RGB camera and an IMU instead of an active stereo depth sensor. Prior zero-shot methods build upon relative MDE methods and propose visual-inertial fine-tuning [9] or sparse feature depth test-time adaptation [10] for metrically accurate depth images. However, the former method requires re-training the MDE networks that may lead to loss of generalization and the latter is evaluated using ground truth data as input.

To bridge these gaps in the state of the art, we contribute (1) a zero-shot metric depth estimation method that leverages the sparse 3D feature map from a visual-inertial navigation system (VINS) for rescaling predicted relative depth to metric depth, (2) analysis of the proposed rescaling techniques in challenging simulation environments, (3) hardware validation of the proposed approach in challenging, real-world

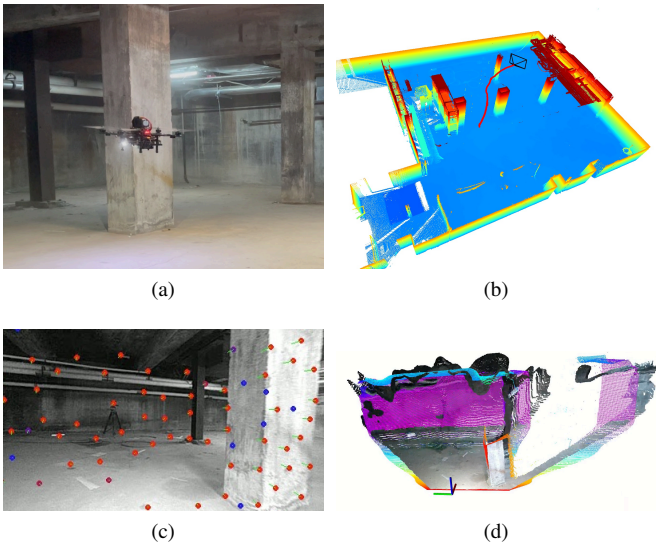


Fig. 1: Image and data corresponding to one hardware experiment to demonstrate collision avoidance during autonomous navigation by using data from a monocular camera and IMU to rescale relative depth measurements from an MDE network and obtain metric depth. (a) illustrates the quadrotor aerial robot navigating in the industrial tunnel environment. (b) illustrates the trajectory plotted in red on top of the environment reconstructed from survey-grade FARO scans. This represents the trajectory for the entire flight trial. The robot uses the proposed approach to select actions that avoid the two pillars in the environment. (c) shows the features tracked in the forward-facing camera, which are used to rescale the predicted image. (d) plots the point cloud generated using our approach in colors ranging from red (closer) to purple (further away) as well as the colorized point cloud from a RealSense sensor generated using active stereo.

* Equal Contributions.

The authors are with the Robotics Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA. {yiy6,xiaoyut,kgoel1,wtabib}@andrew.cmu.edu.

[1]http://realsenseai.com

environments onboard a size, weight, and power (SWaP)-constrained quadrotor aerial system[2] (Fig. 1), and (4) open-source software release of the rescaling method[3].

II. RELATED WORK

Zero-shot methods that output metric depth given a monocular RGB image include ZoeDepth [11], UniDepth [5, 12], Metric3D [13, 14], and Depth Anything [6, 15]. ZoeDepth requires expensive fine-tuning for metric depth estimation if the camera calibration at test-time is different from the one used for training. UniDepth attempts to learn the camera calibration during training for enhanced generalization. However, camera calibration data is required at training time, which means that UniDepth is unable to leverage large datasets of uncalibrated images. Metric3D is able to introduce invariance to camera calibration at the cost of high test-time compute requirements. Depth Anything can leverage uncalibrated image datasets but recommends fine-tuning for high metric accuracy on domain-specific data for downstream applications. However, for many real-world robotics tasks, the environmental conditions are *a priori* unknown (e.g., search and rescue). Therefore, we present an approach that provides zero-shot metrically-accurate depth estimates from learning-based relative depth predictions without expensive environment- or camera-specific fine-tuning.

Wofk et al. [9] present a post-training approach to resolve the scale ambiguity in relative depth estimates and obtain metric depth. Marsal et al. [10], which is most similar to our work, rescale the affine-invariant depth estimates output from an MDE (i.e., Depth Anything [6, 15]). Given a sparse set of 3D points, the approach performs a linear regression to estimate two parameters for rescaling the depth estimates. RANSAC [16] is used to ensure robustness in the regression solution. However, the sparse set of 3D points is generated using ground truth data instead of real sensor data, which may be noisy. The LiDAR sensor is simulated by sampling points along a line in ground truth depth images. To simulate SFM points, the authors match features (e.g., SIFT [17]) across consecutive image frames and triangulate the 3D feature locations using the ground truth poses between the images. In both cases, the sparse 3D feature sets are very accurate because they leverage ground truth information, which is not representative of real-world operation. In contrast, our approach relies on the sparse 3D feature points estimated from a sliding window optimizer [18]. We also do not need extra onboard sensors like heavy LiDARs or hardware triggered stereo cameras. Moreover, the evaluation in [10] is conducted only on publicly available datasets in postprocessing whereas we deploy our approach to a SWaP-constrained aerial robot.

Specific to aerial navigation, Saviolo et al. [7] enhance the depth estimation of the Intel Realsense D435i by feeding the color camera images into DepthAnythingV2 [15] and using the output to fill holes in the depth image created from the time-synchronized IR stereo cameras. The monocular depth estimation network estimates depth but the values are not guaranteed to be correct up-to-scale [8]. To correct for these scale inaccuracies, the authors fit a quadratic polynomial to the known depth in the active stereo depth image and corresponding depth values in the learnt depth image. This polynomial is used to rescale the depth values for the learnt depth image and fill holes in the active stereo depth completion. In contrast, we provide an approach to use only one RGB camera instead of a stereo camera setup and use higher-fidelity but lightweight rescaling methods (e.g., exponential, monotonic splines) for metric accuracy.

III. METHODOLOGY

A. Sparse Feature Depth Map Generation

This section describes how we derive sparse depth maps. We use the monocular visual-inertial navigation system detailed in [18] with minor modifications. We summarize the approach and detail the modifications for our application in the following paragraphs.

Given a monocular RGB image, Shi-Tomasi corners [19] are detected and a minimum distance is enforced between nearby features to enable salient features to be tracked across the image frame. The pyramidal Lucas-Kanade method [20] is used to find feature correspondences between consecutive frames and outliers are rejected using RANSAC [16, 21]. The IMU motion, biases, and feature locations are optimized by minimizing a nonlinear objective function over a sliding window of image keyframes that encodes both IMU- and vision-derived motion constraints. This optimizer provides a sparse 3D feature map with respect to the earliest keyframe camera pose at a rate slower than the camera framerate. To obtain 3D feature depths in the current camera frame, we project the feature map using the high-rate odometry generated via IMU forward-propagation (also referred to as upsampled odometry). Formally, if the earliest keyframe is C_i and the current camera frame is C_j, we can calculate each feature position ${}_{C_j}\mathbf{p}$ from ${}_{C_i}\mathbf{p}$ through the transformations

$$
\begin{aligned}
{}_{W}\mathbf{p} &= \mathbf{R}_{WB_i}\left(\mathbf{R}_{B_iC_i}\,{}_{C_i}\mathbf{p} + \mathbf{t}_{B_iC_i}\right) + \mathbf{t}_{WB_i} \\
{}_{C_j}\mathbf{p} &= \mathbf{R}_{B_jC_j}^{-1}\left((\mathbf{R}_{WB_j})^{-1}({}_{W}\mathbf{p} - \mathbf{t}_{WB_j}) - \mathbf{t}_{B_jC_j}\right),
\end{aligned}
$$

where B and W denote the body and world frames, respectively, $\mathbf{R}_{PQ}$ denotes the rotation matrix and $\mathbf{t}_{PQ}$ the translation vector from a frame P to a frame Q. The quantities $\mathbf{R}_{WB_i}$, $\mathbf{R}_{B_iC_i}$, $\mathbf{t}_{B_iC_i}$, $\mathbf{t}_{WB_i}$, $\mathbf{R}_{B_jC_j}$, $\mathbf{R}_{WB_j}$, $\mathbf{t}_{WB_j}$, and $\mathbf{t}_{B_jC_j}$ are known via the upsampled odometry and extrinsic calibration.

With these temporally aligned feature positions, we obtain the corresponding pixel locations in the image plane using the pinhole camera model. For fractional locations, we choose the surrounding four integer locations and select the smallest relative depth location which represents the nearest pixel to the robot. This is to avoid predicting an obstacle to be further away than it actually is.

[2]A video of the experiments may be found at https://youtu.be/t6FajgB06Vc

[3]https://github.com/rislab/mono_depth_rescaler

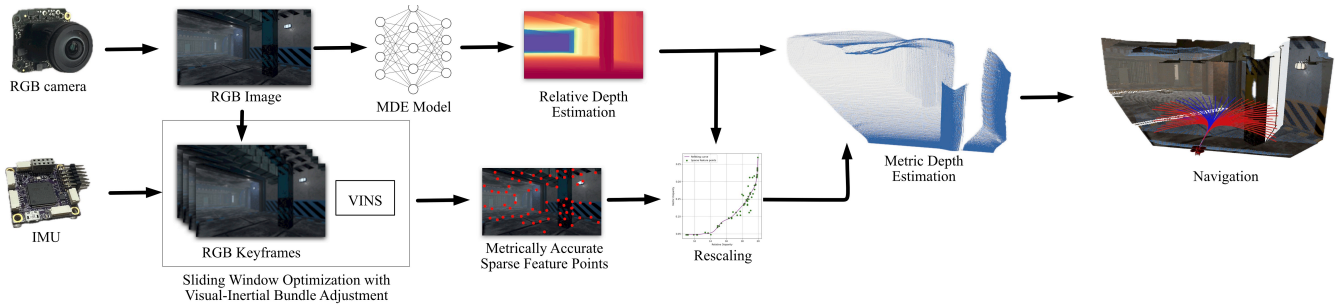


Fig. 2: Overview of the approach to rescale predicted depth from an MDE network using a metrically accurate 3D sparse feature map from a VIN system. The RGB camera image is used by an MDE network to predict a depth image consisting of relative depth estimates. The RGB camera images and IMU data are also used to produce a sparse set of metrically accurate 3D features. We leverage a monotonic spline to rescale the relative depth estimates so that they are metrically accurate. The resulting rescaled depth image is used for navigation.

B. Depth Rescaling

Given an RGB image, let $z_{\text{rel}}(i)$ and $z_{\text{met}}(i)$ be the estimated relative and metric depth values at a pixel i from the learnt depth model and the sparse feature map, respectively. Assuming $z_{\text{rel}}(i), z_{\text{met}}(i) > 0$, we can formulate the depth rescaling problem in terms of disparities $d_{\text{rel}}(i) = \frac{1}{z_{\text{rel}}(i)}$ and $d_{\text{met}}(i) = \frac{1}{z_{\text{met}}(i)}$ [15]. If there are N pixels where the disparity values $(d_{\text{rel}}(i), d_{\text{met}}(i))\, \forall i \in \{1, \ldots, N\}$ are valid, the depth rescaling objective is to derive a scalar-valued function f that maps the relative disparity to metric disparity. This section details rescaling methodologies that leverage several forms of f: polynomial, exponential, smoothing (cubic) splines, monotonic smoothing splines, and monotonic splines.

1) Polynomial: If f is a n-th degree polynomial parameterized by the coefficient vector $\mathbf{a}$, we solve for $\mathbf{a}$ via

$$\min_{\mathbf{a}} \sum_{i}^{N} \left(f(d_{\text{rel}}(i)) - d_{\text{met}}(i)\right)^2. \tag{1}$$

The solution to this least-squares problem is given by

$$\mathbf{a} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{d}_{\text{met}},$$

where $\mathbf{d}_{\text{met}} = [d_{\text{met}}(1), \ldots, d_{\text{met}}(N)]^\top$ and $\mathbf{X}$ is the $N \times (n+1)$ Vandermonde matrix for $\mathbf{d}_{\text{rel}} = [d_{\text{rel}}(1), \ldots, d_{\text{rel}}(N)]^\top$.

2) Exponential: Let f be an exponential function in the form $f(d_{\text{rel}}) = a \cdot e^{b d_{\text{rel}}}$. If $g(d_{\text{rel}}) = \ln f(d_{\text{rel}}) = \ln a + b \cdot d_{\text{rel}} = a' + b \cdot d_{\text{rel}}$, we can use the ordinary least-squares from Eq. (1) using g:

$$\min_{a', b} \sum_{i}^{N} \left(g(d_{\text{rel}}(i)) - \ln d_{\text{met}}(i)\right)^2. \tag{2}$$

Finally, we get the original coefficient a using $a = e^{a'}$.

3) Smoothing Spline: We utilize the cubic C^2 spline function fitting algorithm derived from Dierckx [22]. The details are referenced here for completeness. For a cubic C^2 spline function f, we start with evenly-spaced and sorted knots k_j for $j = \{4, 5, \ldots, n-3\}$ defined along domain $[k_4, k_{n-3}]$ such that: (1) Between each interval (k_j, k_{j+1}), $j = 4, \ldots, n-4$, f is defined as some polynomial with degree 3 or less and (2) f is C^2 continuous across $[k_4, k_{n-3}]$. Using this information, we want to minimize the following criterion:

$$\sum_{r=5}^{n-4} p_r^2$$

subject to the constraint

$$\sum_{i=1}^{N} \left(d_{\text{met}}(i) - f(d_{\text{rel}}(i))\right)^2 \leq S,$$

where p_r is the third derivative jump discontinuity of f at knot position k_r, defined explicitly as $p_r = f^{(3)}(k_r^+) - f^{(3)}(k_r^-)$ that represents smoothness, and $S \geq 0$ is a smoothing hyperparameter that controls the trade-off between minimizing error and maximizing smoothness. A smaller S represents more interpolation towards the data points, while a larger S emphasizes smoothness.

4) Monotonic Smoothing Splines: We leverage the unimodal smoothing formulation from [23] in this section to enforce monotonicity and smoothness. Consider m knots $k_1, \ldots, k_m$. Let the cubic (degree $t = 3$) B-spline basis functions recursively defined by De Boor [24] as

$$\phi_{i,0}(d_{\text{rel}}) = \begin{cases} 1, & k_i \leq d_{\text{rel}} < k_{i+1} \\ 0, & \text{otherwise} \end{cases} \qquad 1 \leq i \leq m-1.$$

For degree $t \geq 1$,

$$\phi_{i,t}(d_{\text{rel}}) = \frac{d_{\text{rel}} - k_i}{k_{i+t} - k_i}\, \phi_{i,t-1}(d_{\text{rel}}) + \frac{k_{i+t+1} - d_{\text{rel}}}{k_{i+t+1} - k_{i+1}}\, \phi_{i+1,t-1}(d_{\text{rel}}),$$

where $1 \leq i \leq m-t-1$. Let β_j be the basis coefficients. Any cubic B-spline function can thus be written as

$$f(d_{\text{rel}}) = \sum_{j=1}^{m} \beta_j\, \phi_{j,3}(d_{\text{rel}})$$

where $\phi_{j,3}$ denotes the cubic basis functions for the jth knot.

To derive the function, we need to find the β_j. We solve this through penalized least squares by minimizing

$$\min_{\boldsymbol{\beta}} \|\mathbf{d}_{\text{met}} - \mathbf{B}\boldsymbol{\beta}\|^2 + \lambda \|\mathbf{D}^{(3)}\boldsymbol{\beta}\|^2 + \kappa \|\mathbf{V}^{1/2}\mathbf{D}^{(1)}\boldsymbol{\beta}\|^2$$

where $\mathbf{B}$ is the basis matrix with entries $B_{ij} = \phi_j(d_{\text{rel}}(i))$, $\mathbf{D}^{(k)}$ is the finite-difference matrix for the kth derivative estimation, $\mathbf{V}$ is a diagonal matrix where $V_{kk} = 1$ if $\mathbf{D}^{(1)}\boldsymbol{\beta} < 0$ and 0 otherwise, λ can be used to prioritize smoothing, and κ is the non-monotonicity penalty.

The first term of this objective ensures that f is a *fitting spline*. The second term enables smoothness by minimizing the third derivative of f (similar to minimizing jerk). The third term biases f towards monotonicity.

5) Monotonic Splines: This formulation is the same as the preceeding section, but we remove the smoothness term:

$$\min_{\boldsymbol{\beta}} \ \|\mathbf{d}_{\text{met}} - \mathbf{B}\boldsymbol{\beta}\|^2 + \kappa\|\mathbf{V}^{1/2}\mathbf{D}^{(1)}\boldsymbol{\beta}\|^2.$$

C. Implementation Details

Depth values outside the range of $0.05\,\text{m}$ to $65\,\text{m}$ are clipped, only counting as valid the pixels that have ground truth depth within that range for each frame. The upper bound is set to $65\,\text{m}$ assuming the 16-bit unsigned integer encoding for depth is represented in millimeters. The maximal representable value without overflow is $65\,\text{m}$, which is also far enough to support analysis for high-speed navigation. The $0.05\,\text{m}$ lower bound excludes invalid or zero measurements.

The sample size, or the number of features, plays an important role in determining the accuracy of the final fitting curve. The required number of sparse features varies depending on the rescaling method. In general, we require that there be at least 10 sparse features, which is empirically determined. If VINS does not provide enough sparse features, we skip the frame with no results being produced. For all the datasets, this occurs only when VINS is initializing in the first frame. Furthermore, the number of knots for monotonic smoothing and monotonic spline rescaling is a hyperparameter that can be tuned for better results depending on the model and constraints. For our experiments, we find that 10 knots works well.

IV. EXPERIMENTAL DESIGN AND RESULTS

A. Simulation Datasets

To benchmark the rescaling methods with ground truth depth, we collected datasets from three photo-realistic [25] environments: *mine*, *sewer*, and *drone dome* (Fig. 3). The *mine* contains small confined hallways with sharp turns that open into a large cavern; the *sewer* environment contains two large concrete rooms connected with a large hallway; the *drone dome* is an outdoor cluttered environment with a small forest of trees and pillars. These environments represent both confined and open environments where quadrotor aerial robots may be expected to operate. The datasets contain VINS system keyframes with associated 3D features. The evaluation datasets include 846 frames and 3D keypoint pairs from the *mine* environment, 707 from the *sewer* environment, and 1015 from the *drone dome* environment.

B. Evaluation Metrics

We compare the rescaling approaches detailed in Section III-B using two measures: the absolute relative depth error and δ_1 error [26]. Let N be the number of valid pixels, $z_{\text{pred}}(i)$ be the predicted depth and $z_{\text{gt}}(i)$ be the ground truth depth for the ith pixel. The Absolute Relative Error (AbsRel) is defined in Eq. (3) and measures how far the predicted value is from the ground truth depth normalized by the ground truth depth. These values are summed and divided by N to provide an average over all predictions. The δ_1 (defined in Eq. (4)) [10] measures the proportion of predictions within 25% of the ground truth depth.

(a) Mine (b) Sewer (c) Drone Dome

Fig. 3: Examples of images used for ablation study derived from photo-realistic Flightmare [25] simulator. The environments strike a balance between confined spaces (see (a)–(b)) and open spaces where the sky may be seen at a distance (see (c)).

$$\text{AbsRel} = \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{|z_{\text{pred}}(i) - z_{\text{gt}}(i)|}{z_{\text{pred}}(i)} \tag{3}$$

$$\delta_1 = \frac{1}{N} \cdot \left| \left\{ i \ : \ \frac{z_{\text{pred}}(i)}{z_{\text{gt}}(i)} < 1.25, \ \frac{z_{\text{gt}}(i)}{z_{\text{pred}}(i)} < 1.25 \right\} \right| \tag{4}$$

C. Rescaling Algorithm Comparison

We conduct ablation studies and evaluate results using Absolute Relative Error and δ_1 on simulated datasets. For these studies, we employ DepthAnythingV2 [6] as our monocular relative depth estimation (MDE) model. DepthAnythingV2 is included given its prior use by Saviolo et al. [7]. To enable real-time operation on constrained quadrotor hardware, we use the DepthAnythingV2 small model with 24.8 M parameters, which strikes a balance between accuracy and inference speed.

Although the DepthAnything model [15] suggests fine-tuning for a target environment to achieve accurate metric depth, this approach is infeasible for robotic systems operating in unknown surroundings. We use the default parameter settings given by the authors of each model, including the pre-processing steps of each RGB image, while accounting for our RGB camera intrinsics. We conduct these evaluations on a desktop with Ubuntu 20.04 (ROS1 Noetic), an Intel i9-14900K processor, an NVIDIA RTX 4090 GPU, and 32 GB of RAM.

Table I shows the result for the proposed rescaling methods, when used with relative depth estimates from DepthAnythingV2. The rescaling strategies are compared using ground truth depth from the simulated depth camera image queried at the pixel locations corresponding to the sparse 3D points from the VINS projected into the image plane. The columns labeled **GT** leverage the ground truth depth data to perform rescaling and provide a baseline comparison for our approach. The **GT** columns are expected to outperform our approach, because our approach does not use ground truth

Fitting Technique	Mine				Sewer				Drone Dome				Weighted Average			
	AbsRel ↓		Delta1 ↑		AbsRel ↓		Delta1 ↑		AbsRel ↓		Delta1 ↑		AbsRel ↓		Delta1 ↑	
	GT	VINS	GT	VINS	GT	VINS	GT	VINS	GT	VINS	GT	VINS	GT	VINS	GT	VINS
deg 1 poly	**0.078**	**0.124**	**0.949**	**0.855**	0.080	0.151	0.937	**0.883**	2.613	0.894	0.682	0.532	1.081	0.436	0.840	0.735
deg 2 poly	0.177	0.180	**0.947**	**0.851**	**0.073**	0.142	0.947	**0.882**	0.738	0.361	0.734	0.562	0.370	0.241	0.863	0.745
deg 3 poly	0.164	0.287	0.935	0.825	0.157	0.257	**0.949**	0.861	20.259	0.457	0.850	0.570	8.105	0.346	0.905	0.734
deg 4 poly	0.255	0.371	0.922	0.810	0.135	0.265	0.944	0.846	**0.144**	0.428	**0.890**	0.562	**0.178**	0.364	**0.915**	0.722
deg 5 poly	0.351	0.374	0.905	0.790	0.207	1.088	0.938	0.828	0.448	0.500	**0.890**	0.549	0.350	0.620	0.908	0.705
exponential	0.177	0.167	0.735	0.732	0.136	0.143	0.817	0.786	0.246	**0.341**	0.566	0.447	0.193	**0.229**	0.691	0.634
smoothing spline	**0.092**	0.136	0.932	0.840	0.074	**0.124**	0.948	0.868	0.407	0.407	0.784	**0.608**	0.212	0.240	0.878	**0.756**
monotonic smoothing spline	0.094	**0.128**	0.927	0.847	0.076	**0.109**	0.944	0.880	0.382	0.446	0.716	**0.607**	0.203	0.248	0.848	**0.761**
monotonic spline	0.094	0.136	0.930	0.833	**0.069**	0.132	**0.956**	0.850	**0.116**	**0.264**	**0.879**	0.606	**0.096**	**0.185**	**0.917**	0.748

TABLE I: Metric depth benchmarking results using the proposed rescaling methodologies and relative depth from DepthAnythingV2. The best performing approach is colored in green and the second best performing approach is colored in orange. Results for both the proposed approach, which rescales the depth using the 3D feature map from VINS, as well as ground truth (labeled as **GT**) are provided. The monotonic spline yields competitive performance in the confined space environments (i.e., *mine* and *sewer*) and also yields superior performance in the open environment (i.e., *drone dome*). Therefore, we leverage this approach for hardware experimentation.

Model	Mine			Sewer			Drone Dome			Weighted Average		
	AbsRel ↓	Delta1 ↑	FPS ↑	AbsRel ↓	Delta1 ↑	FPS ↑	AbsRel ↓	Delta1 ↑	FPS ↑	AbsRel ↓	Delta1 ↑	FPS ↑
Without TensorRT	**0.094**	**0.930**	5	**0.069**	**0.956**	5	**0.117**	**0.879**	5	**0.096**	**0.917**	5
With TensorRT	0.096	0.929	**20**	0.070	0.949	**20**	0.121	0.869	**19**	0.099	0.911	**20**

TABLE II: Inference speed and accuracy with and without TensorRT using monotonic spline rescaling and DepthAnythingV2 on-board an Orin AGX. The key takeaway is the TensorRT model suffers negligible performance degradation while substantially increasing the frame rate for depth prediction. Therefore, we leverage the TensorRT model for hardware experimentation.

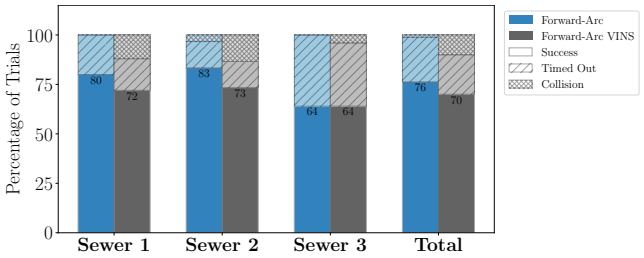


Fig. 4: Performance comparison of autonomous navigation in the simulated sewer environments using our rescaled metric depth estimation approach (shown in grey) and the depth camera data (shown in blue). The proposed approach suffers minor degradation compared to the depth camera image, which is expected as scale is estimated using the fused monocular camera and IMU data.

data. Sparse features in our approach are optimized over a sliding-window of frames, where corner detection introduces errors over time. Since the 3D sparse feature locations rely on corner detection accuracy, these accumulated tracking errors can propagate across the optimization window, leading to degradation in accuracy. Nonetheless, the key takeaway is that using the sparse 3D feature map from VINS yields comparable performance for estimating depth and this is usable for autonomous navigation.

From the weighted average of evaluation metrics across the environments, we observe the monotonic spline rescaling strategy yields the best performance across both confined and open space environments. Because we cannot make any assumptions about the dataset or the shape of the relative-metric disparity distribution and the monotonic spline rescaling strategy performs well across both confined and open space environments, this is the approach we use for the remaining experiments.

D. Autonomous Aerial Navigation in Simulation

The proposed metric depth estimation approach is integrated with the motion primitives-based planner from [27] to enable autonomous aerial navigation. We choose the simulated *sewer* environment for benchmarking navigation performance. We execute the planner at a max velocity of $2\,\mathrm{m/s}$, with a trajectory duration of $2\,\mathrm{s}$, and a robot radius of $1\,\mathrm{m}$ to mitigate potential collisions caused by noise in the estimated depth maps. The Flightmare [25] simulation ran on a host PC equipped with an AMD Ryzen 9950X3D and an RTX 5090 GPU. The proposed depth estimation method, VINS, and the motion planner are executed using a NVIDIA Jetson Orin AGX connected to the host computer via a local area network. Each environment has 5 to 6 start and goal pairs, each pair is run for 5 trials with 120 second time out cutoff.

To obtain higher inference rates, we convert the model into a TensorRT format with FP32 precision and perform benchmarking. Table II presents the trade-off between accuracy and inference speed onboard the NVIDIA Jetson Orin AGX.

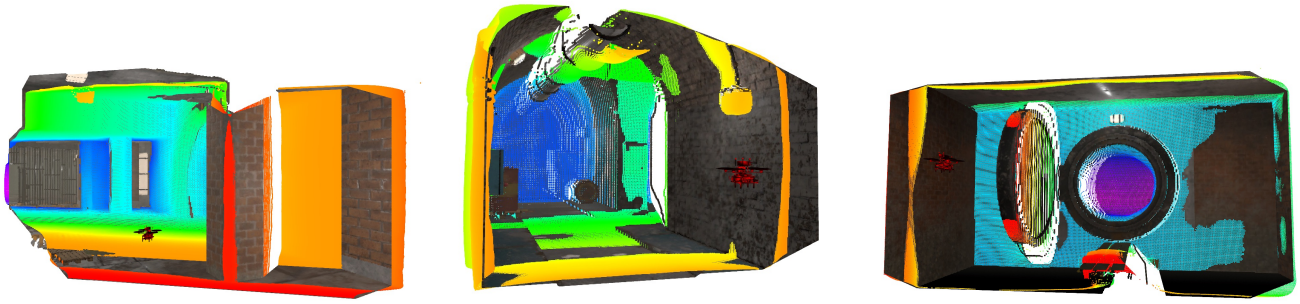
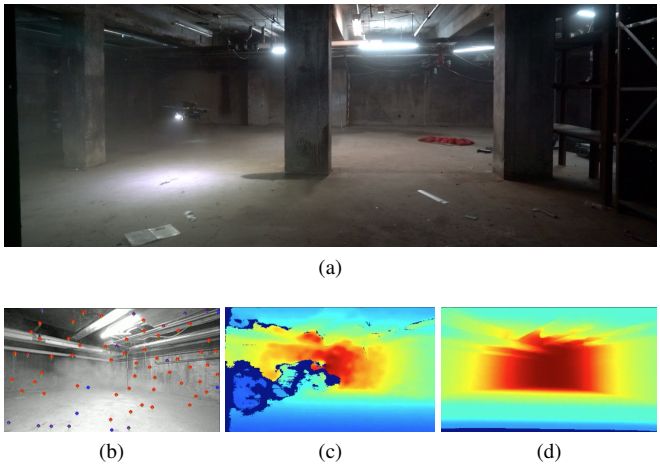


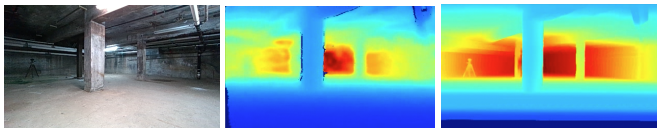
Fig. 5: Representative simulated scenes used to evaluated the proposed approach. The colors ranging from red (closer) to purple (further away) represent the metric depth estimates after rescaling relative to the robots position (shown as a red quadrotor). The predicted depth values closely align with the ground truth, demonstrating the accuracy of the methodology.



(a)

(b) (c) (d)

Fig. 6: Images and data from one of the hardware experiments. (a) provides a third-person view of the robot navigating a dusty industrial tunnel environment. (b) illustrates the corner detections. (c) illustrates how dust affects the active stereo depth image from the RealSense. (d) provides the results from our proposed method.



(a) (b) (c)

Fig. 7: Comparison of environmental details during hardware experiments. (a) RGB image from the RealSense color camera (b) provides a view of the depth image from the RealSense. (c) provides the results from our proposed method.

The results show only a minor accuracy drop, accompanied by a substantial boost in FPS performance.

Quantitative results for navigation performance are shown in Fig. 4. We evaluate the impact of the proposed depth prediction method on (1) the number of collisions incurred and (2) the number of times the robot was able to reach the goal. The *Forward-Arc* case in Fig. 4 uses ground-truth depth data for navigation while the proposed method is labeled *Forward-Arc VINS*. Navigation using estimated depth yields reliable performance with only minimal degradation compared to using ground truth depth. Qualitatively, the predicted point clouds align with the ground truth point clouds (Fig. 5). The number of timeouts and collision rates are comparable between the two methods, demonstrating that the predicted metric depth effectively enables autonomous navigation using a single RGB camera and an IMU instead of the stereo camera used in [7, 27].

E. Real World Deployment

The approach was deployed on a custom-built quadrotor aerial robot equipped with a NVIDIA Orin AGX (32 GB). The forward-facing global shutter RGB camera is from a RealSense D455 sensor. The IMU data comes from the flight controller, which uses a custom version of the PX4 firmware running on an mRo Pixracer. During hardware testing, we ran two VINS instances: one for the depth estimation from the forward-facing camera and another for state estimation using a downward-facing Matrix Vision BlueFox global shutter grayscale camera. All autonomy functions, including obstacle avoidance were run onboard the Orin AGX. Consequently, the load on the CPU was high, which led to slightly reduced depth-estimation inference rate at 15 FPS. This performance remained compatible with system requirements, since the planner operates at 12 Hz, and no systematic failures were observed.

Experiments were conducted in a constrained, dusty catacomb-like environment with concrete pillars (Fig. 6a). The target location was positioned 7 meters from the robot's starting point, behind several pillars, to evaluate obstacle-occluded navigation performance. The robot successfully completed two navigation trials without prior knowledge of the environment. The system demonstrated robustness under dust conditions (Fig. 6). We also observed that our depth estimation method captures fine details. For example, we were able to detect the tripod in the depth estimation output, as shown in Fig. 7.

V. LIMITATIONS

Despite successful navigation, several limitations remain. The proposed depth estimation performance is expected to degrade in open scenes where large portions of the image consist of sky views due to the lack of features or when prominent foreground objects abruptly disappear from the scene. These situations result in degraded metric depth estimates; however, recovery is typically achieved once forward-

facing feature points are re-established. Because there may be chattering in the depth predictions from one frame to the next, planning approaches that select actions close to surfaces (e.g., hug a wall to make a right-hand turn to get to the goal) may be at greater risk of collision. To mitigate for this, a larger collision radius should be used or planning approaches that include gradient information to enable the vehicle to stay far away from obstacles. Furthermore, the method relies heavily on the accuracy of sparse feature points. Depth estimation becomes unreliable when predictions are made outside the depth range represented by the sampled sparse features.

VI. CONCLUSION

In this work, we presented a methodology to predict metric depth from monocular RGB images and an inertial measurement unit (IMU) The approach feeds an RGB image to a monocular depth estimation network without fine-tuning and rescales the inferred depth image using the sparse 3D feature points from a sliding window visual-inertial navigation system. We evaluate several rescaling strategies using diverse simulation data. Our results indicate that DepthAnythingV2 with monotonic splines achieves the most consistent output in both open and confined spaces. Finally, we deploy that pipeline to run navigation in simulation and hardware, validating the depth estimation's robustness through reliable obstacle avoidance without reliance on depth sensors or prior environment knowledge.

Potential directions for future research include optimizing rescaling algorithms to support higher-rate metric depth predictions for faster navigation, improving the accuracy and spread of VINS sparse features using multiple cameras, developing uncertainty-aware weighted rescaling methods and uncertainty-aware planning strategies (e.g., favoring translational motion when depth predictions or sparse features exhibit low confidence).

ACKNOWLEDGEMENTS

The authors would like to thank Jonathan Lee for valuable discussions and insights. This material is based upon work supported in part by the Army Research Laboratory and the Army Research Office under contract/grant number W911NF-25-2-0153.

REFERENCES

[1] W. Tabib, K. Goel, J. Yao, C. Boirum, and N. Michael, "Autonomous cave surveying with an aerial robot," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 1016–1032, 2022.

[2] W. Tabib, K. Goel, J. W. Yao, M. Dabhi, C. Boirum, and N. Michael, "Real-time information-theoretic exploration with gaussian mixture model maps." in *Robotics: Science and Systems*, vol. 2, 2019.

[3] Z. Yang, F. Gao, and S. Shen, "Real-time monocular dense mapping on aerial robots using visual-inertial fusion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 4552–4559.

[4] Y. Lin, F. Gao, T. Qin, W. Gao, T. Liu, W. Wu, Z. Yang, and S. Shen, "Autonomous aerial navigation using monocular visual-inertial fusion," *Journal of Field Robotics*, vol. 35, no. 1, pp. 23–51, 2018.

[5] L. Piccinelli, C. Sakaridis, Y.-H. Yang, M. Segu, S. Li, W. Abbeloos, and L. Van Gool, "Unidepthv2: Universal monocular metric depth estimation made simpler," *arXiv preprint arXiv:2502.20110*, 2025.

[6] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth Anything V2," *Advances in Neural Information Processing Systems*, vol. 37, pp. 21 875–21 911, Dec. 2024.

[7] A. Saviolo, N. Picello, J. Mao, R. Verma, and G. Loianno, "Reactive collision avoidance for safe agile navigation," *arXiv preprint arXiv:2409.11962*, 2024.

[8] J. Mao, R. C. Srinivas, S. Nogar, and G. Loianno, "Time-optimized safe navigation in unstructured environments through learning based depth completion," *arXiv preprint arXiv:2506.14975*, 2025.

[9] D. Wofk, R. Ranftl, M. Müller, and V. Koltun, "Monocular Visual-Inertial Depth Estimation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 6095–6101.

[10] R. Marsal, A. Chapoutot, P. Xu, and D. Filliat, "A simple yet effective test-time adaptation for zero-shot monocular metric depth estimation," *arXiv preprint arXiv:2412.14103*, 2024.

[11] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," *arXiv preprint arXiv:2302.12288*, 2023.

[12] L. Piccinelli, Y.-H. Yang, C. Sakaridis, M. Segu, S. Li, L. Van Gool, and F. Yu, "Unidepth: Universal monocular metric depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 106–10 116.

[13] W. Yin, C. Zhang, H. Chen, Z. Cai, G. Yu, K. Wang, X. Chen, and C. Shen, "Metric3d: Towards zero-shot metric 3d prediction from a single image," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 9043–9053.

[14] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen, "Metric3D v2: A Versatile Monocular Geometric Foundation Model for Zero-Shot Metric Depth and Surface Normal Estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10 579–10 596, Dec. 2024.

[15] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, "Depth anything: Unleashing the power of large-scale unlabeled data," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 10 371–10 381.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[18] J. W. Yao, "Resource-constrained state estimation with multi-modal sensing," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, April 2020.

[19] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 1994, pp. 593–600.

[20] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI'81: 7th international joint conference on Artificial intelligence*, vol. 2, 1981, pp. 674–679.

[21] L. Kneip, M. Chli, and R. Siegwart, "Robust real-time visual odometry with a single camera and an imu," in *Proceedings of the British Machine Vision Conference 2011*. British Machine Vision Association, 2011.

[22] P. Dierckx, "An algorithm for smoothing, differentiation and integration of experimental data using spline functions," *Journal of Computational and Applied Mathematics*, vol. 1, no. 3, pp. 165–184, 1975.

[23] P. H. Eilers, "Unimodal smoothing," *Journal of Chemometrics: A Journal of the Chemometrics Society*, vol. 19, no. 5-7, pp. 317–328, 2005.

[24] C. De Boor, *A practical guide to splines*. springer New York, 1978, vol. 27.

[25] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," in *Conference on Robot Learning*. PMLR, 2021, pp. 1147–1157.

[26] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[27] J. Lee, A. Rathod, K. Goel, J. Stecklein, and W. Tabib, "Rapid quadrotor navigation in diverse environments using an onboard depth camera," in *2024 IEEE International Symposium on Safety Security Rescue Robotics (SSRR)*, 2024, pp. 18–25.