Visual Terrain Relative Navigation: Pose Estimation, Neural Fields, and Verification

By

Dominic Maggio

B.S. Aerospace Engineering
B.S. Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2021

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE IN AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Authored by: Dominic Maggio
Department of Aeronautics and Astronautics
April 12, 2023

Certified by: Luca Carlone
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by: Jonathan P. How
R. C. Maclaurin Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Visual Terrain Relative Navigation: Pose Estimation, Neural Fields, and Verification

by

Dominic Maggio

Submitted to the Department of Aeronautics and Astronautics
on April 12, 2023, in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Aeronautics and Astronautics

Abstract

Visual Terrain Relative Navigation (TRN) is a method for GPS-denied absolute pose estimation using a prior terrain map and onboard camera. TRN is commonly desired for applications such as planetary landings, unmanned aerial vehicles (UAVs), and airdrops, where GPS is either unavailable or cannot be relied upon due to both the possibility of signal loss or outside signal jamming attack. This thesis presents a threefold constribution to visual TRN.

Firstly, due to the high altitude and high speeds of planetary TRN missions, acquiring non-simulation test data oftentimes proves difficult, and thus many datasets used to test TRN systems are from lower altitudes and speeds than what the system would actually be deployed. We present an experimental analysis of visual TRN on data collected from a World View Enterprises high-altitude balloon from an altitude range of 33 km to 4.5 km. We demonstrate less than 290 meters of average position error over a trajectory of more than 150 kilometers. Additionally, we evaluate performance on data we collected by mounting two cameras inside the capsule of Blue Origin's New Shepard rocket on payload flight NS-23, traveling at speeds up to 880 km/h, and demonstrate less than 55 meters of average position error.

Secondly, as accurate terrain map representation is at the core of TRN performance, we explore the question of whether newly emerging Neural Radiance Fields (NeRF) can be efficiently leveraged as a map for visual localization. We propose a NeRF-based localization pipeline coined Loc-NeRF which uses a particle filter backbone to perform monocular camera pose estimation utilizing NeRF.

Thirdly, since TRN is often performed in high-risk missions, we explore the problem of monitoring the correctness of a monocular camera pose estimate at runtime. For this, we again leverage the ability of NeRF to render novel viewpoints and propose a technique coined VERF that incorporates NeRF into a geometrically constrained method to provide assurance on the correctness of a camera pose estimate.

Thesis Supervisor: Luca Carlone
Title: Associate Professor, Aeronautics and Astronautics

Acknowledgments

I want to sincerely thank my advisor Luca. His expertise and enthusiasm have been an essential part of this thesis, and graduate school would not be nearly as fun without him.

I also want to thank all those in SPARK Lab for their collaboration and friendship and for all my friends in undergrad and graduate school.

I want to sincerely thank Draper Laboratory and the Draper Scholar Program for supporting me during my Master's program. The incredible projects and people at Draper have made this program a truly awesome experience. While there are too many people from Draper to thank here, I want to especially thank my Draper advisor Courtney, along with Brett, Ted, Carlos, Rebecca, and Andrew. I also want to sincerely thank the NASA Flight Opportunities Program.

A special thanks to Rou for all the canceled Saturday plans to come to lab and watch me run robot experiments. I also want to thank Father James and all those at St. Benedict's Abbey.

I want to most of all thank my parents Teresa and Gary Maggio. None of this would have been possible without their support, encouragement, and vision. From getting up at 3:45 each morning to open their restaurant and then returning home in the evening to raise cattle in the heat of Louisiana until bed, it goes without saying that my motivation during graduate school has been trying to live up to a small fraction of their dedication. The only time in 37 years the restaurant closed for a non-major holiday was to visit MIT after being admitted. In short, they belong on this thesis much more than I do.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

One of the most fundamental problems in computer vision is visual localization (i.e., estimating the pose of a camera). This is an essential backbone of many robotic and autonomous vehicle applications such as self driving cars, drones, spacecrafts, and robotic manipulators (e.g., robotic arms).

Visual localization can be divided into two paradigms, relative pose estimation and absolute pose estimation. In the relative pose estimation problem, a camera pose is estimated with respect to a prior camera pose. For example, this is the case in visual odometry, where a trajectory of a camera is estimated by tracking the relative movement of automatically computed feature points in a sequence of images. As each newly estimated pose is dependant on prior estimated poses, the trajectory estimate is prone to drift over time.

Absolute localization on the other hand is the task of localizing a camera with respect to a prior map. This often involves determining an optimal association of features (either explicit pixel intensity values or implicit descriptors) between a camera image and a map. If no initial information about the camera's pose is known beforehand, global localization (also known as the kidnapped robot problem) is performed in which the entire space of the map must be searched. If an initial estimate of the camera pose is known, then the search can be reduced to a subsection of the map. In the problem of estimating a camera trajectory for example, in practice an initial guess of the camera pose can oftentimes come from either assuming the camera is close to its prior estimated pose in the trajectory or by gathering a relative estimate of the current camera pose though visual odometry or dead reckoning (i.e., with inertial sensors). Since poses are estimated with respect to a map, absolute localization tends to not be prone to drift. The focus of this thesis will be on the absolute localization problem.

One application of absolute pose estimation is terrain relative navigation (TRN). Terrain relative navigation uses a map of a planetary surface or geographic structure (e.g., canyon) which can be built for example with satellite maps, aerial photography, and ground elevation data. The camera pose is then estimated with respect to the map representation of the terrain. Another example of absolute pose estimation is indoor robotic navigation in which a prior map of the building's interior is available.

Terrain relative navigation is employed in a range of applications such as for drones, airdrops, and spacecraft entry decent and landing (EDL). EDL is a particularly challenging regime for TRN since missions occur at a large range of altitudes (e.g., from a high altitude regime during entry and initial decent to a low altitude regime near the planet's surface as the vehicle approaches landing). Additionally, EDL missions can occur at high speeds as the vehicle decelerates from its orbit.

Because EDL missions occur at high altitude and high speeds regimes, the difficulty of acquiring high-altitude and high-speed data frequently results in TRN methods being tested on simulation data or on data at a lower altitude or lower speed than what a mission would require. This results in both a hesitancy to explore missions far outside the range of testable conditions and extensive manual tuning and extra redundant onboard computation spent to build confidence in lack of realistic test data. To address these shortcomings, in Chapter 2 we present an experimental analysis of visual terrain relative navigation on high altitude data we collect from a World View high altitude balloon ranging from 33 km to 4.5 km and spanning a total distance of over 150 km. We also provide results of performing visual TRN on data we collect onboard Blue Origin's New Shepard rocket which allows us to demonstrate visual TRN in a high speed regime on data at speeds up to 880 km/hr. We additionally provide improvements to the tested TRN method to accommodate high altitude navigation.

Since visual TRN involves matching features between a camera image and a prior built map, the choice and quality of map representation has a significant impact on accuracy. For example two common areas of difficulty with classical map representation (e.g., building a database of pixel patches of known global position from satellite maps) is handling variable lighting and shadows and handling 3D structures such as mountains. Recently, a neural implicit scene representation known as NeRF (Neural Radiance Fields) has gained popularity for visual rendering in computer vision and graphics and has shown the ability to render complex 3D scenes and adapt to variable lighting. This causes us to pose the question of whether NeRF could be used as a map for visual localization. Hence, in Chapter 3 we present a method to use NeRF for localization.

Since TRN is oftentimes used in high-risk missions such as spacecraft EDL, we also pose the question of whether we can provide a measure of assurance that an estimated pose is actually correct. Hence, in Chapter 4 we again leverage NeRF to develop and demonstrate a method that can provide a measure of confidence of whether or not a camera pose estimate is correct. In addition to demonstrating this method on data collected onboard New Shepard, we also demonstrate on data collected from a quadruped robot moving in an indoor environment.

1.1 Related Work

In this section we discuss related work on terrain relative navigation systems, on neural radiance fields with a focus on works bearing relevance for visual localization, and on works characterizing the uncertainty of pose estimation.

1.1.1 Terrain Relative Navigation

The majority of early TRN methods such as the Mars Science Laboratory [37] and NASA's ALHAT Project [7], [3] use radar or lidar. However, due to the high power and weight budget of radar and lidar, cameras have been motivated as an active area of exploration for more recent TRN systems.

The seminal work of Mourikis *et al.* [58] describes a visual-inertial navigation method for Entry, Descent, and Landing (EDL) using an Extended Kalman Filter (EKF) with matched landmarks and tracked feature points in an image. They use inertial navigation results from their entire sounding rocket launch with an apogee of 123 km, and leverage visual methods after the vehicle reaches altitudes below 3800m. Johnson and Montgomery [34] present a survey of TRN methods that use either image or lidar to detect the location of known landmarks.

Singh and Lim [69] demonstrate a visual TRN approach leveraging an EKF for lunar navigation using known crater locations as landmarks. Recently, Downes *et al.* [18] present a deep learning method for lunar crater detection to improve TRN landmark tracking. The Lander Vision System (LVS) [35] used for the Mars 2020 mission uses vision-based landmark matching starting at an altitude of 4200m above the martian surface with the objective of achieving less than 40m error with respect to the landing site. Our analysis in Chapter 2 contains higher altitudes and a larger span on altitudes (4.5 km to 33 km for the balloon dataset).

Dever *et al.*[17] demonstrate visual navigation for guided parachute airdrops using IBAL and a Multi-State Constraint Kalman Filter (MSCKF). Additionally, the work incorporates a lost robot approach to recover from a diverged pose estimate and to initialize the system if the pose is unknown. Steffes *et al.* [72] present a theoretical analysis of three types of visual terrain navigation approaches, namely template matching, SIFT [44] descriptor matching, and crater matching. The work of Lorenz *et al.* [43] demonstrates vision-based terrain relative navigation for a touch and go landing on an asteroid for the OSIRIS-REx mission. Due to extreme computation limits, they used a maximum of five manually selected mapped template features per frame. Mario *et al.* [50] provide additional discussion on ground tests used to prepare the TRN system for the OSIRIS-REx mission. Our balloon dataset in Chapter 2 has much faster rotational motion than what was present during the OSIRIS-REx mission along with camera obstructions.

Steiner *et al.* [73] present a utility-based approach for optimal landmark selection and demonstrates performance on a rocket testbed flight up to 500m. As shadows and variable lighting conditions are a well known challenge for TRN, Smith *et al.* [70] demonstrates the ability to use Blender to enhance a satellite database for different lighting conditions.

1.1.2 Neural Radiance Fields

NeRF was first presented by Mildenhall *et al.* [56] and represents a 3D scene with a neural implicit encoding that can be used to render novel viewpoints of the scene. NeRF is trained using RGB images with known poses. Additional studies have investigated the problem of training NeRF with images whose poses are either unknown or known with low accuracy [83, 41, 54, 93]. These methods take several hours or over a day to train and are intended for building a NeRF as opposed to real-time pose estimation with a trained NeRF. NeRF has also been extended to large-scale [81, 86, 78] and unbounded scenes [94, 6], which has the potential to enable neural representations of large-scale scenes such as the ones typically encountered in robotics applications, from drone navigation to self-driving cars.

Slow training and rendering time has been a longstanding challenge for NeRF, with several recent works proposing computational enhancements. Müller *et al.* [59] use a multi-resolution hash encoding to train a NeRF in seconds and render images on the order of milliseconds. Additionally, some works have utilized depth information to improve rendering time [60, 11], and training time [16, 84, 64]. Related to using depth, Clark [12] uses a volumetric dynamic B+Tree data structure to achieve real-time scene reconstruction and Yu *et al.* [92] use a scene representation based on octrees. Sucar *et al.* [74] proposes iMAP and Zhu *et al.* [98] develop NICE-SLAM which use depth from a stereo camera along with RGB to create a neural implicit map of room-size scenes.

Before the release of Loc-NeRF (discussed in Chapter 3), limited work has been done to leverage NeRF for robotic localization. Yen *et al.* [91] develop iNeRF which inverted the NeRF paradigm by solving for a pose given an image. Adamkiewicz *et al.* [2] develop NeRF-Navigation which uses NeRF for a full autonomy pipeline of localization, planning, and controls. Li *et al.* [40] develop NeRF-Pose which uses PnP with NeRF for object pose estimation by training a pose regression network to predict 2D-3D correspondences. Concurrently and more recently, there have been several NeRF extensions exploring the use of NeRF for localization. Zhu *et al.* [97] propose LATITUDE to perform pose estimation with large-scale scenes. Lin *et al.* [42] use parallelized Monte Carlo Sampling to estimate camera poses. Avraham *et al.* [5] develop Nerfels which use renderable neural codes for camera pose estimation. Moreau *et al.* [57] develop CROSSFIRE which uses PnP for localization with NeRF by training self-supervised feature descriptors and rendering depth directly from a neural renderer.

1.1.3 Certifiable Perception and Runtime Monitoring

Rosen *et al.* [65] develop a certifiable method to estimate a globally optimal solution to the problem of synchronization over the special euclidean group. Yang *et al.* [88, 87] develop certifiable algorithms for outlier robust estimation that produce a certificate of optimality. Garcia-Salguero *et al.* [28, 27] certify the optimality of a relative pose estimate. Zhao *et al.* [96] present an certifiably optimal approach to estimate the generalized essential matrix. Here, we instead focus on monitoring the correctness of the pose estimate, rather than optimality of the estimation backend.

Yang *et al.* [88] and Carlone [10] develop estimation contracts which certify the correctness of a geometric perception problem given conditions are met on the inputs. Talak *et al.* [77] extend certification of correctness for learning based object pose estimation problems. Yang and Pavone [89] provide statistical bounds on object pose estimation given a heatmap predictions of object keypoints. Other works provide confidence metrics to monitor the correctness of perception algorithms without providing a certificate of correctness. Hu and Mordohai [33] provide a survey on confidence metrics for stereo matching. Rahman *et al.* [63] provide a survey on monitoring the correctness of learning-based methods for robotic perception. Antonante *et al.* [4] use a diagnostic graph to formalize detecting and identifying faults in a perception system.

Characterizing the uncertainty of the fundamental matrix by determining its covariance and computing epipolar bands is described in [95, 23, 22, 21, 30, 76, 14]. Brandt [8] uses the fundamental matrix and its uncertainty to estimate the probability that a pair of points will satisfy the true epipolar geometry.

1.2 Thesis Structure and Contributions

This thesis is structured as follows:

- Chapter 2 presents an experimental analysis of visual terrain relative navigation on two challenging real datasets. One onboard a World View Enterprises high-altitude balloon with data beginning at an altitude of 33 km and descending to near ground level (4.5 km) with 1.5 hours of flight time and the other on data we collected onboard Blue Origin's New Shepard rocket on payload flight NS-23, traveling at speeds up to 880 km/hr. We additionally provide improvements to the tested TRN method to handle high-altitude data and to accommodate rapid rotations of the balloon, in some cases over 20 degrees per second. This work was published in the AIAA SciTech Forum in 2023 [47].

- Chapter 3 introduces Loc-NeRF, a real-time six degree-of-freedom vision-based robot localization approach that combines Monte Carlo localization and Neural Radiance Fields (NeRF). We present experiments showing that Loc-NeRF can estimate the pose of a single image without relying on an accurate initial guess, perform global localization on small scales scenes, and achieve real-time tracking with real data collected from a robot moving indoors. This work was published in the proceedings of the International Conference on Robotics and Automation (ICRA) in 2023 [46].

- Chapter 4 presents VERF (Runtime Monitoring of Pose Estimation with Neural Radiance Fields), a collection of two methods for providing runtime assurance on the correctness of a camera pose estimate of a monocular camera without relying on direct depth measurements. Our runtime pose monitoring approach functions independent of how the pose is estimated and runs in less than half a second on a 3090 GPU. We provide results on the publicly available LLFF dataset [55], on real data collected by an A1 quadruped in a room, and on data collected onboard Blue Origin's New Shepard rocket at heights up to 8 km above the ground and at speeds over 800 km/hr.

Chapter 2

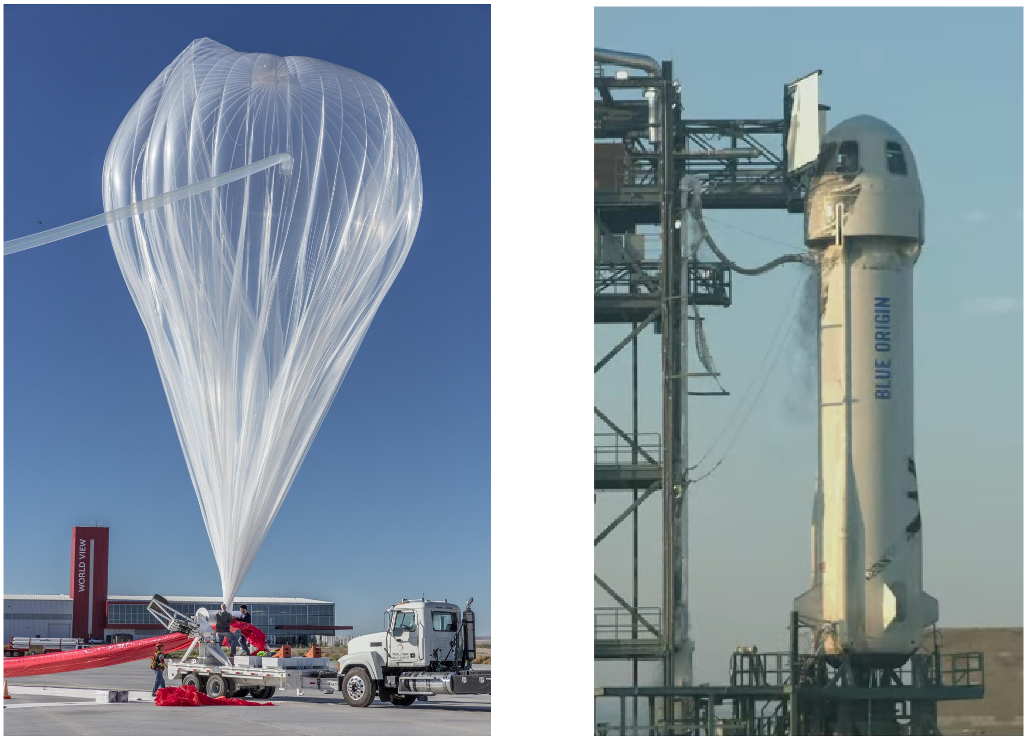
Experimental Analysis of High Altitude Terrain Relative Navigation

This chapter presents an experimental analysis on performing TRN using a camera-based approach aided by a gyroscope for high-altitude navigation by associating mapped landmarks from satellite imagery to camera images. Work in this chapter has been accepted for publication in AIAA SciTech 2023 entitled *Vision-Based Terrain Relative Navigation on High-Altitude Balloon and Sub-Orbital Rocket* [47]. Here, we evaluate performance of both a sideways-tilted and downward-facing camera on data collected from a World View Enterprises high-altitude balloon (Fig. 2-1a) with data beginning at an altitude of 33 km and descending to ground level with almost 1.5 hours of flight time (Fig. 2-2) and on data collected at speeds up to 880 km/h (550 mph) from two sideways-tilted cameras mounted inside the capsule of Blue Origin's New Shepard rocket (Fig. 2-1b), during payload mission NS-23. We also demonstrate the robustness of the TRN system to rapid motions of the balloon which causes fast attitude changes (Fig. 2-3a) and can cause image blur (Fig. 2-3b). Additionally, we demonstrate performance in the presence of dynamic camera obstructions caused by cords dangling below the balloon (Fig. 2-3c), and clouds obstructing sections of the image (Fig. 2-3d).

Sideways-angled cameras are a common choice for TRN applications when mounting a downward camera is either infeasible due to vehicle constraints or would be occluded by exhaust from an engine on vehicles such as a lander or a rocket. Additionally, for planetary landings, a sideways-angled camera allows for a single camera to be used during both the braking phase when the side of the lander faces the surface and during the final descent phase when the bottom of the lander faces the surface (Fig. 2-4). We thus use both a sideways-angled camera and downward-facing camera during our high-altitude balloon flight to separately evaluate the performance of TRN using a camera from each orientation.

We use Draper's Image-Based Absolute Localization (IBAL) [17] software for our analysis. While our datasets have images at a rate of 20Hz, we subsample images by a factor of 10 and hence post-process images at 2Hz in real-time. IBAL could additionally be combined with a nonlinear estimator such as an Extended Kalman Filter (EKF) or a fixed-lag smoother through either a loosely coupled approach using IBAL's pose estimate or a tightly-coupled approach using landmark matches [25]. Since the quality of the feature matches generated by IBAL would affect all these methods, here we limit ourselves to evaluating IBAL as an independent system and also analyze the quality of the feature matches.

In summary, our contributions are as follows. We evaluate performance of both a sideways-tilted and downward-facing camera on data we collected from a World View Enterprises high-altitude balloon with data beginning at an altitude of 33 km and descending to near ground level (4.5 km) with 1.5 hours of flight time. We demonstrate less than 290 meters of average position error over a trajectory of more than 150 kilometers. In addition to showing performance across a range of altitudes, we also demonstrate the robustness of the Terrain Relative Navigation (TRN) method to rapid rotations of the balloon, in some cases exceeding 20° per second, and to camera obstructions caused by both cloud coverage and cords swaying underneath the balloon. Additionally, we evaluate performance on data we collected with two cameras inside the capsule of Blue Origin's New Shepard rocket on payload flight NS-23, traveling at speeds up to 880 km/h, and demonstrate less than 55 meters of average position error. At the same time, we investigate the impact of using a gyroscope in conjunction with IBAL to aid with the challenges of our balloon dataset and show the advantage that even a simple sensor fusion method can provide. Finally, we extend IBAL to incorporate methods to efficiently process high-altitude images when a camera views above the horizon.



(a) Release of high-altitude balloon for data collection. Image: courtesy of World View®Enterprises

(b) Blue Origin's New Shepard rocket carrying Draper experimental payload in the capsule. Image: courtesy of Blue Origin

Figure 2-1: Data collection platforms used for experimental analysis.

The remainder of this chapter is organized as follows: Section 2.1 describes our data collection for both the high-altitude balloon and sub-orbital rocket experiments. Section 2.2 describes the TRN method used in our experiments and Section 2.3 discusses modifications made to address challenges of high altitude navigation. Section 2.4 presents our experiments results for both experiments and lastly Section 2.5 includes an ablation study on the benefits of incorporating a gyroscope for visual TRN in the presence of rapid vehicle rotations.

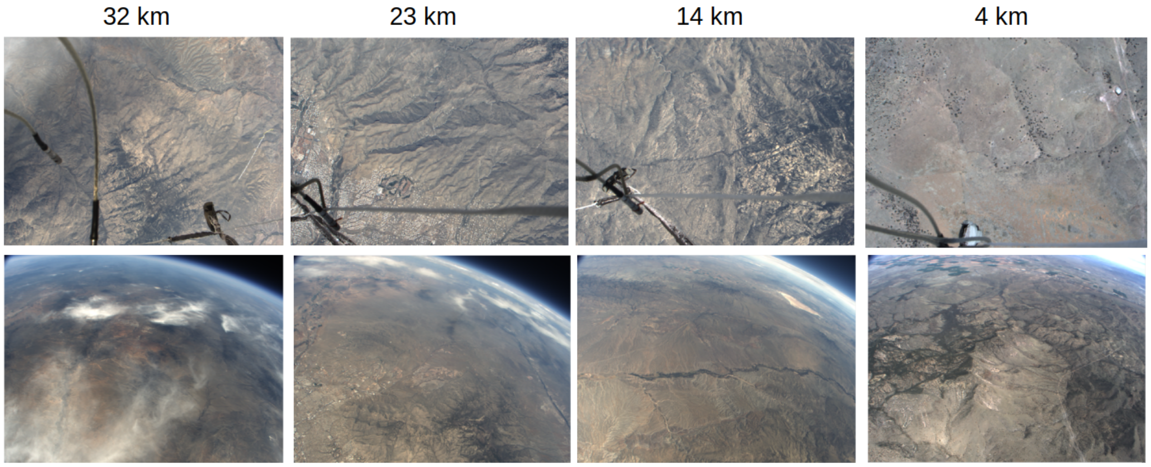
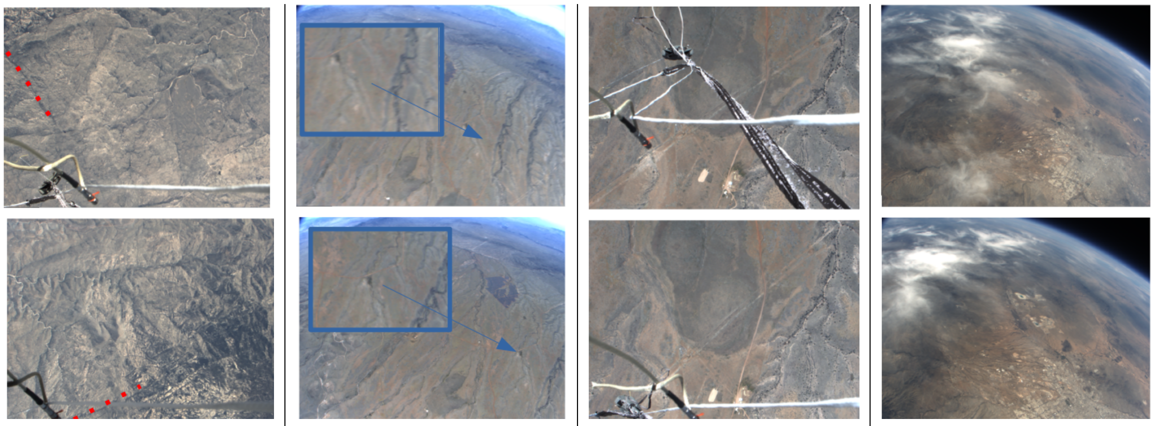


Figure 2-2: Example of images collected at different altitudes (32, 23, 14, and 4 km) from the balloon dataset with the downward-facing camera (top) and sideways-facing camera (bottom).



(a) Rapid rotations, here over 90° in 4 seconds. Red dots show ground reference points between top image and bottom image.

(b) Image blur (top) due to rapid motion compared to crisp image (bottom).

(c) Moving cords in the image. Top and bottom images showing example range of cord motion.

(d) images partially occluded by clouds

Figure 2-3: Different types of TRN challenges in the balloon dataset.

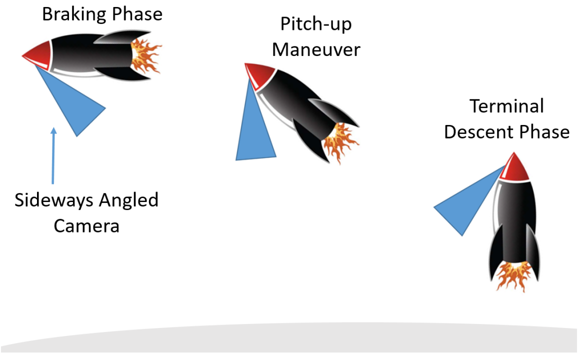


Figure 2-4: Demonstration of a sideways-angled camera viewing the terrain and being used during the braking phase, pitch-up maneuver, and terminal descent phase.

2.1 Data Collection

The collection of both datasets used in chapter was supported by the NASA Flight Opportunities Program. The high-altitude balloon dataset was designed to test TRN on a wide range of high-altitude data and occurred in April of 2019. The New Shepard dataset was intended to test TRN on a high speed vehicle with a flight profile similar to that of a precision landing and occurred in September of 2022. This section will discuss data collection for both experiments.

2.1.1 Balloon Flight

We captured downward and sideways camera images along with data from a GPS and an inertial measurement unit (IMU) on board a World View Enterprises high-altitude balloon shown in Fig. 2-1a, with data recorded up to an altitude of 33 km. We used FLIR Blackfly S Color 3.2 MP cameras for both downward and sideways facing views using 12 mm EFL lens and 4.5 mm EFL lens, respectively. The field of view (FOV) for the downward and sideways camera with their respective lens is 32° and 76°. Both cameras, along with the IMU (Analog Devices ADIS16448) and data logging computer are self contained inside the Draper Multi-Environment Navigator (DMEN) package, shown in Fig. 2-5. Both cameras generated images at 20 Hz with a resolution of 1024 × 768. The IMU logged data at 820 Hz.

Some TRN applications —such as planetary landing— might prefer using a sideways-angled camera, while other applications —such as high-altitude drone flights— may prefer a downward-facing camera. Therefore, we collect data from both a downward and sideways angled camera to allow for IBAL to be evaluated at both these camera angles. Additionally, some planetary landings may also desire a downward-facing camera since it allows the boresight of the camera to be normal to the surface during the terminal descent phase, such as was done for OSIRIS-REx [43].

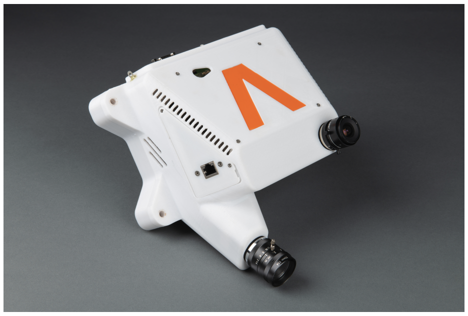


Figure 2-5: Draper Multi-Environment Navigator (DMEN) package: data collection package containing sideways and downward facing cameras, IMU, and logging computer.

2.1.2 New Shepard Flight

We captured images from two sideways-angled cameras with 12.5 mm lens on opposite sides inside the New Shepard capsule which look out the capsule windows. Having two cameras was intended to allow us to study the effects of different cloud cover, terrain, and angle to the sun. We will refer to these cameras as camera 1 and camera 2. We additionally log IMU data from a Analog Devices ADIS16448, and telemetry from the capsule which served as ground truth for our experiment. Data was logged with a NUC mounted inside a payload locker in the capsule. Both cameras generated images at 20 Hz with a resolution of 1024 × 768 and FOV of 31°. The IMU logged data at 820 Hz. The rocket reached speeds up to 880 km/h and an altitude of 8.5 km before a mishap occurred during the NS-23 flight which triggered the capsule escape system.

Figure 2-6 shows our payload locker containing the NUC, IMU, and a power converter which is mounted inside the New Shepard capsule. An ethernet cable and two USB cables transfer telemetry data from the capsule and data from the cameras to the NUC, respectively.

Figure 2-7a shows camera 2 mounted inside the capsule with a sideways-angle and Fig. 2-7b shows the location of both cameras inside the capsule on opposite sides while New Shepard is on the launch pad. Both cameras are mounted at the same tilt angle such that they can view the terrain while not having their FOVs obstructed by components on the rocket. Additionally, a mounting angle was selected to reduce the effects of distortion caused by the windows, and to ensure the cameras did not come in direct contact with the windows.

Distortion effects from the windows were addressed by calibrating the intrinsic parameters of the camera while the camera was mounted in the capsule (i.e., a calibration board was positioned outside the capsule window). We used the Brown-Conrady model [9] which helps account for decentralized distortion caused by the window in addition to distortion from the camera lens. Further evaluation on distortion effects caused by the window of the capsule is left as a topic for future work.

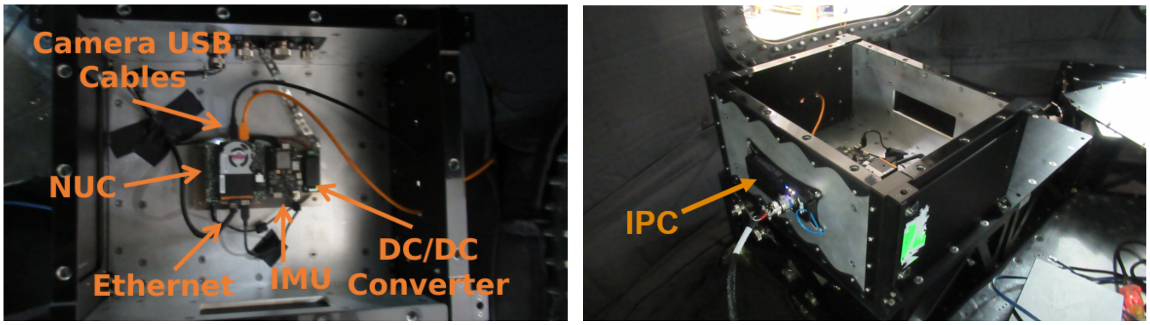
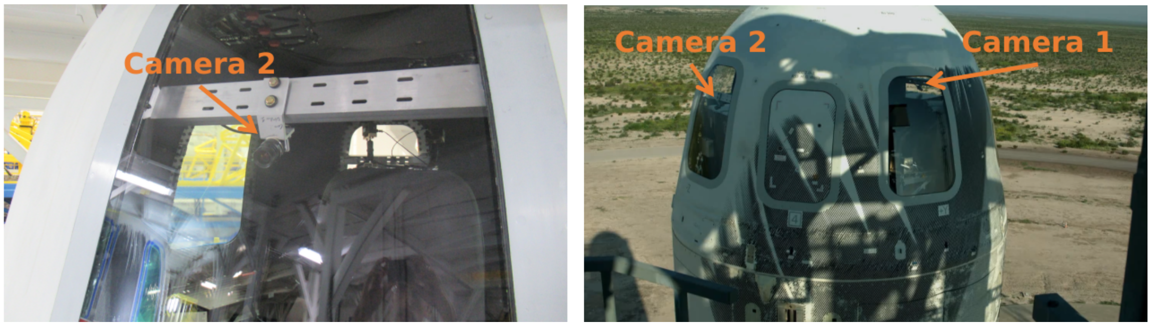


Figure 2-6: Payload locker inside the New Shepard capsule containing a NUC, IMU, DC/DC Converter, and IPC (Integrated Payload Controller). Images courtesy of Blue Origin.



(a) (b)

Figure 2-7: Cameras 1 and 2 mounted inside the New Shepard capsule looking out the capsule windows. Images courtesy of Blue Origin.

As collecting data for the rocket experiment in 2022 is a contribution of this thesis whereas the balloon dataset was collected by others at Draper beforehand in 2019, we will further provide an additional level of detail into the rocket data collection. The data collection for our rocket experiment must be fully automated during the flight. Our data collection payload is loaded into the rocket before flight and remains turned off without power until minutes leading up to launch. Our NUC shown in Fig. 2-6 is responsible for starting sensor drivers for our two cameras and IMU and logging the in flight data.

The NUC is tethered to the rocket through an Integrated Payload Controller (IPC) labeled in Fig. 2-6. Approximately three minutes prior to launch the IPC sends power to the NUC which automatically turns on and starts the sensor drivers for our cameras and IMU and begins logging data. We additionally log telemetry data from the rocket during the flight. Twenty seconds after landing is detected from the received telemetry data, the NUC turns off the camera and IMU and seals the logged data. The NUC stays powered on for 80 seconds after touchdown to ensure a clean shutdown of all sensors and loggers and then shuts itself off.

2.2 Terrain Relative Navigation Method

We use Draper's IBAL software [17] to perform TRN for our datasets. A database of image templates is created in advance from satellite imagery and stored using known pixel correspondence with the world frame. Using satellite images and elevation maps from USGS [1], we automatically select patches of interest from the satellite images and create a collection of templates that serve as 3D landmarks. For each camera image processed by IBAL, IBAL uses an initial guess of the camera pose to predict which templates from the database are in the field of view (FOV) of the camera using a projection from the image plane to an ellipsoidal model of the planet. The templates are then matched to the camera image using cross correlation. The resulting match locations are passed to a 3-point RANSAC [24] (using a Perspective-Three-Point method as a minimal solver) to reject outliers. The output is a list of the inlier matches, their pixel location in the image, and their known location in the world frame that can be passed to a nonlinear estimator or fixed-lag smoother for tightly-coupled pose estimation. A secondary output of RANSAC is an absolute pose estimate found by using the Perspective-n-Point (PnP) algorithm on the set of inliers.

Instead of a tightly-coupled approach, we will use a simpler method to evaluate performance on the balloon and New Shepard datasets. For the balloon dataset, we take the PnP absolute pose estimate directly from IBAL, forward propagate it with the gyroscope measurements, and use it at the next time step as a pose guess for IBAL. We do not use accelerometer data since in the image frame most scene changes for the balloon dataset over a short time span will be due to rotations. This is due to the high altitude and hence large distance between the camera and the Earth's surface. Using the gyroscope to propagate the rotation also allows for reduced computation since we are able to down-sample our camera data by a factor of 10 (2Hz image input to IBAL). Additionally, the gyro allows for robust handling of rapid motions of the balloon and images that have large obstruction from cords which makes generating landmark matches unreliable. An ablation study on incorporating the gyroscope with IBAL is provided in Section 2.5. Since the New Shepard capsule does not experience rapid rotations like the balloon, we did not find it necessary to use the gryoscope to forward propagate the pose estimate for the New Shepard dataset.

We propagate the rotation estimate of the vehicle, $q_{ecef}^{cam_T}$ (i.e., the orientation of the earth-centered, earth-fixed frame w.r.t. the camera frame at time T, represented as a unit quaternion), to the time of the next processed image ($T+1$) with the gyro using second order strapdown quaternion expansion [53]. Using 3-axis gyro measurements $\boldsymbol{\theta}$ and their magnitude $\omega = \|\boldsymbol{\theta}\|$, we compute the orientation $q^{IMU_t}_{IMU_{t+1}}$ between gyro measurements using the following equation

$$q^{IMU_t}_{IMU_{t+1}} = [1 - \frac{\omega^2 \Delta t^2_{IMU}}{8}, \frac{\boldsymbol{\theta}^T \Delta t_{IMU}}{2}] \tag{2.1}$$

where $t+1$ and t represent the time of consecutive IMU measurements occurring Δt_{IMU} seconds apart.

Using the rotations $q^{IMU_t}_{IMU_{t+1}}$ between consecutive IMU timestamps, we can compute the relative rotation $q^{cam_T}_{cam_{T+1}}$ between the camera pose between consecutive images collected at time T and $T+1$:

$$q^{cam_T}_{cam_{T+1}} = \prod_{t=T}^{T+1} q^{cam}_{IMU} \otimes q^{IMU_t}_{IMU_{t+1}} \otimes (q^{cam}_{IMU})^{-1} \tag{2.2}$$

where $\otimes$ is the quaternion product and q^{cam}_{IMU} is the static transform from the IMU frame to the camera frame:

Finally, we can compute the rotation estimate $q^{cam_{T+1}}_{ecef}$ of the vehicle at time $T+1$:

$$q^{cam_{T+1}}_{ecef} = (q^{cam_T}_{cam_{T+1}})^{-1} \otimes q^{cam_T}_{ecef} \tag{2.3}$$

A high level overview of our TRN pipeline for our experiments is shown in Fig. 2-8.

We use a simple yet effective logic for handling short segments in our datasets when PnP is unable to produce a reliable pose, which can be caused by image obstructions or blurry images caused by rapid vehicle motion. If PnP RANSAC selects a small set of inliers (i.e., less than 8) or if the pose is clearly infeasible (i.e., an altitude change between processed images greater than 450 m for the balloon dataset), we reject the pose estimate, keep forward propagating the pose using gyroscope data, and run IBAL with the next available image, ignoring the down-sampling rate.

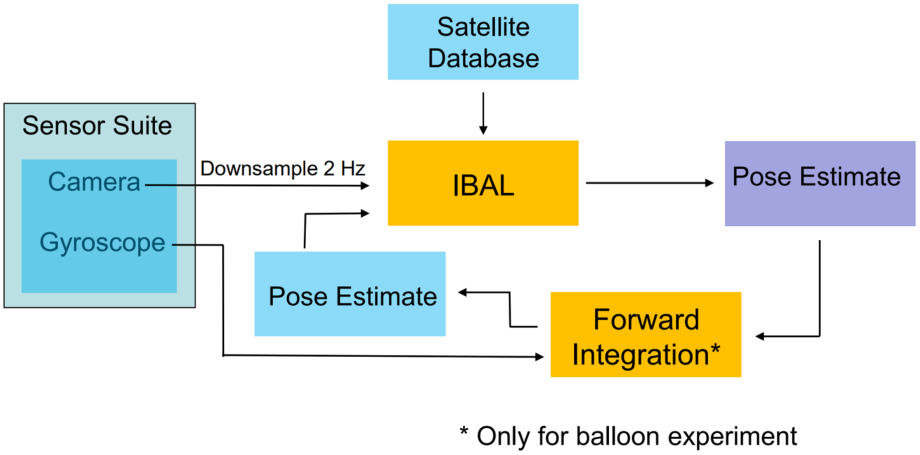


Figure 2-8: High level overview of our TRN pipeline.

2.3 Addressing Challenges of High-Altitude Images

We apply simple and effective methods to address two common challenges we encountered with high-altitude images, namely determining the projection to the ellipsoid when the camera views the horizon, and reducing the number of potential landmarks from the database that have a lower probability of generating good matches when there is a large number of landmarks in view of the camera.

When the horizon is in view of the camera, as is true for the higher altitude images from the sideways camera for the balloon dataset (Fig. 2-2), our baseline method of determining the camera's viewing bounds of the planet's surface is insufficient. Our baseline method is to use an initial estimate of the camera's pose to project each corner of the image to the ellipsoid model. From this, we can create a bounding box on the ellipsoid defined by a minimum and maximum latitude and longitude. However, this is ill-defined if at least one corner of the image falls above the horizon. To resolve this case, if the projection of a corner point does not intersect the ellipsoid we incrementally move the point (in the image space) towards the opposite corner of the image until it intersects the ellipsoid (Fig. 2-9). This process is summarized in Algorithm 1. This process is shown to be effective for our dataset, despite the fact that the approach could fail (see line 15 in Algorithm 1) when the projection of the ellipsoid does not intersect the main diagonals of the image (e.g., when the camera is too far away from Earth or has a large tilt angle). Such a potential case is shown in Fig. 2-10 in which an ellipsoidal body is present in the image but is not intersected by the image diagonal. However, we remark that this case is not present in our dataset and IBAL could be modified to handle this case if a particular mission required it.

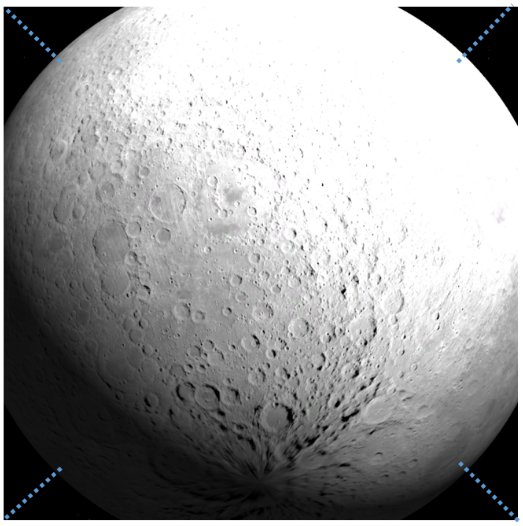


Figure 2-9: Example of our horizon detection method finding the horizon of an ellipsoidal body. Each corner point of the image is incremented towards the opposite corner until the ellipsoid body is intersected.

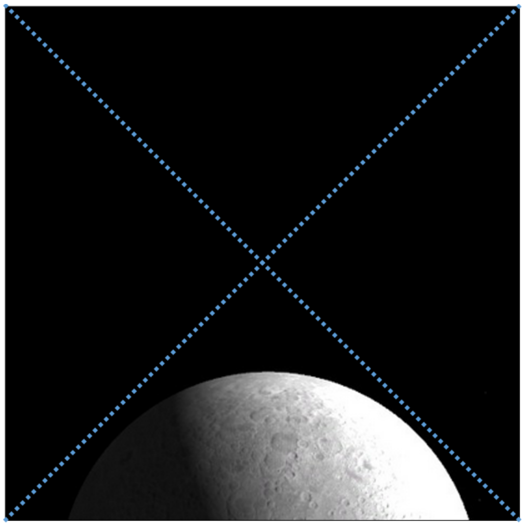


Figure 2-10: An example case where our horizon detection method will not find the horizon since the ellipsoidal body is not intersected by the image diagonals. This case does not appear in our dataset and our method could be trivially extended depending on mission requirements.

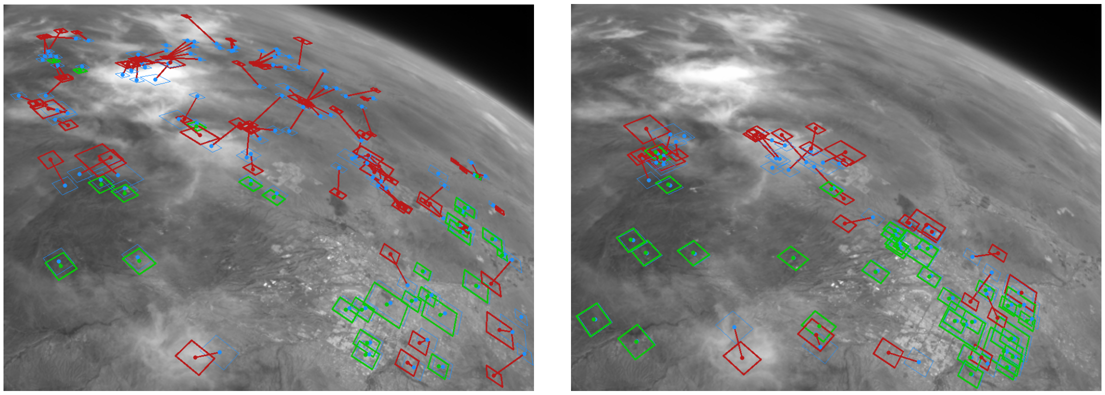
Algorithm 1 Horizon Detection

```
Inputs:
    P    ▷ estimate of camera projection matrix (containing intrinsic and extrinsic parameters)
    $\pi_{WGS84}$    ▷ projection of a pixel coordinate to a 3D point on the surface of the WGS84 model
    $\delta x$    ▷ amount to shift a point by in pixel space (default 10 pixels)
Output: $image\_corners$    ▷ set of four pixel coordinates bounding image
for $x_{corner}, \in image\_corners$ do
    while True do
        X ← $\pi_{WGS84}(P, x_{corner})$
        if X intersects ellipsoid then
            break    ▷ found valid image boundary
        else
            increment $x_{corner}$ towards opposite corner by $\delta x$
        end if
        if $x_{corner}$ outside image then
            return error    ▷ failed to find horizon boundary
        end if
    end while
end for
return $image\_corners$
```

Since we select a maximum number of landmarks based on the landmarks in our satellite database that are in view of the camera, we need additional logic to avoid the possibility of selecting landmarks that mostly fall near the horizon, since these are unlikely to lead to good matches. The ratio of meters per pixels grows rapidly as we approach the horizon, and image matching becomes difficult or impossible near the horizon line due to glare or heavy warping needed to match a shallow surface angle. Additionally, there is significant atmospheric distortion. Removing those landmarks helps avoid unnecessary computation and reduces the number of outliers we pass to RANSAC. Towards this goal, we set a maximum acceptable angle between the boresight of the camera and the surface normal of a landmark and reject landmarks that fail to meet this threshold. To increase the number of potential landmarks that meet our angle requirement, we filter out sections of the camera's FOV projection to the ellipsoid that are unlikely to produce landmarks that meet the angle threshold. This filtering method follows our prior method for intersecting the ellipsoid and uses similar logic. Starting at the first point near each image corner that views the ellipsoid, we find the surface normal by projecting from the image plane to the ellipsoid and move towards the opposite corner of the image until the angle requirement is met. This process is summarized in Algorithm 2 and a corresponding ablation is shown in Fig. 2-11. Notice that without Algorithm 2, more landmarks are selected near the horizon (Fig. 2-11a) where template matching is more difficult resulting in more outliers. Using Algorithm 2 allows IBAL to target regions of the image with more distinguishable features for matching which results in a higher concentration of inliers (Fig. 2-11b).

Algorithm 2 Landmark Angle Filter

```
Inputs:
        P    ▷ estimate of camera projection matrix (containing intrinsic and extrinsic parameters)
        $\pi_{WGS84}$    ▷ projection of a pixel coordinate to a 3D point on the surface of the WGS84 model
        $\alpha_{max}$    ▷ max acceptable angle between camera boresight and normal of a landmark
        $\delta x$    ▷ amount to shift a point by in pixel space (default 10 pixels)
Output: $image\_corners$    ▷ set of four pixel coordinates bounding image
surface_normal() ← function that finds normal vector at a point on the WGS84 model
angle_between() ← function that finds the angle between a camera boresight and a vector
for $x_{corner}, \in image\_corners$ do
    while True do
        X ← $\pi_{WGS84}(P, x_{corner})$
        $x_n \leftarrow surface\_normal(X)$
        $\alpha \leftarrow angle\_between(P, x_n)$
        if $\alpha \leq \alpha_{max}$ then
            break    ▷ found valid image bounary
        else
            increment $x_{corner}$ towards opposite corner by $\delta x$
        end if
        if $x_{corner}$ outside image then
            return error    ▷ failed to meet landmark angle requirement
        end if
    end while
end for
return $image\_corners$
```



(a) Higher concentration of outliers near the horizon without using landmark angle filter. Ratio of inliers to outliers: 0.3

(b) Higher concentration of inliers using landmark angle filter. Ratio of inliers to outliers: 1.3

Figure 2-11: Ablation study for Algorithm 2, which filters regions of the image for landmark matching based on the angle between the surface and the camera boresight. This leads to a higher ratio of inliers to outliers, reducing computation and improving accuracy. Inliers matches are shown in green and outlier are shown in red. Blue shows initial estimate of landmark location based on initial pose estimate before utilizing cross correlation. Images are from sideways camera from balloon dataset.

2.4 Experiment Results

2.4.1 Balloon Flight

We present results from running IBAL with both a sideways-tilted and downward-facing camera aided by gyroscope measurements on altitudes ranging from 33km to 4.5km. Note that we use the term altitude to mean height above the WGS84 ellipsoid. During this time, the system is descending under a parachute. We split our data into 7 segments, each about 15 minutes long, and evaluate our estimated TRN position by comparing with GPS. We manually reseed IBAL at the start of each segment. Results are defined with respect to an East North Up (ENU) frame centered at the landing site of the balloon. Figure 2-12 shows the ground truth trajectory from GPS compared to the trajectory estimates from IBAL with a downward and sideways facing camera. The corresponding plot of absolute position error is shown in Fig. 2-13 for each of the East, North, and Up axes. IBAL is able to achieve an average position error along the up axis of 78 m and 66 m for the entire trajectory with the downward-facing and sideways-tilted camera, respectively, while the balloon travels almost 30 km in elevation. IBAL achieves 207 m and 124 m of average position error for the east and north axis across the entire trajectory of the downward-facing camera, and likewise an average error of 177 m and 164 m along the east and north axis for the sideways camera while the balloon transverses well over 100 km laterally. Figure 2-14 shows total absolute error (defined as the Euclidean distance between the estimate and the GPS position) with respect to flight time and with respect to height above ground level. Average absolute position error for the entire trajectory is 287 m and 284 m for the downward and sideways-tilted camera, respectively. Spikes in position estimates could be diminished using filtering methods such as coupling with an accelerometer or with visual odometry as mentioned in Section 2.2. We run IBAL in real-time on a laptop with an Intel Xeon 10885M CPU. While IBAL is designed to run in real-time on flight hardware, we do not make showcasing run-time performance a focus of this chapter.

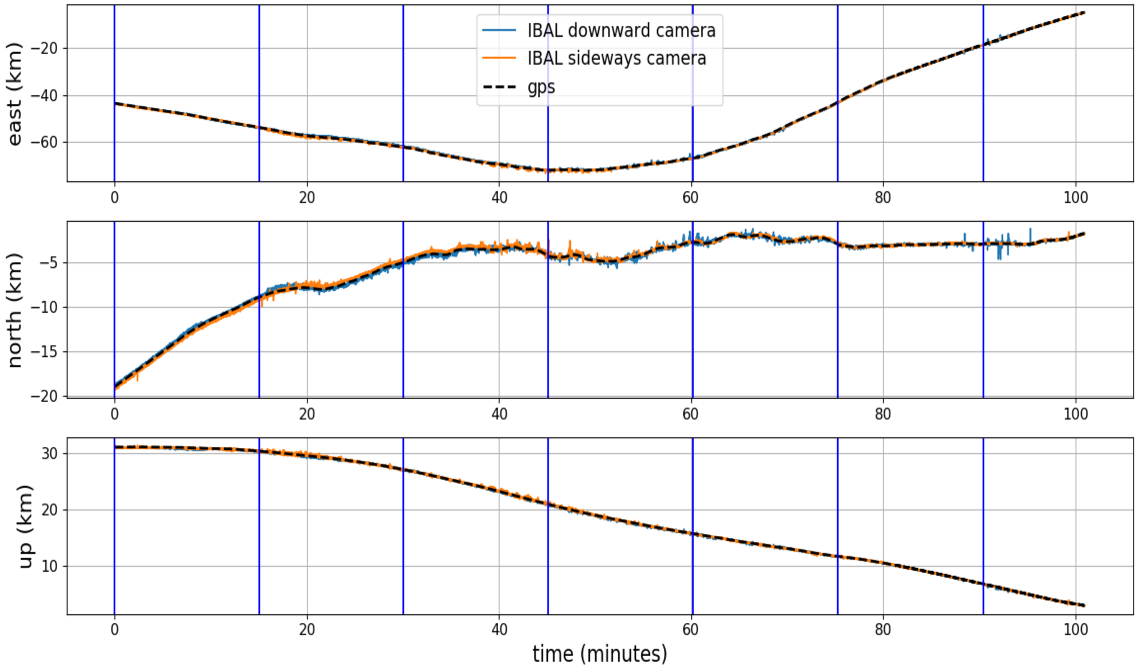


Figure 2-12: IBAL+gyro trajectory estimate vs. GPS for altitude range of 33 km to 4.5 km on balloon dataset. Vertical lines show start of each new data segment.

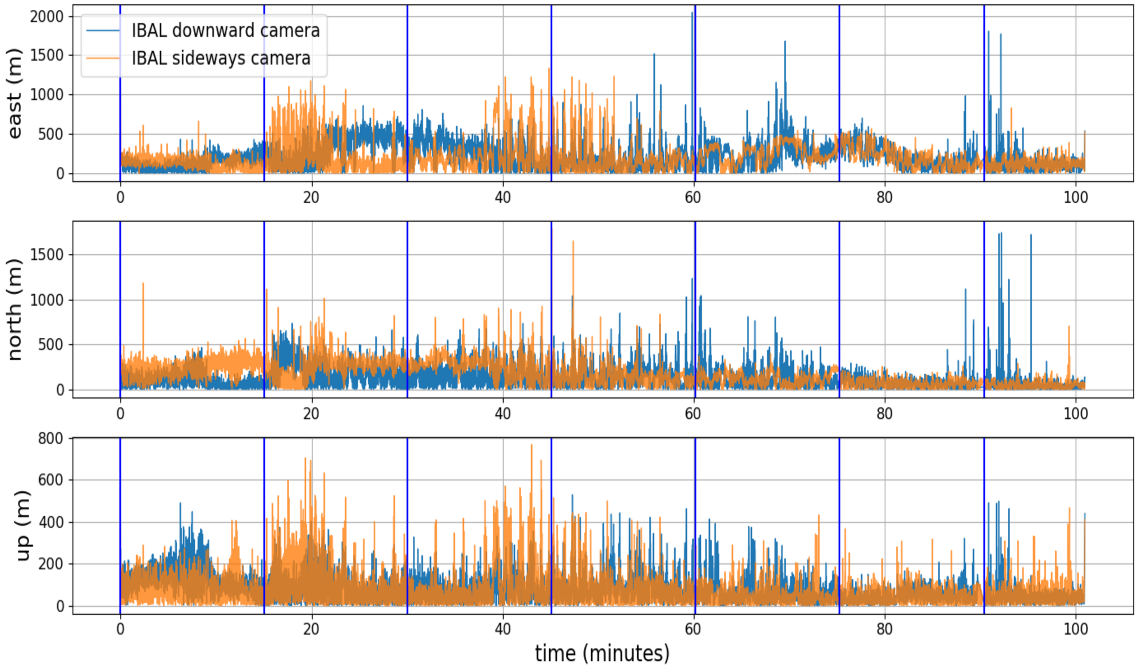


Figure 2-13: IBAL+gyro absolute position error for altitude range of 33 km to 4.5 km on balloon dataset. Vertical lines show start of each new data segment.

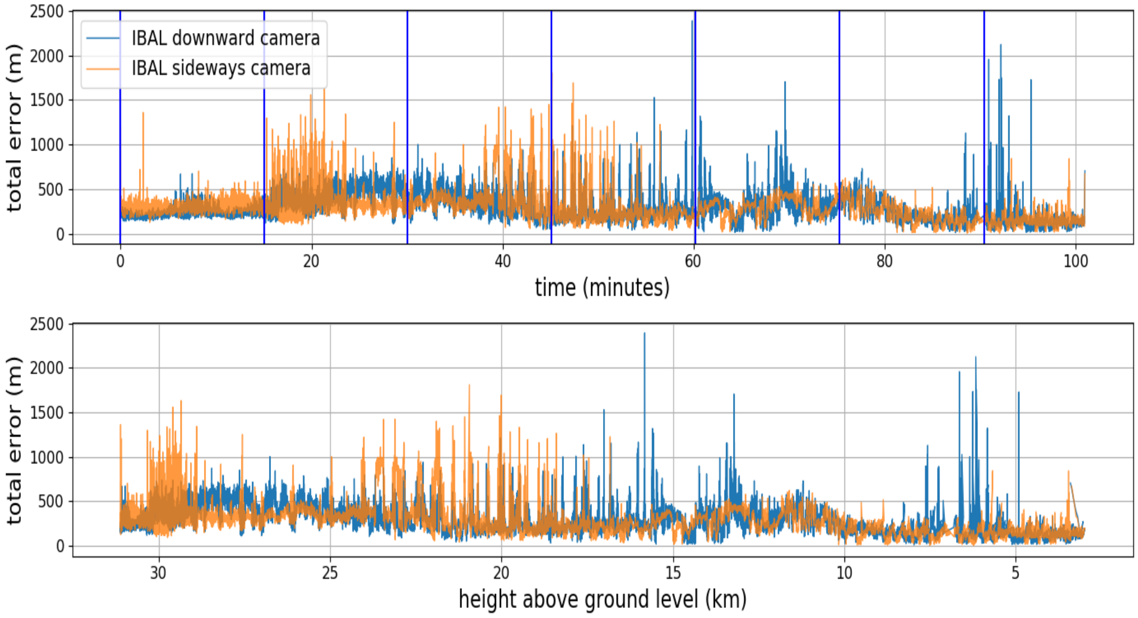
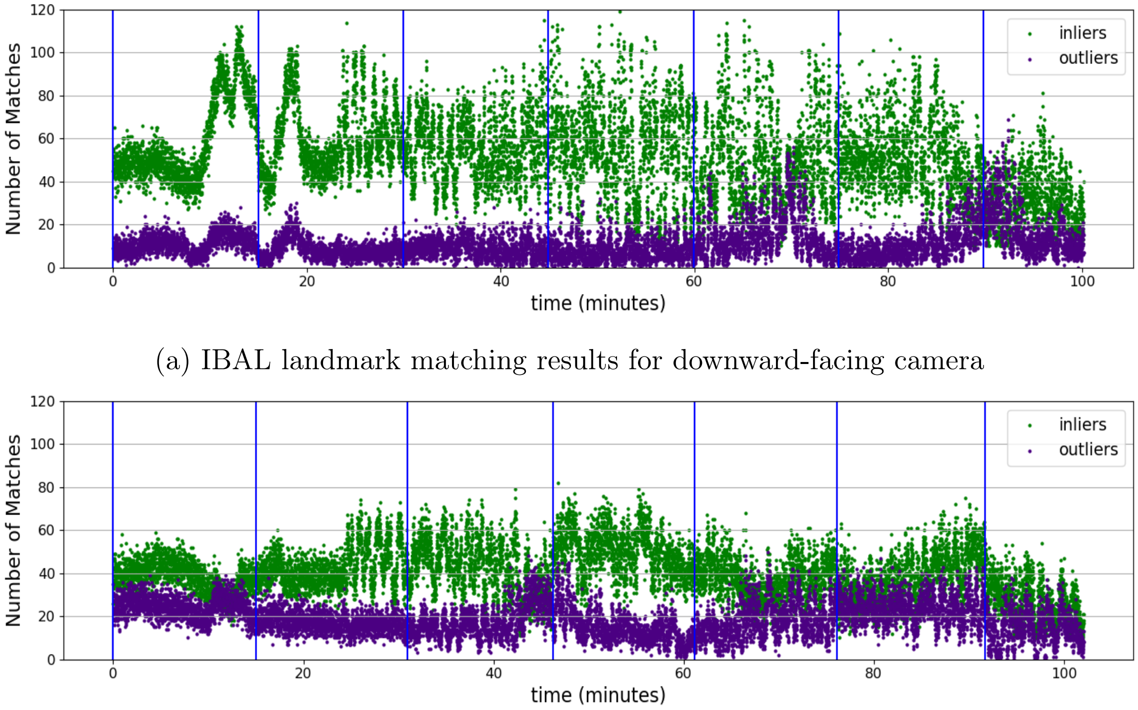


Figure 2-14: IBAL+gyro total trajectory error vs. time and vs. height above ground level on balloon dataset. Error tends to show slight decrease in magnitude at lower altitudes. Vertical lines show start of each new data segment.

We also provide an analysis of the match correlation for both cameras for the entire balloon dataset. Figure 2-15a and Fig. 2-15b show number of inliers and outliers for the downward and sideways facing cameras. After estimating the location of a landmark in the image with cross correlation and peak finding, inliers and outliers are labeled using PnP and RANSAC. There are generally more inliers than outliers which shows the effectiveness of the correlation approach, and that IBAL is able to perform well in the presence of outliers. We observe a greater number of inliers with the downward-facing camera than with the sideways-tilted camera.

Additionally, Fig. 2-16 shows a histogram of the amount of pixel error for the inliers and outliers determined by PnP and RANSAC for both the downward and sideways-tilted cameras. Inlier pixel error is distributed such that most inliers have between 0 and 1 pixel of error as determined by PnP and RANSAC which shows the effectiveness of IBAL's correlation approach. That there is an increase in the ratio of outliers to inliers at lower altitudes. This is due in part to shadows, lack of distinct texture on the ground, and regions with a sparse amount of landmarks in our database. Depending on mission requirements, this issue can be greatly reduced during the landmark database creation process such as by optimizing for landmark template size, ensuring sufficent landmark coverage at low altitudes for all phases of a flight, and by baking shadows into the database as was demonstrated in [70]. However, for the purposes of the balloon experiment in this chapter, we determined our database to be sufficient.

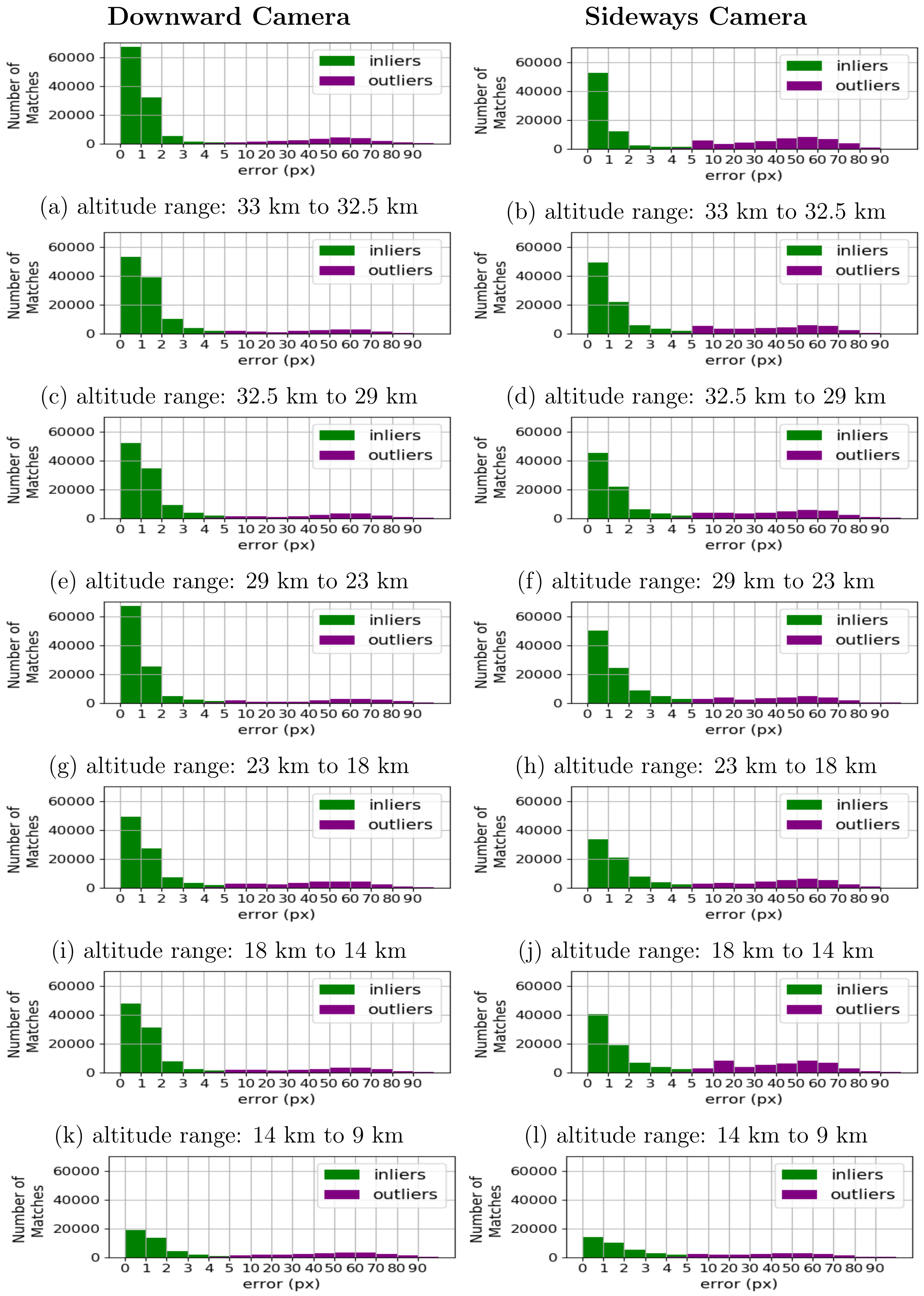
Lastly, we provide visual examples of IBAL matches on a selected subset of frames from the downward and sideways facing cameras. Figure 2-17a shows landmark matches for the downward camera at 13.5 km with inliers shown in green and outliers shown in red. Blue dots show the inital estimate of the landmark locations in the image by using the pose estimated by IBAL's prior pose and the gyro before matching with cross correlation. Figure 2-17b shows matches for the downward camera at 23 km. Cords from the high-altitude balloon are partially in view, but incorrect matches caused by the cords are correctly rejected as outliers. Figure 2-17c and Fig. 2-17d show results for the sideways-tilted camera at 13.5 km and 23 km.



(a) IBAL landmark matching results for downward-facing camera

(b) IBAL landmark matching results for sideways-tilted camera

Figure 2-15: IBAL+gyro number of inliers and outliers for sideways-tilted and downward-facing cameras on balloon dataset for altitude range of 33 km to 4.5 km as determined by PnP and RANSAC. Vertical lines show start of each new data segment. The downward camera tends to have more matches than the sideways-tilted camera.



(a) altitude range: 33 km to 32.5 km

(b) altitude range: 33 km to 32.5 km

(c) altitude range: 32.5 km to 29 km

(d) altitude range: 32.5 km to 29 km

(e) altitude range: 29 km to 23 km

(f) altitude range: 29 km to 23 km

(g) altitude range: 23 km to 18 km

(h) altitude range: 23 km to 18 km

(i) altitude range: 18 km to 14 km

(j) altitude range: 18 km to 14 km

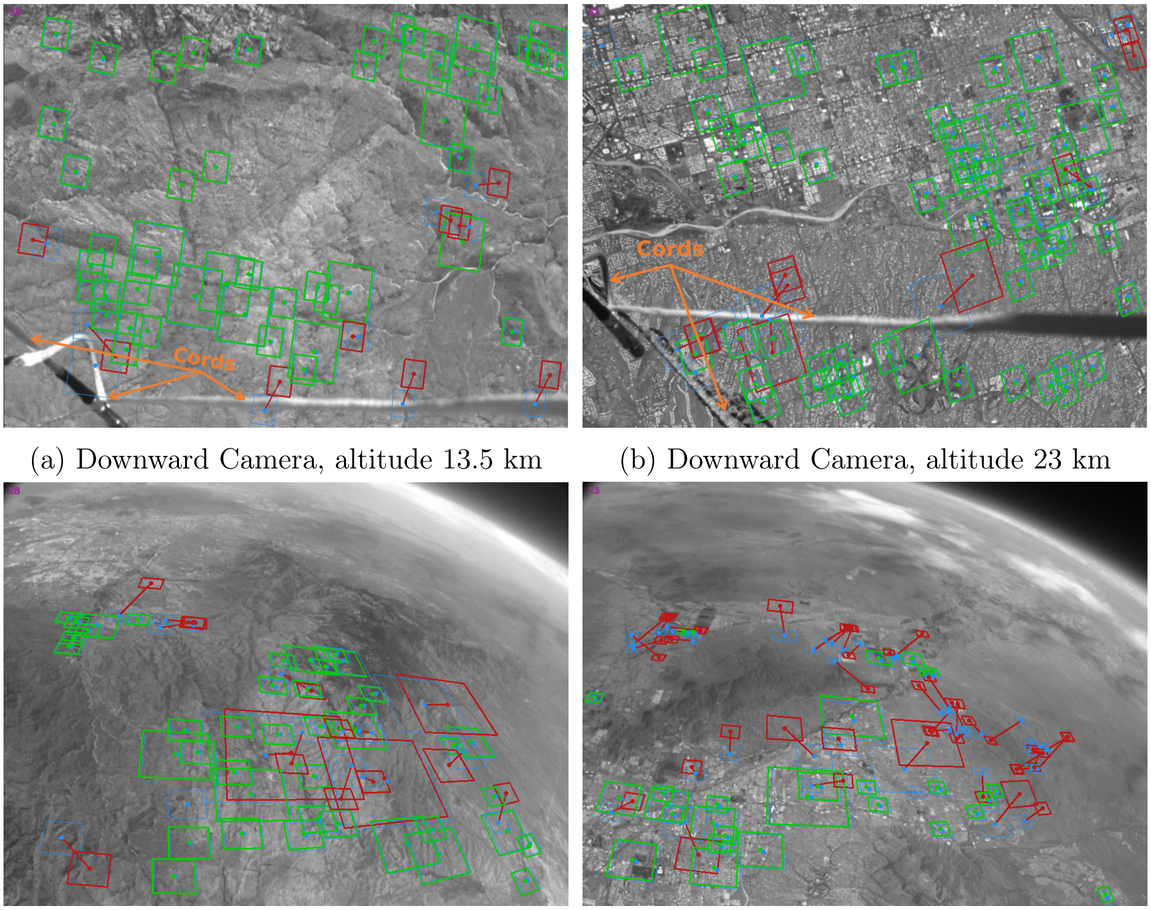
(k) altitude range: 14 km to 9 km

(l) altitude range: 14 km to 9 km

(m) altitude range: 9 km to 4.5 km

(n) altitude range: 9 km to 4.5 km

Figure 2-16: Inlier and outlier pixel error for each segment of balloon dataset. Error is the reprojection error determined by PnP and RANSAC. Left Column: downward camera, Right Column: sideways camera. Rows correspond to different altitude ranges.



(a) Downward Camera, altitude 13.5 km

(b) Downward Camera, altitude 23 km

(c) Sideways Camera, altitude 13.5 km

(d) Sideways Camera, altitude 23 km

Figure 2-17: IBAL landmark match analysis on balloon dataset. Inliers matches are shown in green and outlier are shown in red. Points in blue show initial estimate of landmark location based on initial pose estimate before utilizing cross correlation. Lines connect blue estimate to calculated match location. Landmarks locations covered by the cords are correctly rejected as outliers (top row).

2.4.2 Blue Origin New Shepard Flight

We present results from running IBAL with two cameras (referred to as camera 1 and camera 2) mounted inside the Blue Origin New Shepard capsule. We only show results up to an altitude of approximately 8.5 km since there was a mishap that occurred during flight NS-23 which triggered the capsule escape system. Nevertheless, we are still able to show IBAL working while the rocket achieves nominal speeds up to 880 km/h (550 mph). We seed the initial input image to IBAL using telemetry from New Shepard and then use the previous IBAL pose estimate as the initial pose guess for the next timestep. Unlike the balloon experiment, we do not incorporate the gyroscope measurement to forward propagate the pose estimate since the capsule does not experience significant rotations during its ascent.

We show a similar series of analysis of trajectory error and landmark matches as was presented for the high-altitude balloon experiment. Results are defined with respect to an ENU frame centered at the launch pad. Figure 2-18 shows absolute error for each of the East, North, and Up axes by comparing the position estimate of IBAL with GPS. Figure 2-19 shows total absolute error with respect to flight time and with respect to height above ground level. IBAL's total position error estimate is below 120 m for the duration of the dataset, and that error with camera 2 is as low as 10 m when the rocket is at an altitude of 3.5 km. Average absolute position error for the entire trajectory is 54 m and 34 m for camera 1 and camera 2, respectively. Both cameras show similar performance with IBAL, and slight differences in performance can be explained by the cameras being located on opposite sides of the capsule (and thus viewing different terrain) and by potential unaccounted distortion effects in the camera calibration.

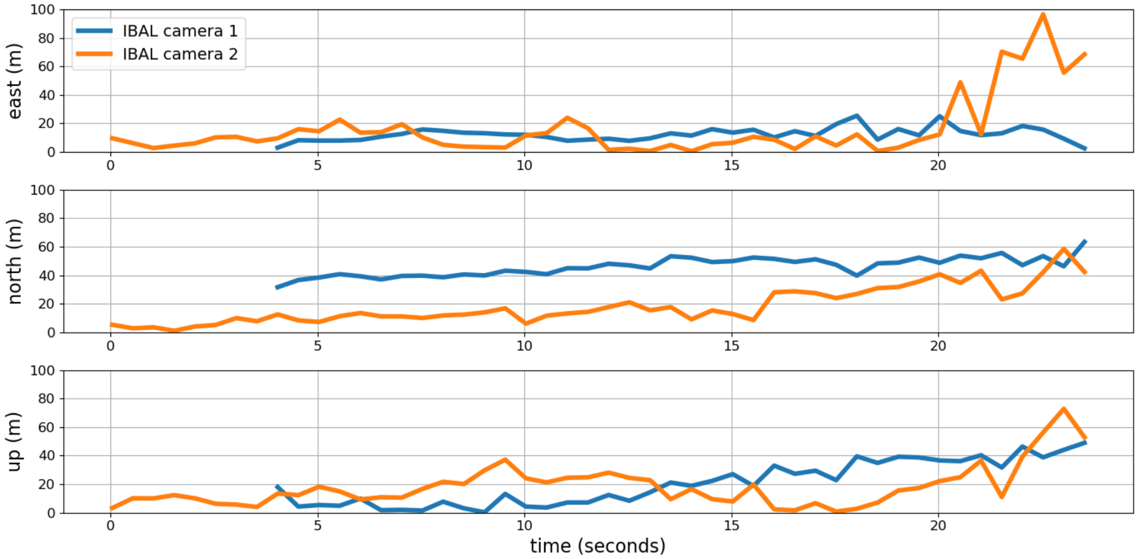


Figure 2-18: IBAL absolute position error on New Shepard dataset: altitude range of 3.5 km to 8.5 km.

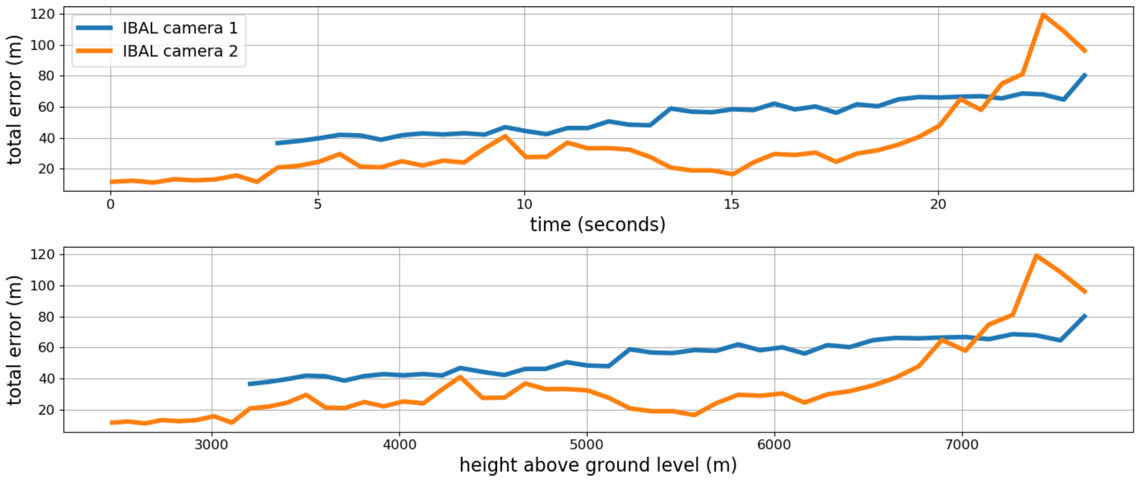
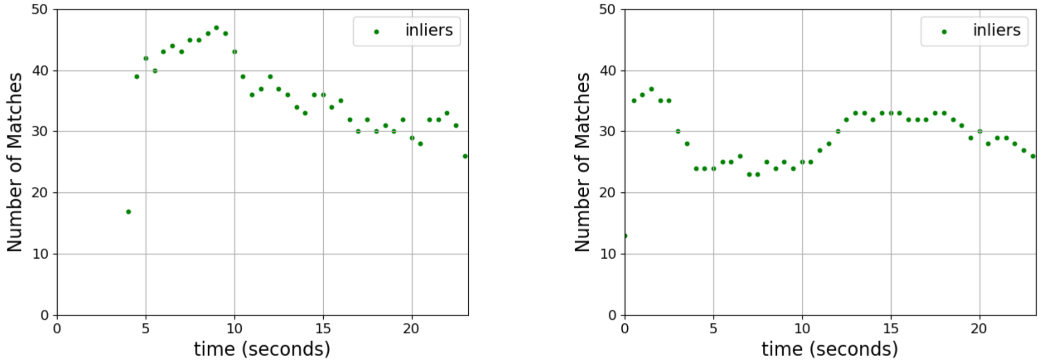


Figure 2-19: IBAL total trajectory error vs. time and height above ground level on New Shepard dataset. Total error is less than 120 m while reaching speeds up to 880 km/h and a peak altitude of 8.5 km.

We also provide an analysis of match correlation for both cameras. Since each processed frame only had at most 2 matches identified as outliers by PnP and RANSAC, we do not include match analysis for outliers in our results. Fig. 2-20a and Fig. 2-20b show number of inliers for both cameras. Fig. 2-21 shows a histogram of the amount of pixel error for the inliers determined by PnP RANSAC for both cameras. Similarly to the results from the balloon flight, pixel error for a majority of the inliers is less than two pixels.

We provide visual examples of IBAL matches on a frame from both cameras in Fig. 2-22. Matches labeled as inliers are shown in green, while outliers are shown in red. There is only one outlier present in the processed image from camera 1 (Fig. 2-22a) and no outliers in the image from camera 2 (Fig. 2-22b).

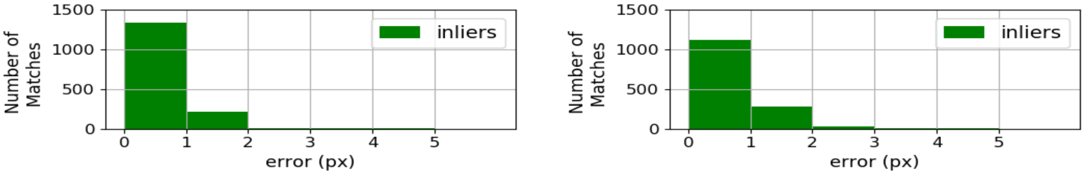
Lastly, we remark on one difficulty of the New Shepard dataset. A mountain range is in view of camera 2 which makes landmark matching more difficult near the latter portion of the dataset as the mountain comes into the camera's FOV (Fig. 2-23). This is due to the presence of shadows in the mountain that may not be consistent with shadows present in the time of day the database imagery was collected. Additionally, the 2D-2D homography assumption which we use to warp landmark templates into the image for correlation begins to break down when 3D structures such as mountains are viewed from low altitudes. Work with database creation such as [70] along with advances in IBAL not mentioned in the chapter can be used to reduce these issue for low altitude navigation over mountains.



(a) IBAL landmark matching results for camera 1

(b) IBAL landmark matching results for camera 2

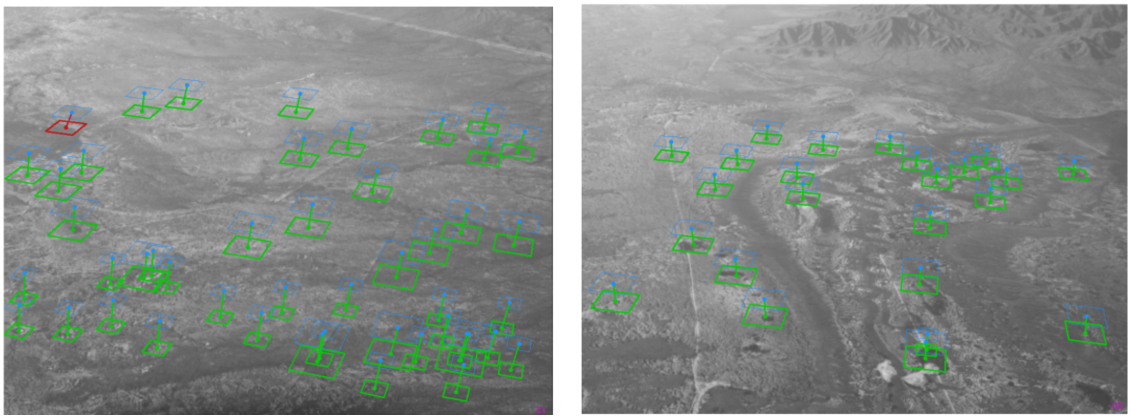
Figure 2-20: IBAL number of inliers and outliers for cameras 1 and 2 on New Shepard dataset as determined by PnP and RANSAC. The data corresponds to an altitude range between 3.5 km and 8.5 km.



(a) Camera 1

(b) Camera 2

Figure 2-21: Inlier pixel error distribution for Cameras 1 and 2 on New Shepard dataset.



(a) IBAL inlier and outlier matches for camera 1 on New Shepard dataset at an altitude of 6.4 km

(b) IBAL inlier and outlier matches for camera 2 on New Shepard dataset at an altitude of 6.4 km

Figure 2-22: IBAL inlier and outlier matches for cameras 1 and 2 on New Shepard dataset. Inliers matches are shown in green and outlier are shown in red. Blue shows initial estimate of landmark location based on initial pose estimate before utilizing cross correlation. Lines connect blue estimate to calculated match location. Images have been rotated by 180 ° for visual appeal.

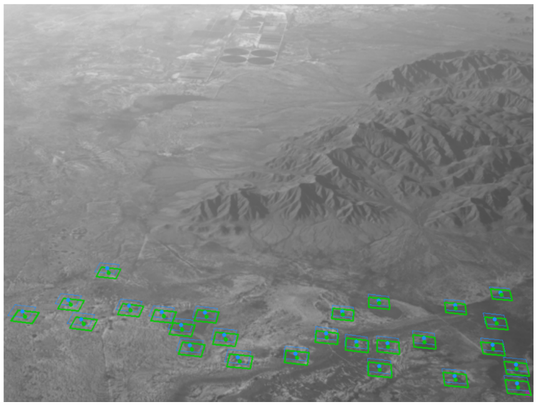


Figure 2-23: IBAL Camera 2 viewing a mountain range on New Shepard dataset. Inliers matches are shown in green. Blue shows initial estimate of landmark location based on initial pose estimate before utilizing cross correlation. Lines connect blue estimate to calculated match location. Image has been rotated by 180 ° for visual appeal.

2.5 Gyroscope Incorporation Ablation Study

We provide an ablation study of forward propagating the IBAL pose estimate with a gyroscope for the high-altitude balloon dataset as mentioned in Section 2.2. The benefits of incorporating the gyroscope data is two-fold. Firstly, since the balloon experiences rapid rotations, in some cases exceeding 20° per second, the gyro provides a more accurate initial guess of the balloon's pose for IBAL, which reduces the frequency at which images must be to used to estimate the pose, hence reducing computation. Additionally, if landmark match quality is temporarily insufficient (typically on the order of 1 to 3 seconds) for PnP and RANSAC, which can be caused for example by significant obstruction by the cords below the balloon, the gyro allows the pose estimate to be carried over until good landmark matches can be found.

Table 2.1 shows the benefits of using the gyro with our balloon dataset. Using the downward-facing camera, we show the percentage of each of the seven data segments IBAL is able to successfully complete with and without incorporating the gyroscope. We also test on two different rates of image processing, noting that while one could partially compensate the lack of gyroscope measurements by increasing the rate of image processing, that strategy is only effective at high altitudes in our dataset.

	33-32.5 km	32.5-29 km	29-23 km	23-18 km	18-14 km	14-9 km	9-4.5 km
4 Hz w/ gyro	100	100	100	100	100	100	100
2 Hz w/ gyro	100	100	100	100	100	100	100
4 Hz w/o gyro	100	100	96	3	3	1	1
2 Hz w/o gyro	100	100	63	0	0	1	1

Table 2.1: Ablation study showing the benefit of incorporating gyroscope measurements with IBAL on each of the seven altitude segments of the balloon dataset for different rates of image processing. Results show the percent of each dataset segment IBAL successfully processes using images from the downward camera.

Chapter 3

Loc-NeRF: Monte Carlo Localization using Neural Radiance Fields

This chapter presents Loc-NeRF, a real-time vision-based robot localization approach that combines Monte Carlo localization and Neural Radiance Fields (NeRF). Work in this chapter has been accepted for publication in the International Conference on Robotics and Automation (ICRA) in 2023 as part of a paper entitled *Loc-NeRF: Monte Carlo Localization using Neural Radiance Fields* [46]. Our system uses a pre-trained NeRF model as the map of an environment and can localize itself in real-time using an RGB camera as the only exteroceptive sensor onboard the robot. We make our code publicly available [1].

Classical approaches for camera pose estimation typically address the task of visual localization by adopting a multi-stage paradigm, where keypoints are first detected and matched between each image frame and the map (where the latter is stored as a collection of images with the corresponding keypoints and descriptors), and six degree-of-freedom (DoF) poses are estimated using Perspective-n-Point (PnP) algorithms [49, 39, 38]. However, such methods are sensitive to the quality of the keypoint matching and require storing a database of images as the map representation. Additionally, variation in lighting conditions or non-Lambertian surfaces (i.e., reflective surfaces whose light emittance depends on the viewing angle) reduce the ability to

[1] `https://github.com/MIT-SPARK/Loc-NeRF`

match features between a stored map and sensor images. In Chapter 2 we present an experimental validation of a classical visual navigation approach for terrain relative navigation whose map is built using a collection of images patches extracted from satellite and elevation maps. Each patch is labeled with GPS position and elevation. Since map representation is a core driver of accuracy in the pose estimation for TRN and the broader field of visual localization, we propose a method which explores the potential of NeRF to be used as a map for visual localization.

Neural Radiance Fields (NeRF) have gained significant popularity for visual rendering in computer vision and graphics as they can encode both 3D geometry and appearance of an environment [56]. NeRFs are fully-connected neural networks trained using a collection of monocular images to approximate functions taking 3D positions as inputs and returning RGB values and view density (the so called "radiance") as output. NeRF can then be used in conjunction with ray tracing algorithms to synthesize novel views [56]. NeRF has even been extended to address challenging rendering problems involving non-Lambertian surfaces, variable lighting conditions [51], and motion blur [45] which we believe makes it particularly attractive for visual localization.

However, at the time Loc-NeRF [46] was published, limited work has shown the potential for NeRF to be used for robotics. In this chapter we benchmark with the current state of the art available at the time Loc-NeRF was made public, with more recent works mentioned in Section 1.1. Prior approaches for NeRF-based localization require both a good initial pose guess and significant computation, making them impractical for real-time robotics applications. By using Monte Carlo localization as a workhorse to estimate poses using a NeRF map model, Loc-NeRF is able to perform localization faster than the state of the art and without relying on an accurate initial pose estimate. In addition to testing on synthetic data, we also run our system using real data collected by a Clearpath Jackal UGV Fig. 3-1 and demonstrate for the first time the ability to perform real-time and global localization with neural radiance fields (albeit over a small workspace).

Before Loc-NeRF, existing literature on NeRF-based localization is sparse. Yen-Chen *et al.* [91] propose iNeRF, the first method to demonstrate pose estimation

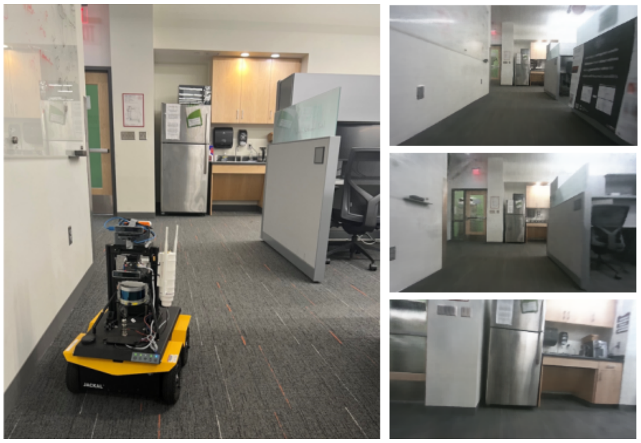


Figure 3-1: Real-time experiments with Loc-NeRF using a Clearpath Jackal UGV (left) equipped with a Realsense d455 camera. Examples of NeRF renderings near the beginning, middle, and end of the experiment (right).

by "inverting" a NeRF; iNeRF estimates the camera pose by performing local optimization of a loss function quantifying the per-pixel mismatch between the map and a given camera image. Adamkiewicz *et al.* [2] propose NeRF-Navigation, which demonstrates the possibility of using NeRF as a map representation across the autonomy stack, from state estimation to planning.

In summary, our contributions are as follows. Following the same research thrust as iNeRF and NeRF-Navigation, we present *Loc-NeRF*, a 6DoF pose estimation pipeline that uses a (particle-filter-based) Monte Carlo localization [15] approach as a novel way to extract poses from a NeRF. More in detail, we design a vision-based particle-filter localization pipeline, that (i) uses NeRF as a map model in the update step of the filter, and (ii) uses visual-inertial odometry or the robot dynamics for highly accurate motion estimation in the prediction step of the filter. The proposed particle-filter approach allows pose estimation with poor or no initial guess, while allowing us to adjust the computational effort by modifying the number of particles. We present experiments showing that Loc-NeRF can: (i) estimate the pose of a single image without relying on an accurate initial guess, (ii) perform global localization on small scenes, and (iii) achieve real-time tracking with real-world data (Fig. 3-1).

The rest of this chapter is organized as follows. Section 3.1 provides a high level overview of NeRF. Section 3.2 presents the structure of Loc-NeRF. Section 3.3 evaluates Loc-NeRF on three types of experiments: benchmarking with iNeRF on pose estimation from a single image, benchmarking with NeRF-Navigation on simulated drone flight data, and real-time navigation with real-world data on a small scale scene.

3.1 NeRF Preliminaries

NeRF [56] uses a multilayer perceptron (MLP) to store a radiance field representation of a scene and render novel viewpoints. NeRF is trained on a scene given a set of RGB images with known poses and a known camera model. At inference time, NeRF renders novel views by predicting the density σ and RGB color $\boldsymbol{c}$ of a point in 3D space given the 3D position and viewing direction of the point. To predict the RGB value of a single pixel, NeRF projects a ray $\boldsymbol{r}$ from the center point of the camera, through a pixel in the image plane.

Then n_{coarse} samples are uniformly generated along the ray and n_{fine} samples are selected based on the estimated σ of the coarse samples. Volume rendering is then used to estimate the color value $\mathcal{C}(\boldsymbol{r})$ for the pixel:

$$\mathcal{C}(\boldsymbol{r}) = \int_{z_{\text{near}}}^{z_{\text{far}}} T(\boldsymbol{r}, z)\sigma(\boldsymbol{r}, z)\boldsymbol{c}(\boldsymbol{r}, z)dz \tag{3.1}$$

where z_{near} and z_{far} are bounds on the sampled depth z along the ray $\boldsymbol{r}$ and $T(\boldsymbol{r}, z)$ is given by:

$$T(\boldsymbol{r}, z) = \exp\left(-\int_{z_{near}}^{z} \sigma(\boldsymbol{r}, z')dz'\right). \tag{3.2}$$

The reader is referred to [56] for a more detailed description.

3.2 Loc-NeRF: Monte Carlo Localization using Neural Radiance Fields

We now present Loc-NeRF, a real-time Monte Carlo localization method that uses NeRF as a map representation. Given a map $\mathcal{M}$ (encoded by a trained NeRF), RGB input image $\mathcal{I}_t$ at each time t, and motion estimates $\mathcal{O}_t$ between time $t-1$ and time

t, Loc-NeRF estimates the 6DoF pose of the robot $\boldsymbol{X}_t$ at time t. In particular, Loc-NeRF uses a particle filter to estimate the posterior probability $\mathbb{P}\left(\boldsymbol{X}_t \mid \mathcal{M}, \mathcal{I}_{1:t}, \mathcal{O}_{1:t}\right)$, where $\mathcal{I}_{1:t}$ and $\mathcal{O}_{1:t}$ are the sets of images and motion measurements collected between the initial time 1 and the current time t, respectively.

Monte Carlo localization [15] relies on a particle filter and models the posterior distribution $\mathbb{P}\left(\boldsymbol{X}_t \mid \mathcal{M}, \mathcal{I}_{1:t}, \mathcal{O}_{1:t}\right)$ as a weighted set of n particles:

$$S_t = \left\{\langle \boldsymbol{X}_t^i, w_t^i \rangle \mid i = 1, ..., n\right\} \tag{3.3}$$

where $\boldsymbol{X}_t^i$ is a 3D pose (represented as a 4×4 transformation matrix in our implementation) associated to the i-th particle, and $w_t^i \in [0, 1]$ is the corresponding weight. The particle filter then updates the set of particles at each time instant (as new images and odometry measurements are received) by applying three steps: prediction, update, and resampling.

3.2.1 Prediction Step

The prediction step predicts the set of particles S_t at time t from the corresponding set of particles S_{t-1} at time $t-1$, given a measurement $\mathcal{O}_t$ of the robot motion between time $t-1$ and time t; the measurement is typically provided by some odometry source (*e.g.,* wheel or visual odometry) or obtained by integrating the robot dynamics; in our implementation, we either use visual-inertial odometry or integrate the robot dynamics, depending on the experiment. When a measurement of the robot's relative motion $\mathcal{O}_t$ is received, the set of particles can be updated by sampling new particles using the motion model $\mathbb{P}\left(\boldsymbol{X}_t \mid \boldsymbol{X}_{t-1}, \mathcal{O}_t\right)$. While the particle filter can accommodate arbitrary motion models, here we adopt a simple model that updates the pose of each particle according to the motion $\mathcal{O}_t$ and then adds Gaussian noise to account for odometry errors:

$$\boldsymbol{X}_t = \boldsymbol{X}_{t-1} \cdot \mathcal{O}_t \cdot \boldsymbol{X}_\epsilon \quad , \quad \boldsymbol{X}_\epsilon = \operatorname{Exp}\left(\boldsymbol{\delta}\right), \tag{3.4}$$

where $\boldsymbol{X}_\epsilon$ is the prediction noise, $\operatorname{Exp}(\cdot)$ is the exponential map for SE(3) (the Special Euclidean group), and $\boldsymbol{\delta} \in \mathbb{R}^6$ is a normally distributed vector with zero mean and covariance $\operatorname{diag}(\sigma_R^2 \cdot \mathbf{I}_3, \sigma_t^2 \cdot \mathbf{I}_3)$, where σ_R and σ_t are the rotation and translation noise standard deviations, respectively.

3.2.2 Update Step

The update step uses the camera image $\mathcal{I}_t$ collected at time t to update the particle weights w_t^i. According to standard Monte Carlo localization [15], we update the weights using the measurement likelihood $\mathbb{P}(\mathcal{I}_t \mid \boldsymbol{X}_t^i, \mathcal{M})$, which models the likelihood of taking an image $\mathcal{I}_t$ from pose $\boldsymbol{X}_t^i$ in the map $\mathcal{M}$. We use a heuristic function to approximate the measurement likelihood as follows:

$$w_t^i = \left(\frac{M}{\sum_{j=1}^{M} (\mathcal{I}_t(\boldsymbol{p}_j) - C(\boldsymbol{r}(\boldsymbol{p}_j, \boldsymbol{X}_t^i)))^2} \right)^4 \tag{3.5}$$

where $\boldsymbol{r}(\boldsymbol{p}_j, \boldsymbol{X}_t^i)$ computes the ray emanating from pixel $\boldsymbol{p}_j$ when the robot is at pose $\boldsymbol{X}_t^i$, and $\mathcal{I}_t(\boldsymbol{p}_j)$ is the image intensity at pixel $\boldsymbol{p}_j$. Intuitively, eq. (3.5) compares the collected image $\mathcal{I}_t$ with the image $\mathcal{C}(\boldsymbol{r})$ predicted by the NeRF map and assigns low weights to particles where the two images do not match. For efficient computation, we compute the weight update (3.5) only using a subset of M pixels randomly sampled from $\mathcal{I}_t$. Weights are then normalized to sum up to 1.

3.2.3 Resampling Step

After the update step, we resample n particles from the set S_t with replacement, where each particle is sampled with probability w_t^i. As prescribed by standard particle filtering, the resampling step allows retaining particles that are more likely to correspond to good pose estimates while discarding less likely hypotheses.

3.2.4 Computational Enhancements and Pose Estimate

Particle Annealing. To improve convergence of the filter and reduce the computational load, we automatically adjust the prediction noise (σ_R, σ_t) and the number of particles n over time. As shown in Section 3.3, this leads to computational and accuracy improvements. The prediction noise and number of particles are updated as shown in Algorithm 3. Our particle annealing approach is similar in spirit to the KLD-based approach from [26]. In particular, we use the standard deviation of the particles' position σ_{S_t} to characterize the spread of the particles in the filter at time t and reduce the prediction noise and the number of particles (initially set to $\sigma_{R,\text{init}}$, $\sigma_{t,\text{init}}$, and n_{init}) when the spread falls below given thresholds (α_{refine} and $\alpha_{\text{super-refine}}$ in Algorithm 3).

Algorithm 3 Particle Annealing

```
Input: $\sigma_{R,\text{init}}, \ \sigma_{t,\text{init}}, \ \sigma_{S_t}, \ n_{\text{init}}$
  $\sigma_R \leftarrow \sigma_{R,\text{init}}$
  $\sigma_t \leftarrow \sigma_{t,\text{init}}$
  $n \leftarrow n_{\text{init}}$
  if $\sigma_{S_t} < \alpha_{\text{super-refine}}$ then
      $\sigma_R \leftarrow \frac{\sigma_{R,\text{init}}}{4}, \quad \sigma_t \leftarrow \frac{\sigma_{t,\text{init}}}{4}, \quad n \leftarrow n_{\text{reduced}}$
  else if $\sigma_{S_t} < \alpha_{\text{refine}}$ then
      $\sigma_R \leftarrow \frac{\sigma_{R,\text{init}}}{2}, \quad \sigma_t \leftarrow \frac{\sigma_{t,\text{init}}}{2}, \quad n \leftarrow n_{\text{reduced}}$
  else
      $\sigma_R \leftarrow \sigma_{R,\text{init}}, \quad \sigma_t \leftarrow \sigma_{t,\text{init}}$
  end if
```

Obtaining a Pose Estimate from the Particles. Besides computing the set of particles, Loc-NeRF returns a single pose estimate $\hat{\boldsymbol{X}}_t$ that is computed as a weighted average of the particle poses. In particular, the position portion of $\hat{\boldsymbol{X}}_t$ is simply the weighted average of the positions of the particles in S_t. The rotation portion of $\hat{\boldsymbol{X}}_t$ is found by solving the geodesic L_2 single rotation averaging problem. The reader is referred to [29] and [48] for details on rotation averaging.

3.3 Experiments

We evaluate Loc-NeRF on three sets of experiments: (i) pose estimation from a single image using the LLFF dataset [55] given either a poor initial guess or no initial guess, where we benchmark against iNeRF [91] (Section 3.3.1), (ii) pose estimation over time using synthetic data from Blender [13], where we benchmark against NeRF-Navigation [2] (Section 3.3.2), and (iii) a full system demonstration where we perform real-time pose tracking using data collected by a Clearpath Jackal UGV (Section 3.3.3).

3.3.1 Single-image Pose Estimation: Comparison with iNeRF

Setup. To show Loc-NeRF's ability to quickly localize given a camera image and from a poor initial guess, we use the same evaluation protocol used in iNeRF [91]. Using 4 scenes (Fern, Fortress, Horns, and Room) from the LLFF dataset [55], we pick 5 random images from each dataset and estimate the pose of each image. For this experiment, both Loc-NeRF and iNeRF use the same pre-trained weights from NeRF-Pytorch [90]. As in [91], we give iNeRF an initial pose guess $\boldsymbol{X}_{\text{iNeRF}}$. The rotation component of $\boldsymbol{X}_{\text{iNeRF}}$ is obtained by randomly sampling an axis from the unit sphere and rotating about that axis by a uniformly sampled angle between [-40°, 40°] with respect to the ground truth rotation. The position portion of $\boldsymbol{X}_{\text{iNeRF}}$ is obtained by uniformly perturbing the ground truth position along each axis by a random amount between [-0.1 m, 0.1 m]. We set iNeRF to use 2048 interest region points ($M = 2048$) as suggested in [91]. Interest regions are found using keypoint detectors and sampling from a dilated mask around those keypoint.

Since Loc-NeRF uses a distribution of particles, we uniformly distribute the initial particles' poses using:

$$\boldsymbol{X}_0^i = \boldsymbol{X}_{\text{iNeRF}} \cdot \text{Exp}\left(\boldsymbol{\delta}\right) \tag{3.6}$$

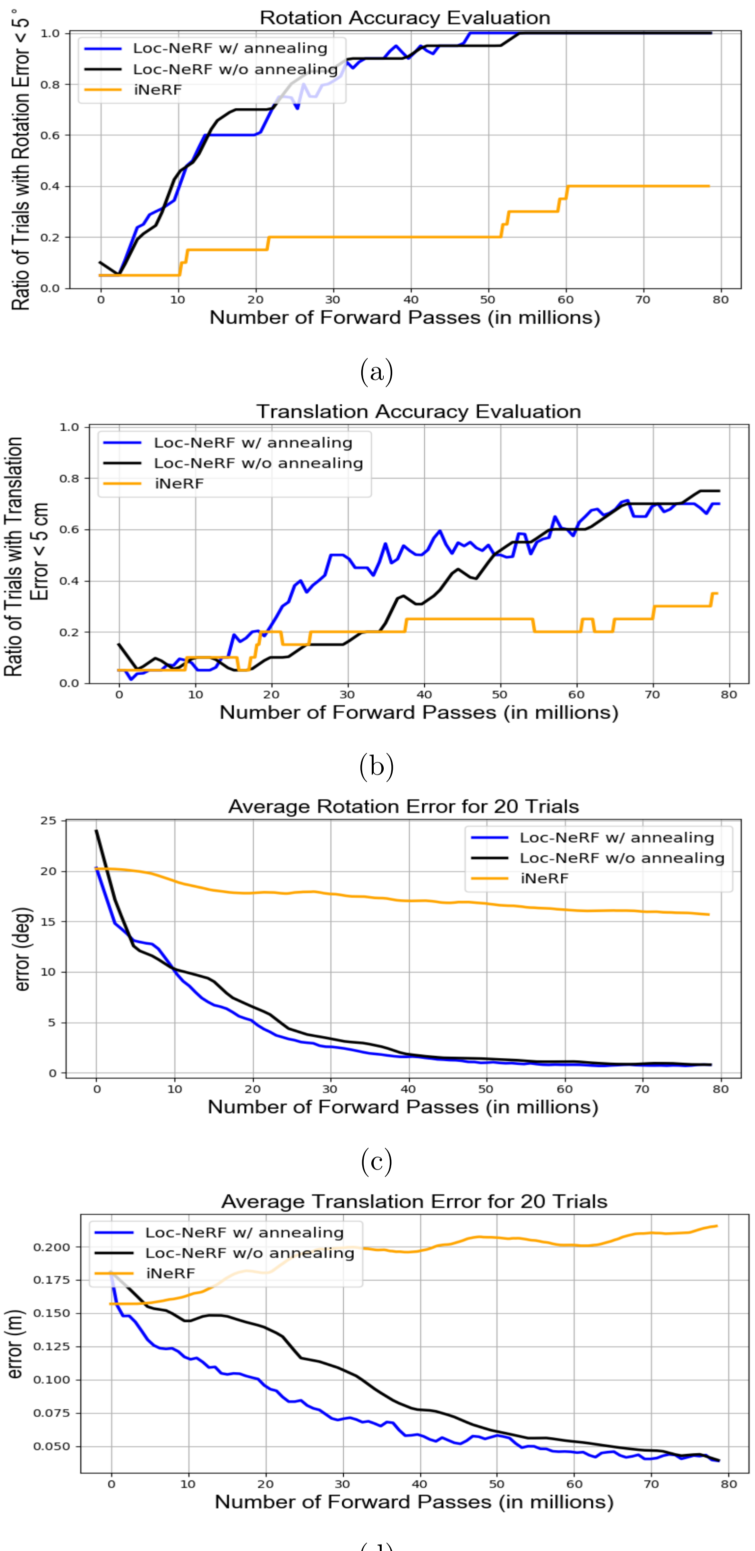
where the entries corresponding to the rotation component and the translation component of $\boldsymbol{\delta}$ are sampled from a uniform distribution in the range [-40°, 40°] and [-0.1

m, 0.1 m], respectively. Since we only test on a static image, we set the motion model of Loc-NeRF to be a zero-mean Gaussian distribution whose standard deviation decreases according to Algorithm 3. Loc-NeRF is initialized with 300 particles which reduces to 100 during annealing. We set Loc-NeRF to use 64 ($M = 64$) randomly sampled image pixels per particle.

Results. We plot the fraction of estimated poses with position and rotation error less than 5 cm and 5° in Fig. 3-2a and Fig. 3-2b, respectively. Since the computational cost of an iNeRF iteration is different from an iteration of Loc-NeRF (due to number of particles and different values of M) we plot performance against the number of NeRF forward passes. Loc-NeRF achieves higher accuracy than iNeRF in terms of both position and rotation.

We also plot the average rotation error and average position error for all 20 trials in Fig. 3-2c and Fig. 3-2d respectively. In our experiments, the position estimate from iNeRF would occasionally diverge or reach a local minimum and thus the average position error for iNeRF actually increases over time. On a laptop with an RTX 5000 GPU, the update step for Loc-NeRF runs at 0.6 Hz for 300 particles which then accelerates to 1.8 Hz during annealing when the number of particles drops to 100. Loc-NeRF runs approximately 55 seconds per trial. As an ablation study of our annealing process (Algorithm 3), we also include results of Loc-NeRF without annealing. Using annealing shows the most benefit for position accuracy and allows update steps to occur at a faster rate due to the decreased number of particles.

We also demonstrate for the first time that global localization can be performed with NeRF. We repeat a similar experiment as before with LLFF data except now we generate an offset translation by translating the ground truth position along each axis by a random amount between [-1 m, 1 m] and generate a random distribution of particles in a $2 \times 2 \times 2$ m cube about that offset. We then sample the yaw angle from a uniform distribution in $[-180°, +180°]$, while we initialize the roll and pitch to the ground truth; the latter is done to mimic the setup where we localize using visual-inertial sensors, in which case the IMU makes roll and pitch directly observable. Note that Loc-NeRF still optimizes the particles in a full 6DoF state. We increase the initial



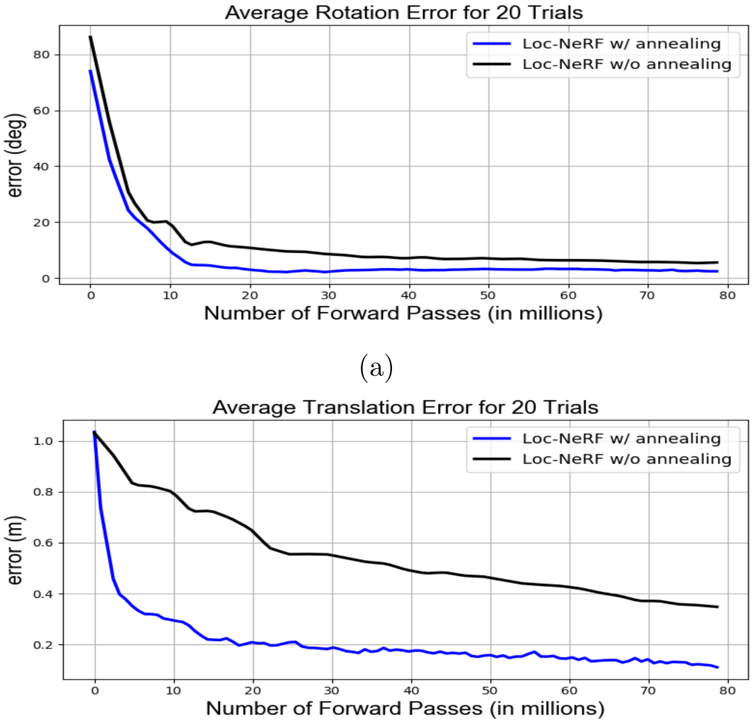
(a)

(b)

(c)

(d)

Figure 3-2: Evaluation of Loc-NeRF and iNeRF on 20 camera poses from the LLFF dataset. As an ablation study of our annealing step, we also include results of Loc-NeRF without using Algorithm 3. (a) Ratio of trials with rotation error $< 5°$. (b) Ratio of trials with translation error < 5 cm. (c) Average rotation error. (d) Average translation error.

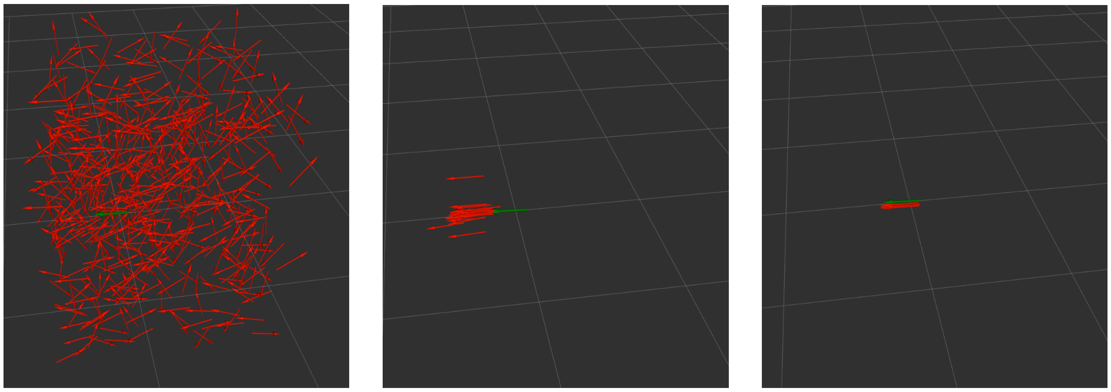


(a)

(b)

Figure 3-3: Evaluation of Loc-NeRF on 20 camera poses from the LLFF dataset without an initial guess for the unknown pose. (a) Average rotation error. (b) Average translation error.

number of particles to 600 which drops to 100 during annealing and reduce M to 32. Results of average rotation and translation error from 20 trials are provided in Fig. 3-3a and Fig. 3-3b. Loc-NeRF is able to converge to an accurate pose estimate while performing global localization. The annealing process is shown to enable significant improvement for position accuracy and also improves rotation accuracy. iNeRF is unable to produce a valid result for global localization and is thus not included in the figure. We also provide a visualization of the distribution of particles in Fig. 3-4 at three points in the optimization process (the initial distribution, after 15 update steps, and after the final update step) for one of the global localization tests.



(a) Initial particle distribution.

(b) Particle distribution after 15 update steps.

(c) Particle distribution after the last update step - 150 for this test.

Figure 3-4: Example of Loc-NeRF particles (red) converging to ground truth pose (green). Tiles are 1 m x 1 m.

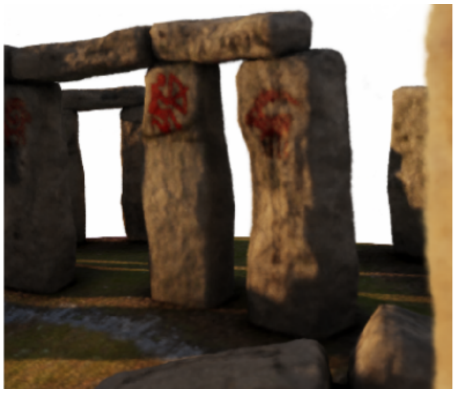


Figure 3-5: Example of NeRF rendering of a scene from Stonehenge.

3.3.2 Pose Tracking: Comparison with NeRF-Navigation

Setup. NeRF-Navigation [2] performs localization using simulated image streams of Stonehenge recreated in Blender, as if they were collected by a drone flying across the scene (Fig. 3-5). For this experiment, both Loc-NeRF and NeRF-Navigation use the same pre-trained weights from torch-ngp [79]. We use the same trajectory and sensor images for evaluating Loc-NeRF and NeRF-Navigation. The prediction step for Loc-NeRF uses the same dynamical model estimate of the vehicle's motion that NeRF-Navigation uses for their process loss. For each image, we run Loc-NeRF for the equivalent number of forward passes as NeRF-Navigation. In particular, we run NeRF-Navigation for 300 iterations per image with $M = 1024$. We use 200 particles

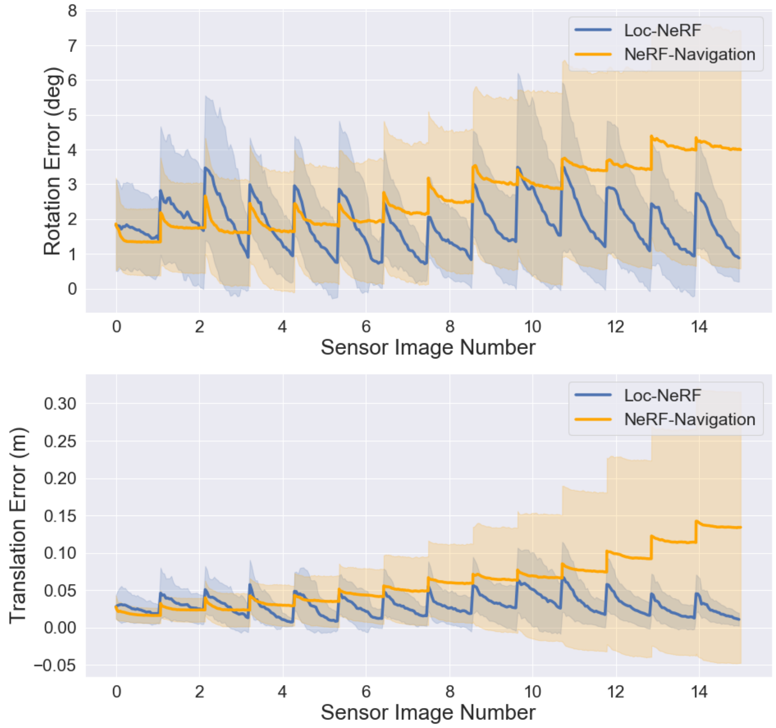


Figure 3-6: Translation and rotation error of Loc-NeRF and NeRF-Navigation averaged over 18 trials. The shaded area shows one standard deviation above and below the mean error. The area between each sensor image number shows the optimization steps. Spikes at the beginning of each sawtooth show error when an image is first received and the pose is forward propagate with a dynamics model, and the bottom of each sawtooth represents the final pose estimate after optimization. For a fair comparison, both methods run the same number of forward passes for each camera image.

for Loc-NeRF with $M = 64$ and run 24 update steps per image.

Results. Fig. 3-6 shows position and rotation error respectively for a simulated drone course over 18 trials. Note that since NeRF-Navigation uses a similar photometric loss as iNeRF —which requires a good initial guess— we assume the starting pose of the drone is well known even though that is not a requirement for Loc-NeRF. The process loss of NeRF-Navigation gives it added robustness to portions of the trajectory where the NeRF rendering is of lower quality. However, Loc-NeRF is still able to achieve lower errors for both position and rotation on average and is able to recover from inaccurate pose estimates.

3.3.3 Full System Demonstration

Finally, we demonstrate our full system running in real-time on real data collected by a robot. We pre-train a NeRF model using NeRF-Pytorch [90] with metric scaled poses and images from a Realsense d455 camera carried by a person. To run Loc-NeRF, we use a Realsense d455 as the vision sensor mounted on a Clearpath Jackal UGV. The prediction step for Loc-NeRF is performed using VINS-Fusion [62]. We log images and IMU data from the Jackal and then run VINS-Fusion and Loc-NeRF simultaneously on a laptop with an RTX 5000 GPU.

We initialize particles across a 1 × 0.5 × 3.5 m area with a uniformly distributed yaw in [-180°,+180°] and uniformly distributed roll and pitch in [-2.5°,+2.5°] (again, the latter are directly observable from the IMU). Loc-NeRF receives data at the real-time rate. The prediction step runs at the nominal VIO rate of 15 Hz. Loc-NeRF starts with 400 particles which reduces to 150 during particle annealing. We set M to 32. With 400 particles the update step runs at approximately 0.9 Hz and then accelerates to 2.5 Hz with 150 particles during annealing. In this experiment, the particles quickly converge enough to trigger the annealing stage after about 6 update steps.

To qualitatively demonstrate that Loc-NeRF converges to the correct pose, we render a full image from NeRF using the pose estimated by Loc-NeRF and compare it with the corresponding camera image. We provide results from this test in Fig. 3-7 at selected points in the trajectory.

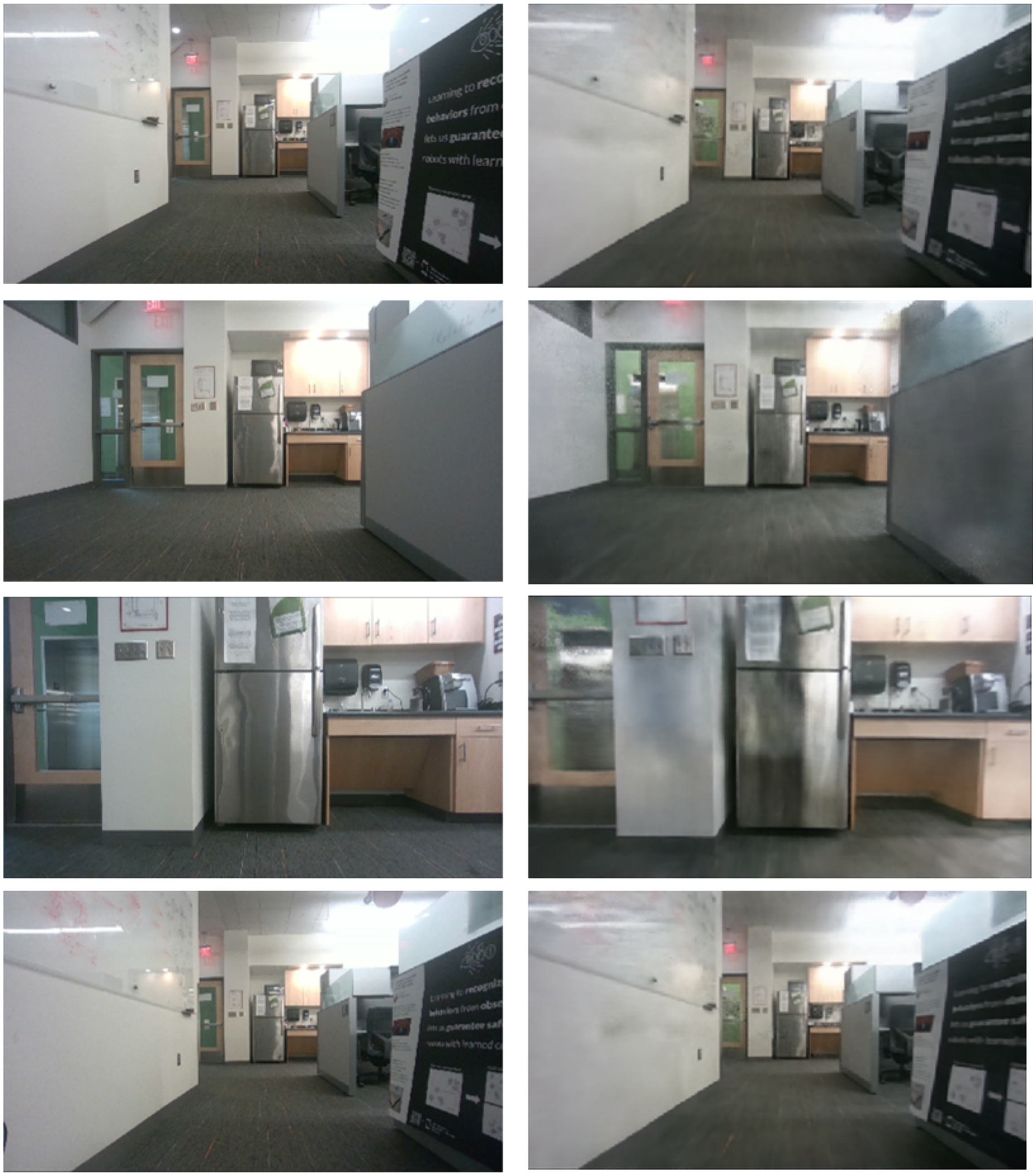


Figure 3-7: Left Column: true images viewed by the camera. Right Column: NeRF-rendered images using the pose estimate from Loc-NeRF. Images correspond to update steps number 20, 40, 60, and 100 which occur at 13, 20, 28, and 44 seconds into the experiment, respectively.

Chapter 4

VERF: Runtime Monitoring of Pose Estimation with Neural Radiance Fields

Estimating the pose of a camera from a monocular image is a fundamental problem in computer vision. However, limited work has been done to independently monitor the accuracy of the estimated pose and detect incorrect estimates without having direct access to depth information of the scene. This need is motivated by the growing use of monocular camera localization in high-stakes scenarios such as self-driving [82], spacecraft entry decent and landing [35, 43, 47, 19], and precision robotics tasks [49].

Recent works such as [91], [2], [46], [42], [97], [5] explore the use of NeRF [56] (Neural Radiance Fields) for camera pose estimation. As an example, in Chapter 3 we introduce Loc-NeRF which uses NeRF as a map of an environment and utilizes a particle filter backbone to output a pose estimate of a provided sensor image. However, there is no clear and reliable measure to determine if the outputted pose is correct - where we define correct as being within some acceptable distance ϵ of the true pose. To overcome this limitation, in this chapter we propose VERF, a collection of two approaches coined VERF-PnP and VERF-Light. VERF uses the sensor image already present in the pose optimization phase to provide assurance that the pose estimate is correct. We additionally require a NeRF model of the scene, but NeRF

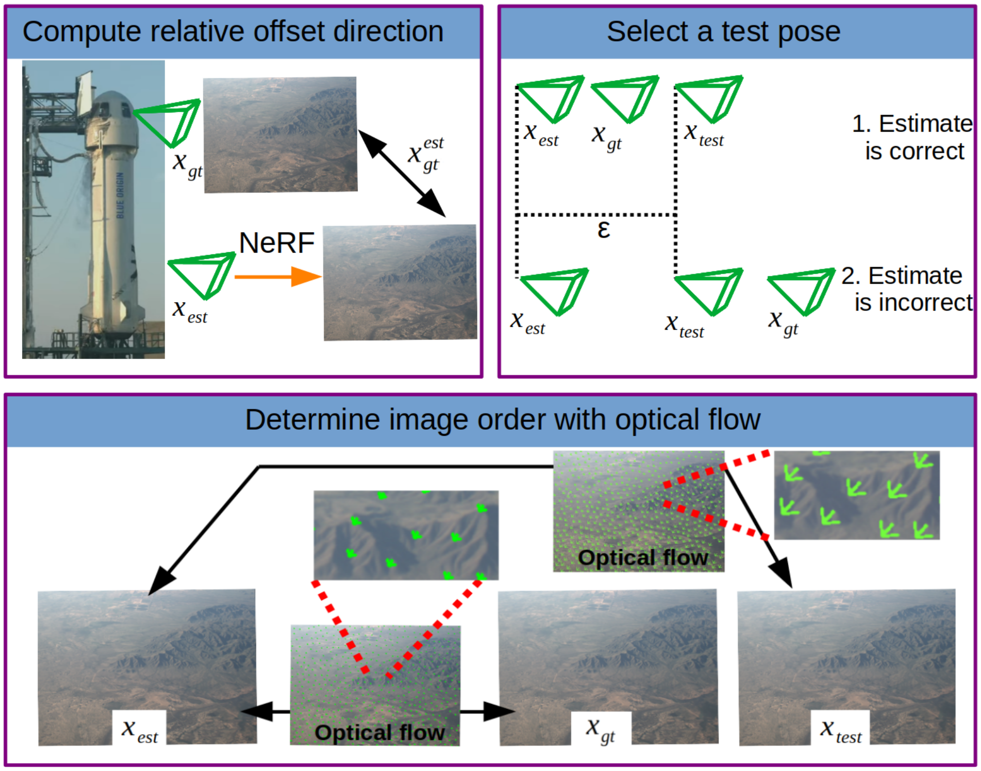


Figure 4-1: Three main phases of VERF-Light. First, the relative error of a pose estimate up to scale is found by comparing a sensor image (collected at the ground truth pose, $\boldsymbol{x}_{gt}$,) to a NeRF image rendered at the pose estimate, $\boldsymbol{x}_{est}$. Next, a test pose, $\boldsymbol{x}_{test}$, is selected at an ϵ distance from the estimated pose such that all three poses are co-linear. Determining if the pose estimate is correct is lastly done by estimating the order of the three poses by comparing optical flow between the three corresponding images.

does not need to be used to produce the pose estimate being monitored which allows VERF to be used for pose monitoring regardless of the pose estimation method.

VERF-PnP renders a stereo pair of images with NeRF, one of which is at the estimated pose and the other at a given baseline, and uses the Perspective-n-Point (PnP) solver with RANSAC [24] to estimate the relative offset to the sensor image.

VERF-Light uses a different methodology which can be stated concisely as follows. We first render an image with NeRF at the estimated pose, $\boldsymbol{x}_{est}$, and use it to determine the relative translation up to scale between the estimated pose and the ground truth pose, $\boldsymbol{x}_{gt}$. To overcome scale ambiguity we render a test image at a pose $\boldsymbol{x}_{test}$ which is at a distance ϵ from the estimate pose in the direction of the sensor image. If the camera origin of these three images are co-linear with no rotation, then we show that we can compare optical flow fields between the three images to determine the order of the camera centers and hence the correctness of the pose estimate (Fig. 4-1). To enable assurance in the presence of noise, we incorporate an estimate of optical flow error and add outlier rejection using geometric constraints to compute a measure of confidence instead of a binary decision. We remark that as the rotation error can be directly observed between the sensor image and the image rendered at the estimated pose, we only focus our attention on determining the quality of the position estimate.

In summary, our contributions are as follows. We provide a collection of two approaches —VERF-PnP and VERF-Light— to estimate whether a monocular pose estimate is correct without requiring depth measurements of the scene. Our runtime pose monitoring approach functions independent of how the pose is estimated and runs in less than half a second on a 3090 GPU. We provide results on the publicly available LLFF dataset [55], on real data collected by an A1 quadruped robot in a room, and on data collected onboard Blue Origin's sub-orbital New Shepard rocket at heights up to 8 km above the ground and at speeds over 800 km/hr. In doing so, we demonstrate the potential of VERF to perform in challenging real-world conditions.

The rest of this chapter is organized as follows. Section 4.1 provides relevant notation and preliminary concepts. Our two methods are presented in Section 4.2 and Section 4.3. In Section 4.4 we evaluate the methods on three types of experiments: LLFF, A1 robot, and sub-orbital rocket. Extra results and studies are included in Appendix A, Appendix B, and Appendix C.

4.1 Notation and Preliminaries

Notation. We will use lowercase symbols (e.g., ϵ) to represent scalars, bold lowercase letters (e.g., $\boldsymbol{x}$) for vectors, and bold uppercase letters (e.g., $\boldsymbol{E}$) for matrices. Sets will be represented with capital calligraphic fonts (e.g., $\mathcal{R}$). Unit vectors and homogeneous vectors are denoted with a bar and tilde (e.g., $\bar{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}$) respectively. Estimated quantities are shown with a caret (e.g., $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{E}}$). We express the 2-norm of a vector as

$\|\cdot\|$.

Let $\boldsymbol{r}_i = (x, y)$ be a coordinate in an image I_i. The sensor image will be referred to as I_{gt} as it is taken by a camera at the true pose. The estimated and test images will be referenced as I_{est} and I_{test}. Let $\boldsymbol{v}(\boldsymbol{r}_i)_{I_i,I_j}$ be the optical flow vector at point $\boldsymbol{r}$ in some image i to the corresponding point in some image j such that $\boldsymbol{r}_i + \boldsymbol{v}(\boldsymbol{r}_i)_{I_i,I_j} = \boldsymbol{r}_j$. $\boldsymbol{E}_{i,j}$ is the essential matrix associated with some images i and j. $[\boldsymbol{a}]_\times$ is the skew-symmetric matrix such that $\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b}$

The Essential Matrix. Assuming points have been calibrated using the camera intrinsic matrix $\boldsymbol{K}$, the essential matrix $\boldsymbol{E}_{i,j}$ relates corresponding homogeneous coordinates $\tilde{\boldsymbol{r}}_i$, $\tilde{\boldsymbol{r}}_j$ in two images with the following constraint:

$$(\tilde{\boldsymbol{r}}_j)^T \boldsymbol{E}_{i,j} \tilde{\boldsymbol{r}}_i = 0. \tag{4.1}$$

$\boldsymbol{E}_{i,j}$ describes the relative pose transform between two cameras defined with a rotation matrix $\boldsymbol{R}$ and translation $\boldsymbol{t}$ up to scale as:

$$\boldsymbol{E}_{i,j} = \boldsymbol{R}[\boldsymbol{t}]_\times. \tag{4.2}$$

Decomposing $\boldsymbol{E}$ to recover $\boldsymbol{t}$ and $\boldsymbol{R}$ yields four solutions, of which only one satisfies the cheiral inequalities [32] which in summary state that triangulated points must lie in front of the two cameras. Since eq. (4.1) does not restrict scale, $\boldsymbol{E}_{i,j}$ along with a point $\boldsymbol{r}_i$ constrains a corresponding point $\boldsymbol{r}_j$ in I_j to a line known as the epipolar line.

Problem formulation. Our objective is to determine if a given position estimate is within some acceptable error bound, ϵ, from the true position:

$$\|\boldsymbol{x}_{est} - \boldsymbol{x}_{gt}\| < \epsilon. \tag{4.3}$$

All we assume are available is the position estimate $\boldsymbol{x}_{est}$, the sensor image I_{gt}, and a NeRF model whose weights are trained on a scene containing I_{gt}.

4.2 VERF-PnP

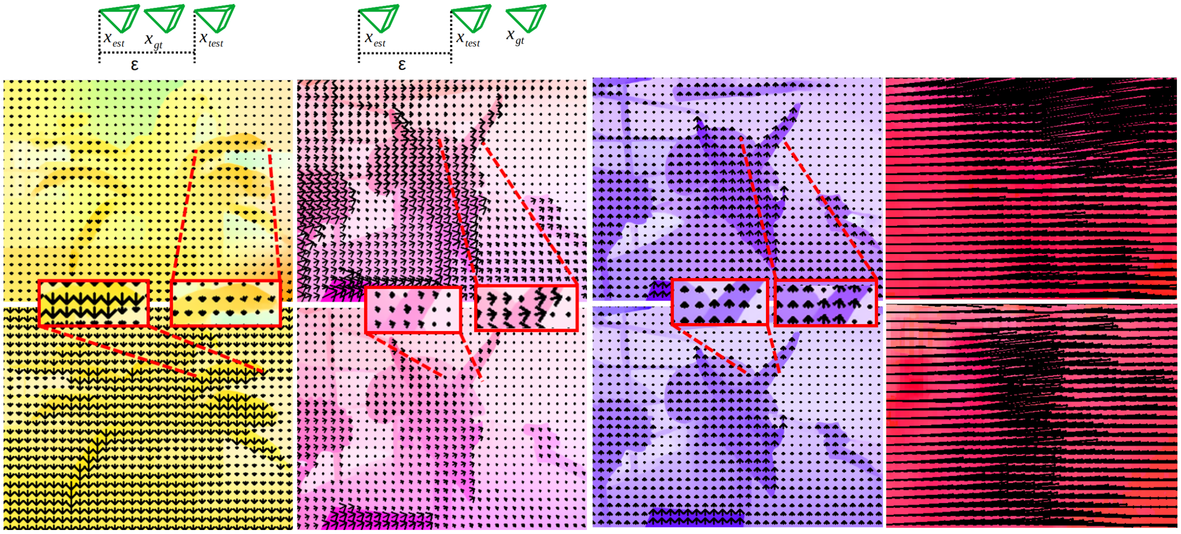
Here we present a simple yet effective method to estimate the correctness of a pose estimate using NeRF. We leverage NeRF to render a pair of stereo images to perform PnP. We first render an image I_{est} at the estimated pose $\boldsymbol{x}_{est}$. Since the true pose $\boldsymbol{x}_{gt}$ is by definition the camera position corresponding to I_{gt}, the verification constraint in eq. (4.3) can be satisfied by showing that the metric offset between $\boldsymbol{x}_{gt}$ and $\boldsymbol{x}_{est}$ is less than ϵ. Towards this goal, we render a second image I_{right} at $\boldsymbol{x}_{right}$ by translating 2ϵ to the right with respect to $\boldsymbol{x}_{est}$. The image pair I_{est} and I_{right} whose poses are both known can then be used as a classical stereo pair of images. We compute the optical flow between these two images using RAFT [80] and use good features to track [68] to get sparse optical flow from RAFT's dense optical flow field. Likewise, we find the correspondences between I_{est} and I_{gt} for the same sparse points with RAFT. We then triangulate the 3D location of the sparse points by knowing $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{right}$ and finally apply PnP with RANSAC [24] to estimate the transform $\hat{\boldsymbol{x}}_{gt}^{est}$ between $\boldsymbol{x}_{est}$ and the unknown $\boldsymbol{x}_{gt}$. Our level of confidence in the accuracy of $\boldsymbol{x}_{est}$ is then estimated as follows:

$$\mathbb{P}(\|\hat{\boldsymbol{x}}_{gt}^{est}\| < \epsilon). \tag{4.4}$$

We model $\|\hat{\boldsymbol{x}}_{gt}^{est}\|$ as a random variable whose mean value is the estimated position from PnP and standard deviation is manually selected. We will show in Section 4.4 the effectiveness of VERF-PnP despite its simplicity.

4.3 VERF-Light

VERF-Light can be divided into three phases (Fig. 4-1): computing the relative offset between $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{gt}$ up to scale, selecting a test position $\boldsymbol{x}_{test}$ distance ϵ from $\boldsymbol{x}_{est}$ and co-linear with the latter two poses, and computing a quality of assurance that eq. (4.3) is met by using an application of the cheiral constraint. In particular, we leverage the fact that given three images from camera poses that are co-linear and



(a) True position error is 1.9 cm. The flow field should allow for concluding that the pose estimate is correct with high confidence. Order of camera positions shown above.

(b) True position error is 10 cm. The flow field should allow for concluding that the pose estimate is incorrect with high confidence. Order of camera positions shown above.

(c) True position error is 5.6 cm. Pose can potentially be verified as correct, but should be done with low confidence.

(d) True position error is much larger than ϵ such that there are no clear correspondences between images.

Figure 4-2: Example of optical flow between I_{est}, I_{gt}, and I_{test} for pose estimates with a correctness condition of $\epsilon = 5cm$. **Top row**: optical flow between I_{est} and I_{gt}. **Bottom row**: optical flow between I_{est} and I_{test}.

with the same rotation, their order along the line they belong to can be determined by comparing the optical flow fields between them. For this arrangement, the flow fields between I_{est} and I_{gt} will be in the same direction as the flow field between I_{est} and I_{test}, and the order of the three positions $\boldsymbol{x}_{est}$, $\boldsymbol{x}_{gt}$, and $\boldsymbol{x}_{test}$ can be estimated by comparing the magnitude of corresponding vectors between the two flow fields. Now if $\boldsymbol{x}_{gt}$ falls between $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{test}$ in such ordering, we can conclude that the error of $\boldsymbol{x}_{est}$ is less than ϵ.

Motivation. Figure 4-2 shows four example conditions that VERF-Light could potentially encounter. In Fig. 4-2a the flow field should provide confidence that the estimated pose is correct. First, the two optical flow fields have similar directions (and hence the same epipole) which validates our assumption of $\boldsymbol{x}_{est}$, $\boldsymbol{x}_{gt}$, and $\boldsymbol{x}_{test}$ being co-linear. Secondly, the magnitude of the optical flow between I_{est} and I_{test} (which have camera centers ϵ apart) is significantly greater than the corresponding flow between I_{est} and I_{gt} meaning that $\boldsymbol{x}_{gt}$ falls between $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{test}$ and hence the estimated pose is within ϵ of the true pose. In Figure 4-2b, the estimate can safely be labeled as incorrect as there is consistent and clear evidence from the flow field that the flow between I_{est} and I_{test} is less in magnitude than the flow field between I_{est} and I_{gt} and again that the three perspectives are co-linear. Figure 4-2c on the other hand does not allow drawing strong conclusion. In this case there should be reduced confidence in the verification decision as the flow field is roughly the same and differences may be only the result of noise. Figure 4-2d should be determined to be an incorrect pose but because of a different reason than Fig. 4-2b - here a cue that the pose is wrong is because no clear correspondences can be found between I_{est} and I_{gt}.

4.3.1 Computing Relative Error Direction of Position Estimate

We use NeRF along with $\boldsymbol{x}_{est}$ to render an image I_{est} which is the image that the camera would see if its center were at $\boldsymbol{x}_{est}$. We use RAFT to compute the dense optical flow between I_{gt} and I_{est}, and use good features to track [68] to extract a set of n pixel coordinates $\boldsymbol{r}_{est}$ (the set of points is subsequently written as $\mathcal{R}$ for brevity) to get sparse optical flow between the two images.

We can use the 5-point algorithm [61] with RANSAC to determine the essential matrix $\boldsymbol{E}_{est,gt}$. RANSAC will attempt to search for a $\hat{\boldsymbol{E}}_{est,gt}$ such that a maximum number of points in $\mathcal{R}$ have sampson distance (a geometric constraint related to eq. (4.1)) less than δ. In short, the sampson distance [66] is an approximation of error to the epipolar line for two corresponding points. The unique solution to extracting the relative position $\hat{\bar{\boldsymbol{x}}}_{gt}^{est}$ up to scale from $\hat{\boldsymbol{E}}_{est,gt}$ is found using the cheiral constraints with maximum consensus. Any points whose correspondence are not part of the maximum consensus or whose sampson distance is larger than δ are removed from the set of inliers $\mathcal{R}$ reducing the set of points to $\boldsymbol{r}_{est} \in \mathcal{R}' \in \mathcal{R}$ where n' is the number of points currently labeled as inliers.

4.3.2 Computing Location of Test Position

We now calculate a test position, $\boldsymbol{x}_{test}$, that is distance ϵ from $\boldsymbol{x}_{est}$ and co-linear with I_{est} and I_{gt}:

$$\boldsymbol{x}_{test} = \boldsymbol{x}_{est} + \epsilon \hat{\bar{\boldsymbol{x}}}_{gt}^{est}. \tag{4.5}$$

The condition for verification, eq. (4.3), can now be stated as:

$$\|\boldsymbol{x}_{est} - \boldsymbol{x}_{gt}\| < \|\boldsymbol{x}_{est} - \boldsymbol{x}_{test}\| = \epsilon \tag{4.6}$$

where the exact pose of $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{test}$ are known and chosen to be ϵ apart. Note that since the positions are collinear by construction, the condition $\|\boldsymbol{x}_{est} - \boldsymbol{x}_{gt}\| < \|\boldsymbol{x}_{est} - \boldsymbol{x}_{test}\|$ is the same as requiring that these positions are ordered as $\boldsymbol{x}_{est}, \boldsymbol{x}_{gt}, \boldsymbol{x}_{test}$ along the line they belong to. We render a new image I_{test} at $\boldsymbol{x}_{test}$ using NeRF.

4.3.3 Determining Verification Score

We again use RAFT to compute the dense optical flow, this time between I_{est} and I_{test} and get sparse optical flow $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{test}}$ for coordinates $\boldsymbol{r}_{est} \in \mathcal{R}'$.

We now consider several properties given our particular choice of $\boldsymbol{x}_{test}$. The first is that it is unnecessary to compute $\boldsymbol{E}_{est,test}$ as we directly know it without error from the true poses of $\boldsymbol{x}_{est}$ and $\boldsymbol{x}_{test}$. Furthermore, it is simply the same as our estimate of $\boldsymbol{E}_{est,gt}$ since $\boldsymbol{x}_{est}$, $\boldsymbol{x}_{gt}$, and $\boldsymbol{x}_{test}$ are aligned and co-linear. This is summarized in the following relation:

$$\hat{\boldsymbol{E}}_{est,gt} = \boldsymbol{E}_{est,test}. \tag{4.7}$$

Determining whether eq. (4.6) is satisfied now reduces to solving an image ordering problem for $I_{est}, I_{gt}, I_{test}$ outlined visually in Fig. 4-1. If $\mathcal{R}'$ contains only true, noiseless inliers, the image ordering problem could now be solved using an application of the cheiral constraint:

$$\begin{gathered} \|\boldsymbol{x}_{est} - \boldsymbol{x}_{gt}\| < \epsilon \implies \\ \forall \boldsymbol{r}_{est} \in \mathcal{R}', \|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}\| < \|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{test}}\| \end{gathered} \tag{4.8}$$

$$\begin{gathered} \|\boldsymbol{x}_{est} - \boldsymbol{x}_{gt}\| > \epsilon \implies \\ \forall \boldsymbol{r}_{est} \in \mathcal{R}', \|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}\| > \|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{test}}\| \end{gathered} \tag{4.9}$$

Equations (4.8) and (4.9) state that for noiseless optical flow fields, the condition of correctness in (4.6) implies the optical flow vector relating a point $\boldsymbol{r}_{est}$ to its corresponding point in I_{gt} should be of less magnitude than the flow vector relating $\boldsymbol{r}_{est}$ to its corresponding point in I_{test}. The two corresponding vectors are in same direction since the three poses are co-linear and hence the points $\boldsymbol{r}_{gt}$ and $\boldsymbol{r}_{test}$ corresponding to $\boldsymbol{r}_{est}$ are bound to the same epipolar line.

However, in the presence of noise and false inliers, we must consider the possibility that the epipolar constraint in eq. (4.1) is not exactly satisfied and hence $\mathcal{R}'$ may contain false inliers, the location of points $\boldsymbol{r}_{gt}$ and $\boldsymbol{r}_{test}$ along the epipolar line $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{I_{est}}$ are perturbed by noise, and that $\hat{\boldsymbol{E}}_{est,gt}$ differs from $\boldsymbol{E}_{est,gt}$.

A primary source of error in our proposed verification method is the calculation of optical flow. Our estimate of the optical flow for any particular point can be expressed as follows:

$$\hat{\boldsymbol{v}}(\boldsymbol{r}_i)_{ij} = \boldsymbol{v}(\boldsymbol{r}_i)_{ij} + \boldsymbol{o}_{ij} + \boldsymbol{\gamma}_{ij} \tag{4.10}$$

where $\|\boldsymbol{\gamma}_{ij}\| \leq \delta$ and $\boldsymbol{o}_{ij}$ is 0 if $\hat{\boldsymbol{v}}(\boldsymbol{r}_i)_{ij}$ is an inlier with sampson distance less than δ. Otherwise, in the case of an outlier, $\boldsymbol{o}_{ij}$ is any arbitrary value such that $\hat{\boldsymbol{v}}(\boldsymbol{r}_i)_{ij}$ can exist at any location in the image. By computing the sampson distance of each $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{test}}$ w.r.t. $\hat{\boldsymbol{E}}_{est,gt}$, we can filter out points with error larger than δ. Note this does not check for error along the epipolar line. We additionally filter out points which are not part of the cheiral set of maximum consensus. We again prune out any points whose correspondences have been labeled as outliers from a set of size n' to a set of n'', i.e. $\boldsymbol{r}_{est} \in \mathcal{R}'' \in \mathcal{R}'$.

Lastly, we project all of $\hat{\boldsymbol{r}}_{I_{gt}}$ and $\hat{\boldsymbol{r}}_{test}$ to the epipolar line defined by $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{I_{est}}$ yielding $\hat{\hat{\boldsymbol{r}}}_{gt}$ and $\hat{\hat{\boldsymbol{r}}}_{test}$ such that pairs of corresponding points satisfy eq. (4.1).

Computing the verification score. Now we must estimate the confidence, q, that the optical flow for corresponding points between I_{est} to I_{test} is greater than the ones between I_{est} to I_{gt}, i.e., $\|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}\| < \|\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{test}}\|$. Using the corresponding optical flow vectors from $\boldsymbol{r}_{est}$ to the projected points $\hat{\hat{\boldsymbol{r}}}_{gt}$ and $\hat{\hat{\boldsymbol{r}}}_{test}$ we define the following confidence score:

$$q = \frac{1}{n''}\sum_{i=1}^{n''} \mathbb{P}(\|\hat{\hat{\boldsymbol{v}}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}\| < \|\hat{\hat{\boldsymbol{v}}}(\boldsymbol{r}_{est})_{I_{est},I_{test}}\|). \tag{4.11}$$

Explicitly, (4.11) is solved with (4.12) using the Normal CDF with a user-specified variance V. Standard deviation is set to a reasonable value of pixel error (e.g. 0.5). A logical and straightforward heuristic for the standard deviation could come from using the already computed sampson error, but this was determined in our experiments to not add value. As a results, we rewrite (4.11) as:

$$q = \frac{1}{n''}\sum_{i=1}^{n''} \Phi\left(\frac{\hat{\hat{\boldsymbol{v}}}(\boldsymbol{r}_{est})_{I_{est},I_{test}} - \hat{\hat{\boldsymbol{v}}}(\boldsymbol{r}_{est})(\boldsymbol{r}_{est})_{I_{est},I_{gt}}}{\sqrt{V[\hat{\hat{\boldsymbol{v}}}(\boldsymbol{r}_{est})_{I_{gt}}]}}\right) \tag{4.12}$$

where Φ is the Normal CDF.

The assurance score mimics a probability, however due to simplifying assumptions such as approximating optical flow uncertainty and potential errors in computing the essential matrix, we do not claim it to be a true probability.

4.4 Experiments

We now present results of running VERF-PnP and VERF-Light on three types of environments ranging from small scale indoor scenes to a rocket trajectory spanning 8 km. For all experiments, we use torch-ngp [79] as our NeRF model. To get experimental sensor images we use randomly selected images from the NeRF training set. For each image, we generate a pose estimate to be checked for correctness by adding a random offset to corresponding ground truth position. To get a diverse distribution of correct and incorrect poses, we randomly selected either a low or high error regime when generating offsets.

In addition to comparing the two proposed methods, we include a simple baseline method that we will refer to as Disparity Check. For this, we simply compute the optical flow between I_{est} and I_{gt} and determine the mean disparity from sparse flow. A naive approach is to assume low disparity means a correct pose estimation whereas a high disparity points to an incorrect pose. We use a folded normal distribution which computes a confidence level of correctness given a mean disparity. All experiments use a standard deviation of 4 pixels for the folded normal distribution. Since this method makes no efforts to handle scale ambiguity, we will show that it does not generalize well for varying scene size.

We pick a 0.5 cutoff confidence level for each method to estimate if the pose is correct or not. To show the generalizability of VERF, for all experiments we use the same standard deviation in (4.12) for VERF-Light (0.5 pixels) and the same standard deviation for VERF-PnP in (4.4). Likewise, the same RANSAC, RAFT, and good features to track parameters are used for all experiments.

4.4.1 LLFF dataset

Setup. We first evaluate VERF on 4 scenes (Fern, Fortress, Horns, and Room) from the LLFF dataset [55]. We pick 250 randomly selected views from the training set of images for each scene to serve as the sensor image I_{gt} and for each image randomly generate a choice for $\boldsymbol{x}_{est}$. We downscale I_{gt} to 504×378 and render the same resolution images when using NeRF. For these 1000 tests, we set ϵ to be 5 cm.

Results. In Fig. 4-3 we show the level of confidence VERF computed that the position error is less than ϵ compared to the actual position error for each test. As expected, confidence levels approach 1 as the position error is well less than ϵ and approach 0 when the position error is much greater than ϵ. On a 3090 GPU, total time to produce a verification from VERF-Light is on average 0.4 seconds with 0.25 seconds of that used for NeRF rendering and is on average 0.35 seconds for VERF-PnP with the same time used for rendering since each method renders two NeRF images.

A summary of results is provided in Table 4.1. Similar performance is observed by VERF-Light and VERF-PnP with most misclassifications occurring for pose estimates with errors near epsilon. Additionally, to test more extreme cases, in Table 4.2 we show results on 100 tests for all three methods for estimated poses with no error and for estimated poses with very large error compared to epsilon (error randomly selected between 1 m to 2.5 m). Here all three methods correctly classify all cases.

	Disparity Check	VERF-PnP	VERF-Light
True Positives	146	415	381
True Negatives	572	494	545
False Positives	5	83	32
False Negatives	277	8	42
Total Correct	**72%**	**91%**	**93%**

Table 4.1: Summary of results for all proposed methods on 1000 tests on LLFF dataset. Classification is made with a 0.5 confidence score cutoff.

	Disparity Check	VERF-PnP	VERF-Light
Total Correct, large error	**100%**	**100%**	**100%**
Total Correct, no error	**100%**	**100%**	**100%**

Table 4.2: Summary of results for all proposed methods on 100 tests of LLFF dataset for position errors much larger than $\epsilon = 5$ cm (errors range from 1 m to 2.5 m) and on poses with no position error.

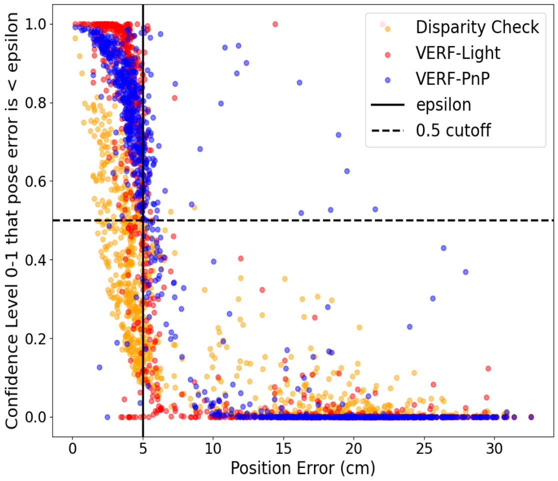


Figure 4-3: VERF confidence level that for 1000 randomly sampled position estimates for LLFF scenes error is less than $\epsilon = 5$ cm

4.4.2 A1 Quadruped

Setup. We train a NeRF (Fig. 4-4) using RGB images collected with a realsense d455 mounted on a Unitree A1 quadruped robot (Fig. 4-4). The robot transverses around a table at varying distances to the table in a motion capture room. Training images and sensor images are $downscaledto 640 \times 360$. Ground truth poses are estimated with COLMAP [67]. To correct from the ambiguous scale from COLMAP, we use vicon odometry to add metric scale to the poses. We again randomly select 1000 images with replacement from the dataset as sensor images and generate a random pose estimate for each image to be verified.

Results. We pick epsilon to be 5 cm and observe similar results as with the LLFF experiment with nearly all VERF mistakes occurring for position errors near the value of epsilon. Results are summarized in Table 4.3 and shown visually in Fig. 4-5. Disparity Check is shown to generalize poorly for different scale scenes as

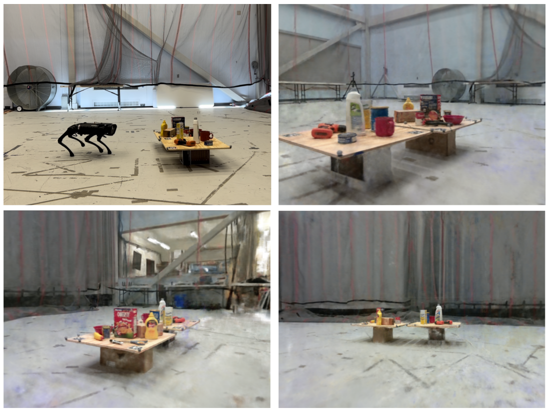


Figure 4-4: A1 quadruped rocket collecting monocular RGB data for NeRF training and VERF evaluation (top left). Three example NeRF rendered views using weights trained by camera data collected onboard an A1 robot.

most of its errors are false negatives for the LLFF experiment whereas most of its errors are false positives for the A1 experiment. Again checking that VERF handles potential edge cases of either no or large error (1 m to 10 m) on 100 tests, we find all methods correctly classify all cases.

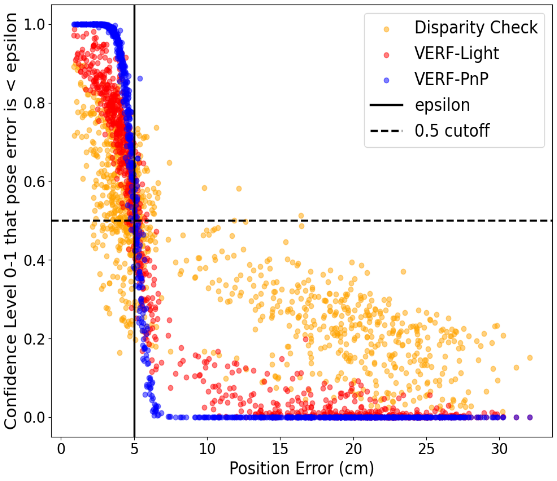


Figure 4-5: VERF confidence level that for 1000 randomly sampled position estimates of the A1's pose, error is less than $\epsilon = 5$ cm

	Disparity Check	VERF-PnP	VERF-Light
True Positives	304	418	411
True Negatives	513	561	551
False Positives	66	18	28
False Negatives	117	3	10
Total Correct	**82%**	**98%**	**96%**

Table 4.3: Summary of results for all proposed methods on 1000 tests of A1 robot dataset. Classification is made with a 0.5 confidence score cutoff.

4.4.3 Sub-Orbital Rocket

Setup. Here we demonstrate the potential for VERF to be used in a highly complex scenario such as for precision spacecraft navigation. This experiment uses data we collected for [47] (discussed in Chapter 2) in which we mounted two cameras inside the capsule of Blue Origin's New Shepard rocket which point out the capsule windows towards the terrain Fig. 2-7.

We train on 140 images collected during the rocket's ascent from an altitude range of approximately 0.2 to 8 km above ground level during which the rocket reaches a speed up to 880 km/hr. We do not include data at higher altitudes as there was a mishap during flight NS-23 which triggered the capsule escape system. The curious reader can refer back to Chapter 2 for more details of our flight data collection.

For simplicity, we train on images collected during the flight and use estimated poses from COLMAP as ground truth. In practice, a NeRF could be trained from prior satellite maps as was done in [86]. Again, similar to the A1 experiment, VERF is run on a scaled NeRF model and we provide metric scale to the COLMAP reconstruction from ground truth poses of the training images - in this case from GPS inside the rocket's capsule.

Results. We pick 40 m for epsilon since this is on the order of typical spacecraft landing accuracy for planetary exploration [35]. A summary of results is shown in Table 4.4 and visually in Fig. 4-7. VERF-PnP performs notably stronger than VERF-Light on this dataset which we believe to be caused by inaccuracies in the essential matrix estimation due to the scene being approximately planar at high altitudes.

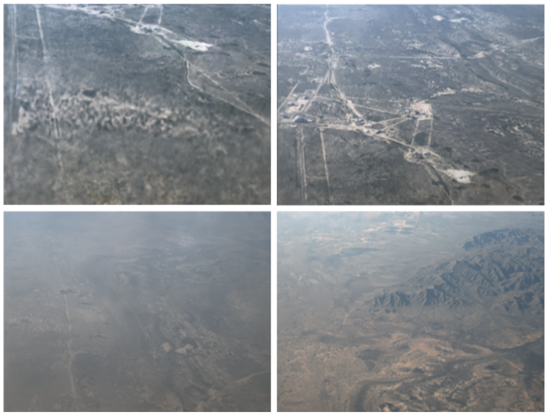


Figure 4-6: Example of four NeRF rendered views from sub-orbital rocket ascent from an altitude range of approximately 1 km to 8 km.

Appendix A provides a study on the effects of error in the essential matrix on VERF-Light. Again checking that VERF handles potential edge cases of either no or large error (500 m to 4000 m) on 100 tests, we find all methods correctly classify all cases.

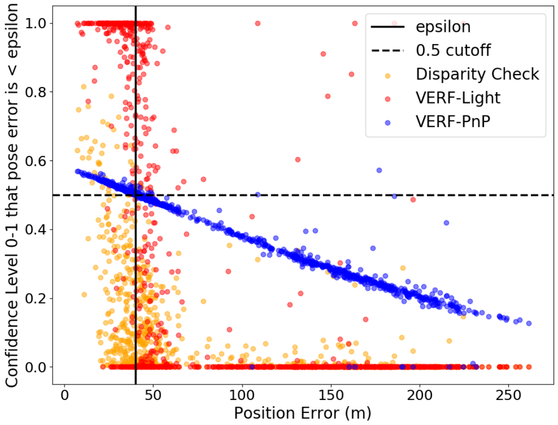


Figure 4-7: VERF confidence level that for 1000 randomly sampled position estimates of the rocket's pose, error is less than $\epsilon = 40$ m

Additionally, as VERF must perform well across a wide range of altitudes for the rocket dataset, we show that VERF-PnP and VERF-Light perform well across all altitudes while Disparity Check does not generalize well. To further demonstrate, in Fig. 4-8 we pick estimated poses with error 15 m (with epsilon again set to 40 m) and run all three methods on sequential images during launch. Figure 4-8 shows that the performance of Disparity Check is dependent on altitude (switching its decision from incorrect to correct after 4 km) while VERF-PnP and VERF-Light perform consistently throughout the rocket's accent.

	Disparity Check	VERF-PnP	VERF-Light
True Positives	38	259	214
True Negatives	738	710	640
False Positives	2	30	100
False Negatives	222	1	46
Total Correct	**78%**	**97%**	**85%**

Table 4.4: Summary of results for all proposed methods on 1000 tests of rocket dataset. Classification is made with a 0.5 confidence score cutoff.

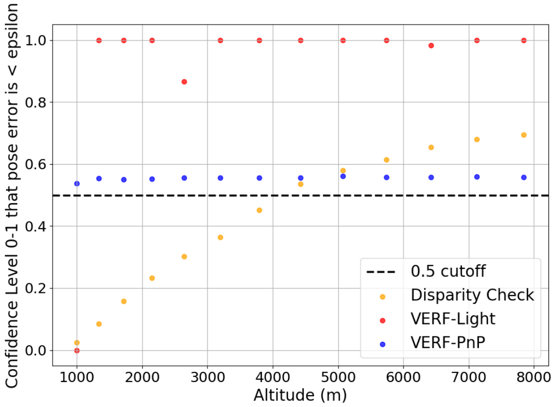


Figure 4-8: VERF confidence level vs altitude for rocket dataset with fixed 15 m of position error. Epsilon is selected as 40 m. Disparity Check is shown to not generalize with varying scene scale while VERF-PnP and VERF-Light performance are independent of the rocket's altitude.

Chapter 5

Conclusion and Future Work

In this thesis, we present a threefold contribution for terrain relative navigation while also developing general-purpose tools for robotic perception by exploring the use of NeRF for visual localization. We report on the performance of a vision-based terrain relative navigation method on data ranging from 4.5 km to 33 km on a high-altitude balloon dataset and on data collected onboard Blue Origin's New Shepard rocket. We evaluate performance of both a sideways-tilted and downward-facing camera for the balloon dataset and two sideways-tilted cameras on the New Shepard dataset. We observe less than 290 meters of average position error on the balloon data over a trajectory of 150 kilometers and with the presence of rapid motions and dynamic obstructions in the field of view of the camera. Additionally, we report less than 55 m of average position error on the New Shepard dataset while reaching an altitude of 8.5 km and a max nominal speed of 880 km/h. As future work, we plan to fly again onboard the New Shepard rocket and capture camera data from ground level to an altitude of over 100 km.

We have presented Loc-NeRF, a Monte Carlo localization approach that uses a Neural Radiance field (NeRF) as a map representation. We show how to incorporate NeRF in the update step of the filter, while the prediction step can be done using existing techniques (e.g., visual-inertial navigation or by leveraging the robot dynamics). While Loc-NeRF presents itself as a promising approach to remove dependency of classical TRN map representations, we have only shown small scale experiments of

Loc-NeRF and leave a large-scale demonstration as future work. Future work is also to leverage larger NeRF models such as [81] and [78] and incorporate modifications to handle global localization on large-scale scenes such as those encountered during terrain relative navigation missions.

Lastly, we present VERF, a collection of two methods (VERF-PnP and VERF-Light) to monitor the accuracy of a monocular camera pose estimate using NeRF. Experiments have shown the effectiveness of VERF on scene scales ranging from small rooms to kilometer scale outdoor scenes. Our method functions independently of how the pose is estimated (i.e., NeRF does not have to be used for pose estimation) and can provide a level of assurance in under half a second. VERF uses geometric constraints to provide a confidence level on the correctness of a pose estimate as opposed to having to learn model uncertainty. Additionally, we provide preliminary formulation of VERF-Full which is a further geometrically constrained VERF. One limitation of VERF is that it is currently intended to be a local verification approach in the sense that if an arbitrarily large epsilon were used, it is possible for NeRF rendered images to be outside the range of the trained NeRF or fail to match features to the sensor image leading to a false assumption of an incorrect pose.

Appendix A

Effects of Essential Matrix Error on VERF-Light

Here we study the effects of the accuracy of the essential matrix estimation for VERF-Light. We repeat each of the three experiments with the same setup except we now provide VERF-Light with the true essential matrix $\boldsymbol{E}_{est,gt}$. Table A.1 shows notable improvements on all experiments with VERF-Light correctly classifying 99% of pose estimates. The most significant improvement is on the rocket dataset. Not only does accuracy go from 86% to 99% but as shown in Fig. A-1, the confidence levels follow a cleaner distribution. We believe this to be caused by the approximate planar scene from high altitudes. This study thus shows the potential to improve VERF-Light with a more effective essential matrix estimation method.

	LLFF	A1	Rocket
True Positives	415	420	256
True Negatives	574	567	735
False Positives	3	12	5
False Negatives	8	1	4
Total Correct	**99%**	**99%**	**99%**

Table A.1: Summary of results on running VERF-Light on all experiments using true essential matrix.

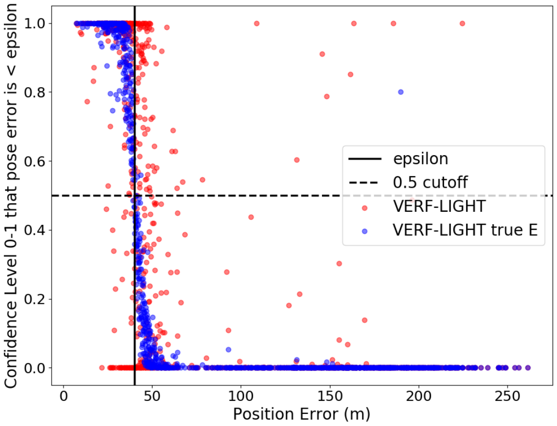


Figure A-1: VERF-Light confidence level for 1000 randomly sampled position estimates for rocket data that their error is less than $\epsilon = 40m$.

Appendix B

Estimating Metric Error with VERF-PnP

A logical question to pose is since PnP can estimate metric distance, how well can VERF-PnP estimate the true error instead of just estimating correctness with respect to an epsilon value. With an estimate of the true error, the pose estimate can then be corrected. Figure B-1 and Fig. B-2 show position errors before an after being corrected in this fashion by VERF-PnP with errors decreasing by an order of magnitude.

For each experiment there were a small number of pose estimates omitted (1 for Fig. B-1 and 9 for Fig. B-2) as PnP diverged. One potential option to automatically check and prevent this is to only accept the updated pose if VERF-Light predicts that the corrected pose is less than epsilon.

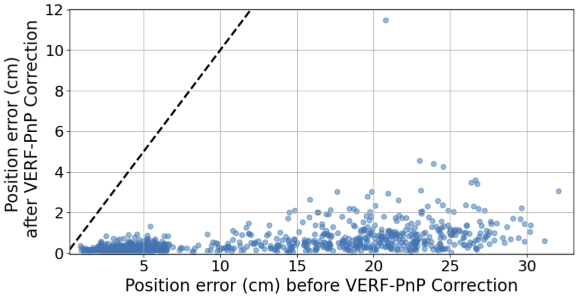


Figure B-1: Position errors before and after being corrected using position error estimate from VERF-PnP. Results shown for 1000 tests from A1 dataset.

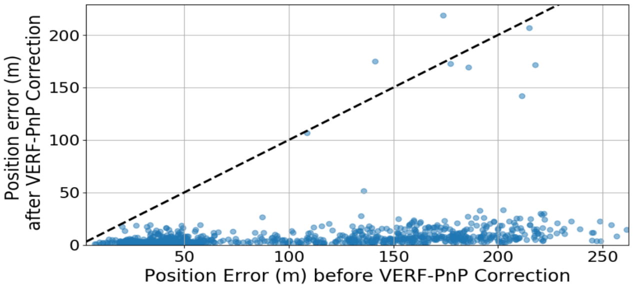


Figure B-2: Position errors before and after being corrected using position error estimate from VERF-PnP. Results shown for 1000 tests from Rocket dataset.

Appendix C

VERF with Trifocal Constraints and Epipolar Uncertainty

This section presents two additions to VERF-Light to make an extended and more theoretically rigorous method referred to as VERF-Full. Namely, VERF-Full makes an additional effort to prune outliers by inducing a trifocal constraint and provides a more reasoned quantification of the uncertainty of optical flow by factoring in the uncertainty of the estimated essential matrix, $\hat{\boldsymbol{E}}_{est,gt}$. In practice, preliminary results show similar experimental performance from VERF-Light and VERF-Full and the reader looking for a simple and effective approach should be tempted towards VERF-Light or VERF-PnP. The remainder of this section leverages two concepts from multi-view geometry: the trifocal tensor and the epipolar band. We assume that we have a set of points $\boldsymbol{r}_{est} \in \mathcal{R}''$ (Section 4.3.3) from VERF-Light.

C.1 Overview of the Trifocal Tensor

The trifocal tensor is attributed to Spetsakis and Aloimonos [71] and Weng et al. [85]. Sun [75] presents a noise analysis of trifocal transfer and the trifocal tensor is critically reviewed by Julià and Monasse [36]. A study of its properties are presented by Martyushev [52].

The trifocal tensor relates geometric constraints between three views using a 3 ×

3×3 tensor. Given three camera views defined by their 3×4 projection matrices $[\boldsymbol{I}|\boldsymbol{0}], [\boldsymbol{A}|\boldsymbol{a}_4], [\boldsymbol{B}|\boldsymbol{b}_4]$ where a_i b_i are the columns of 3×3 matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, the trifocal tensor $T = [T_1, T_2, T_3]$ can be written as follows:

$$T_i = a_i b_4^T - a_4 b_i^T. \tag{C.1}$$

Points $\boldsymbol{r}_i$, $\boldsymbol{r}_j$, and $\boldsymbol{r}_k$ in three views can then be related through trifocal point transfer as follows:

$$[\boldsymbol{r}_j]_\times \left(\sum_{a \in 1,2,3} (\boldsymbol{r}_i)_a T_a \right) [\boldsymbol{r}_k]_\times = \boldsymbol{0}_{3\times3}. \tag{C.2}$$

The reader is referred to [31] and [36] for a detailed review of the trifocal tensor.

C.2 VERF-Full: The Trifocal Constraint

Until now we have not checked for what could be arbitrarily large error of $\hat{\boldsymbol{r}}_{gt}$ and $\hat{\boldsymbol{r}}_{test}$ along the epipolar lines, only that they are close to their defined epipolar line. This is because the epipolar constraint restricts a point in one view to a line in a second view (4.1). By constructing a trifocal transfer constraint of the form of (C.2) we can check the validity of $\hat{\boldsymbol{r}}_{test}$. To get three views of known poses to construct the trifocal tensor we render another test image I_{test2} at an arbitrary pose $\boldsymbol{x}_{test2}$. In practice we pick $\boldsymbol{x}_{test2}$ defined as follows:

$$\boldsymbol{x}_{test2} = \boldsymbol{x}_{est} - \epsilon \hat{\bar{\boldsymbol{x}}}_{gt}^{est}. \tag{C.3}$$

We now use the property that a triplet of points in three views are in correspondence if and only if the trifocal constraint is satisfied [20]. Hence, we use the trifocal transfer constraint from eq. (C.2) with the three known poses of $\boldsymbol{x}_{est}$, $\boldsymbol{x}_{test}$, and $\boldsymbol{x}_{test2}$ along with the points $\boldsymbol{r}_{est} \in \mathcal{R}''$, $\hat{\dot{\boldsymbol{r}}}_{test}$, and $\hat{\boldsymbol{r}}_{test2} = \boldsymbol{r}_{est} + \hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est}, I_{test2}}$. Note we project $\hat{\boldsymbol{r}}_{test}$ to the epipolar line defined by $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{est}$ yielding $\hat{\dot{\boldsymbol{r}}}_{test}$. For points where the constraint is met, we assume $\hat{\dot{\boldsymbol{v}}}(\boldsymbol{r}_{est})_{I_{est}, I_{test}}$ to be a true inlier. All other points get pruned, again reducing our set of inliers to $\boldsymbol{r}_{est} \in \mathcal{R}''' \in \mathcal{R}''$.

To build confidence that $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}\boldsymbol{r}_{est} \in \mathcal{R}'''$ contains true inliers, we compute the optical flow between I_{gt} and I_{test} and utilize the following relation:

$$\boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{gt}} - \boldsymbol{v}(\boldsymbol{r}_{est})_{I_{est},I_{test}} = \boldsymbol{v}(\boldsymbol{r}_{gt})_{I_{gt},I_{test}}. \tag{C.4}$$

Including noise, we get the following constraint for $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}$ to be reasonably considered an inlier:

$$\|\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}} - \dot{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{test}} - \hat{\boldsymbol{v}}(\boldsymbol{r}_{gt})_{I_{gt},I_{test}}\| \leq 2\delta \tag{C.5}$$

where we have assumed that $\dot{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{test}}$ is known now without noise and the maximum allowable error for $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}$ and $\hat{\boldsymbol{v}}(\boldsymbol{r}_{gt})_{I_{gt},I_{test}}$ to be considered an inlier is δ (4.10).

Any points that do not meet the condition in eq. (C.5) are removed, reducing the set of inliers for the final time to $\boldsymbol{r}_{est} \in \mathcal{R}'''' \in \mathcal{R}'''$. We now project all of $\hat{\boldsymbol{r}}_{gt}$ to the epipolar line defined by $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{est}$ yielding $\hat{\dot{\boldsymbol{r}}}_{gt}$.

C.3 VERF-Full: Estimating the Uncertainty of Optical Flow

Given a known location of $\boldsymbol{r}_{est}$, we must estimate the uncertainty of its corresponding location $\hat{\dot{\boldsymbol{r}}}_{gt}$ in I_{gt} which we will approximate by a Gaussian distribution which restricts that $\hat{\dot{\boldsymbol{r}}}_{gt}$ lie on the epipolar line defined with $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{est}$.

Each Gaussian distribution will be set such that the expected value is the location of the point - i.e., $\hat{\dot{\boldsymbol{r}}}_{gt}$. For the standard deviation, while VERF-Light uses a fixed pre-selected value, we want VERF-Full to both estimate the uncertainty of the optical flow vector and the uncertainty of the essential matrix. Error in $\hat{\boldsymbol{E}}_{est,gt}$ would cause the three poses $\boldsymbol{x}_{est}, \boldsymbol{x}_{gt}, \boldsymbol{x}_{test}$ to not be co-linear, reducing our ability to determine their order by directly comparing optical flow fields of their respective images. We collectively approximate both of these sources of error by setting the standard deviation to the width of the epipolar band at each correspondence between $\boldsymbol{r}_{est}$ and $\hat{\boldsymbol{r}}_{gt}$.

Assuming the location of $\boldsymbol{r}_{est}$ is known exactly (which is fair considering we choose the points in I_{est}), the covariance matrix of the epipolar line in I_{gt}, $\boldsymbol{l}_{gt}$, defined from $\hat{\boldsymbol{E}}_{est,gt}\tilde{\boldsymbol{r}}_{est} \in \mathcal{R}''''$, is:

$$\boldsymbol{L} = \frac{\partial \bar{\boldsymbol{l}}}{\partial \boldsymbol{E}} \boldsymbol{G} (\frac{\partial \bar{\boldsymbol{l}}}{\partial \boldsymbol{E}})^T \tag{C.6}$$

where $\boldsymbol{l}_{gt}$ is normalized to $\bar{\boldsymbol{l}}_{gt}$ and $\boldsymbol{G}$ is the covariance matrix of $\hat{\boldsymbol{E}}_{est,gt}$ and is computed using the analytical method described in [95] and [23].

The 3×3 conic coefficent matrix $\boldsymbol{C}$ can now we written as:

$$\boldsymbol{C} = \bar{\boldsymbol{l}}_{gt}\bar{\boldsymbol{l}}_{gt}^T - k^2\boldsymbol{L}, \tag{C.7}$$

where k is a confidence level for the χ^2 distribution $\chi^2(k, 2)$. The reader is referred to [30] [76] for a more detailed review of computing the epipolar band.

Finally, the standard deviation is found by approximating it equal to the width of the band at $\hat{\boldsymbol{r}}_{gt}$. We now compute the final score using (4.12) where the standard deviation is the epipolar band width computed for each point and we use $\boldsymbol{r}_{est} \in \mathcal{R}'''' \in \mathcal{R}''$ in place of $\boldsymbol{r}_{est} \in \mathcal{R}''$. A potential alternative to using the width of the epipolar band is to consider a probability distribution for each optical flow vector $\hat{\boldsymbol{v}}(\boldsymbol{r}_{est})_{I_{est},I_{gt}}$ by using the epipolar pdf defined by Brandt [8].

Bibliography

[1] U.S. Geological Survey. https://apps.nationalmap.gov/downloader/, 2022.

[2] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022.

[3] Farzin Amzajerdian, Larry Petway, Glenn Hines, Bruce Barnes, Diego Pierrottet, and George Lockard. Doppler lidar sensor for precision landing on the moon and mars. In *2012 IEEE Aerospace Conference*, pages 1–7, 2012.

[4] P. Antonante, H. Nilsen, and L. Carlone. Monitoring of perception systems: Deterministic, probabilistic, and learning-based fault detection and identification. *arXiv preprint: 2205.10906*, 2022. blue(pdf).

[5] Gil Avraham, Julian Straub, Tianwei Shen, Tsun-Yi Yang, Hugo Germain, Chris Sweeney, Vasileios Balntas, David Novotny, Daniel DeTone, and Richard Newcombe. Nerfels: Renderable neural codes for improved camera pose estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5061–5070, June 2022.

[6] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[7] Tye M. Brady, E. S. Bailey, Timothy P. Crain, and Stephen C. Paschall. Alhat system validation. *8th International ESA Conference on Guidance, Navigation and Control Systems*, 2011.

[8] Sami S. Brandt. On the probabilistic epipolar geometry. *Image and Vision Computing*, 26(3):405–414, 2008. 15th Annual British Machine Vision Conference.

[9] Dean Brown. Decentering distortion of lenses. In *Photogrammetric Engineering*, pages 444–462, 1966.

[10] L. Carlone. Estimation contracts for outlier-robust geometric perception. *arXiv preprint: 2208.10521*, 2022. blue(pdf).

[11] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo. In *Intl. Conf. on Computer Vision (ICCV)*, pages 14124–14133, 2021.

[12] Ronald Clark. Volumetric bundle adjustment for online photorealistic scene capture. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6124–6132, 2022.

[13] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[14] Thiago L. T. da Silveira and Claudio R. Jung. Perturbation analysis of the 8-point algorithm: A case study for wide fov cameras. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 11749–11758, 2019.

[15] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo Localization for mobile robots. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1999.

[16] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[17] Chris Dever, Lei Hamilton, Rob Truax, Leonard Wholey, and Keith Bergeron. Guided-airdrop vision-based navigation. *24th AIAA Aerodynamic Decelerator Systems Technology Conference*, 2017.

[18] Lena Downes, Ted J. Steiner, and Jonathan P. How. Deep learning crater detection for lunar terrain relative navigation. *AIAA Scitech 2020 Forum*, 2020.

[19] C. D. Norman et al. Autonomous navigation performance using natural feature tracking during the osiris-rex touch-and-go sample collection event. *The Planetary Science Journal*, 3(5):101, may 2022.

[20] O. Faugeras and T. Papadopoulo. A nonlinear method for estimating the projective geometry of 3 views. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 477–484, 1998.

[21] O.D. Faugeras. *Three-dimensional computer vision: A geometric viewpoint.* The MIT press, Cambridge, MA, 1993.

[22] O.D. Faugeras and Q.T. Luong. *The geometry of multiple images.* The MIT press, Cambridge, MA, 2001. with contributions from T. Papadopoulo.

[23] Olivier Faugeras, Zhengyou Zhang, Cyril Zeller, and Gabriella Csurka. Characterizing the Uncertainty of the Fundamental Matrix. Technical Report RR-2560, INRIA, June 1995.

[24] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981.

[25] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Trans. Robotics*, 33(1):1–21, 2017. arxiv preprint: 1512.02363, blue(pdf), technical report GT-IRIM-CP&R-2015-001.

[26] Dieter Fox. KLD-Sampling: Adaptive particle filters. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 14. MIT Press, 2001.

[27] Mercedes Garcia-Salguero, Jesus Briales, and Javier Gonzalez-Jimenez. Certifiable relative pose estimation. *Image and Vision Computing*, 109:104142, 2021.

[28] Mercedes Garcia-Salguero and Javier Gonzalez-Jimenez. Fast and robust certifiable estimation of the relative pose between two calibrated cameras. *arXiv preprint arXiv:2101.08524*, 2021.

[29] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *IJCV*, 103(3):267–305, 2013.

[30] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

[31] R.I. Hartley. Lines and points in three views and the trifocal tensor. *Intl. J. of Computer Vision*, 22(2):125– 140, 1997.

[32] Richard I. Hartley. Chirality. *Intl. J. of Computer Vision*, 26:41–61, 1998.

[33] Xiaoyan Hu and Philippos Mordohai. A quantitative evaluation of confidence measures for stereo vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 34(11):2121–2133, 2012.

[34] Andrew E. Johnson and James F. Montgomery. Overview of terrain relative navigation approaches for precise lunar landing. In *IEEE Aerospace Conference*, pages 1–10, 2008.

[35] Andrew Edie Johnson, Seth B. Aaron, Johnny Chang, Yang Cheng, James F. Montgomery, Swati Mohan, Steven Schroeder, Brent E. Tweddle, Nikolas Trawny, and Jason Xin Zheng. The lander vision system for mars 2020 entry descent and landing, 2017.

[36] Laura Fernàndez Julià and Pascal Monasse. A critical review of the trifocal tensor estimation. In *Pacific-Rim Symposium on Image and Video Technology*, 2017.

[37] Anup Katake, Chrisitian Bruccoleri, Puneet Singla, and John L. Junkins. LandingNav: a precision autonomous landing sensor for robotic platforms on planetary bodies. In David P. Casasent, Ernest L. Hall, and Juha Röning, editors, *Intelligent Robots and Computer Vision XXVII: Algorithms and Techniques*, volume 7539 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 75390D, January 2010.

[38] Lei Ke, Shichao Li, Yanan Sun, Yu-Wing Tai, and Chi-Keung Tang. GSNet: joint vehicle pose and shape reconstruction with geometrical and scene-aware supervision. In *European Conf. on Computer Vision (ECCV)*, pages 515–532. Springer, 2020.

[39] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *Intl. J. of Computer Vision*, 81(2):155, 2009.

[40] Fu Li, Hao Yu, Ivan Shugurov, Benjamin Busam, Shaowu Yang, and Slobodan Ilic. Nerf-pose: A first-reconstruct-then-regress approach for weakly-supervised 6d object pose estimation, 2022.

[41] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-adjusting neural radiance fields. In *Intl. Conf. on Computer Vision (ICCV)*, 2021.

[42] Yunzhi Lin, Thomas Müller, Jonathan Tremblay, Bowen Wen, Stephen Tyree, Alex Evans, Patricio A. Vela, and Stan Birchfield. Parallel inversion of neural radiance fields for robust pose estimation, 2022.

[43] David A. Lorenz, R. D. Olds, Alexander May, Courtney Mario, Mark E. Perry, Eric Edward Palmer, and Michael G. Daly. Lessons learned from osiris-rex autonomous navigation using natural feature tracking. *IEEE Aerospace Conference*, pages 1–12, 2017.

[44] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. J. of Computer Vision*, 60(2):91–110, 2004.

[45] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V. Sander. Deblur-NeRF: Neural radiance fields from blurry images. *arXiv preprint arXiv:2111.14292*, 2021.

[46] D. Maggio, M. Abate, J. Shi, C. Mario, and L. Carlone. Loc-NeRF: Monte carlo localization using neural radiance fields. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2022. blue(pdf),blue(video).

[47] D.R. Maggio, C. Mario, B. Streetman, T.J. Steiner, and L. Carlone. Vision-based terrain relative navigation on high altitude balloon and sub-orbital rocket. In *AIAA SciTech Forum*, 2023.

[48] J.H. Manton. A globally convergent numerical algorithm for computing the centre of mass on compact lie groups. In *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference*, volume 3, pages 2211–2216 Vol. 3, 2004.

[49] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, 2019.

[50] C. E. Mario, C. J. Miller, C. D. Norman, E. E. Palmer, J. Weirich, O. S. Barnouin, M. G. Daly, J. A. Seabrook, D. A. Lorenz, R. D. Olds, R. Gaskell, B. J. Bos, B. Rizk, and D. S. Lauretta. Ground testing of digital terrain models to prepare for osiris-rex autonomous vision navigation using natural feature tracking. *The Planetary Science Journal*, 3(5):104, may 2022.

[51] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[52] Evgeniy Martyushev. On some properties of calibrated trifocal tensors. *Journal of Mathematical Imaging and Vision*, 58:321–332, 06 2017.

[53] Richard A McKern. *A Study of Transformation Algorithms For Use in a Digital Computer.* PhD thesis, MIT Instrumentation Laboratory, 1968.

[54] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. GNeRF: GAN-based Neural Radiance Field without Posed Camera. In *Intl. Conf. on Computer Vision (ICCV)*, 2021.

[55] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019.

[56] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *arXiv preprint arXiv:2003.08934*, 2020.

[57] Arthur Moreau, Nathan Piasco, Moussab Bennehar, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Crossfire: Camera relocalization on self-supervised features from an implicit representation, 2023.

[58] Anastasios I. Mourikis, Nikolas Trawny, Stergios I. Roumeliotis, Andrew E. Johnson, Adnan Ansar, and Larry Matthies. Vision-aided inertial navigation for spacecraft entry, descent, and landing. *IEEE Transactions on Robotics*, 25(2):264–280, 2009.

[59] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[60] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021.

[61] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(6):756–770, 2004.

[62] Tong Qin, Jie Pan, Shaozu Cao, and Shaojie Shen. A general optimization-based framework for local odometry estimation with multiple sensors. *arXiv preprint: 1901.03638*, 2019.

[63] Quazi Rahman, Peter Corke, and Feras Dayoub. Run-time monitoring of machine learning for robotic perception: A survey of emerging trends. *IEEE Access*, PP:1–1, 01 2021.

[64] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 12892–12901, 2022.

[65] D.M. Rosen, L. Carlone, A.S. Bandeira, and J.J. Leonard. SE-Sync: a certifiably correct algorithm for synchronization over the Special Euclidean group. *Intl. J. of Robotics Research*, 2018. arxiv preprint: 1611.00128, blue(pdf).

[66] Paul D Sampson. Fitting conic sections to "very scattered" data: An iterative refinement of the bookstein algorithm. *Computer Graphics and Image Processing*, 18(1):97–108, 1982.

[67] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[68] J. Shi and C. Tomasi. Good Features to track. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.

[69] Leena Singh and Sungyung Lim. On lunar on-orbit vision-based navigation: Terrain mapping, feature tracking driven ekf. *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2012.

[70] Kyle W. Smith, Nicholas Anastas, Andrew Olguin, Matthew Fritz, Ronald R. Sostaric, Sam Pedrotty, and Teming Tse. Building maps for terrain relative navigation using blender: an open source approach. *AIAA SCITECH 2022 Forum*, 2022.

[71] Minas E. Spetsakis and John Aloimonos. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–183, jun 1990.

[72] Stephen R. Steffes, Fredy Monterroza, Lylia Benhacine, and Courtney Mario. Optical terrain relative navigation approaches to lunar orbit, descent and landing. *AIAA Scitech 2019 Forum*, 2019.

[73] Ted J. Steiner, Tye M. Brady, and Jeffrey A. Hoffman. Graph-based terrain relative navigation with optimal landmark database selection. In *2015 IEEE Aerospace Conference*, pages 1–12, 2015.

[74] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew Davison. iMAP: Implicit mapping and positioning in real-time. In *Intl. Conf. on Computer Vision (ICCV)*, 2021.

[75] Zhaohui Sun. Method for error analysis of trifocal transfer, May 5 2016. US Patent application Publication 2004/008617 A1.

[76] Frédéric Sur, Nicolas Noury, and Berger marie odile. Computing the uncertainty of the 8 point algorithm for fundamental matrix estimation. *British Machine Vision Conf. (BMVC)*, 09 2008.

[77] Rajat Talak, Lisa Peng, and Luca Carlone. Certifiable 3D object pose estimation: Foundations, learning models, and self-training. *arXiv preprint: 2206.11215*, Jan. 2023. blue(pdf).

[78] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 8248–8258, June 2022.

[79] Jiaxiang Tang. Torch-ngp: a pytorch implementation of instant-ngp, 2022. https://github.com/ashawkey/torch-ngp.

[80] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. *ArXiv*, abs/2003.12039, 2020.

[81] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-NERF: Scalable construction of large-scale nerfs for virtual fly-throughs. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 12922–12931, June 2022.

[82] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang. The ApolloScape open dataset for autonomous driving and its application. *IEEE Trans. Pattern Anal. Machine Intell.*, 2019.

[83] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021.

[84] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. NerfingMVS: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Intl. Conf. on Computer Vision (ICCV)*, pages 5610–5619, 2021.

[85] Juyang Weng, Thomas Huang, and Narendra Ahuja. Motion and structure from line correspondences: Closed-form solution, uniqueness, and optimization. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(3), 1992.

[86] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. BungeeNeRF: Progressive neural radiance field for extreme multi-scale scene rendering. In *European Conf. on Computer Vision (ECCV)*, 2022.

[87] H. Yang and L. Carlone. Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization. *IEEE Trans. Pattern Anal. Machine Intell.*, 2022. blue(pdf).

[88] H. Yang, J. Shi, and L. Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *arXiv preprint: 2001.07715*, 2020. blue(pdf).

[89] Heng Yang and Marco Pavone. Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation, 2023.

[90] Lin Yen-Chen. NeRF-pytorch. `https://github.com/yenchenlin/nerf-pytorch/`, 2020.

[91] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[92] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *Intl. Conf. on Computer Vision (ICCV)*, 2021.

[93] Jiahui Zhang, Fangneng Zhan, Rongliang Wu, Yingchen Yu, Wenqing Zhang, Bai Song, Xiaoqin Zhang, and Shijian Lu. VMRF: View matching neural radiance fields, 2022.

[94] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. NeRF++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020.

[95] Z.Y. Zhang. Determining the epipolar geometry and its uncertainty - a review. *Intl. J. of Computer Vision*, 27(2):161–195, March 1998.

[96] Ji Zhao, Wanting Xu, and Laurent Kneip. A certifiably globally optimal solution to generalized essential matrix estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[97] Zhenxin Zhu, Yuantao Chen, Zirui Wu, Chao Hou, Yongliang Shi, Chuxuan Li, Pengfei Li, Hao Zhao, and Guyue Zhou. Latitude: Robotic global localization with truncated dynamic low-pass filter in city-scale nerf, 2022.

[98] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. NICE-SLAM: Neural implicit scalable encoding for slam. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022.