ZERO-SHOT GROUND VEHICLE NAVIGATION ENABLED BY AERIAL-BASED DIGITAL TWIN SCENE RECONSTRUCTIONS

by

Desiree J. M. Fisker

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
University of Toronto Institute for Aerospace Studies

Zero-Shot Ground Vehicle Navigation Enabled by Aerial-Based Digital Twin Scene Reconstructions

Desiree J. M. Fisker
Master of Applied Science

University of Toronto Institute for Aerospace Studies
2025

Abstract

Furthering autonomy is a priority for many unmanned ground vehicle (UGV) operations, as minimizing human presence in remote or potentially hazardous environments reduces risk and lowers costs. In this thesis, we present Virtual Teach and Repeat (VirT&R): an extension of the Teach and Repeat (T&R) framework that enables GPS-denied, zero-shot autonomous ground vehicle navigation in untraversed environments. VirT&R leverages aerial imagery to realistically simulate an environment, allowing a pilot to virtually control a modelled UGV to define a path through it, replacing the traditional method of teaching. A pose graph is then created by associating submaps made from a virtually generated point cloud along the defined path, which enables the real robot to execute the desired mission in the actual environment.

Our focus was to rigorously test and establish VirT&R as a functional extension by deploying it in different environments and seasons, for both Radar and Lidar T&R, with different scene reconstruction tools. Over 33 km of autonomous driving demonstrated that smooth, reliable path-tracking can be achieved with similar closed-loop performance to existing T&R implementations without requiring an operator to manually pilot the UGV to teach a route in an environment.

Acknowledgements

I want to express my utmost gratitude to Tim Barfoot and Melissa Greeff for their mentorship and guidance throughout my time as a master's student. Tim's seasoned insight, and leadership in evaluating research topics and methodologies, as well as in deciding when to pivot or proceed, were invaluable. Melissa's detailed support and feedback during my research also greatly contributed to the further development of my problem solving and writing skills. Together, Tim and Melissa continually raised the bar to ensure research was conducted and presented at a high standard. This taught me many valuable lessons that are pertinent to success, the most important being to fail fast and iterate.

The Defence Research Department of Canada supported this work via grant funding and actively collaborated by mirroring the experiments conducted with the algorithm developed in this thesis at their testing site in Suffield, Alberta. Thank you to Jack Collier for organizing and facilitating the use of the Argobot and the CFB Suffield site for expanded testing of our work. I also want to thank my colleagues, Alec Krawciw and Sven Lilge, for sharing their expertise with me and for supporting me through many of the firsts in an academic career. Additionally, thank you to Alec, Sven, Hunter Ma, and Anthony Becca, for collaborating on publications and help with experiments. As well, I thank my colleagues and friends from UTIAS and the U of T Robotics Institute for their comradery. Furthermore, I could not have done all this without my close friends and housemates Timo, Julia, and Chris, who have seen me through thick and thin, and have always been there for me, thank you all. As well, I'd like to thank my parents Pauline and Chris for their unwavering love and support, and everyone at Skydive Toronto who I got to work and jump with over my years in Toronto for being amazing people and making sure my life was a good balance of work and play.

Lastly, I would like to thank the musical stylings of David Bowie, ABBA, Queen, Electric Light Orchestra, Fleetwood Mac, Radiohead, The Offspring, and others in those genres for keeping my spirits high during long nights spent gathering data for my experiments. I am so grateful that the two years I spent with ASRL will be part of the good old days I will get to look back on.

Contents

List of Tables

List of Figures

List of Acronyms

AGL	above ground level
CSV	comma separated value
DoF	degrees of freedom
DSM	Digital Surface Models
EXIF	Exchangeable Image File
FMCW	Frequency Modulated Continuous Wave
FoV	field of view
GCP	Ground Control Point
GNSS	Guidance Navigation Satellite System
GPS	Global Positioning System
GPU	Graphic Processing Unit
GS	Gaussian Splatting
ICP	Iterative Closest Point
LIDAR	Light Detection and Ranging
LPIPS	Learned Perceptual Image Patch Similarity
LT&R	Lidar Teach and Repeat
MLP	Multi-Layer Perceptron
MPC	Model Predictive Control
MVS	Multi View Stereo
NeRF	Neural Radiance Field
PnP	Perspective-n-Point
PoV	point of view
PTE	Path Tracking Error
PSNR	Peak Signal to Noise Ratio
RADAR	Radio Detection and Ranging
RANSAC	Random Sample Consensus
RGB	Red Green Blue
RMSE	Root Mean Squared Error
ROS	Robot Operating System
RT&R	Radar Teach and Repeat

SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SfM	Structure-from-Motion
SSIM	Structural Similarity Index Measure
SVD	Singular Value Decomposition
TSDF	Truncated Signed Distance Function
T&R	Teach and Repeat
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
URDF	Unified Robot Description Format
UTIAS	University of Toronto Institute for Aerospace Studies
UWB	ultra-wideband
VI-SLAM	Visual Inertial Simultaneous Localization and Mapping
VIO	Visual Inertial Odometry
VirT&R	Virtual Teach and Repeat
VirLT&R	Virtual Lidar Teach and Repeat
VirRT&R	Virtual Radar Teach and Repeat
VTRN	Visual Terrain Relative Navigation
VT&R	Visual Teach and Repeat
VT&R3	Visual Teach and Repeat 3

Notation

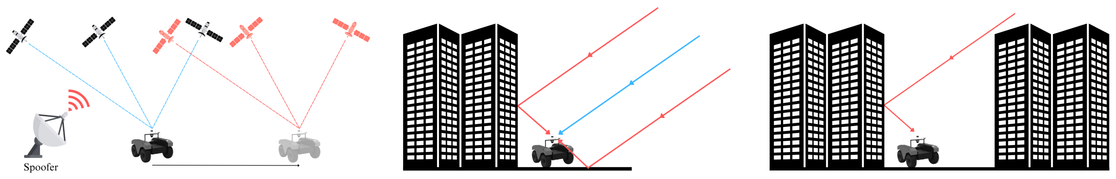
a	This font is used for quantities that are real scalars
$\mathbf{a}$	This font is used for quantities that are real column vectors
$\mathbf{A}$	This font is used for quantities that are real matrices
$\mathbf{1}$	The identity matrix
$\underrightarrow{\boldsymbol{\mathcal{F}}}_a$	A vectrix representing a reference frame in three dimensions
$SE(3)$	The special Euclidean group, a matrix Lie group used to represent poses
$\mathfrak{se}(3)$	The Lie algebra associated with $SE(3)$
$\mathbf{T}_{ba}$	A matrix in $SE(3)$ that transforms vectors from frame $\underrightarrow{\boldsymbol{\mathcal{F}}}_a$ to $\underrightarrow{\boldsymbol{\mathcal{F}}}_b$
$\exp(\cdot^\wedge)$	A Lie algebra operator mapping from $\mathfrak{se}(3)$ to $SE(3)$
$\ln(\cdot)^\vee$	A Lie algebra operator mapping from $SE(3)$ to $\mathfrak{se}(3)$

Chapter 1

Introduction

1.1 Motivation

The field of autonomous robot navigation is well-established, having produced numerous algorithms tailored to different sensing modalities and use cases, ranging from engineering and civil operations [1], to agricultural analysis [2], to humanitarian and emergency response [3]. A core capability that enables these operations is accurate and robust localization. In many cases, localization via route or waypoint following is facilitated by Global Navigation Satellite Systems (GNSS); however, connectivity is often subject to interruption or failure due to factors such as intentional jamming or spoofing, multipath effects, environmental interference, and accumulated errors. Thus, designing solutions that avoid these issues is of great interest, as many of the most valuable applications of autonomous vehicles—such as underground mining, military operations, and remote or extraterrestrial exploration—take place in environments where GNSS support is either unavailable or highly unreliable.



(a) Signal Jamming/Spoofing (b) Multipath Effects (c) Environmental Interference

Figure 1.1: A visualization of common issues faced when using GNSS in different scenarios.

Vision-based localization has been a prominent solution to replace or supplement GNSS, as cameras are small, lightweight, passive sensors unaffected by the factors that limit GNSS. Popular methods, such as Visual Inertial Odometry (VIO) [4], Visual Inertial Simultaneous Localization and Mapping (VI-SLAM) [5], and Visual Terrain Relative Navigation (VTRN) [6], rely on techniques derived from foundational concepts in state estimation, such as dead-reckoning and sensor-based pose estimation. While they can excel at providing scalable and robust real-time localization, these tools are still limited by their perception of the world, which can be negatively impacted by common factors such as varying lighting conditions, weather, reflective surfaces, airborne particulates, viewpoint alterations, and subpar scene coverage or occlusions. Additionally, they are constrained by significant storage requirements, computational demands, odometry drift, error accumulation, and a limited field of view (FoV).

As technology advances, many sensors are becoming more readily accessible, increasing the research effort towards exploiting the benefits of different modalities. Supplementing a passive sensor, such as a camera, with active sensors such as Lidar or Radar may increase the complexity of a navigation pipeline, but it introduces the idea of creating a framework that uses different sensors to overcome environmental constraints such as lighting or weather to enhance autonomous capabilities. A summary of various sensor pros and cons can be found in Table 1.1.

Table 1.1: A comparison of common sensors used for mapping and localization on autonomous vehicles.

	Vision	**Lidar**	**Radar**
Pros	✓ Works regardless of most geometries ✓ Human interpretable	✓ Accurate ✓ Dense pointcloud	✓ Multiple returns per measurement ✓ Weather ✓ Long range
Cons	✗ Weather/Lighting ✗ Appearance changes	✗ Weather	✗ Noisy ✗ Sparse

The Visual Teach and Repeat 3 (VT&R3)[1] [7] framework provides a complete autonomy stack and addresses many of the mentioned practical challenges by simplifying navigation tasks into path-following routines. Through a learning phase (the *teach pass*) where a robot is manually piloted in an environment to generate a pose graph (the *map*) embedded with rich sensor data, and an autonomous route-following phase (the *repeat pass*) where the robot uses live sensor data to localize to the map, highly precise navigation is achieved. This approach has been demonstrated successfully on UGV platforms equipped with a 3D Lidar [8–10], a Radar [11], and Red Green Blue (RGB) vision sensors [7], as well as on UAV platforms using RGB vision sensors [12, 13]. T&R does have an inherent operational drawback; it requires a human operator to manually drive the robot through the environment during the teach pass to create the pose graph map while ensuring route traversability. This requirement becomes impractical or even dangerous in remote, hazardous, or inaccessible use case scenarios, such as facility inspections, battlefields, mining operations, or extraterrestrial exploration.

To overcome this limitation, we propose Virtual Teach and Repeat (VirT&R), visualized in Figure 1.2. Our algorithm modifies the teach pass of the existing T&R framework and leverages the affordable and lightweight camera sensors commonly found on commercial drones to capture de-

[1] https://github.com/utiasASRL/vtr3

tailed aerial images of outdoor environments. Advanced reconstruction tools then use these images to create realistic simulations and localization data of the captured scene, which can be customized for different sensor modalities. Figure 1.3 shows a generated mesh and localization layer data for Lidar and Radar-based VirT&R .

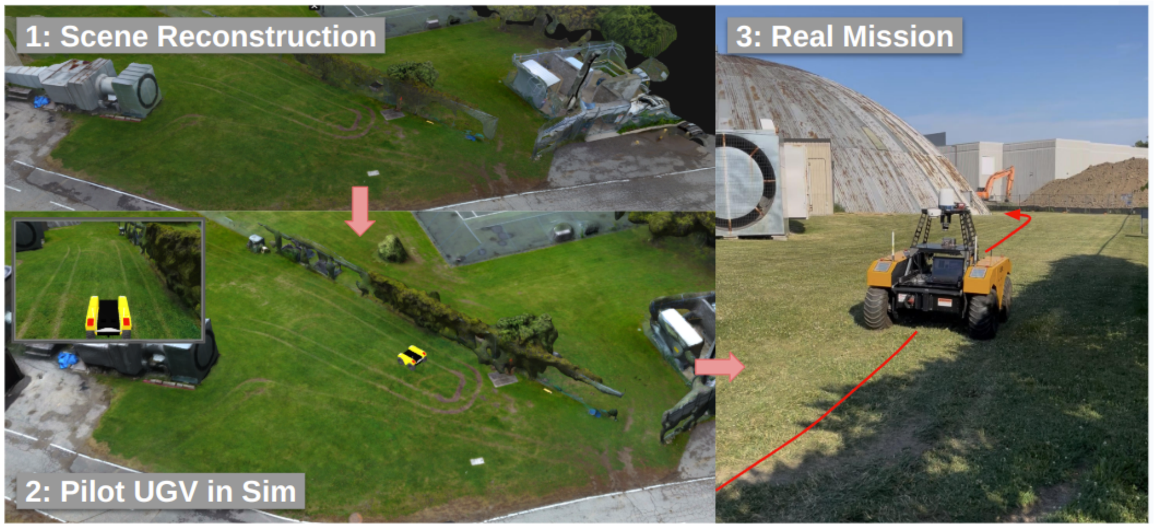


Figure 1.2: We present an architecture for navigating previously untraversed environments that uses 3D reconstruction tools to simulate the environment and create high-fidelity localization data from aerial imagery of the target environment. Shown here are the main steps of the framework, dubbed Virtual Teach and Repeat. The three steps are: 1. A scene reconstruction based on aerial imagery, a virtual driving simulation for virtual path definition, and the execution of an online mission with T&R.

We explore both learned and classical reconstruction methods to elucidate their respective strengths and limitations. This comparison further informs optimal tool selection for real-world navigation scenarios, ensuring that the most detailed and realistic data is used for reliable localization, regardless of lighting or environmental conditions, and enhances both operational safety and cost-effectiveness for unmanned missions through the use of multiple modalities and vehicle platforms.

1.2 Contributions

The primary contribution of this thesis is a new method for conducting the teach pass. Our novel architecture replaces the existing manually piloted teach pass with an offline, virtual, high-fidelity pilot simulation that allows a user to pilot their UGV through the environment to define a virtual path, which is then populated with localization data also obtained from the detailed 3D reconstruction created with the previously captured aerial imagery. This notably expands the operational functionality and usefulness of the Teach and Repeat framework in various ways by combining multiple platforms [14, 15] and bridging the gap between sensor modalities [15–17] to exploit their respective advantages. To our knowledge, this represents the first successful demonstration of zero-shot, closed-loop autonomous driving in outdoor environments using aerial scene reconstructions paired with ground-based Lidar or Radar localization. We also conducted several small studies throughout the evolution of VirT&R to verify that the most suitable tools, parameters, metrics, and evaluation methods were used.

Thus, we present the following contributions as a part of the development of the Virtual T&R framework and the methodology used to assess its performance in varied terrains, utilizing traditional Lidar and Radar T&R [9, 11] as comparative baselines:

- A novel Virtual Teach and Repeat framework, integrated with the T&R codebase[2] via a published repository[3] that provides the following new capabilities:
  a. Navigate through an environment previously untraversed by either a human or the UGV autonomously without GPS assistance;
  b. Conduct multiple teach passes in a desired environment with only one aerial survey into said environment for image capture, as opposed to manually piloting a new path in the physical environment each time a new route is to be defined;
  c. Interactively and explore the target environment in simulation from multiple perspectives ahead of virtually piloting the UGV to determine the path.
- An experimental methodology for quantitatively comparing the accuracy and repeatability of VirT&R across the sim-to-real gap with real-world benchmarks using physical markings;
- Rigorous evaluation of the proposed architecture in various environments and seasons, with Lidar and Radar sensors, that consisted of the following sub-studies:
  a. An ablation study to determine optimal hyperparameters for the learned scene reconstruction method;
  b. An analysis of the ability to assess VirT&R performance with pseudo-GPS data created for virtual teach passes;
  c. A comparison of classical and learned 3D reconstruction methods.

The remainder of this thesis is organized as follows. Chapter 2 presents a detailed background of the fields and related work relevant to this thesis. Chapters 3 and 4 describe the methodology, implementation, and evaluation methods of the VirT&R pipeline. The various experiments and results are outlined in Chapters 5 and 6. Finally, Chapter 7 presents an analysis of our findings, outlines future work, and highlights the lessons learned.

[2]https://github.com/utiasASRL/vtr3
[3]https://github.com/desifisker/virtual_teach_vtr_wrapper

1.2.1 Associated Material

Publications

Portions of the technical work and writing presented in this thesis were submitted for publication with help from co-authors; however, all written content in this work was produced independently.

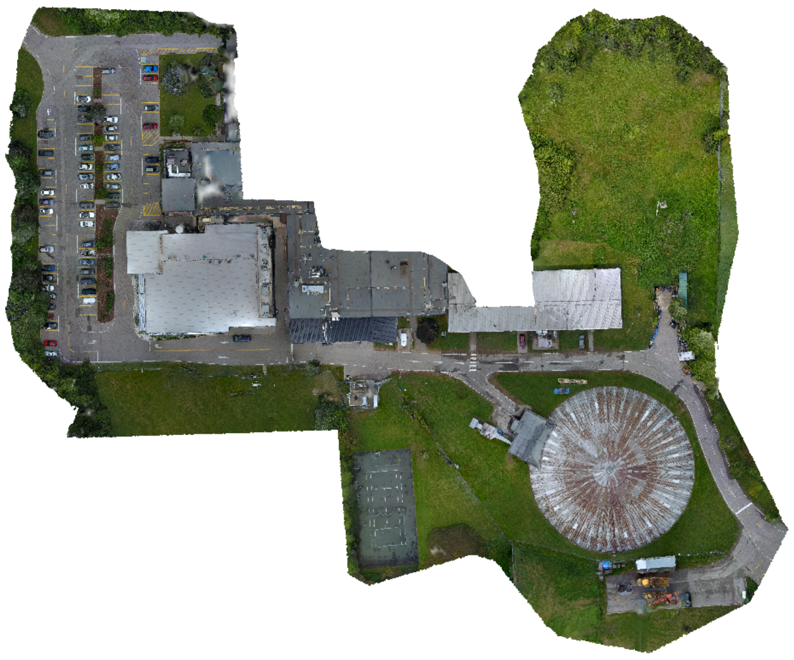
- Fisker D, Krawciw A, Lilge S, Greeff M, and Barfoot T D. "UAV See, UGV Do: Aerial Imagery and Virtual Teach Enabling Zero-Shot Ground Vehicle Repeat". In Proceedings of the IEEE/RSJ International Conference on Intelligent Robot Systems (IROS), to appear. Hangzhou, China, 19-25 October 2025.

Code

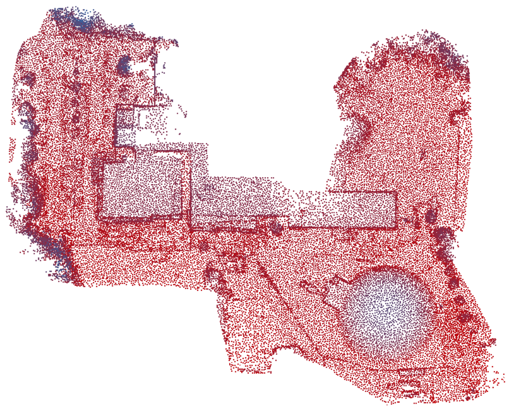
The code for the vtr_virtual_teach package that interfaces with the existing T&R codebase, along with a dockerized setup of the required supporting software and custom scripts for VirT&R, is available at https://github.com/desifisker/virtual_teach_vtr_wrapper, and the official T&R codebase is available at https://github.com/utiasASRL/vtr3.

Video

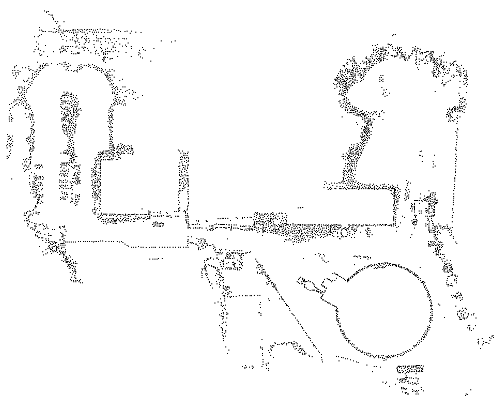
- Virtual Teach and Repeat https://www.youtube.com/watch?v=0C8iAYP3atM



(a) Photo-Textured Triangular Mesh Exported from Pix4D



(b) Virtual Lidar Point Cloud Created from Pix4D Point Cloud and Coloured by Elevation in the Z Direction



(c) Virtual Radar Point Cloud Created from Pix4D Point Cloud

Figure 1.3: Visualization of the high-fidelity photo-textured mesh, virtual Lidar point cloud, and virtual Radar scan extracted point cloud data products produced from the reconstruction step in the VirT&R pipeline that enable virtual teach map creation.

Chapter 2

Related Works

2.1 State Estimation Overview

This section introduces foundational concepts and conventions used throughout this thesis for camera pose and state estimation, point cloud alignment, and rigid-body transformation theory. These topics form the basis for our various mapping, reconstruction, and localization modules, as well as Teach and Repeat itself.

2.1.1 Three-Dimensional Geometry and Localization

Vehicles that translate and rotate have three degrees of freedom (DoFs) for rotation and three degrees of freedom for translation. Understanding these DoFs is crucial, as we deal with multiple types of moving vehicles, in addition to their sensor measurements. We refer to their six DoF configuration as a *pose*, which consists of both position and orientation. This section introduces the mathematical tools used to represent poses and transformations throughout Chapters 3, 4, 5, and 6 where we describe the VirT&R framework, explain how we obtain path-tracking error, detail our attempt at creating pseudo-GPS for evaluation, as well as the results of the pipeline. For detailed derivations, refer to Chapter 7 of State Estimation for Robotics [18].

A Lie group is a set of elements with an operation that combines any two elements to form a third in the same set. It is also a differentiable manifold where the elements in a matrix Lie group are matrices themselves. Rotations are represented using the special orthogonal group:

$$\mathrm{SO}(3) = \left\{ \mathbf{C} \in \mathbb{R}^{3\times 3} \,\middle|\, \mathbf{C}\mathbf{C}^T = \mathbf{1}, \det(\mathbf{C}) = 1 \right\}. \tag{2.1}$$

Rigid-body transformations are represented in the special Euclidean group:

$$\mathrm{SE}(3) = \left\{ \mathbf{T} = \begin{bmatrix} \mathbf{C} & \mathbf{r} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \,\middle|\, \mathbf{C} \in \mathrm{SO}(3), \mathbf{r} \in \mathbb{R}^3 \right\}. \tag{2.2}$$

Equations 2.1 and 2.2 must be true while satisfying the required group axioms: closure, associativity, identity, and invertibility.

Transformations between coordinate frames in this work follow the robotics convention, where a transformation $\mathbf{T}_{ab} \in \mathrm{SE}(3)$ transforms a homogeneous point expressed in frame $\mathcal{F}_b$ to frame $\mathcal{F}_a$ such that $\mathbf{r}_a = \mathbf{T}_{ab}\mathbf{r}_b$.

We define the following frames, commonly referred to throughout this work:

- $\mathcal{F}_{\mathrm{i}}$: The fixed inertial frame (world frame);
- $\mathcal{F}_{\mathrm{m}}$: The map frame (the pose graph map);
- $\mathcal{F}_v$: The vertex frame, a local pose within the map;
- $\mathcal{F}_r$: The robot frame, located at the robot's base;
- $\mathcal{F}_s$: The sensor frame (e.g., Lidar or Radar), rigidly mounted on the robot.

In T&R, a trajectory is encoded in a pose graph as a series of temporally ordered vertices, each associated with a timestamp and a relative transformation (an *edge*) connecting to its predecessor. These relative transforms are elements of $\mathrm{SE}(3)$ and are defined as transformations from the previous pose to the current pose ($\mathbf{T}_{ab}$), all resolved in the local map frame $\mathcal{F}_m$. A visualization of a pose graph can be seen in Figure 2.1

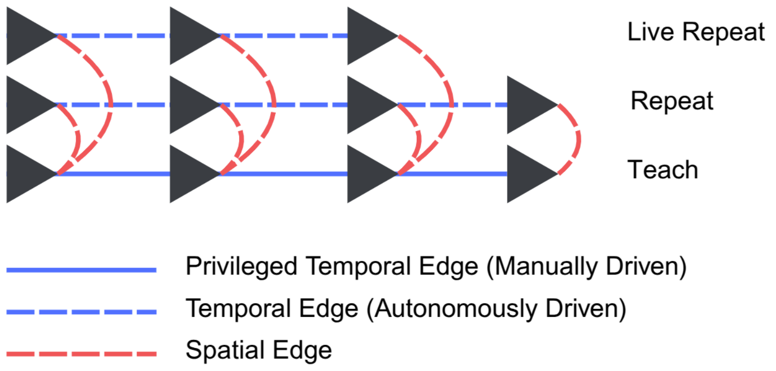

Live Repeat
Repeat
Teach
Privileged Temporal Edge (Manually Driven)
Temporal Edge (Autonomously Driven)
Spatial Edge

Figure 2.1: Visualization of the pose graph structure used in T&R. Shown here are the Privileged Edge (teach pass), a repeat Temporal Edge localizing to the Privileged Edge, and a subsequent repeat Temporal Edge in progress, also localizing to the Privileged Edge.

The pose graph in VirT&R is built from a list of time-stamped $\mathrm{SE}(3)$ transformation matrices obtained from the relative motion defined by driving a UGV model in a high-fidelity simulation of the environment as opposed to driving it in the real world as in traditional T&R. The first transformation is set to be identity to define the origin, and subsequent transformations (assumed to be perfect odometry) are used as edges to yield each vertex's pose in the map frame. These transformations are expressed as robot-to-robot transforms and can be resolved in the world frame.

At each vertex, a submap point cloud is optionally created to save data taken by onboard sensors, which creates a representation of the environment at that instant. These submaps, created during a traditional or virtual teach, are what is used by the T&R when running a repeat pass on a vehicle to perform localization. Submaps are made by transforming the virtually generated environment point cloud into the current sensor frame (where the sensor is currently located in the virtual point cloud of the environment) and taking a small cropping of it. To do this, the environment point cloud is re-based using the known starting position in the world frame such that the first vertex becomes the origin (i.e., $\mathbf{T}_{v,0} = \mathbf{I}$) in the map frame, and the remaining transformations are compounded relative to this new origin.

The submaps are then saved to the map in the local vertex frames, allowing us to maintain the topometric map representation and recover our position using that information. The full sensor-to-map transformation is computed as

$$\mathbf{T}_{s,m} = \mathbf{T}_{s,r}\mathbf{T}_{r,v}\mathbf{T}_{v,m}, \tag{2.3}$$

where:

- $\mathbf{T}_{s,r}$ is the fixed extrinsic calibration between sensor and robot base;
- $\mathbf{T}_{r,v}$ is identity, since each vertex pose is defined in the robot frame;
- $\mathbf{T}_{v,m}$ is the absolute pose of the current vertex with respect to the map.

Each vertex also stores a pointer to the most recent vertex with a submap (that is Identity if a submap is located at the vertex in question), which are created based on spatial and rotational thresholds (e.g., 1.5 m or 30° of motion). The relative transformation between the current vertex and the current submap at a prior vertex is computed as follows:

$$\mathbf{T}_{\text{submap vertex,current vertex}} = \mathbf{T}_{\text{current vertex},m}\mathbf{T}^{-1}_{\text{submap vertex},m}. \tag{2.4}$$

This relative pose is saved and used later to help initialize a live scan against the stored submap during localization. Given a set of 3D landmark observations $\{\mathbf{p}_v\}$ from a local vertex submap frame $\boldsymbol{\mathcal{F}}_v$ at time t_k, and corresponding landmarks $\{\mathbf{p}_s\}$ from a live sensor frame $\boldsymbol{\mathcal{F}}_s$ at time t_{k+1}, we seek the transformation $\mathbf{T}_{k+1,k} \in \mathrm{SE}(3)$ that best aligns them.

For both the vertex and sensor frames, we have M measurements of the set of landmark points P given in the respective frames $\mathbf{r}_v^{p_i v}$ and $\mathbf{r}_s^{p_i s}$, where $i = 1 \ldots M$. The goal is to find the rotation matrix, $\mathbf{C}_{vs}$, and translation, $\mathbf{r}_{vs}$, that will align the two sets of points. We also define w_i as the scalar weights for each point.

Following the derivation in State Estimation for Robotics [18], we can formulate the alignment as a least-squares problem over rotation $\mathbf{C} \in \mathrm{SO}(3)$ and translation $\mathbf{r} \in \mathbb{R}^3$, minimizing

$$J(\mathbf{C}, \mathbf{r}) = \frac{1}{2}\sum_{i=1}^{M} w_i \|\mathbf{y}_i - \mathbf{C}(\mathbf{p}_i - \mathbf{r})\|^2, \tag{2.5}$$

under the constraint $\mathbf{C} \in \mathrm{SO}(3)$. A closed-form solution can be obtained by computing weighted centroids, forming a cross-covariance matrix, and solving for $\mathbf{C}$ using Singular Value Decomposition (SVD). This provides a non-iterative estimate of $\mathbf{T}_{vs}$ when correspondences are known and well-behaved. A visualization of a robot and onboard sensor observing a point in the world frame (not the map frame as previously used) is shown in Figure 2.2.

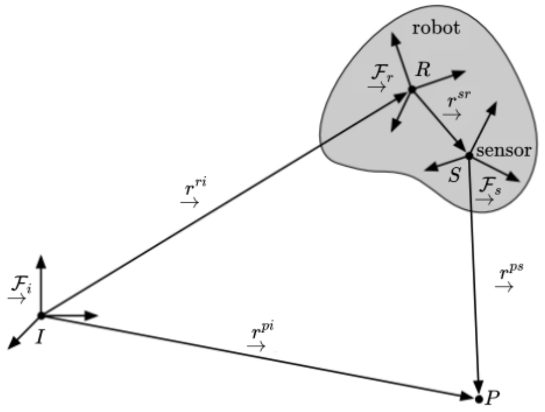

robot
$\underrightarrow{\mathcal{F}}_r$
R
$\underrightarrow{r}^{sr}$
sensor
S
$\underrightarrow{\mathcal{F}}_s$
$\underrightarrow{r}^{ri}$
$\underrightarrow{r}^{ps}$
$\underrightarrow{\mathcal{F}}_i$
I
$\underrightarrow{r}^{pi}$
P

Figure 2.2: A definition of reference frames for a moving robot with an onboard sensor observing a point, P, in the environment, which is a frame itself.

When incorporating the alignment of point clouds with differing resolutions and noisy geometric accuracy (e.g., Lidar to virtual point cloud or Radar to virtual point cloud, as done in Chapter 3), we use the point-to-plane Iterative Closest Point (ICP) formulation [19], which minimizes the distance from each source point to the tangent plane of its target, meaning the signed distance from each point in the live scan to the tangent plane of its matched neighbour in the vertex submap is minimized. This is more robust to variation in density and better respects local surface geometry than point-to-point ICP. The error term becomes

$$\mathbf{e}_{\text{loc},j} = \mathbf{n}_j^\top \left(\mathbf{T}_{vs} \cdot \mathbf{q}_j - \mathbf{p}_j\right), \tag{2.6}$$

where $\mathbf{q}_j \in \mathbb{R}^3$ is a point from the live scan (in frame $\underrightarrow{\mathcal{F}}_s$), $\mathbf{p}_j \in \mathbb{R}^3$ is the closest point in the vertex submap (in frame $\underrightarrow{\mathcal{F}}_v$), $\mathbf{n}_j$ is the surface normal at $\mathbf{p}_j$, and $\mathbf{T}_{vs} \cdot \mathbf{q}_j$ is the live point transformed into the vertex frame.

The corresponding cost function is

$$J_{\text{loc}}(\mathbf{T}_{vs}) = \sum_j \left[\mathbf{n}_j^\top \left(\mathbf{T}_{vs} \cdot \mathbf{q}_j - \mathbf{p}_j\right)\right]^2. \tag{2.7}$$

As this cost function is nonlinear in $\mathbf{T}_{vs}$, it is solved iteratively. At each iteration, we linearize the transform using the Lie group SE(3), which allows us to update the estimate in a structure-preserving way:

$$\mathbf{T}_{vs}^{(k+1)} = \exp\big(\boldsymbol{\xi}^\wedge\big) \cdot \mathbf{T}_{vs}^{(k)}, \tag{2.8}$$

where $\boldsymbol{\xi} \in \mathbb{R}^6$ is a twist vector (3 for rotation, 3 for translation), and $(\cdot^\wedge)$ maps the vector into a matrix in $\mathfrak{se}(3)$, and $\exp(\cdot)$ maps it to $\mathrm{SE}(3)$ using the matrix exponential.

Conversely, the Lie Algebra, $\mathfrak{se}(3)$, is mapped to using the matrix logarithm, $\log(\cdot)$, from $\mathrm{SE}(3)$. For any transform $\mathbf{T} \in \mathrm{SE}(3)$, $\log(\mathbf{T}) \in \mathfrak{se}(3)$ is a $4{\times}4$ matrix that encodes the corresponding twist vector; the operator $(\cdot)^{\vee} : \mathfrak{se}(3) \to \mathbb{R}^6$ then converts this matrix to the vector representation $\boldsymbol{\xi} = \begin{bmatrix}\boldsymbol{\phi} \ \mathbf{t}\end{bmatrix}^{\top}$, where $\boldsymbol{\phi} \in \mathbb{R}^3$ is the rotation vector and $\mathbf{t} \in \mathbb{R}^3$ is the translation vector.

To compute $\boldsymbol{\xi}$, we derive the first-order Taylor expansion of the residuals and solve the normal equations:

$$\mathbf{A}\boldsymbol{\xi} = \mathbf{b}, \tag{2.9}$$

where $\mathbf{A}$ is the Jacobian of the residuals, and $\mathbf{b}$ is the residual vector. This linear system is solved at each iteration until convergence.

The ICP algorithm alternates between:

1. Correspondence assignment: finding the closest point and normal for each $\mathbf{q}_j$.

2. Transform update: solving for $\boldsymbol{\xi}$ and updating $\mathbf{T}_{vs}$.

This process repeats until the update magnitude $\|\boldsymbol{\xi}\|$ falls below a set threshold.

After convergence, the final transformation $\hat{\mathbf{T}}_{vs}$—consisting of rotation and translation—aligns the live scan to the submap under the point-to-plane metric. In T&R, this transformation is used as the relative pose estimate between the sensor frame $\underrightarrow{\mathcal{F}}_s$ and the localization vertex frame $\underrightarrow{\mathcal{F}}_v$.

The optimization is initialized with a prior estimate $\check{\mathbf{T}}_{vs}$ and regularized with a prior term:

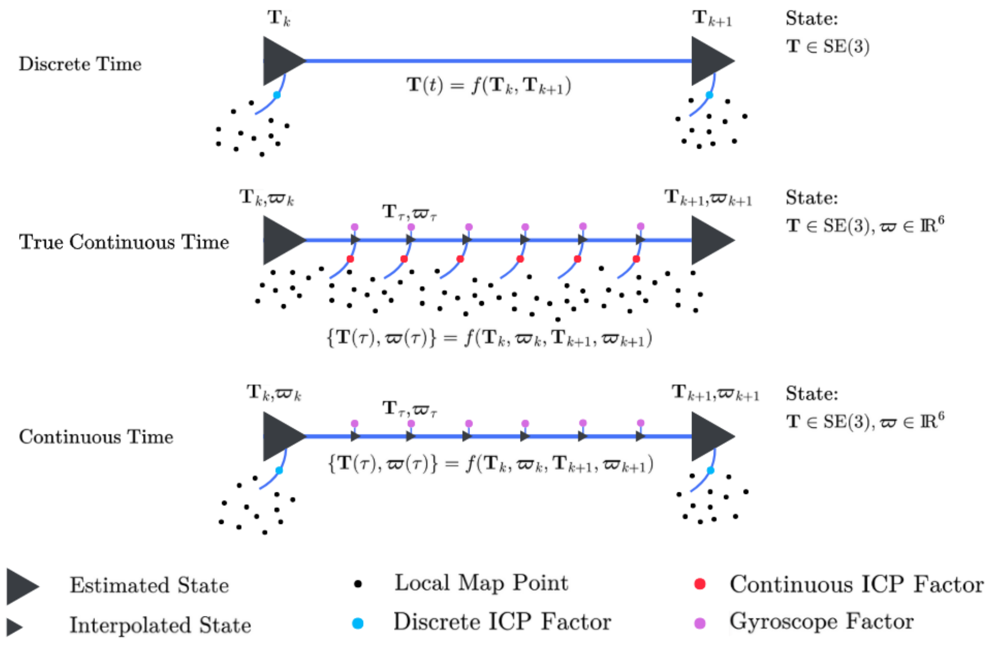
$$\phi_{\text{pose}} = \left\| \log\left(\check{\mathbf{T}}_{vs}^{-1}\hat{\mathbf{T}}_{vs}\right)^{\vee} \right\|^2_{\mathbf{Q}_{vs}}, \tag{2.10}$$

where $\mathbf{Q}_{vs}$ is the prior covariance computed by compounding the uncertainties of intermediate edges. Since all point clouds are motion-compensated, temporal distortion is not a concern in the residuals.

It should be noted that, while this description of the pose graph and localization is helpful, there are many variants of T&R that differ slightly in their exact formulation. The above logic applies to Lidar T&R (LT&R) as of March 2025. Part of this thesis involved switching to a newly developed, upgraded version of LT&R for our second experiment, which changed the formulation from discrete time ICP to continuous time ICP and incorporated single-axis gyro measurements to provide a high-rate yaw prior to keep heading drift below 0.5° between updates as well. Also of note, in the continuous ICP formulation, we assume that the points in the Lidar scan are still collected discretely, but we utilize the higher-frequency gyroscope measurements in a continuous manner. This is done for computational reasons, as the many thousands of points in a single Lidar scan make the problem intractable for real-time operation in our case.

In the upgraded LT&R pipeline, the goal of finding $\mathbf{T}_{k+1,k} \in \mathrm{SE}(3)$ remains the same, but incorporates more information. We grow the state, previously just $\mathbf{T} \in \mathrm{SE}(3)$ (which is $\boldsymbol{\xi} \in \mathbb{R}^6$) to also include the velocities $\boldsymbol{\varpi} \in \mathbb{R}^6$, effectively creating an $\mathbb{R}^{12}$ state. Now instead of finding $\mathbf{T}_{k+1,k}$, we obtain the global pose and velocity at the interpoltated time, τ, which makes the state become $\{\mathbf{T}(\tau), \boldsymbol{\varpi}(\tau)\} = f(\mathbf{T}_k, \boldsymbol{\varpi}_k, \mathbf{T}_{k+1}, \boldsymbol{\varpi}_{k+1})$ to incorporate the higher frequency gyroscope measurements continuously. The Lidar points are assumed to be collected in quantized semi-frequent packets, de-skewed by the time-stamped angular velocities added to the state. The point-to-plane ICP error is then minimized with an analytically propagated Gaussian Process motion prior, delivering continuous-time $\mathrm{SE}(3)$ estimates that seed the local map builder [9].

This formulation is more akin to how Radar T&R (RT&R) functions (but with Doppler vectors injected into the ICP Jacobian to correct for Radar's radial velocity bias [9]), as there are far fewer points extracted from a Radar polar image, meaning RT&R has the compute to consider them continuously. A visualization of the difference is shown below in Figure 2.3.



$\mathbf{T}_k$
$\mathbf{T}_{k+1}$
State:
$\mathbf{T} \in \mathrm{SE}(3)$
Discrete Time
$\mathbf{T}(t) = f(\mathbf{T}_k, \mathbf{T}_{k+1})$
$\mathbf{T}_k, \boldsymbol{\varpi}_k$
$\mathbf{T}_\tau, \boldsymbol{\varpi}_\tau$
$\mathbf{T}_{k+1}, \boldsymbol{\varpi}_{k+1}$
State:
$\mathbf{T} \in \mathrm{SE}(3), \boldsymbol{\varpi} \in \mathbb{R}^6$
True Continuous Time
$\{\mathbf{T}(\tau), \boldsymbol{\varpi}(\tau)\} = f(\mathbf{T}_k, \boldsymbol{\varpi}_k, \mathbf{T}_{k+1}, \boldsymbol{\varpi}_{k+1})$
$\mathbf{T}_k, \boldsymbol{\varpi}_k$
$\mathbf{T}_\tau, \boldsymbol{\varpi}_\tau$
$\mathbf{T}_{k+1}, \boldsymbol{\varpi}_{k+1}$
State:
$\mathbf{T} \in \mathrm{SE}(3), \boldsymbol{\varpi} \in \mathbb{R}^6$
Continuous Time
$\{\mathbf{T}(\tau), \boldsymbol{\varpi}(\tau)\} = f(\mathbf{T}_k, \boldsymbol{\varpi}_k, \mathbf{T}_{k+1}, \boldsymbol{\varpi}_{k+1})$
Estimated State
Interpolated State
Local Map Point
Discrete ICP Factor
Continuous ICP Factor
Gyroscope Factor

Figure 2.3: This visualization shows the formulation of our discrete and continuous time optimization frameworks. Grey triangles represent the state at a timestep, blue and red dots represent discrete or continuous Lidar measurements, and purple dots represent high-frequency gyroscope measurements.

In the first row, for discrete time, we see local map points from a Lidar scan accumulated at times t_k and t_{k+1}, indicating that we assume all points are collected at the same discrete timestep and contribute a single ICP factor. In the second row, for true continuous time, we see a continuous accumulation of local map points and ICP factors across interpolated timesteps, indicating we consider the Lidar points as the sensor reads them, as well as their velocities, which are used for motion compensation. The purple measurements, also placed at interpolated states, are high-frequency gyroscope measurements. This method is what we use for RT&R (except using points extracted from Radar scans, along with gyroscope measurements, instead of Lidar points). The continuous time formulation in the third row is what we use for LT&R, which is less computationally intensive, as it only considers the gyroscope measurements continuously, and assumes the Lidar points are gathered discretely, as in the first row.

Similarly to before the upgrade, repeat localization uses discrete ICP for LT&R and continuous

ICP for RT&R to carry out the live scan-to-submap alignment. A localization registration is declared valid when the inlier ratio exceeds a certain percentage and the posterior lateral covariance is under a set translation threshold; otherwise, the vehicle continues to dead-reckon until either a valid match is found or a set drift budget is exhausted.

Overall, this formulation provides accurate and robust alignment for localization, even when the point clouds originate from different sensing modalities or exhibit resolution and quality mismatches, as is typical in field robotics applications.

2.1.2 Scene Reconstruction

In this section, we aim to describe how photogrammetry tools, such as COLMAP and Pix4D, recover accurate geometric representations of real-world scenes from a collection of images, which is crucial for the pipeline described in Chapter 3. At a high level, this is achieved by estimating camera poses (positions and orientations) alongside scene structure through bundle adjustment. The optimization problem simultaneously refines the 3D coordinates of feature points and the corresponding camera parameters by minimizing reprojection errors.

Various camera models exist, each with different benefits and use cases, the most notable being the stereo and monocular configurations. While stereo camera models utilize two spatially offset cameras to compute disparity to obtain depth, monocular models infer depth indirectly, requiring careful geometric modelling and triangulation. The majority of monocular photogrammetry applications rely heavily on pinhole camera models, specifically the simple pinhole model, due to its computational efficiency and ease of integration.

Along with these models, various conventions exist across the state estimation, robotics, and computer vision literature. This section will elucidate how they are similar and which we will be using. We begin with a perspective camera using a frontal projection model (simple pinhole), as defined in Chapter 6 of State Estimation for Robotics [18] and seen in Figure 2.4.

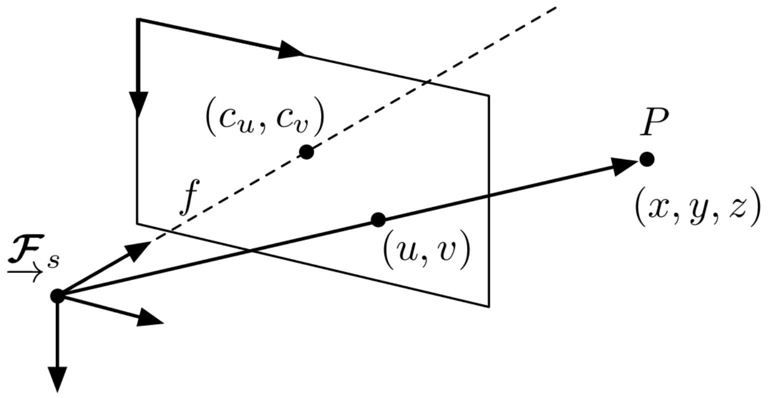


Figure 2.4: Illustrated is a perspective camera using a frontal projection model from State Estimation for Robotics [18]. Realistically, the image plane is behind the camera and the frontal projection model is used to avoid working with a flipped image.

Camera images can be distorted by lens effects, so we assume that undistortion (or calibration) has been applied to ensure images appear as if they had come from an idealized pinhole camera. Given a point in the camera sensor frame, $\boldsymbol{\mathcal{F}}_s$, with coordinates, $\mathbf{p}_s = \begin{bmatrix} x & y & z \end{bmatrix}^T$, it can be projected into image frame coordinates with the perspective camera model, $\mathbf{s} : \mathbb{R}^3 \to \mathbb{R}^2$, to obtain $\mathbf{q} = \begin{bmatrix} u & v \end{bmatrix}^T$ as follows:

$$\mathbf{q} = \begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{s}(\mathbf{p}) = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{\mathbf{P}} \underbrace{\begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{2.11}$$

Here, $\mathbf{P}$ represents the projection matrix and the $\mathbf{K}$ matrix holds the intrinsic camera parameters of focal lengths, f_u and f_v, and optical centre coordinates, (c_u, c_v).

The frontal projection model illustrates the image plane in front of the camera to avoid flipped image coordinates during derivations. Thus, mapping a 3D point, $\mathbf{p}_s = \begin{bmatrix} x & y & z \end{bmatrix}^T$, in the camera sensor frame into a 2D point, $\mathbf{q} = \begin{bmatrix} u, v \end{bmatrix}^T$, in the image frame is described by the same projection function:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \propto \mathbf{K} \cdot [\mathbf{R} \,|\, \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix},$$

where the proportionality accounts for the normalization in the z dimension. Scene reconstruction tools such as COLMAP and Pix4D typically rely on this simple pinhole camera model where intrinsics are stored and refined alongside pose estimates during bundle adjustment as visualized in Figure 2.5 adapted from State Estimation for Robotics [18].

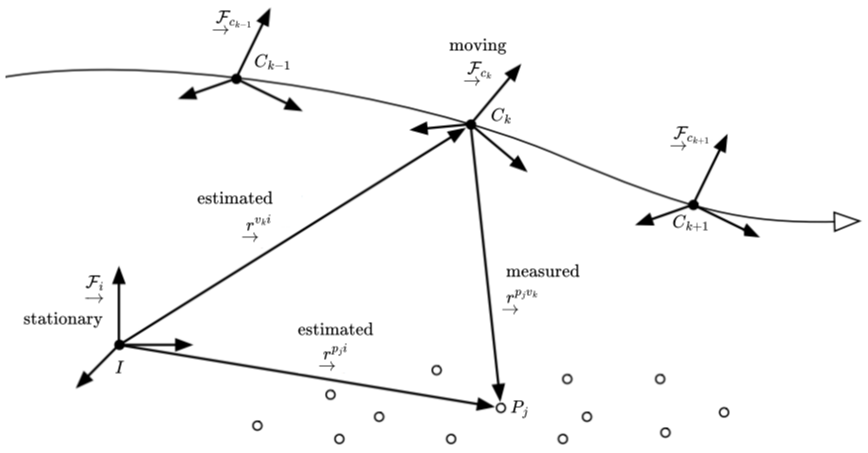

$\underset{\rightarrow}{\mathcal{F}}_{c_{k-1}}$
C_{k-1}
moving
$\underset{\rightarrow}{\mathcal{F}}_{c_k}$
C_k
$\underset{\rightarrow}{\mathcal{F}}_{c_{k+1}}$
C_{k+1}
estimated
$\underset{\rightarrow}{r}^{v_k i}$
measured
$\underset{\rightarrow}{r}^{p_j v_k}$
$\underset{\rightarrow}{\mathcal{F}}_i$
stationary
estimated
$\underset{\rightarrow}{r}^{p_j i}$
I
P_j

Figure 2.5: A definition of reference frames in a bundle adjustment problem where moving frames showcase how a camera could move through a scene, taking images to capture it. Relative poses of the moving camera frames are estimated with respect to the stationary frame, using observations like point P, in the world frame.

The rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ together form the extrinsic parameters that describe a camera pose, represented compactly as an element of the Special Euclidean group $\mathrm{SE}(3)$, described earlier, which defines rigid body transformations in 3D:

$$\mathbf{T}_{is} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \in \mathrm{SE}(3), \tag{2.12}$$

where $\mathbf{T}_{is}$ maps a homogeneous point ($\tilde{\mathbf{p}}_s = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$) from the camera sensor coordinate frame to the image coordinate frame. Thus, using homogeneous coordinates, the image frame point $\mathbf{p}_i$ is:

$$\mathbf{p}_i = \mathbf{R}\mathbf{P}_s + \mathbf{t} = \mathbf{T}_{is} \cdot \mathbf{P}_s^{\text{hom}}. \tag{2.13}$$

The projection of $\mathbf{p}_s = \begin{bmatrix} x & y & z \end{bmatrix}^T$ onto the image plane gives normalized image coordinates for a focal length of 1:

$$\mathbf{x}_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x/z \\ y/z \end{bmatrix}, \tag{2.14}$$

which are then converted to pixel coordinates through the intrinsic matrix:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix}. \tag{2.15}$$

Therefore, for any known 3D camera sensor frame point $\mathbf{P}_s$, the corresponding 2D projection in an image captured from camera pose $\mathbf{T}_{is}$ is obtained through this chain of transformations. Conversely, to recover the position of a camera $\mathbf{p}_s$ observing many 2D points, photogrammetry systems solve for the $\mathbf{T}_{si}$ that best reprojects known or estimated 3D points $\mathbf{P}_i^s$ to their 2D image locations, minimizing the reprojection error:

$$E = \sum_j \left\| \mathbf{p}_s^j - \pi(\mathbf{T}_{si} \cdot \mathbf{P}_i^j) \right\|^2, \tag{2.16}$$

where $\pi(\cdot)$ denotes the camera projection function defined by the intrinsic matrix.

Photogrammetry techniques estimate parameters iteratively through feature correspondence, triangulation, pose graph optimization, and bundle adjustment. Accurate camera pose estimation is an essential component of photogrammetry and 3D mapping, as well as downstream path planning and navigation, meaning the process described here is fundamental for autonomous robot operation.

2.2 Overview of Teach and Repeat

Teach and Repeat is a navigation framework that allows robots to autonomously repeat paths previously traversed under manual control, with applications spanning transportation, mining, and planetary exploration, where navigating predefined paths significantly reduces complexity and improves safety. T&R achieves GPS-denied navigation through an environment by making use of sensors such as accelerometers, gyroscopes, cameras, Inertial Measurement Units (IMUs), Lidar, and Radar sensors to estimate the position and motion of the vehicle. One of the unique benefits of T&R is the emphasis on using a modular local topometric map representation, which avoids unbounded drift without requiring global consistency and provides efficient localization through the use of a small window of point clouds tied to odometry, thereby reducing the computational load. This flexibility has enabled deployments across different platforms, including Unmanned Aerial Vehicles (UAVs) [12, 13]. However, cross-modal T&R (e.g., Radar-to-Lidar) has only recently been explored [9, 20] and still requires manual teaching in the target environment.

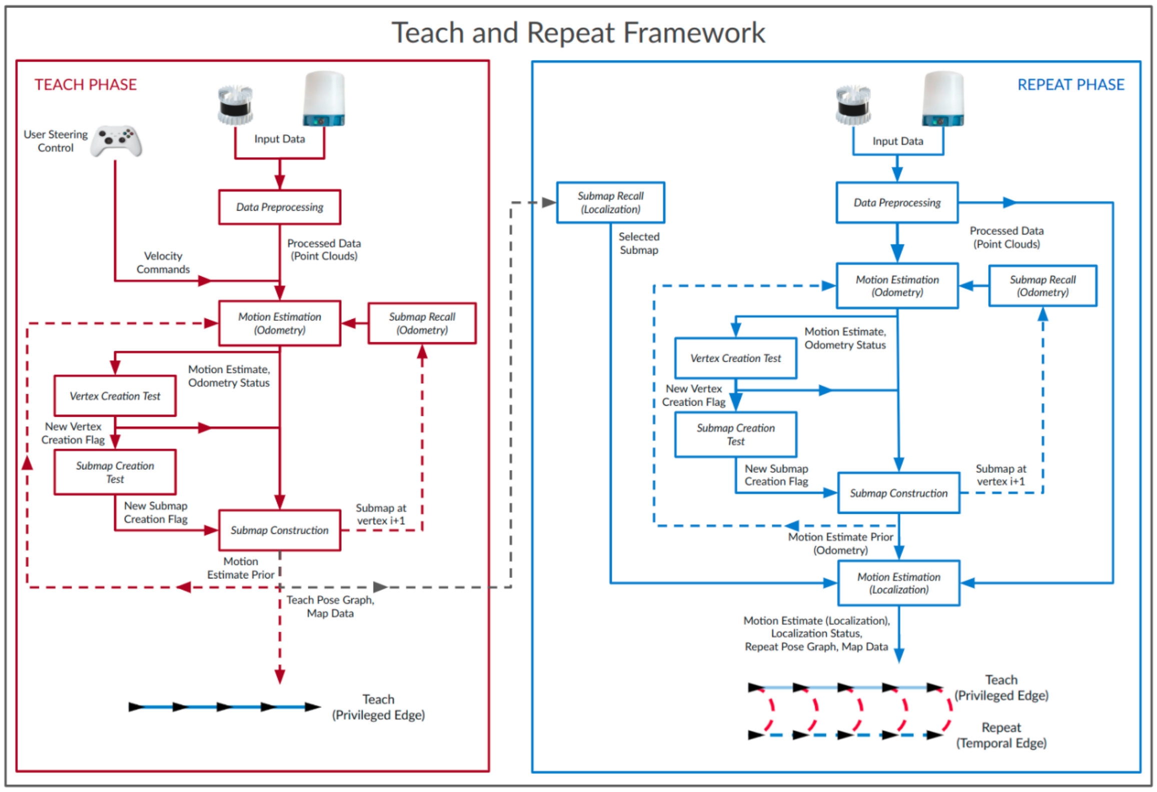
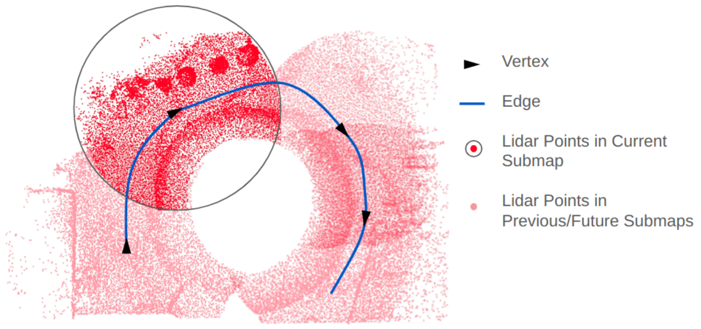


Figure 2.6: This block diagram shows how the Teach Phase and Repeat Phase are connected in the Teach and Repeat Framework. During both phases, an odometry module is run to track UGV motion, and in the Repeat Phase, this is supplemented by a localization module that compares the submaps of the Pivileged Edge to what the UGV is currently observing.

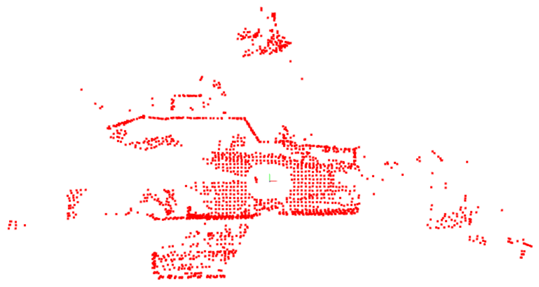
2.2.1 Topological Local Map Representation

A core insight behind Teach and Repeat is that all sensing pipelines—stereo vision, Lidar, Radar, or any future sensor capable of providing a data-rich representation of the world—share the same underlying map structure. By decoupling sensing from memory, T&R can combine new and diverse front-end algorithms while maintaining a single, robust navigation backbone—a characteristic we exploit in this thesis.

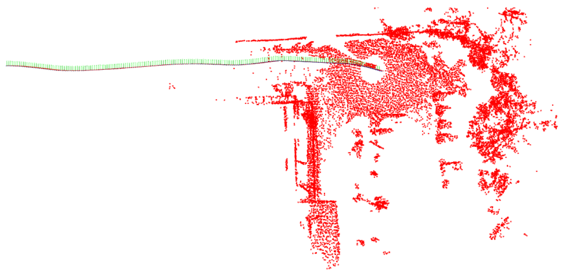
The T&R map stores routes through an environment as relative $\mathrm{SE}(3)$ transformations in a pose graph (see Chapter 2.1), as can be seen illustrated in Figure 2.7a. The local topometric map philosophy used to represent the environment asserts that each vertex in a map defines a local frame, which represents the robot's pose along the path at a given moment, and is associated with a local metric submap of the environment. Edges are the encoded $\mathrm{SE}(3)$ transformations (with optional covariance) between successive vertex frames in a map. Spatial edges connect spatially close frames (from localization), and temporal edges connect time-adjacent frames (from odometry). Sensor data from local vertex frames is aggregated into compact local vertex submaps during traversal, that live in the map frame $\mathcal{F}_m$, where each vertex keeps a rigid transform $\mathbf{T}_{mv}$ to its closest local submap, allowing live scans to be registered in a bounded, high-overlap workspace instead of the entire graph. Figure 2.7b and 2.7c show a stored Lidar scan at the beginning of a path in a pose graph and then later on in the pose graph as well.


Vertex
Edge
Lidar Points in Current Submap
Lidar Points in Previous/Future Submaps

(a) Visualization of Submaps Centred at Vertices in a Pose Graph



(b) Submap at the Beginning of a Route



(c) Submap Further Along in a Route

Figure 2.7: A visualization of what a pose graph map looks like. (a) Submaps that represent the environment centred at vertices are connected by temporal edges that create the route. Sequential submaps stored at vertices in a teach map pose graph connected by edges. (b) A real Ouster OS-128 beam Lidar sensor scan stored at the first vertex in a pose graph map, which is represented by a small orthogonal axis. (c) A subsequent real Lidar sensor scan saved at a vertex further along the path in the pose graph map, where the path can be seen trailing behind the scan represented by many small orthogonal axes.

Additionally, because the graph is made of local vertex frames, it stays numerically stable over kilometre-scale missions and does not require a globally consistent frame, while preserving bounded drift, path-network scalability, and real-time repeatability. The submaps and relative edges also help control memory usage and numerical conditioning. All teach passes in an environment or network collectively form what is known as the *privileged edge*—the authoritative network of traversable paths used as the reference for planning and localization. Repeat passes mirror the pose graph construction behaviour of linking sequential vertices with edges and embedding sensor data as submaps during online operation. However, the repeat passes use the established privileged edge from the teach map to localize and then follow the route through the environment, thereby growing a non-privileged chain while tracking along the privileged path [11].

2.2.2 Teach Phase

Autonomous operation with T&R always begins with a teach pass in which a human operator manually pilots the UGV along a desired route in an environment. The pilot's goal is to expose the vehicle to every viewpoint it will see again, so sharp turns, narrow corridors, or areas prone to future obstruction are best avoided or treated with extra care. If there are any large environmental changes in the future that impede the taught path, such as construction, parked cars, snowbanks, or large puddles, the path will not be able to be followed, and a new one will need to be taught manually again.

While the vehicle traverses the environment, raw exteroceptive measurements stream into an odometry module that delivers continuous $\mathrm{SE}(3)$ motion estimates. Odometry, in general, refers to any system, sensor, or technique used to estimate the position of the vehicle. Unlike localization, which measures relative motion against map features, odometry predicts the vehicle's position based on its motion over time. Having initialized the vehicle's pose in the x-y plane, sensors can then be used to estimate the vehicle's motion with respect to its initial position.

Every new odometry update is integrated into the pose graph that forms the topometric local privileged map. When the accumulated translation or rotation between the most recent vertex and the live pose exceeds configurable thresholds, or when localization quality indicators drop below set bounds, the system creates a new vertex. Sensor data gathered over a short temporal window around that vertex then gets combined into the associated local submap. Multiple consecutive vertices can share the same local map, using pointers from current vertices to the last vertex that has an associated submap when a separate set of configurable thresholds are not exceeded, so the storage footprint scales with environment complexity rather than with odometry sample rate [21]. If the operator drives past a location that already lies in the pose graph, the localization module attempts to register the live local scan to the nearest stored map. Successful alignment creates a spatial edge that ties the new trajectory back into the existing network, closing the loop.

Thus, the output of the teach pass is the privileged pose graph map, whose vertices encode the vehicle's manually piloted path and include local submaps that encode the geometry or appearance needed for future localization. Crucially, because each vertex carries the transform to the local map frame, subsequent repeat passes can register live sensor data locally as well, even when the global environment is vast or GPS-denied.

2.2.3 Repeat Phase

A repeat pass begins by loading the privileged pose graph map produced in an earlier teach. The operator manually pilots the vehicle to a physical location in the environment that corresponds to any vertex on the desired route—this need not be the original start position—and initializes the T&R user interface with the map, an approximate starting vertex on the route, and one or more goal vertices. Once the repeat is initialized, the vehicle assumes control, and the autonomy stack proceeds through three tightly coupled threads: global path planning, metric localization, and model-predictive tracking control [21, 22].

The planner performs a graph search across all privileged vertices and spatial edges, selecting the shortest traversable sequence regardless of the direction in which it was first taught. The leading vertex in that sequence defines the initial local submap against which live sensor data is registered. Localization then operates in a prediction-correction loop. High-rate odometry propagates the pose estimate, and at a lower rate, the localization module aligns the most recent live sensor input with the closest stored submap using a sensor-appropriate registration method. Successful alignment yields an $\mathrm{SE}(3)$ correction that resets the odometry drift and is immediately fused into the pose graph as a non-privileged vertex, creating the repeat path within the pose graph map without altering the privileged backbone.

The corrected pose is projected onto $\mathrm{SE}(2)$ for a real-time Model Predictive Controller (MPC) that tracks the graph waypoint stream while respecting the set velocity, curvature, and corridor constraints [11]. If localization quality degrades, odometry continues with dead-reckoning for a fixed distance while repeated attempts are made to recover. A successful realignment resumes normal operation, whereas an exhausted budget triggers a controlled halt that awaits pilot intervention or a restart.

2.2.4 Supported Sensor Modalities

T&R has been extended to various sensors, including RGB vision, Lidar, and Radar. The modular design of T&R supports flexible combinations of estimation and mapping strategies across sensor modalities. Lidar implementations enhance robustness under variable lighting conditions, while Radar increases reliability in snow or dense vegetation. The current T&R framework supports Lidar, Radar, and RGB stereo vision sensors, along with integrated IMU or gyroscope measurements in its official codebase. We will focus on describing the details of the existing Lidar and Radar T&R modalities, as they are the most actively used sensors and are the most pertinent to this thesis.

The exteroceptive modalities of an Ouster OS-1 128 beam Lidar sensor [16] and a Navtech Frequency Modulated Continuous Wave (FMCW) Radar sensor [17] can also be supplemented with a single–axis (yaw-rate) gyroscope that is embedded in the Ouster Lidar. Although the pose graph map structure and the MPC are sensor-agnostic, the localization and odometry modules differ in how motion is extracted and scan-to-map registration is performed during teach and repeat passes, given the different world representations provided by each sensor.

A Lidar sensor emits rapid laser pulses and measures each pulse's round-trip time-of-flight to infer range. Knowing the instantaneous azimuth ϕ and elevation θ of the rotating laser, the range r

converts directly to Cartesian coordinates,

$$\mathbf{p}_i = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r\cos\theta\cos\phi \\ r\cos\theta\sin\phi \\ r\sin\theta \end{bmatrix},$$

producing a dense, metrically accurate snapshot of the surrounding surfaces in the form of a detailed 3D point cloud.

When using RT&R, the Radar sensor first forms a polar range–azimuth image, in which each pixel at index (r, α) records the reflected power from range bin r and bearing α. After clutter suppression and peak detection (with algorithms such as CFAR [23], K-Strongest [24], or CFEAR [25]), significant returns are retained as 2-D targets. These detections convert into Cartesian ground-plane coordinates via

$$x = r\cos\alpha, \qquad y = r\sin\alpha, \qquad z \approx 0,$$

where the elevation component is typically neglected, as UGV Radars have limited vertical resolution. Doppler processing enables each point to also carry a radial velocity estimate, resulting in a more sparse, motion-aware 2D point cloud that remains robust in environments with dust, snow, or fog.

Notably, Lisus and Burnett [20] combined Lidar and Radar modalities to present the first cross-modal localization in the T&R framework, which enables a privileged map taught with Lidar to be repeated by the vehicle using a Radar sensor. T&R handles this by retaining Lidar submaps from a teach pass and aligning the live Radar target cloud directly to them with a generalized ICP that models differing point densities. This implementation incorporates learned per-point weighting masks predicted by a U-Net architecture from the incoming polar Radar image, which down-weights noise while highlighting stable man-made structures. After weighting, a point-to-plane objective identical to the standard Lidar case is solved.

The work in this thesis becomes the second T&R variant to tackle cross-modality, doing so indirectly to facilitate the inclusion of sensors with vastly different data outputs. Instead of working with two geometric point-cloud sensors, we begin with purely visual data to create a reconstruction of the world, which enables us to obtain a geometric point cloud localization layer and mesh. Lidar and Radar-like submaps are created from the virtually generated point cloud so that, during the repeat phase, the UGV can localize its real-time Lidar or Radar live scan directly against this synthetic geometry (described further in Chapter 3.4). This effectively closes the loop from pixels to range returns, enabling cross-modality between three different sensors.

2.3 Photogrammetry

Photogrammetry is the process of creating a 3D reconstruction from overlapping 2D images, usually by following a two-stage structure-from-motion (SfM) to multi-view stereo (MVS) workflow [26–28]. SfM detects and matches local features (e.g., SIFT[29], SuperPoint[30]) across images to estimate camera intrinsics and poses [31]. Well-established open-source and commercial tools all implement variations of this paradigm, differing mainly in front-end feature extraction, pose optimization, dense matching strategy, and export formats. Common data products produced from photogrammetry pipelines include comprehensive statistical information on the processing, an orthomosaic (usually geo-referenced if that information is provided), a dense RGB point cloud, and a detailed high-fidelity mesh of the captured environment.

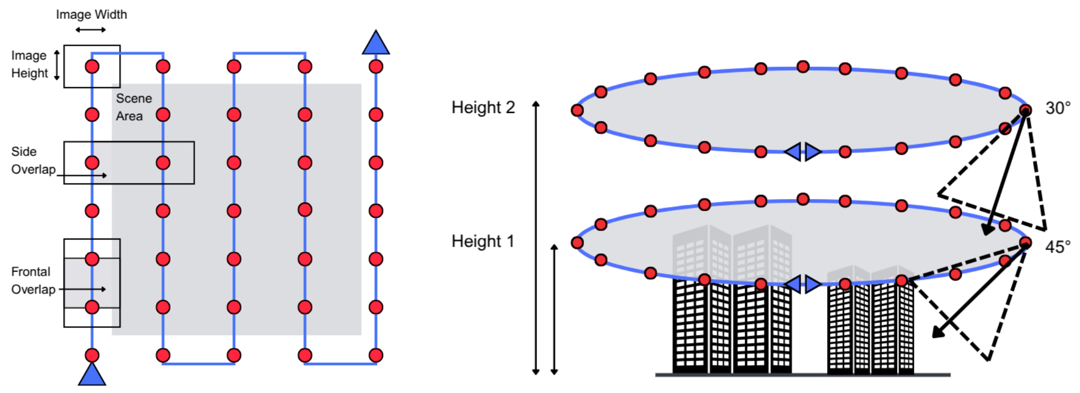
One such tool is COLMAP, a classical photogrammetry pipeline that transforms a collection of overlapping RGB images into a 3D model. It first performs structure-from-motion (SfM), detecting and matching local features across images to estimate camera intrinsics, extrinsics, and a sparse 3D point cloud [31]. It then applies multi-view stereo (MVS) to compute depth maps, which are fused with the reconstruction, leveraging pixel-wise view selection to improve accuracy [28, 32]. This tool is relied on for camera pose estimation priors in several other world reconstruction tools like Neural Radiance Field (NeRF) models described in Chapter 2.4.1, such as Instant-NGP and Nerfstudio [33, 34]. By default, the COLMAP reconstruction is rooted in an arbitrary Euclidean reference frame with unknown scale and orientation; absolute scale can be resolved later by introducing ground-control points (GCPs) or by optimizing against provided image GPS poses.

Polycam is a commercial photogrammetry tool that uses a sequence of images or video frames of a scene in an SfM-to-MVS pipeline to produce a dense point cloud, followed by surface reconstruction and texture mapping to produce a detailed textured mesh [35]. When image EXIF metadata contains GPS information, Polycam propagates those pose priors through bundle adjustment, yielding an automatically scaled and approximately geo-referenced reconstruction without additional user input.

Another well-known, commercial photogrammetry suite is Pix4D [36]. This software is designed to provide industry-quality mapping and inspection of scenes it reconstructs, utilizing sequential monocular imagery, which is most often captured by a UAV. After automated feature detection and bundle adjustment, Pix4D performs hybrid depth-map fusion and volumetric meshing, with built-in tools for geo-referencing, GCP adjustment, and radiometric calibration [36]. Its emphasis on rigorous camera calibration and scalable cloud processing has made it an industry standard for engineering-quality orthomosaics and Digital Surface Models (DSMs). If per-image GPS tags are present, Pix4D directly produces metric-scaled, geo-aligned outputs—refined further when precise GCPs are available—making it a turnkey option for aerial survey workflows.

All three pipelines share the SfM to MVS backbone but occupy different niches. Polycam targets rapid on-device captures with minimal user intervention, Pix4D delivers survey-grade mapping with extensive geo-processing utilities, and COLMAP provides camera pose estimates used to initialize NeRF training (but cannot produce a sufficiently detailed point cloud and mesh itself). Selecting among them depends on the scene scale, desired accuracy, and downstream integration—in our case, COLMAP's explicit camera pose export provides ground-truth initialization for downstream NeRF optimization.

Regardless of the chosen software, there are some accepted best practices in the field of photogrammetry that dictate specific patterns of aerial flights to capture data, depending on the type of environment to be reconstructed. As well, a high percentage of overlap between subsequently captured images is required. In general, it is sufficient to have 75% frontal overlap (with respect to flight direction) and at least 60% side overlap (between flight lines) when flying a rectangular grid pattern at a constant height for scenes populated with distinct geometry (suburban areas or varied landscapes) [37]. With particularly challenging scenes that include mostly forested, grassy, snowy, or other complex objects, such as tall towers, a circular ‘orbit’ pattern is recommended [37]. This orbit should be performed at multiple altitudes with a varying camera pitch that becomes more nadir-oriented at higher altitudes and more oblique at lower altitudes. A higher overlap of at least 85% is also recommended in addition to ensuring that any vegetation in the scene is not moving or swaying due to wind. Figure 2.8 shows depictions of these patterns.



(a) Rectangular Grid Pattern

(b) Circular ‘Orbit’ Pattern

Figure 2.8: Visualizations of two different common photogrammetry flight patterns from recommendations by Pix4D [37]. (a) A rectangular grid pattern is flown at a constant height for scenes populated with distinct geometry. (b) A circular pattern is flown at multiple heights and different camera angles, and is better suited for more challenging scenes.

Although not required, the use of GCP markers placed throughout the survey region can significantly improve the geo-referencing of the output point cloud and mesh. As visible points with known GPS locations, GCPs enhance the accuracy of airborne photogrammetry by associating identifiable visual references with precise coordinates. If GPS is not available for GCPs or for images taken from a drone, photogrammetry tools default to producing results that are arbitrarily scaled.

Successful reconstruction demands careful image capture tailored to a given method’s strengths and limitations. For example, COLMAP requires tens to hundreds of high-resolution RGB images with at least 60–80% overlap, abundant surface texture, consistent exposure, and intrinsic calibration may be provided or internally estimated [31]. Polycam supports up to 2,000 images (depending on the user’s subscription level), captured by walking or panning around the scene, and also requires an overlap of more than 60% [35]. Pix4D’s recommended acquisition pattern is a nadir grid or oblique double-grid with forward/side overlap of 75%, supplemented by ground-control points when metricly scale or global globally-alignment is required [37].

2.4 Volumetric Rendering

Volumetric rendering techniques have been rapidly advancing the field of computer vision. Particularly, NeRF and Gaussian Splatting (GS) techniques have revolutionized continuous volumetric scene representation. NeRFs utilize a Multi-Layer Perceptron (MLP) neural network to predict colour and density at any 3D location and viewing direction [38], whereas GS represents the scene as an explicit collection of 3D Gaussian Primitives defined by their centre, covariance (orientation and scale), and opacity. Both methods of reconstruction were investigated and considered for use in the pipeline described later in Chapter 3. However, given prior experience and success with NeRF implementations, as well as the ease of integration, we chose to pursue using NeRFs in our research.

2.4.1 Neural Radiance Fields

Neural Radiance Fields [38] encode a real-world scene as a continuous volumetric field that can be rendered to synthesize photorealistic images from novel viewpoints. Compared with classical geometric representations such as point clouds, meshes, or voxels [39], NeRFs provide a lightweight continuous representation. These characteristics have rapidly popularized NeRFs for scene capture, and have recently motivated their use in robotics for mapping, localization, and simulation [40, 41].

Most NeRF implementations require a moderate number of images (typically 20–200 depending on scene size) covering the scene from diverse angles, without motion blur, and with at least 70% overlap between adjacent frames for successful reconstruction. Accurate camera intrinsics and pose estimates—obtained via an initialization from COLMAP—are essential, as are uniform lighting and exposure to facilitate reliable training [34].

FlyNeRF [42] is an example of drone-based data acquisition for high-quality 3D scene reconstruction. Large-scale NeRF representations have been addressed by Block-NeRF [43] and Mega-NeRF [44], which scale NeRF models for city-sized environments by decomposing them into multiple sub-networks, improving rendering efficiency and fidelity.

NeRF's potential for localization and SLAM has also been widely explored. NeRF-SLAM [45] integrates NeRF into a monocular SLAM framework, enhancing real-time scene reconstruction. NICE-SLAM and NICER-SLAM [40, 46] extend functionality by using monocular, multi-level local geometric cues for improved mapping and tracking. NICE-SLAM was also improved upon by incorporating depth uncertainty and motion information to obtain better tracking accuracy and robustness [47].

Additionally, several works have focused on real-time localization with NeRFs, using Monte Carlo localization for vision-based global pose estimation [48], neural signed distance functions to enhance Lidar odometry and mapping [49], and couplings with visual-inertial odometry pipelines for pose estimation [50, 51]. Rapid-Mapping [52] integrates Lidar and visual data to create high-fidelity scene reconstructions while maintaining real-time performance. Most applicable to our work, LocNDF [53] learns a Neural Distance Field to create a localization map from Lidar data. However, LocNDF does not directly employ NeRFs to produce their localization layer, does not incorporate a different sensor for scene capture, and relies on a rough initial GPS location for the ICP algorithm. Another approach related to our work is NeuRAD [54], which uses NeRFs to create a simulation tool for testing autonomous driving sensors. While they leverage NeRFs to extract sensor readings in simulation, they do not perform real-world localization. In contrast to related work, we develop a framework that leverages NeRFs for both simulation generation and map generation, enabling real-time localization with a ground vehicle in a physical environment.

Complementary to map-centric pipelines, NeRF inversion methods such as iNeRF [41], Parallel Inversion [55], and PI-NeRF [56] recover a single image's 6 DoF pose by minimizing its photometric residual against a pre-trained radiance field. These approaches demonstrate sub-second inference on modern Graphic Processing Units (GPUs), yet evaluations so far remain self-referential; they seldom benchmark against classical registration, leaving an exciting open question for future work.

Recent open-source toolkits such as Nerfstudio [34] streamline NeRF data processing, training, and evaluation. In our pipeline, we utilize Nerfstudio's seminal model called Nerfacto, which implements several key advancements from many other leading NeRF models like Instant-NGP's [33] hash grid encoding, enabling rapid training, large-scale scene modelling, and preservation of the input reconstruction's coordinate system. Additionally, camera pose refinement, proposal sampling, and per-image appearance embeddings accelerate convergence and improve output fidelity [34].

2.5 UGV and UAV Collaboration

UAV-UGV collaboration strategies have been employed for many aspects of autonomous operations, such as estimating a drone's position relative to a ground vehicle [57, 58], and using a SLAM framework with ultra-wideband (UWB) ranging units to enhance UGV localization accuracy [59].

Combined mapping efforts such as [60] involve a UAV-UGV system that employs a two-layer exploration strategy of coarse UGV-based mapping and fine UAV-based 3D reconstruction. Similarly, a cooperative exploration method has been developed, where a UAV provides aerial situational awareness and relative positioning using fiducial-tag tracking [61]. UAV-UGV path planning and obstacle mapping have also seen considerable advancements, including a multi-UAV stereo vision system for obstacle mapping [62] and a UAV-assisted path planning framework that utilizes aerial imagery to detect obstacles and generate safe routes [63]. The novel map registration pipeline in AgriColMap [64] involves leveraging visual information and dense optical flow estimation between UAV and UGV maps consisting of grid-based multimodal environments with a DSM and vegetation index for precision agriculture applications.

Other areas of this field include leveraging UAVs to scan terrain ahead of a moving UGV [65], integrating aerial and ground Lidar for collaborative risk mapping to enable real-time terrain assessment for off-road UGVs [66], and employing a UAV-guided UGV navigation system to optimize guidance viewpoints and trajectory planning in cluttered environments [67]. Notably, these works focus on simultaneous UAV and UGV operation and communication, not on creating localization map priors for later UGV navigation.

More closely related to our work is a cooperative SLAM system [68], which allows a UGV with a range sensor to localize itself within a reference map made by a UAV with a monocular camera, but strictly uses elevation maps built independently by the UGV and UAV to maintain invariance across sensors and viewpoints for live global localization demonstrated in only indoors. Additionally, a neural cost-mapping approach [69] also derives localization layer data using satellite imagery for NeRF modelling beforehand in an initial cost-mapping phase. Semantic information from crops and weeds observed by both a UAV and UGV has also be used to reduce visual ambiguity when using an aerial map as a prior for UGV localization [70], with success over long periods of time in farm fields.

However, none of these methods use a UAV and monocular camera exclusively for generating a digital twin environment offline, prior to UGV operation, to be used for pilot simulation and subsequent mission execution via localization with a Lidar and Radar sensor in large-scale outdoor environments without GPS.

Chapter 3

Virtual Teach and Repeat

This section describes the methodology for our Virtual Teach and Repeat algorithm. As mentioned in Chapter 1.1, the goal of this thesis is to expand current T&R capabilities so operations can be performed in new and more useful or efficient ways. The main advantages provided by VirT&R include removing the requirement of an operator to manually pilot the specific UGV through the desired environment to define a path, as well as the need to use the same model of UGV or sensor to repeat to an existing Teach map. Multiple paths through an environment can also be created once the aerial mapping and reconstruction are done, as opposed to manually piloting a new path for the UGV in the environment every time a different route is required. Additionally, with VirT&R, a teach pose graph can be created in only 1-2 mins after the path is driven in simulation for either Lidar or Radar T&R, given only one reconstruction of the environment made from aerial imagery.

3.1 System Overview

The VirT&R pipeline detailed in this section has three distinct phases to enable a UGV to repeat a route that was never physically taught: generating world representations, virtual map creation, and integration with Teach and Repeat. Respectively, these phases cover offline mapping, offline pose graph generation, and online navigation. We replace the traditional teach pass found in existing work [9] with the first two phases of the VirT&R, and the pipeline is built using the infrastructure in the T&R framework to ensure seamless integration of the virtual map. Our full VirT&R pipeline can be seen in 3.1, with the three main phases colour-coded for clarity.

The process of making a virtual teach map begins with offline mapping via UAV flights through the desired environment to acquire aerial imagery of the site so the images (optionally geo-referenced) can be used as input data to either of the two world reconstruction tools (Nerfstudio or Pix4D) previously described in Chapter 2.3 and 2.4.1. The resulting mesh and point cloud are used to virtually define any desired path through the environment and provide the localization layer that T&R will use for online navigation. The repeat phases of traditional T&R and VirT&R are identical, as this work focuses on presenting an expansion to the algorithm [9] by enabling a new way to teach. During the online navigation phase, the UGV localizes its live Lidar or Radar scans to point cloud

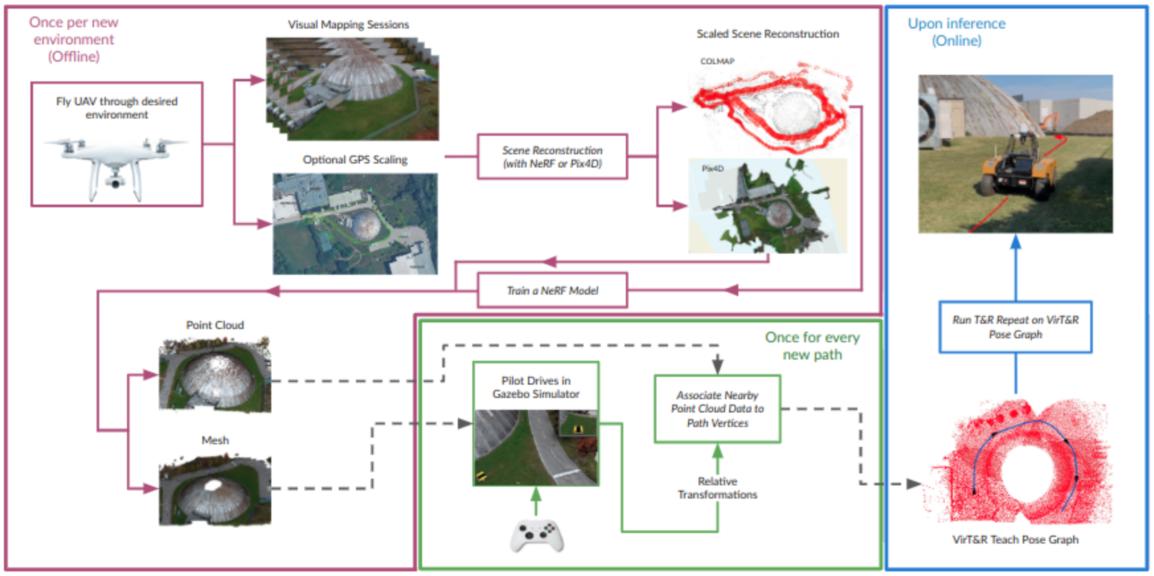

Once per new environment (Offline)
Fly UAV through desired environment
Visual Mapping Sessions
Optional GPS Scaling
Scene Reconstruction (with NeRF or Pix4D)
Scaled Scene Reconstruction
COLMAP
Pix4D
Train a NeRF Model
Point Cloud
Mesh
Once for every new path
Pilot Drives in Gazebo Simulator
Associate Nearby Point Cloud Data to Path Vertices
Relative Transformations
Upon inference (Online)
Run T&R Repeat on VirT&R Pose Graph
VirT&R Teach Pose Graph

Figure 3.1: Overview of the UAV-based mapping and UGV navigation pipeline. (1) Offline Mapping (magenta): A UAV captures images and optional GPS data, which are processed for reconstruction with either NeRF or Pix4D to generate point clouds and meshes. (2) Pose Graph Generation (green): The desired path is piloted in simulation, and the point cloud is used to associate nearby points with path vertices. (3) Online Navigation (blue): A UGV follows the path using T&R for real-world localization and navigation.

submaps in the virtual teach map generated from reconstruction tools and prior UAV imagery.

3.2 Generating World Representations

This section reviews how a desired scene for autonomous ground vehicle operation is captured by a UAV and reconstructed using 2D RGB aerial imagery, while also examining the optimal hyper-parameters and settings for the specific tools used. Through work on VirT&R, we experimented with multiple tools for scene reconstruction, focusing on NeRFs and classic photogrammetry with Pix4D, as they proved to be the most successful. Once a reconstruction of the scene has been created, regardless of the tool used, a point cloud and photo-textured mesh are extracted.

3.2.1 Scene Capture and Datasets

Creating high-quality datasets that will accurately represent the desired real-world scene begins with understanding the various input requirements and processing pipelines of the world reconstruction tools introduced in Chapter 2. For both Nerfstudio and Pix4D, the best results are obtained by following the well-known best practices also outlined in Chapter 2.

When using the NeRF method for reconstruction, the process begins with using COLMAP to obtain the camera poses throughout the scene. COLMAP relies on overlapping, well-exposed, still images from varied viewpoints to determine the camera intrinsics, match keypoints, estimate camera poses, and triangulate a sparse 3D point cloud. Then, the NeRF pipeline uses these camera pose estimates and the images in the MLP training. In general, this method demands diverse angular coverage and consistent lighting conditions throughout the duration of the aerial scene capture.

Pix4D also requires overlapping imagery to reconstruct textured meshes and dense point clouds directly, and it also requires the camera intrinsics to be provided. Across both methods, locking camera exposure, white balance, and focus parameters, as well as ensuring images are not blurry and have at least 70% overlap between adjacent frames (see Figure 3.2), forms the basis of our method for creating a successful dataset.

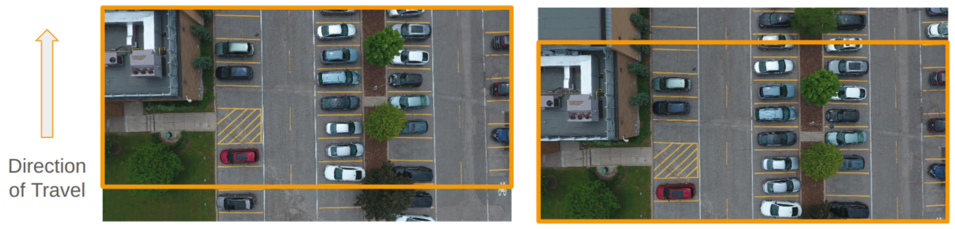


Figure 3.2: An example of approximately 80% overlap between images taken sequentially as a part of a dataset gathered for reconstruction with NeRF and Pix4D.

In addition to using the above analysis of the requirements for the respective methods, general recommendations for creating custom NeRF datasets were also taken into account when refining our method. Although specific guidelines are sparse, it is accepted that most NeRF implementations train best when a rough dome-shape of input images is provided [33].

It should be noted that both overcast and clear sunny conditions allowed datasets to be made that produced high-quality meshes and point clouds, with stationary shadows posing no issue. However, footage taken during the morning and evening golden hours, as well as on days where varied cloud cover led to moving shadows or quickly changing light levels, made for datasets that produced a poor or blurry, inaccurate reconstruction of the scene, sometimes failing to reproduce any recognizable structures at all. Additionally, it was found that reconstructions suffered substantially when large patches of undisturbed snow were present, which is a documented limitation of both NeRF and photogrammetry methods in general. The lack of visually identifiable texture, depth information, and the increased light reflection in snowy scenes resulted in blurrier NeRF renderings as well as bumpier point clouds and meshes, but not to the degree that they were unusable.

Experimentally, we found that datasets with between 600 and 1,500 images of at least 960 x 540 resolution were required for COLMAP camera pose estimation and NeRF model training. However, further refinement of our scene capture process and the inclusion of Pix4D as a reconstruction method after the initial experiment at the University of Toronto Institute for Aerospace Studies (UTIAS) campus, we found that using between 300 and 700 images of 4000 x 2250 resolution was ideal. Any additional images at higher resolutions yielded diminishing returns in terms of time spent training and visual quality.

To ensure that each UAV flight produced a good dataset for VirT&R, we developed and tested a specific method for flying through a scene to capture it, based on existing photogrammetry techniques, accepted NeRF image capture guidelines, and the significant factors previously outlined in this section. We use three concentric laps through the environment on a given survey flight, where we focus on flying around the scene's approximate centre with the loose goal of creating a dome-shape of images around it. The first pass is flown at approximately 20 m above ground level (AGL), with a 45° inward-facing sweep achieved by angling the camera obliquely and yawing around the scene centre. This pass captures distant and intermediate structures with pronounced baseline variation, which is essential for robust feature matching and wide-angle context to help align images from closer perspectives. The second pass is a 10 m AGL nadir-facing pass aligned with the intended UGV trajectory: straight-down imagery ensures dense, orthographic ground coverage for accurate terrain modelling and height mapping. Finally, the third pass consists of a 5–8 m AGL forward-facing point-of-view (PoV) circuit tracing the desired UGV route at close range, revealing occluded details, narrow corridors, and fine surface textures. Together, these passes balance scale, parallax, and occlusion coverage to satisfy both Pix4D's photogrammetric requirements and NeRF's viewpoint diversity needs. A visualization of the flight path can be seen in Figure 3.3.

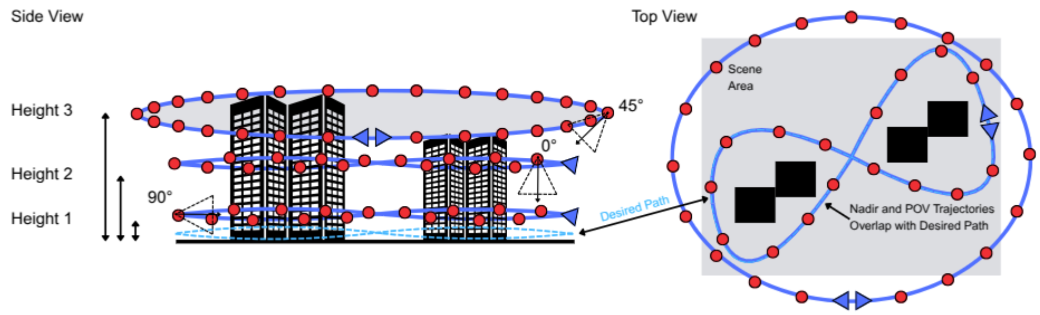
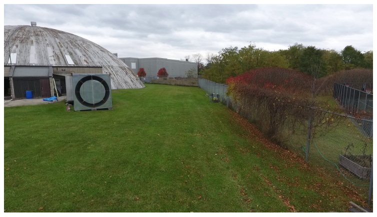
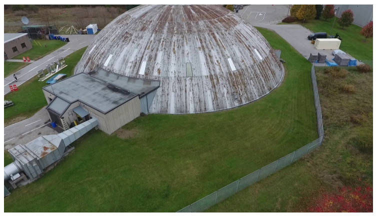
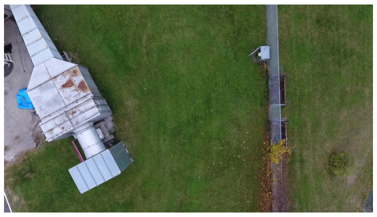


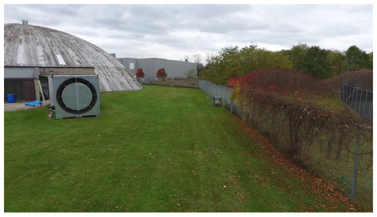
Figure 3.3: Depiction of the three laps through a given environment flown by a UAV that forms the survey flight for VirT&R and ensures a detailed capture of the scene.

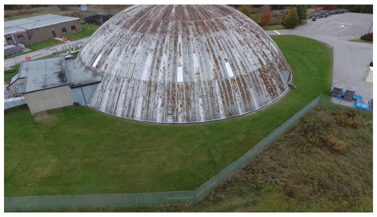
Examples of images taken during the three described laps of the UAV survey flight for the UTIAS Mars Dome Loop environment we captured can be seen below in Figures 3.4.


(a) PoV Lap (time = t_k)


(b) Oblique Lap (time = t_k)


(c) Nadir Lap (time = t_k)


(d) PoV Lap (time = t_{k+1})


(e) Oblique Lap (time = t_{k+1})

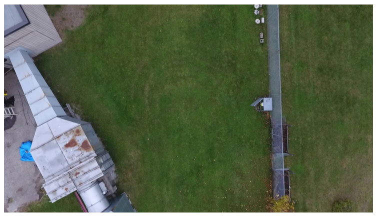

(f) Nadir Lap (time = t_{k+1})

Figure 3.4: Images captured by the DJI Phantom 4 Drone to create the VirT&R Mars Dome Loop Dataset. The three viewpoints (PoV, Oblique, and Nadir) can be seen here. Within the columns of the figure, the images at the respective viewpoints are the actual subsequent frames captured by the drone at the correct approximate 75% overlap between some time t_k and the next time t_{k+1}.

3.2.2 NeRF Reconstruction and Parameter Optimization

The chosen NeRF implementation used with VirT&R is Nerfstudio's Nerfacto model, described previously in Chapter 2.4.1. As with other NeRF implementations, an initialization of camera poses from COLMAP is required as an input, along with the images themselves. The dataset created from flying a UAV through an environment is organized into a project folder and processed through COLMAP's feature detection, pair-wise matching, and incremental pose estimation to produce a camera trajectory.

During this process, COLMAP converts every image to a gray-scale version with a maximum edge of 2,000 pixels, then detects up to 16,384 Difference-of-Gaussian keypoints per view (a threshold dictated by GPU strength). Each keypoint is refined through affine-shape adaptation and normalized using domain-size pooling, which together enhance repeatability across viewpoint and scale changes, allowing for the computation of a 128-dimensional SIFT descriptor for every refined keypoint. There are multiple matching methods available in COLMAP, including the sequential matcher, exhaustive matcher, and spatial matcher. We primarily use the exhaustive matcher; however, using the spatial matcher, if GPS information is included with the images for experimental evaluation, speeds up the processing time as GPS coordinates for each image aid in determining the camera poses by predicting which pairs are likely to observe the same part of the scene, and also results in a scaled reconstruction. The exhaustive matcher attempts pairwise matching between all image pairs and retains only correspondences that survive the set geometric verification thresholds (we found the default to be sufficient). For every candidate pair, a matcher uses Lowe's ratio test [29] (default 0.8) and symmetric cross-checking to generate putative correspondences, then filters them with a guided-epipolar Random Sample Consensus (RANSAC) that keeps only inlier matches supporting a minimum of eight SIFT correspondences and an inlier ratio above 0.10. Then, by initiating the mapping procedure, an initial image pair with the richest set of spatial matches triangulates an initial seed model, and then repeatedly (i) localizes the next best image by Perspective-n-Point, (ii) triangulates new points, and (iii) bundle-adjusts the jointly optimized camera parameters and 3D points. Images with a match graph that is too weak (fewer than eight verified correspondences to the current model) are omitted; therefore, if camera coverage is poor, the resulting model may contain gaps.

The reconstruction is then passed through COLMAP's model aligner, where the GPS coordinates of every image (intentionally downscaled by a factor of 100 to enable faster NeRF training by creating a smaller scene) are treated as an object-space coordinate system. A similarity transform (three rotations, three translations, one uniform scale) is fit by minimizing the reprojection error between the reconstructed camera centres and their provided coordinates, using a one-pixel inlier threshold. This deliberately miniaturized model trains dramatically faster in Nerfstudio as it is small enough to use their standard model of Nerfacto, whereas the real-world scaled version would require their Nerfacto-Large model, that increases training time by approximately five hours. Rescaling the point cloud and mesh by the preset factor of 100, after NeRF training, ensures we remain metrically faithful to the environment. Figure 3.5 shows the final result of running COLMAP to obtain the camera poses in a given scene.

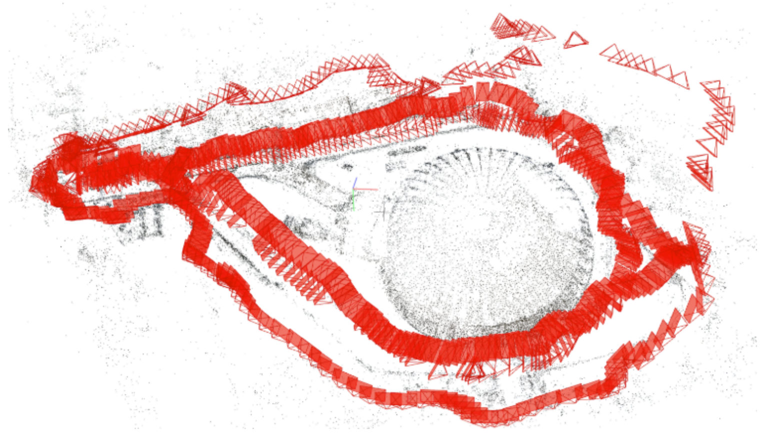


Figure 3.5: The final estimated camera poses as a result of COLMAP's SfM pipeline. Red frustums represent camera locations in the scene and their orientations. The sparse point cloud produced here is the result of the points used in triangulation to obtain the image poses.

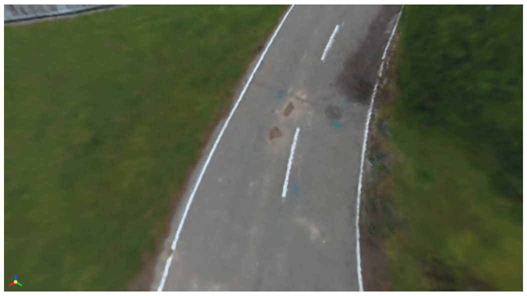
The COLMAP reconstruction and optional GPS scaling [31] used to create the inputs for the NeRF model took 43 minutes for the largest scene (approximately 150 m x 100 m in the real world) with 1,555 960 x 540 resolution images. This initial reconstruction provides the required pose input (in the form of .bin files of 3D points, cameras, and poses). Along with the raw images from this reconstruction, the Nerfstudio MLP can now be trained. The default Nerfacto architectural choices enabled training with an NVIDIA GeForce RTX 3090 GPU to take 22 minutes for the largest scene (100 m x 150 m) with the 1,555 960 x 540 resolution images. These parameters were used for the initial UTIAS experiment in Chapter 5, as the resulting mesh and point cloud were of sufficient quality. Examples of different views from a NeRF reconstruction of one of the datasets we used with VirT&R can be seen below in Figure 3.6.

To maximize the fidelity of the mesh and point cloud from NeRF reconstructions for localization and potentially improved path tracking, we performed an ablation study on the hyperparameters of Nerfacto. Machine learning models such as NeRF MLPs have many hyperparameters that can significantly affect performance, and finding the right hyperparameters for a given scenario is crucial, as an inappropriate setting can result in blurry reconstructions or convergence failure. Extracting explicit geometric representations like point clouds and meshes from a NeRF model is non-trivial, as the representation is implicit, which further emphasizes the importance of parameter optimization for this specific use case.

Exhaustively enumerating their combinations is intractable, so a carefully staged ablation study is required to locate critical regions of the search space while keeping compute budgets reasonable. We employ a three-stage study: (i) a broad exploratory search to reveal globally important factors, (ii) a series of targeted grid sweeps to refine promising regions of the hyperparameter space, and (iii) a final Bayesian optimization that jointly tunes the narrowed ranges. Throughout, performance is assessed on a fixed validation set using Peak Signal-to-Noise Ratio (PSNR), Learned Perceptual Image Patch Similarity (LPIPS), and Structural Similarity Index Measure (SSIM). The PSNR pro-

(a) First view

(b) Second view


(c) Third View

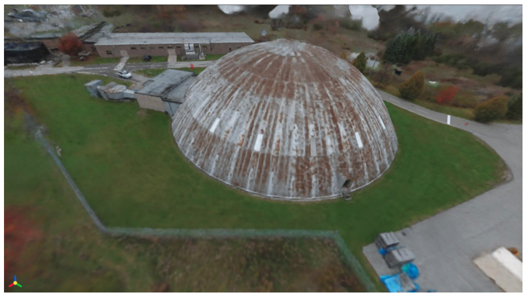

(d) Fourth View

Figure 3.6: Several views of the NeRF volumetric rendering-based reconstruction of the UTIAS Mars Dome Loop dataset, created from aerial imagery taken by a drone and given to the software. Shown here is the volumetric rendering visualized in Nerfstudio. The detail captured in this dataset is consistent with all other datasets made by Nerfstudio with default parameters, where the spray paint marks, concrete cracks, grass texture, and many other details are visible.

vides a logarithmic measure of per-pixel fidelity (higher is better); The SSIM captures luminance, contrast, and structural agreement that correlates with human perception (higher is better); and the LPIPS leverages deep feature embeddings to quantify perceptual similarity (lower is better). Using all three provides a balanced view of radiance-field quality, ensuring that parameter choices that improve numeric PSNR at the cost of perceived realism are rejected.

Initially, a random sweep (sweep 1) sampled eight key knobs across broad ranges detailed in Table 3.1 using a fixed training schedule (20,000 steps) and identical data for every run, guaranteeing metric comparability.

Table 3.1: Search ranges for all hyperparameters in the first, large, random sweep.

Parameter	**Values**
Hidden-layer width h_{dim}	$\{64, 128, 256, 512\}$
Hash-grid depth L	$\{12, 14, 16\}$
Features per level F	$\{2, 4\}$
Hash-table size $\log_2 \|T\|$	$\{19, 21, 23\}$
Maximum resolution $\max_{\text{res}}$	$\{2048, 4096, 8192\}$
Rays per batch N_{rays}	$\{4096, 8192, 16384\}$
Samples per ray N_{samples}	$\{48, 64, 96\}$
Distortion-loss multiplier λ_{dist}	$\{0, 0.002, 0.01\}$
Field learning rate η_{field}	$\{5 \times 10^{-3}, 10^{-2}, 2 \times 10^{-2}\}$
Camera learning rate η_{cam}	$\{10^{-4}, 10^{-3}\}$

Insights from the random sweep were distilled into five smaller, topic-focused grid searches, as shown in Table 3.2, which were conducted using 50,000 steps this time. A summary of each sweep and what their respective evaluation metric results were can be found in Table 3.3.

Table 3.2: Hyperparameter dimensions explored in Sweeps 2–6.

Sweep #	**Hyper-parameter focus**
2	Optimizer learning rates (fields and camera)
3	Network capacity (hidden-layer width and hash-grid depth)
4	Ray-batch size versus per-ray sample count trade-off
5	Hash-grid spatial resolution parameters
6	Distortion-loss multiplier jointly with network capacity

Table 3.3: Best validation metrics per sweep and the salient hyperparameter values that produced them.

Sweep #	**PSNR** $\uparrow$	**SSIM** $\uparrow$	**LPIPS** $\downarrow$	h_{dim}	L	η_{field}
1 (random)	19.76	0.371	0.711	128	16	0.001
2 (LR)	**21.06**	**0.448**	0.784	64	16	0.005
3 (capacity)	20.38	0.438	**0.673**	512	14	0.001
4 (rays $\times$ samples)	20.30	0.430	0.772	64	16	0.001
5 (grid res)	20.52	0.437	0.771	64	16	0.001
6 (distortion)	20.64	0.437	0.793	512	12	0.001

Our smaller sweeps revealed that low-width networks (64–128 units) yielded the highest PSNR, but wider networks (256–512 units) consistently improved perceptual LPIPS and SSIM. Sixteen hash-grid levels were favoured in four of the six sweeps, while a small distortion-loss weight of 0.002 provided a measurable PSNR bump without hurting SSIM.

Guided by the results from sweeps 2-6, a Bayesian optimization sweep (sweep 7) was launched with a 100-run budget. The discrete search space retained only the hyperparameter values that appeared in the top decile of the previous sweeps, along with a few midpoints to facilitate smooth interpolation. An early-termination scheduler pruned poorly performing runs after 3,000 iterations, allowing the sweep to converge within two days. The winning configuration attained a PSNR $=$ 22.83 dB, SSIM $= 0.442$, and LPIPS $= 0.715$, compared to PSNR $= 20.31$ dB, SSIM $= 0.432$, and LPIPS $= 0.788$ obtained from using the default NeRF parameters. Table 3.4 summarizes the default and winning parameters.

For large-scale aerial surveys captured by UAVs, the state-of-the-art Aerial-NeRF [71] reports 23.52 dB, 0.742 SSIM, 0.261 LPIPS on their presented SCUTic drone dataset that covers a large, outdoor, university campus. Thus, the Nerfacto model, for which we determined optimal parameters, approaches their PSNR while remaining within one order of magnitude in perceptual scores, despite being trained on a smaller compute budget and a different scene domain. This configuration will henceforth be used for all NeRF training throughout the remainder of the thesis. A visual qualitative comparison of Nerfacto model scene rendering after training can be seen in Figure 3.7.

Table 3.4: Summary of the hyperparameters varied in the ablation study, what the optimal parameters were found to be for our scene size and quality/quantity of images, and what the default parameters were.

Hyper-parameter	**Optimized**	**Default**
Hidden width h_{dim}	256	64
Hash levels L	14	16
Features per level F	4	2
$\log_2 \lvert T \rvert$	19	19
Maximum resolution $\max_{\text{res}}$	8192	2048
Rays per batch N_{rays}	16384	4096
Samples per ray N_{samples}	64	48
Distance regularizer λ_{dist}	0.002	0.002
Field LR η_{field}	3.5×10^{-3}	5×10^{-4}
Camera LR η_{cam}	1×10^{-4}	1×10^{-3}

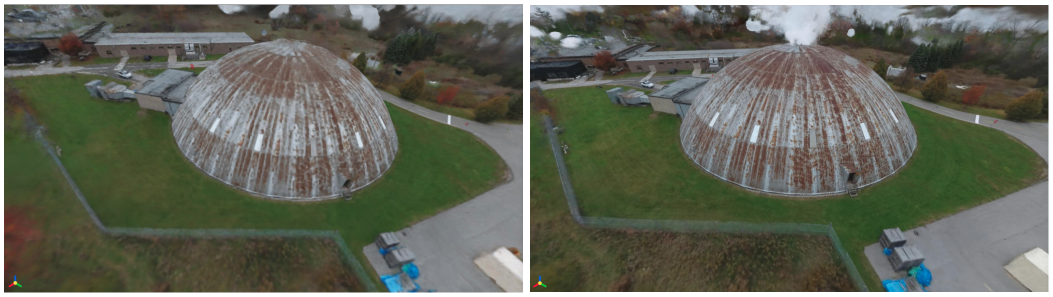
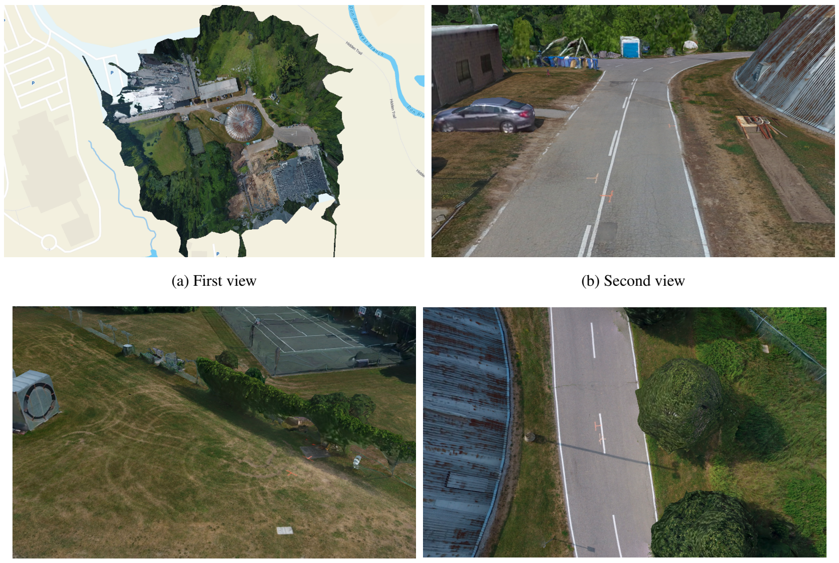

(a) Default Hyperparameters (b) Optimized Hyperparameters

Figure 3.7: Comparison of visual quality between a NeRF volumetric rendering of the same scene trained with and without optimal parameters. (a) shows the non-optimized NeRF training result. (b) Shows the NeRF trained using optimal parameters identified in the ablation study. The quality of the optimized NeRF is slightly improved, particularly in fine details and surfaces.

3.2.3 Photogrammetry Reconstruction

Of the investigated photogrammetry tools, the commercial software, Pix4D Cloud Advanced [36], was chosen to complement the learning-based NeRF reconstruction method. This program executes the entire reconstruction pipeline server-side once a given image set and optional GCP information are uploaded. Pix4D also provides precise global position and orientation through its pipeline, with built-in features that visualize the reconstruction results over a real-world global map when GPS information is available. Although this is not necessary for the VirT&R pipeline, which is capable of providing accurate, GPS-denied, zero-shot navigation via a local topometric map approach, it does facilitate more accurate virtual GPS data creation for absolute path-tracking assessment, not possible with the NeRF reconstruction method due to a build up of error from the reconstruction as discussed in Chapter 4.2.

The process of creating a scene reconstruction with Pix4D Cloud Advanced [36] begins with the same aerial data capture flights as the NeRF method does. After uploading images to the software, a few key parameters are set. Pix4D requires classification of the type of viewpoints captured in the images uploaded, specifically whether they are mostly nadir-facing or contain a variety of nadir and oblique-facing views (as our datasets do). Additionally, the GCP point information can be provided optionally. When processing begins, the platform inspects EXIF tags, removes corrupted or duplicate files, and converts each photograph to a pyramidal JPEG for accelerated keypoint detection. Scale-invariant features are extracted in a multi-threaded stack, then robust pairwise matches are found using an adaptive RANSAC and bundle adjustment. Next, the internal camera parameters are refined together with every exterior orientation. From here, depth is interpolated for every pixel seen in at least two calibrated views, and the image positions (when provided) are treated as observations with user-defined standard deviation, anchoring the model in the selected output coordinate system (we use WGS84, as does the NovAtel GPS on the Warthog). Views of a photogrammetry reconstruction of the Mars Dome scene using Pix4D are shown in Figure 3.8.


(a) First view (b) Second view

(c) Third View (d) Fourth View

Figure 3.8: Several views of the Pix4D photogrammetry-based reconstruction of the UTIAS Mars Dome Loop dataset, created from aerial imagery taken by a drone. Shown here is the mesh visualized in Pix4D. The detail captured in this dataset is consistent with all other datasets made by Pix4D, where the spray paint marks, concrete cracks, grass texture, and many other details are visible.

The resulting point cloud inherits the absolute geo-referenced pose where each point (x, y, z) lies directly in the global frame (if GPS positions are provided). Advanced parameters enable the user to adjust the minimum number of matches, voxel subsampling, depth-fusion aggressiveness, and implement a Region-of-Interest to restrict computation to a specified processing area, thereby trimming noise outside the region of interest. The resulting mesh is created in an arbitrary frame, as many 3D applications use a local frame. Therefore, Pix4D writes the mesh in relative coordinates: vertex positions are centred on a temporary origin and are stored without the global transformation. When the mesh is viewed alongside the point cloud inside Pix4D, the platform implicitly applies the transform used for the points, but on export, the user must re-embed that transform if global coordinates are required, as mentioned.

Pix4D took approximately 75 minutes to generate the reconstruction of a 150 x 100 m scene with 604 4000 x 2250 resoultion images shown in Figure 3.8. This is slightly longer than the NeRF method (which has two steps that take around an hour combined) but is more streamlined and predictable.

3.2.4 Localization Layer Rendering

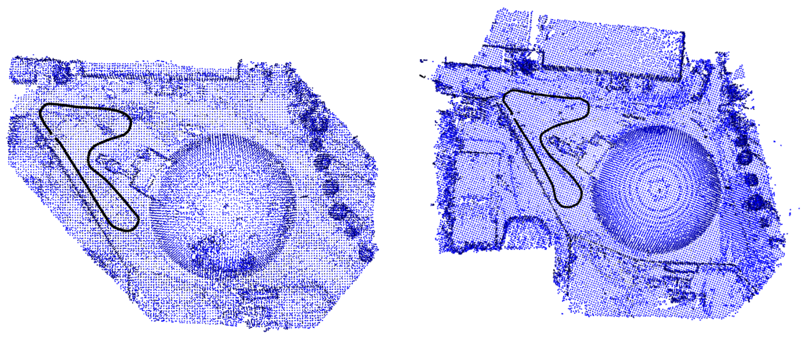
The two key representations from the virtual world reconstruction: a dense point cloud and a photo-textured mesh, are essential for localization and navigation tasks. The point cloud provides a rich representation of the environment and acts as the teach map's localization layer, while the mesh facilitates a visually and physically accurate pilot simulation for trajectory planning and path definition. An example of what the mesh, virtual Lidar point cloud, and virtual Radar point cloud look like can be found in Figure 1.3 near the beginning of this thesis in Chapter 1. In VirT&R, the localization layer is derived directly from the virtual world reconstruction point cloud, rather than a sliding window of raw sensor scans that are saved to a vertex in the pose graph to provide a dense, locally metrically accurate map of the scene.

The localization layer for any T&R variant is the geometric world representation that the UGV uses to localize itself in an environment. The amalgamation of several point clouds (either from a Lidar sensor or extracted from a polar image taken by a Radar sensor) creates an accurate representation of a small, local portion of the environment along the taught route (a submap) and is what gets used as the reference when ICP alignment is performed during a repeat for localization. To replicate this for both Radar and Lidar sensors, we use the virtually generated point cloud of the whole environment and take small croppings of it (see section 3.4.1), based on the sensor modality, along the virtually defined path from the pilot simulation to mimic the result of a submap at a vertex.

Once a NeRF has converged, Nerfstudio exposes its learned volumetric scene $\big(\mathbf{x}, \mathbf{d}\big) \mapsto (\sigma, \mathbf{c})$, allowing the field to be sampled densely and converted into explicit geometry. For point cloud extraction, Nerfstudio launches millions of virtual rays that stochastically cover the axis-aligned bounding box of the scene. Each ray is marched until the accumulated opacity, computed as $\alpha = 1 - \exp(-\sigma \Delta t)$, exceeds a user-defined threshold (default 0.7). The first hit is accepted as a surface point, its colour is taken directly from the NeRF's RGB prediction, and the local surface normal is either estimated by finite-differencing the density field or read from the network if it was trained with normal prediction enabled [34]. Mesh extraction follows a parallel but more elaborate pipeline. The Truncated Signed Distance Function (TSDF) fusion method renders dense depth maps from a spherical set of virtual cameras, integrates them into a truncated signed-distance volume, and then applies Marching Cubes to obtain a watertight manifold.

In Pix4D, a MVS densification routine that reprojects the calibrated images at multiple resolutions, interpolates depth through semi-global matching, and fuses the depth hypotheses into a single, coloured point cloud. Once the densified cloud is available, Pix4D triangulates the point set using an adaptive Delaunay-based meshing engine, fills small gaps, decimates and smooths the data according to user-defined quality presets, and finally projects the original photographs onto the mesh to create a high-resolution texture atlas.

Figure 3.9 shows a NeRF (trained using optimal parameters) and a Pix4D point cloud made by providing aerial imagery of a desired environment to the respective tools. It is these point clouds that act as the localization layer in VirT&R, as they are what the submaps in the virtual teach maps come from.



(a) NeRF (b) Pix4D

Figure 3.9: Visualizations of the (a) NeRF and (b) Pix4D generated point clouds of the environment captured by aerial imagery that act as the source for creating submaps for the virtual teach maps. These point clouds have already been voxel-downsampled by a factor of 0.9 from their approximately one million points. The path of black dots over the point clouds is created from the poses recorded by driving a robot in the virtual driving simulation to define a path through the environment.

We found that ensuring the reconstruction methods return a point cloud with a density of approximately 100 points per square meter in it is crucial as, for scenes of approximately 100 m x 100 m, one million points produced a detailed enough representation with sufficient points for localization with point-to-plane ICP in the smaller submap croppings after a voxel downsampling by a factor of 0.9 to ensure convergence within a feasible runtime. This translates to the virtually generated submap point clouds having approximately 7,000 points to simulate a Lidar scan and 200 points to simulate a Radar scan.

3.3 Virtual Map Creation

The creation of the virtual teach pose graph map involves assembling all the necessary information we have produced into a traditional teach map format offline, before a mission is run. This includes initializing values and providing metadata, such as the rough latitude and longitude of the environment, sensor frame information, and assigning covariances for transformations that the T&R framework expects to find, so that we can easily utilize the existing architecture to manufacture a pose graph of our virtually taught paths in the same format. Our virtual map creation method gives the operator the same situational awareness as standing behind the vehicle with a handheld remote, but from the comfort and safety of a computer. Features such as third-person camera viewing, the ability to fly hundreds of meters ahead to examine terrain without moving the robot or physically going there, and the high-fidelity photo-textured appearance enable reliable terrain assessment long before the first real wheel touches the ground.

3.3.1 Pilot Simulation

As mentioned, all teach passes for VirT&R are generated offline. Two methods have been developed for virtually defining a path through the environment for the UGV to follow: an interactive pilot simulation with a high-fidelity model of the UGV, and a waypoint clicking tool that is useful for long paths.

In the interactive driving simulation method, the virtually generated mesh is imported into Gazebo [72] twice: once as a collision-aware static mesh for physics and once as a high-resolution visual mesh, so that every bump, berm, salt stain, and shadow appears visible to the user. The UGV is spawned into Gazebo using its standard URDF, preserving the standard joint frames and wheelbase. The simulated IMU and wheel joints are left active, allowing the vehicle to settle realistically on sloped or uneven terrain. A floating chase-camera is rigidly added to the URDF behind and above the UGV, centred at a 45° angle to provide a useful, supplemental third-person-view. In practice, this vantage point, combined with the photorealistic mesh and the ability to dynamically pan through the environment, enables the pilot to make the same miniscule terrain judgments (e.g. avoiding ruts, choosing grass over mud) and large-scale planning decisions that would normally require a physical presence.

Piloting the UGV is enabled by connecting Gazebo with Robot Operating System (ROS) so that velocity commands can be communicated using the published Joy topic and pose can be logged. The UGV is driven through the mesh with an Xbox joystick controller whose axes are mapped to linear and angular velocity commands. During the virtual teach pass, we subscribe and convert the Warthog pose to an $\mathrm{SE}(3)$ matrix. The relative transformations of the path the robot took through the environment are logged and can be compounded using $\mathrm{SE}(3)$ rigid body transformations to obtain the vertices that traverse the environment, as well as providing the trajectory to link together vertices and ensure new submaps are saved at certain distance and orientation thresholds. Relative transformations were recorded at 20 Hz and saved to a time-stamped comma separated value (CSV) file, providing sufficient vertices for a pose graph. A view of the Gazebo simulation can be seen in Figure 3.10.

The waypoint clicking tool consists of a lightweight Python OpenGL viewer that renders the same mesh in an interactive window, allowing waypoints to be clicked to define the path. Left-clicking creates a waypoint, displayed as a small coloured sphere; when at least four points exist, a cubic B-spline is fit that minimizes the squared curvature subject to zero end slopes. The spline is then upsampled at 0.2 m arc-length intervals, and the discrete poses are written to a CSV file in the same relative transform format as is used with the Gazebo ROS version. The initial position and subsequent relative transformations are logged to a CSV file in the same manner as the Gazebo simulation method does. The use of the waypoint definition tool is illustrated in Figure 3.11a, which presents a top-down view, and Figure 3.11b, which provides a closer PoV-type view of the waypoints and path.

Whether the path was driven with the joystick or clicked to create waypoints, the downstream T&R modules are agnostic. Once the CSV file is produced, which records the initial starting pose

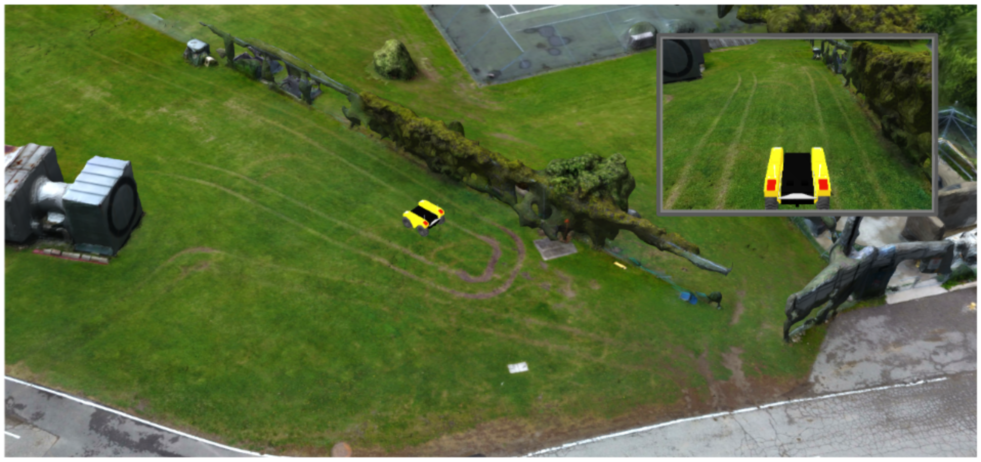
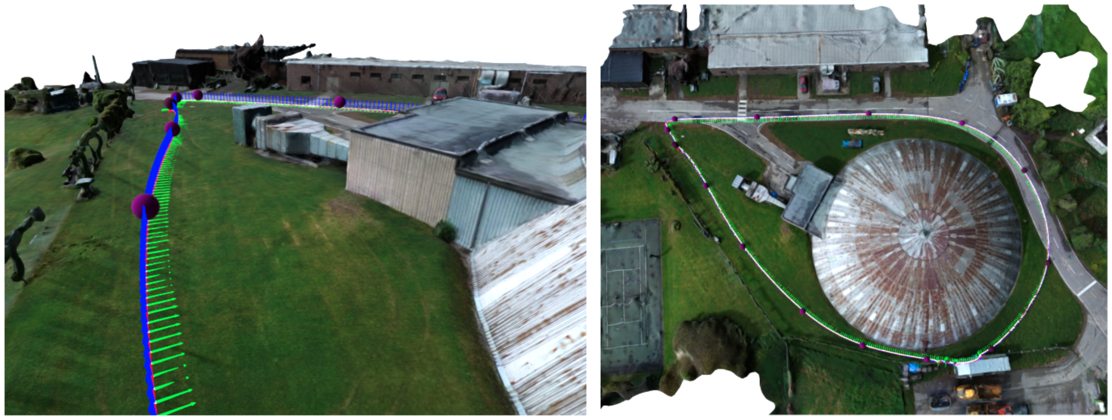


Figure 3.10: The URDF model of the Clearpath Warthog is being driven in the Gazebo pilot driving simulation that uses the triangular mesh virtually generated from the aerial scene reconstruction tools as the physical surface for traversal. The window in Gazebo shows the larger interactive view for panning around and the fixed PoV view of the robot placed at a 45° angle above and behind the robot.



(a) Close-Up of Waypoints and Path

(b) View of Entire Path Defined by Waypoints

Figure 3.11: Views from the waypoint-based virtual path definition tool created for more efficient path generation, particularly for long paths. (a) Shows the interactive window panned closer to the spline fit between the purple waypoints used to define the path. (b) Shows the interactive window panned further to capture the whole path defined by the spline fit between the purple waypoints.

of the robot in the mesh frame and subsequent relative transformations, integration with T&R is facilitated easily because the edge transformations are stored and accumulated to provide vertex poses.

3.3.2 Submap Association

The VirT&R map making algorithm uses the CSV of relative transformations and timestamps that comes from the pilot simulation, as well as the point cloud from the world reconstruction. The whole point cloud is first transformed into the vertex frame. This is done by pre-multiplying it's origin by the absolute starting pose of the robot on the mesh in the Gazebo pilot simulation, which is recorded in the CSV file; hence the origin of the point cloud now coincides with the origin of the pose-graph, and every subsequent absolute pose can be applied directly to the cloud. The starting position in the T&R code is actually set to identity, but recording the absolute position in the Gazebo frame allows us to start recording in any location on the mesh, as opposed to needing to be at the origin, and still have the path line up in the correct spot when overlaid on the point cloud.

Then, each of the k time-stamped relative transformations becomes a vertex in the new pose graph; the first vertex of the path, $v_k, k = 0$, is set as the origin (0,0,0) and every subsequent time-stamped transformation $\mathbf{T}_{k+1,k}$ is added to the pose graph as a temporal edge with zero covariance, which get accumulated to obtain the absolute pose of current vertex which also gets saved to the pose graph. This allows downstream T&R modules to interpret the virtual path as perfect odometry.

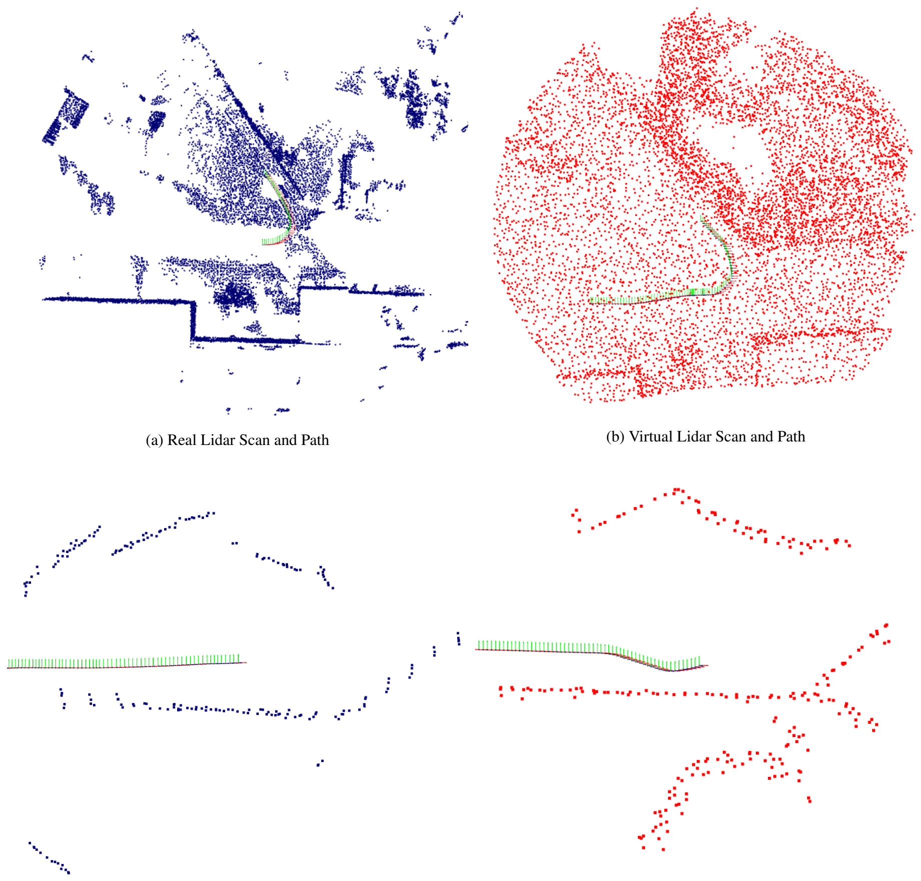
Iterating over the vertices in chronological order, rotation and translation thresholds (tuned in previous versions of T&R) determine whether the current vertex should create a new submap. If it is the first vertex in the sequence, the answer is automatically yes. Otherwise, the transform between the current vertex and the last vertex with an associated submap is formed; its translation norm and Euler-angle norm are compared with the standard T&R thresholds of 1.5 m or $30°$. Exceeding either threshold triggers a new submap to be instantiated and the last pointer to be updated; failing means the vertex will merely reference the most recent vertex submap using a pointer that represents the transformation to that populated vertex. This strategy replicates the vertex/submap density produced by live teaching and guarantees that, at runtime, localization queries never exceed the map update rate used during real-world field trials.

Whether a new submap was created or not, a pointer is created, containing the current vertex identifier, the identifier of the vertex that owns the relevant submap, and the relative transform between the two vertices. This pointer enables the localization pipeline to skip directly to the correct submap from a live sensor scan, without traversing the graph, in cases where the real mission is not initiated at the starting vertex in the real world. Figure 3.12 shows what real and virtual sensor scans look like for a Radar and Lidar sensor, highlighting the differences between them. Most notably, are the differences in range covered by the real and virtual scans, the density and distribution of points, and the presence of radial noise seen in the real Radar scan that isn't present in the virtual scan.

Upon completion, the directory contains a pose graph whose sparsely spaced vertices contain Lidar or Radar submaps with point clouds generated from a reconstruction of the world, designed to reassemble the real sensor data. Meanwhile, intermediate vertices carry only pointers and the necessary transforms for real-time localization. Because the data layout is indistinguishable from a conventional teach pass, the T&R stack can seamlessly replay the graph unmodified during an autonomous repeat.

3.4 Integration with Teach and Repeat

Online navigation using a virtual teach map is achieved using the existing repeat portion of the T&R framework, which allows the UGV to localize itself relatively using its onboard Lidar or Radar sensors. During a real mission, live sensor scans of the environment surrounding the UGV are matched to the virtual submaps using the ICP algorithm [19]. Once localization is established,


(a) Real Lidar Scan and Path

(b) Virtual Lidar Scan and Path

(c) Real Radar Scan and Path

(d) Virtual Radar Scan and Path

Figure 3.12: Visual comparison of real and virtually generated Lidar and Radar scans from teach maps used for online navigation, along with the path through the environment visualized as small orthogonal axes at vertices in the map. Respectively, the real and virtual scans are from approximately the same locations.

the UGV can follow the virtually taught path, repeating as if it had been physically taught with traditional T&R. The odometry, localization, MPC, and state machine modules of T&R that run during a repeat, as outlined previously in Chapter 2.1 and 2.2, remain unchanged and process the virtual teach map in the same manner as any traditional map.

3.4.1 Localizing with Virtual Submaps

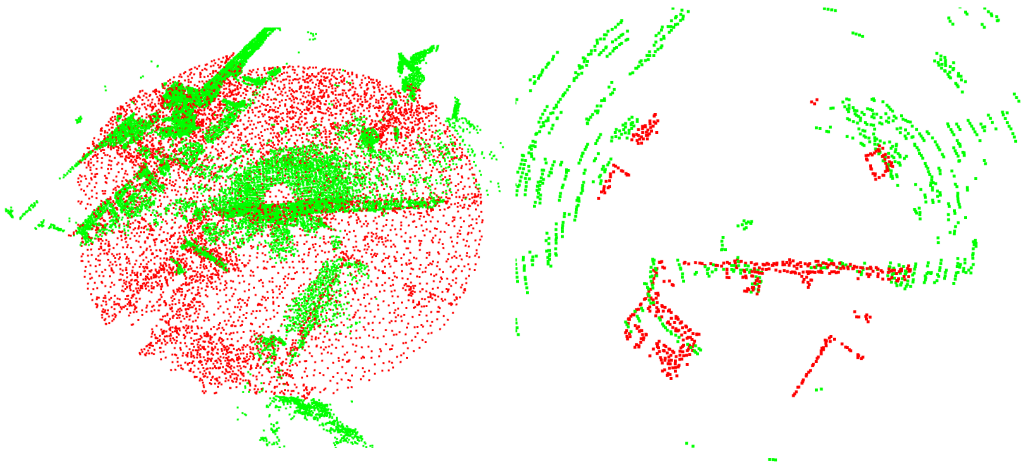
In general, during the repeat pass, odometry provides a high-rate but drifting guess of the robot trajectory. When the next pose graph vertex is reached, the teach vertex closest to it is used to build a pose prior to initialize the live scan localization. The live scan is then aligned to the chosen submap using point-to-plane ICP, which minimizes the distance between corresponding live scan and nearest neighbour vertex submap points by finding the closest point and normal. A more detailed explanation can be found in Chapter 2.1.

Ensuring that the virtually generated point cloud of the environment is cropped into submaps during the virtual teach submap association process in a way that allows for ICP alignment to take place is the crux of this problem and is what enables our cross-modality. A common initial step for creating virtual Lidar and Radar submap point clouds is to voxel downsample the entire virtual point cloud (with approximately 100 points per square meter) by a factor of 0.9. Then, for Virtual Lidar T&R, submaps use a $r = 40$ m, $h = 30$ m cylinder, centred at the sensor frame origin, to determine the points to be included in a submap for a given vertex. Point normals are copied so that localization may employ point-to-plane ICP, which is properly seeded by the outgoing edge transform defined in the virtual driving simulator and added to the pose graph, as it would be with a traditional teach map.

The virtual Radar submap cropping uses a smaller radius and height, $r = 20$ m and $h = 0.1$ m, with the point cloud expressed in the Radar sensor frame through the fixed transform, $\mathbf{T}_{sr}$, between the robot and sensor frames. This ensures submaps are made according to how a Radar beam would be cast out into the environment from the robot, and projected down to a 2D plane at the height of the Radar, mimicing how traditional RT&R produces the point cloud submaps from 2D polar images. This is also crucial to ensure a similar level of ground strike artifacts. Points are further constrained to the Radar boresight cone by filtering out points with an elevation angle exceeding 2° ($|\arctan(z/r_{xy})| > 2^\circ$). We then perform per-azimuth first-return selection (bin width $\Delta\theta \approx 0.9^\circ$), keeping only returns within $0.1\,\mathrm{m}$ of the nearest range to suppress shadows/occlusions. We then add a small Gaussian jitter in range and azimuth to generate short forward echo streaks, and points in the cropped and projected 2D point cloud are also only kept within the tuned threshold of 7 cm to any adjacent neighbour to mimic how a Radar sensor would not be able to see through solid objects.

While this is also true for Lidar-based T&R, we found the larger radius for virtual Lidar maps, which resulted in the inclusion of more structure, was helpful as it meant more identifiable surfaces were visible and there were enough correct points to perform alignment, whereas it caused issues with RT&R, likely due to the smaller number of points in the submaps and the Radar noise resulting in incorrect ICP convergence.

Localization using virtual submaps for both Radar and Lidar T&R is then done with point-to-plane ICP on deskewed live point clouds to register the motion-undistorted scan against the local virtual submap, again seeded by the outgoing edge transform retrieved from the graph. When ICP converges, the resulting transform $\mathbf{T}_{rm}$ is fused with odometry and written back to the graph as a non-privileged edge; otherwise, the robot continues to dead-reckon until the next live scan. The resulting localization residual that gets fused with the internal odometry estimate in the state estimator, allows the model-predictive controller to track the virtual reference path with centimetre-level lateral accuracy—providing a secondary evaluation metric to assess the performance of VirT&R with the respective conventional T&R method. Once localization is established, the UGV follows the virtual teach path, repeating as if it had been taught in a traditional T&R system. A visualization of Lidar and Radar localization during a real repeat mission is shown in Figure 3.13.


(a) Lidar Submap Localization

(b) Radar Submap Localization

Figure 3.13: Visualization of live sensor scans from real repeats (in green), that have localized to virtually generated teach submaps (in red) for both Lidar and Radar sensors.

3.5 Novel Contributions

This chapter presents the bulk of the novel contributions in this thesis by describing the algorithm developed to extend the capabilities of Teach and Repeat.

- We present the structure of a novel Virtual Teach and Repeat framework, that provides the following advantages:
    a. autonomous navigation through previously untraversed environments without need for previous human and UGV presence or GPS assistance;
    b. creation of multiple teach passes in a desired environment given only one prior venture into the environment with a UAV;
    c. interactive exploration of the entire target environment through a digital twin simulation used for pilot simulation and path definition.
- We conduct an ablation study to determine the optimal hyperparameters for a NeRF model used to generate localization data for a UGV with a Lidar or Radar sensor.

Chapter 4

Evaluation Methods and Metrics

Across modern Teach & Repeat work—RGB vision, Lidar, Radar, and now virtual variants—the methods for quantifying performance have converged on a small set of metrics that tease apart localization quality, controller accuracy, and overall robustness. Performance is evaluated with respect to the robot's ability to retrace the taught path; pure localization error is not reported, because the vehicle's objective is to remain on the reference trajectory rather than to recover its absolute pose in some global frame.

4.1 Existing Methods

The primary metric used is lateral path-tracking error (PTE), assessed as the perpendicular distance from the robot to the closest segment of the discrete teach path. It is reported as a root-mean-squared error (RMSE) over the entire repeat, along with the single worst-case (maximum) deviation to identify outliers. For the majority of modern T&R testing, we use a NovAtel OEM7 GPS [73] enabled UGV robot, together with a separate base station, to provide accurate GPS pose estimates and find the absolute PTE. Since, ultimately, we care about the robot's ability to follow a path, we measure the PTE and do not compute localization error directly. GPS poses are recorded during the teach pass as 3D positions without orientation. For each GPS pose recorded in a repeat pass, we find the closest point in the teach data and compute the distance between them. Specific to Radar assessment, since the Navtech Radar is a 2D sensor, it is typical to restrict our comparison to $\mathrm{SE}(2)$ by omitting vertical (z-axis) errors and reporting heading error as the rotation error.

The secondary assessment available and used mainly in GPS-denied or indoor tests is T&R's own internal signed-distance estimate between the live robot pose and the privileged teach vertex. Agreement between internal and external estimates (when available) is used as a sanity check. Any differences can give some sense of the localization accuracy, as the internal estimate is based solely on robot odometry, and the GPS-measured data also reflects localization. Data for this internal estimate comes from the teach and repeat pose graph paths.

4.2 Virtual Teach and Repeat Assessment Methods

The method for evaluating accuracy and path-tracking performance in VirT&R differs from prior work [7–9, 74, 75] because the teach phase is conducted entirely in a virtual environment. As such, we cannot gather the GPS data for teach paths commonly used to quantify PTE.

4.2.1 Physical Marker-Based Absolute Lateral Path Tracking Error

The lack of GPS data evaluation led us to develop a robust, physical marker-based sim-to-real path-tracking assessment method. Physical markings in the environment are the most suitable tool to assess the UGV's ability to track the virtual teach path because they ensure the PTE accounts for our own errors in piloting the UGV in simulation, as well as localization and control errors, not any that may result from the scene reconstruction process. This is due to the markings appearing in the drone imagery, the scene reconstruction model, and the photo-textured mesh, meaning any drift or distortion from scene reconstruction is applied equally to the markings, the virtual teach path driven alongside them, and the point cloud used for localization. A visualization of how reconstruction error is implicitly cancelled out can be seen in Figure 4.1, where the use of local, potentially drifted, submaps derived from the virtually generated point cloud are localized to the real environment, enabling the desired virtually taught path to be repeated accurately.

We use spray-paint markings as our reference for path generation in simulation as these markings appear in the drone imagery, the reconstruction, and the photo-textured mesh—so any drift or distortion in the scene is applied equally to the markings, the simulated teach path (which is driven along those drifted markings), and the point cloud map used for localization. Then, when the UGV repeats, it localizes and drives relative to the real paint marks because the simulated path was driven along the same (drifted) marks on the mesh, leading the UGV to naturally drive along the true-world marks, with drift being implicitly accounted for, thus not contributing to the measured error.

To implement this methodology, a teach pass is conducted with the sensor the virtual map will use (which acts as our baseline), and the robot is periodically stopped to spray paint marks near the rear-right tire. These marks are then captured in the reconstruction and piloted alongside of

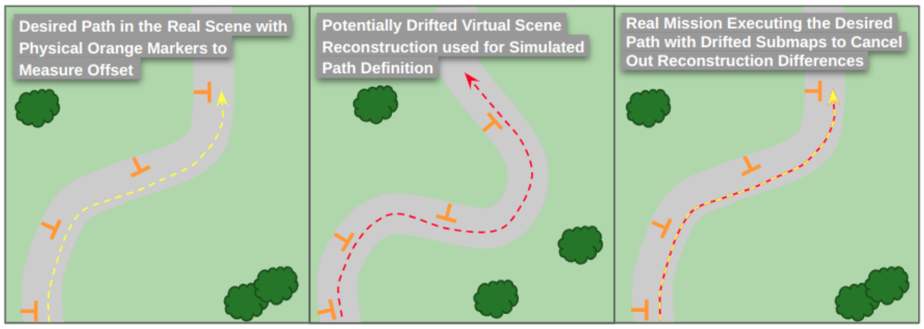

Desired Path in the Real Scene with Physical Orange Markers to Measure Offset
Potentially Drifted Virtual Scene Reconstruction used for Simulated Path Definition
Real Mission Executing the Desired Path with Drifted Submaps to Cancel Out Reconstruction Differences

Figure 4.1: A visualization of a training image with spray paint markings and the desired Teach path, an exaggeration of what potential drift could do to the photo-textured mesh and the driven path, and what the real world mission would be as drift is equally applied to the markings, path, mesh, and point cloud correctly recovering the teach path upon a Repeat.

as closely to the distance they were painted from the wheel as possible to generate a comparable VirT&R teach pass so offsets could be measured on the real route. Fig. 4.2 illustrates the jig used to apply and measure the marks during the real repeat.

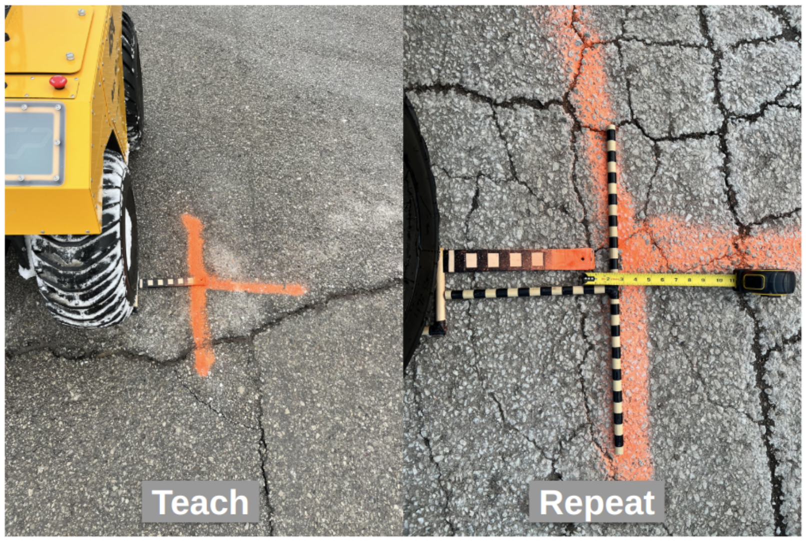


Figure 4.2: Left: Use of the jig to repeatedly make spray paint marks next to the rear right tire for assessment of path tracking accuracy during an LT&R teach path. Right: Measurement of the spray paint mark during a repeat of the virtual teach path. Repeats were also done using standard T&R to obtain measurement offsets for comparison.

To validate performance, we repeat to the T&R and VirT&R teach maps, respectively, so that mark offset measurements can be taken for both algorithms, enabling a direct and absolute lateral path-tracking comparison. Although aligning the robot with the marks in the simulation is inherently imperfect, this challenge mirrors the real-world difficulty of precisely following a physical reference path. It is also of note that the comparison of VirT&R and T&R with offset measurements to spray paint markings does not actually compare the exact same path. The path driven in the real scene to place the marks is not exactly the same as what is driven in the Gazebo simulation. In the simulator, the pilot follows the spray paint markings as accurately as possible, but this is not exact, just as it would be difficult to pilot a UGV to exactly follow a path in the real world. Thus, this assessment of lateral path-tracking error is also subject to piloting error. While the paths are different, we observed the offsets to be small, particularly when close to the markings.

To account for human error in the pilot simulation, a ruler model was attached to the Warthog in the Gazebo, allowing measurements from the Warthog's tire to the spray paint mark to be taken in the simulation as well. By factoring in the jig used to place the spray paint marks, we can better quantify the lateral PTE by subtracting the error attributable to pilot imprecision from the VirT&R measurements, providing a more accurate benchmark of performance and the sim-to-real gap in creating real missions.

Additionally, there are several factors in the process of measuring the offset to spray paint marks that contribute uncertainty in the measurement. Firstly, the measuring tape used is in centimeters with an error of $\pm$ 0.05 cm (half the nearest division), which contributes a standard uncertainty of

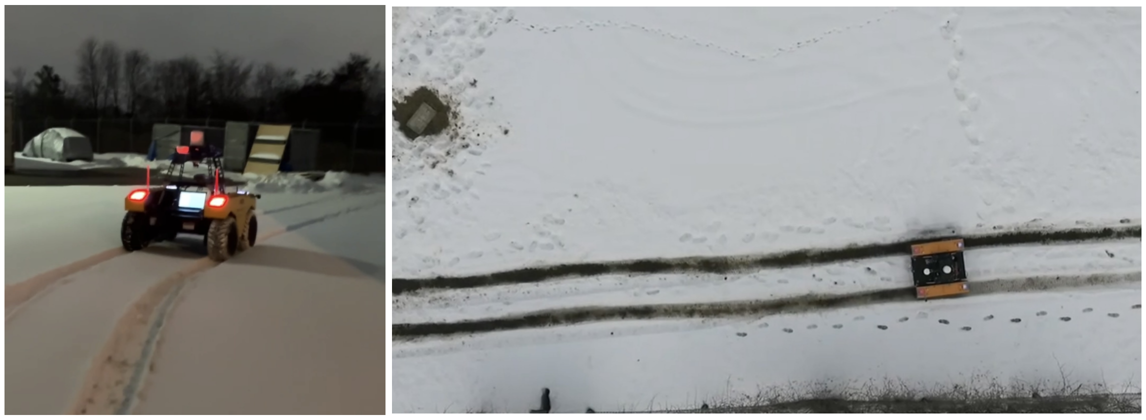
0.03 cm when divided by $\sqrt{3}$ as it is a uniform distribution. Additionally, because the measurments are taken by measuring the distance between the jig that fits right against the tire (assumed to have no error) and a wooden T-shaped dowel aligned with the spray paint marking to ensure a perpendicular measurment is taken, there is a factor of human error in aligning the dowel with the spray paint mark. Several measurements taken with the dowel placed each time resulted in an error of 0.25 cm, which leads a standard uncertainty of 0.01 cm. Lastly, and most significantly, is the error contributed to the PTE measurement created by stopping the UGV at the spray paint mark. The pausing of the UGV's autonomous path following sometimes results in a shift in position of the tire or slight jerky motion as it is an abrupt stop. In taking the measurements, the largest error due to the unpredictable shift was approximately 7 cm, although it was often negligible. It should be noted that this error cannot be directly measured as it is created randomly by stopping the robot's motion and does not occur if the robot is not stopped. Thus the worst-case scenario contributes an uncertainty from stopping the robot of $\frac{7}{\sqrt{3}}$ which gives 4.04 cm. The overall combined standard uncertainty for the measured offset of the rear right tire to the spray paint mark to quantify the lateral PTE is then found by adding the contributing factors in quadrature, resulting in a $\pm$ 4 cm uncertainty on the measurement.

4.2.2 Relative Repeat Path Deviation with Virtual Teach Submaps

In lieu of GPS data and the continuous path assessment it provides, the internally estimated PTE (signed distances between the vertices in the pose graph chains) for two repeats to a teach map was used to determine the relative deviation between repeats to virtually generated teach maps. Comparing two repeat paths is a useful metric in addition to spray paint marking offset measurements, as it provides a continuous assessment of the path and compares the exact same path to itself, rather than a path taught in real life and replicated in simulation to the best of a pilot's abilities.

Given that this internally estimated metric has been used in other T&R evaluations between a teach and repeat pass, it is deemed to be a useful assessment. The goal of this metric is not only to supplement the spray paint marking offset measurements for PTE but to ensure that virtual submaps provide reliable localization. By analyzing the signed distance between the vertices in the pose graphs of two repeats, we observe both localization and control error, and can quantitatively gauge if the geometry reconstructed from the aerial images of the environment is accurate enough to provide correct ICP alignment each time. The findings from this assessment method can often be supported with physical evidence of good path tracking, such as subsequent repeats driving exactly in tire tracks through the snow as seen in Figure 4.3, or through puddles, grass, and along the painted lines of a road.

We can also use this metric to observe repeatability over time. We found no identifiable increase in relative repeat RMSE or max error when comparing repeats that had cars moving in different spots, melted snow banks, or a completely different arrangement of cars in a parking lot. However, attempting to repeat a path in the summer using a NeRF-generated virtual teach map created in the winter resulted in the robot swerving and jolting unpredictably at a level inconsistent with what we know constitutes a good repeat. The virtual point cloud from the NeRF reconstructions



(a) PoV of The Warthog Repeating in its Tracks (b) Aerial View of The Warthog Repeating in its Tracks

Figure 4.3: Visual evidence of low relative repeat deviations during real repeat missions using virtually generated teach maps on the Warthog. Relative repeat deviation can be qualitatively evaluated by observing no drift outside of tracks in the snow created by previous repeats.

is qualitatively observed to be slightly less geometrically accurate than what Pix4D can provide. As we hadn't yet worked with Pix4D in the winter, we have no direct comparison of how large seasonal changes would affect the photogrammetry reconstruction method. It appears that the lack of snow and increased vegetation from seasonal changes everywhere in a scene is enough to hinder performance when using NeRF-derived submaps. Given that the internally estimated PTE is used in traditional T&R PTE assessments, and that we found it to be within the uncertainty of the measured lateral RMSE for the Mars Dome and UTIAS Parking Lot Loops in the initial VirT&R validation tests, we continue to use it in our expanded testing.

It should be noted that this estimate does not directly quantify localization error, as it is based solely on the path taught and the path repeated. To bridge this gap, we also recorded GPS data for repeat paths done with VirT&R so the relative deviation can be assessed continuously in this manner as well. If the internal T&R estimate reports a smaller error than that found using the GPS data of two repeats, the localization of the virtual scan to the live scan is likely the dominant error factor. If the internally estimated relative deviance is similar to the GPS-measured relative deviance, it is likely that the controller is contributing more to the error. In general, small, near-zero centimetre-level errors for both metrics, whether used to assess a virtual teach and a virtual repeat or the deviation between two virtual repeats, indicate that we are repeating within the realm of noise for localization and control. This intuition further highlights the benefit of using these metrics.

Additionally, we tend to place a higher value on the spray paint marking measurements as they more concretely showcase the PTE obtained during the experiments, anchored in reality. This notion is supported by the irrefutable evidence of low relative repeat deviation we observed during our initial validation VirT&R experiment in the winter, as the robot was repeating right in its tire tracks in the snow (see Figure 4.3 above).

4.2.3 Pseudo-GPS Data for Absolute Lateral Path Tracking Error

As mentioned in Chapter 4.1, the T&R absolute PTE is typically assessed by comparing recorded GPS for the teach and repeat passes. This is not immediately possible with VirT&R as, regardless of the reconstruction method, there is no way to record GPS data for the virtual teach pass.

In an effort to enable this assessment, we attempted to create ‘pseudo-GPS’ for virtual teach passes. The path driven in simulation is recorded as an absolute pose with respect to the mesh–which can be geo-referenced if that information is provided during the reconstruction phase–and subsequent relative transformations. This path is used in the teach pose graph with a covariance of zero, as we assume it is perfect odometry. Thus, it is theoretically possible to compound all the transformations that are applied to the geo-referenced digital twin reonstruction’s mesh as a result of putting it into the Gazebo driving simulation to determine how to go from the real-world GPS frame–established by the aerial imagery GPS poses–to the Gazebo simulation frame where the path is defined. This would then recover the virtually defined path in Earth-relative coordinates, providing an equivalent dataset with which we can compare the logged repeat NovAtel GPS data.

The following equations are used to recover the path driven in simulation on a mesh made using COLMAP and NeRF for GPS evaluation:

$$\mathbf{T}_{WG} = \mathbf{T}_{WC}\,\mathbf{T}_{CN}\,\mathbf{T}_{NG}, \tag{4.1}$$

$$\mathbf{p}_{W} = \mathbf{T}_{WG}\,\mathbf{p}_{G}\ , \tag{4.1a}$$

where W denotes the world frame, G denotes the Gazebo frame, C denotes the COLMAP geo-referenced frame, and N denotes the NeRF frame. The same is done for Pix4D, where the equations are slightly simpler due to Pix4D directly reporting a single GPS offset translation as an output.

$$\mathbf{T}_{WG} = \mathbf{T}_{WP}\,\mathbf{T}_{PG}, \tag{4.2}$$

$$\mathbf{p}_{W} = \mathbf{T}_{WG}\,\mathbf{p}_{G}\ , \tag{4.2a}$$

where W denotes the world frame, G denotes the Gazebo frame, and P denotes the Pix4D geo-referenced frame.

We used (4.1)–(4.1a) and (4.2)–(4.2a) to obtain the path driven on the NeRF and Pix4D-generated meshes in Earth relative coordinates respectively so they could be compared to the GPS recorded by the NovAtel GPS sensor on the Warthog during a VirT&R repeat pass.

The resulting plots can be seen in Figure 4.4 for NeRF and Pix4D, respectively. The path-tracking RMSE and maximum error obtained by comparing the VirT&R teach pass pseudo-GPS to the NovAtel GPS recorded during a repeat for the Parking Loop were found to be 0.475 m and 2.914 m for NeRF reconstructions and 0.324 m and 1.057 m for Pix4D reconstructions.

The internally estimated RMSE and maximum error for these VirT&R runs are 0.152 m and 0.434 m for the NeRF reconstruction and 0.069 m and 0.332 m for the Pix4D reconstruction. Following the intuition detailed in Chapter 4.2.1, we can conclude that localization error is the domi-

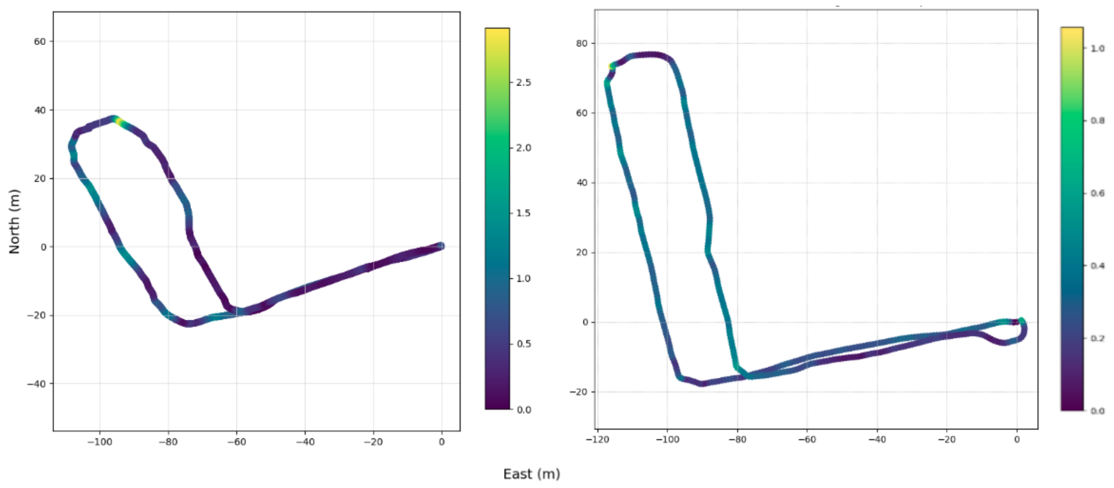


Figure 4.4: Plots of NeRF (left) and Pix4D (right) generated virtual teach pseudo-GPS vs Real Repeat GPS for Parking Loop missions shown in a relative ENU frame as a heat map of absolute lateral Path Tracking Error.

nating contributor to the overall PTE. However, given that the RMSE and maximum errors reported between a teach and a repeat for both NeRF and Pix4D are relatively high compared to the internal estimate, and there is no way of knowing what portion of the elevated error reported is real or is from the compounded transformations used to create the data, thus it is not practical to use pseudo-GPS to assess a PTE that is nearly 2.5-3.5 times smaller.

The result of the effort towards pseudo-GPS for virtual teach paths is that we determined too many sources of error are compounded through factors such as GPS receiver differences, a weeks-long timespan between GPS data collection, reconstruction inaccuracies, NeRF model training, and GPS post-processing for any meaningful assessment to be done.

4.3 Novel Contributions

This chapter establishes the methodology used to evaluate Virtual Teach and Repeat.

- We present a novel experimental methodology for quantitatively evaluating the lateral path-tracking error of Virtual Teach and Repeat across the sim-to-real gap and against existing real-world benchmarks.
- We analyse the ability to create and use pseduo-GPS data generated from a digital twin environment used for path definiton as a baseline for assessing the lateral path-tracking error of a UGV as measured by a real-world GPS system.

Chapter 5

Preliminary Virtual Teach and Repeat Validation

To validate the proposed VirT&R pipeline, we designed an experiment that allows us to benchmark its performance against the current version of Lidar Teach and Repeat, using the methods outlined in Chapter 4.2. Our experiment includes capturing various scenes with different geometries around the UTIAS campus, producing virtual teach maps for routes driven in simulation, and conducting real missions along the virtually defined routes. The current state of the LT&R pipeline as of this experiment in February 2025 did not include the continuous time formulation of Lidar ICP and gyroscope modifications mentioned in Chapter 2.1.1 and 6.1.1 that were used in the subsequent experiments. Additionally, VirT&R had not yet been expanded to work with a Radar sensor.

5.1 Testing Routes

VirT&R was tested on four routes in Fig. 5.1 at the UTIAS campus. The UTIAS Parking Loop and UTP Loops primarily consist of paved paths, buildings, and cars. The Mars Dome route features a partially grass and snow-covered path, slight elevation changes, increased vegetation, and fencing, in addition to buildings, paved roads, and cars. Detailed visual cues, such as salt stains, tire tracks, puddles, and shadows embedded in the NeRF models enabled the pilot in the Gazebo simulation to conduct high-fidelity terrain assessment.

The Mars Dome was particularly difficult to capture and to traverse, as large patches of 0.5 m of snow build-up hindered NeRF training and required a path to be shovelled for the UGV. This resulted in some parts of the route having less than 30 cm of lateral clearance during tight turns. Additionally, in the UTP Loop, two routes were taught and repeated to showcase one of the benefits of the virtual teach pass: multiple routes through one environment without any new scene data capture.

A total of 12.4 km of autonomous driving was conducted to demonstrate the consistent efficacy of VirT&R and qualitatively assess its performance. Table 5.1 presents the environments and

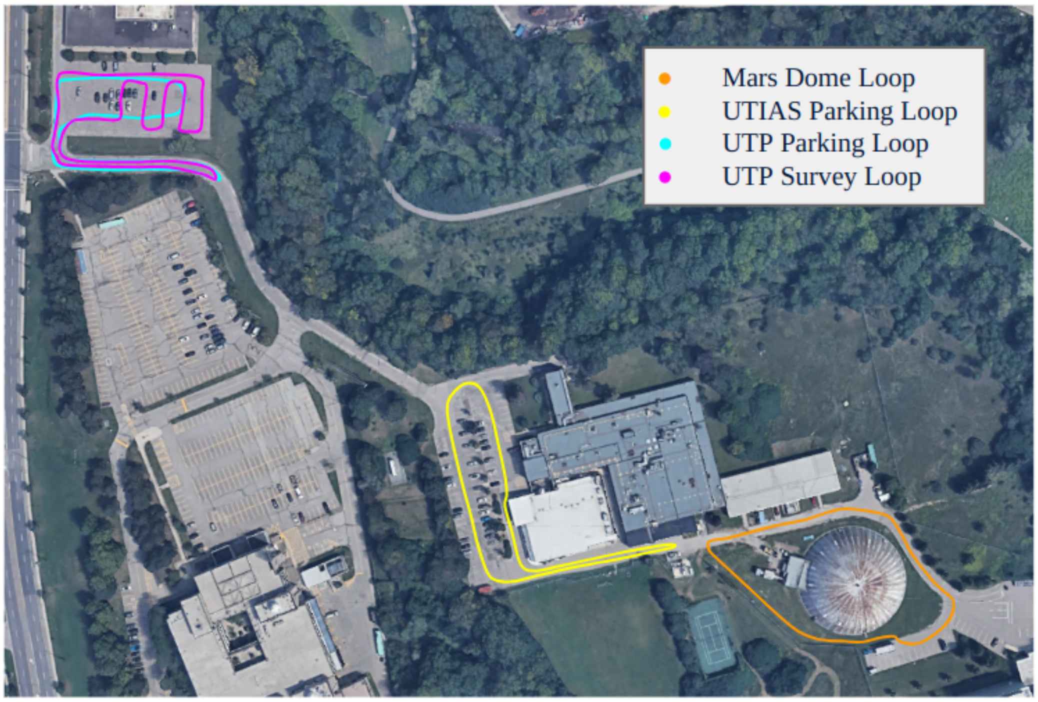


Figure 5.1: Visualizations of all four loops on which VirT&R was evaluated, consisting of urban settings with buildings, cars and paved paths, as well as slightly more off-road settings with vegetation, trees, fencing, and grassy paths.

corresponding distances driven for teaches and repeats respectively. The Mars Dome and UTIAS Parking Loops are evaluated using spray paint markings to obtain an absolute lateral path-tracking error, as permission was obtained to mark the pavement only in those locations. Two example repeats for each of the four routes are also compared to each other to quantify T&R's relative repeat path deviation in a continuous manner when using a virtual teach map. Additionally, five repeats were performed using the baseline algorithm on the Mars Dome and Parking Loops, resulting in an overall total of 15.6 km of autonomous driving.

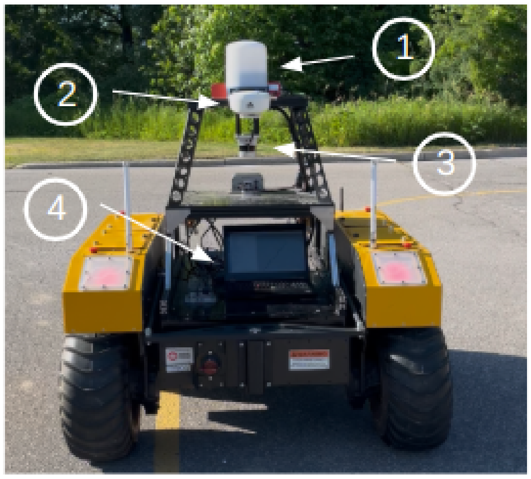
Table 5.1: Breakdown of the Teaches and Repeats done as a part of the initial VirT&R validation experiments at UTIAS.

Path	**Teach Distance**	**# and Distance T&R Repeats**	**# and Distance VirT&R Repeats**
Mars Dome Loop	290 m	5 (1.45) km	10 (2.9 km)
UTIAS Parking Loop	350 m	5 (1.75) km	10 (3.5 km)
UTP Parking Loop	366 m	-	10 (3.7 km)
UTP Survey Loop	463 m	-	5 (2.3 km)
Total	1469 m	10 (3.2) km	35 (12.4 km)

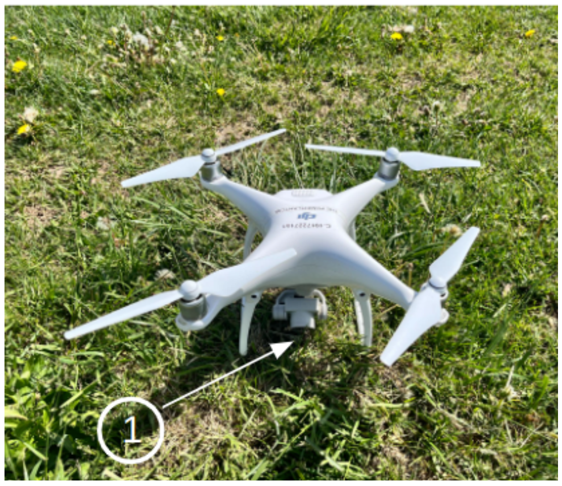
5.2 Experimental Platform

The Clearpath Warthog UGV [14] is the UGV platform used for testing the VirT&R algorithm. The Warthog runs T&R during the online mission execution phase using the virtually generated pose graph map. We have equipped our Warthog with an Ouster OS1-128 Lidar [16] operating in 512 x 64 scan mode at 10 Hz, a Navtech FMCW Radar, a NovAtel OEM7 GPS system [73], and the onboard laptop running Teach and Repeat is a Lenovo Thinkpad with an Intel Core i7-9750H CPU @ 2.60 GHz × 12. The Warthog can be seen labelled in Figure 5.2a.

For the aerial surveying flights done to collect imagery for the VirT&R pipeline, we used a DJI Phantom 4 Pro [15] that has a 1-inch CMOS sensor with 20MP effective pixels, a fixed f/2.8 aperture, and has an 84° fixed field of view for its 3-axis stabilized, gimballed camera. Images were obtained by recording HD videos with the Phantom (1920 x 1080), and geo-tagged using the time they were taken in the video and the starting GPS location of the drone. The Phantom can be seen labelled in Figure 5.2b.



(a) Warthog Components: (1) Navtech FMCW Radar Sensor, (2) NovAtel OEM7 GPS, (3) Ouster OS1-128 Lidar, (4) Lenovo Thinkpad



(b) Phantom Components: (1) Gimballed 1-inch 20 MP camera

Figure 5.2: Labelled images of the Clearpath Warthog and DJI Phantom 4 at UTIAS.

We chose to record video at the expense of higher-resolution imagery for initial experiments, allowing us to tune the number of images provided to COLMAP and NeRF without needing to recapture the scene. We found that images taken at 2.5 Hz (approximately every 0.5 m on average) provided an overlap of roughly 80% on average, which was sufficient for the pipeline to produce good reconstructions. The tradeoff for resolution was not a hindrance, as we also found COLMAP's pose estimation suffered when using images with a resolution above 960 x 540.

5.3 Results

Qualitative assessment of repeat performance on all routes reveals that the robot can reliably follow the virtual path over long distances, navigate narrow openings, complete tight turns, and consistently hit specific marks. The NeRF meshes provide sufficient detail to infer scene characteristics—including potential dynamic elements—and the point clouds accurately represent the environment, even in predominantly flat areas. The UGV follows a smooth route and is resilient to both stationary and moving vehicles, pedestrians, and melted snowbanks.

5.3.1 Physical Markers

To quantitatively compare VirT&R to LT&R, we implemented our physical marking-based evaluation methodology to obtain the lateral path-tracking error, since GPS data is only reliably usable for repeat paths and is not available for the virtual teach pass. Measurements for the VirT&R and LT&R repeat methods were taken five times for each mark on the Mars Dome and UTIAS Parking Lot Loops, using identical controller parameters, and can be found plotted in Figure 5.3.

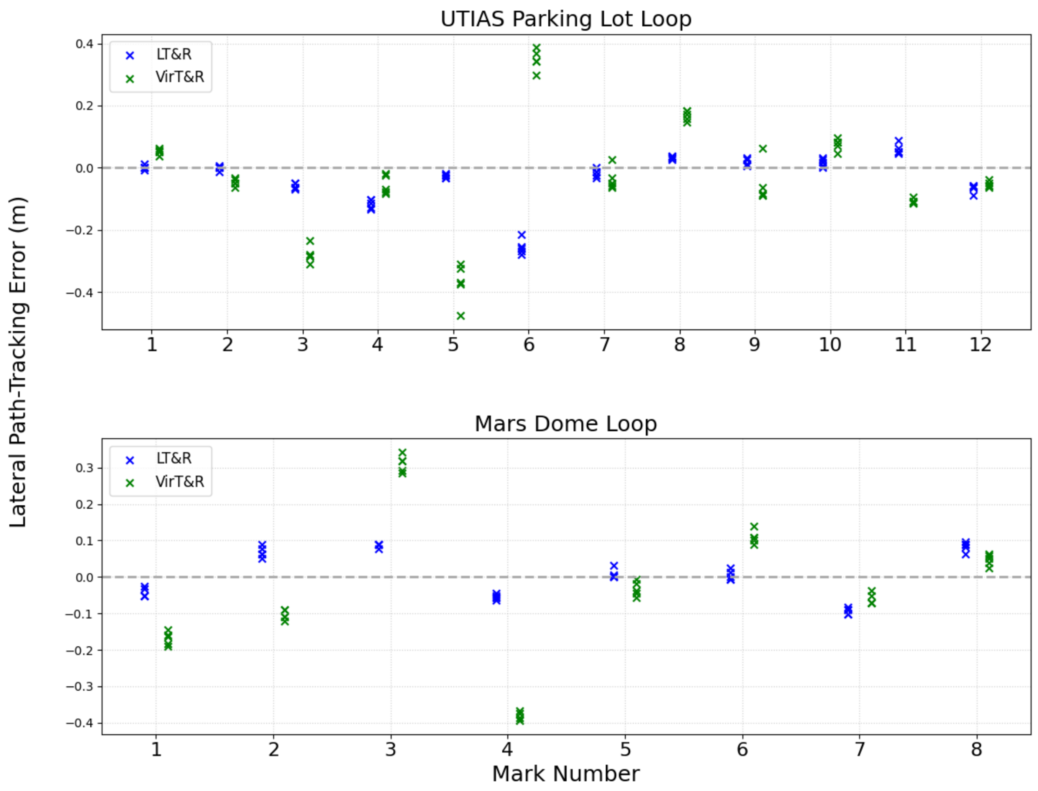


Figure 5.3: Lateral path-tracking error distribution for five measurements of each mark on the UTIAS Parking Lot and Mars Dome Loops taken for both VirT&R and LT&R.

Due to the nature of physically measuring these marks, an uncertainty of ±4 cm has been associated with the readings to account for the measuring tape's precision as well as the human error in placing the jig and stopping the Warthog next to the mark, as described in 4.2.1. Thus, we expect the PTE of baseline LT&R repeats we measure, seen in Figure 5.3, to agree with recently published data within this uncertainty envelope [11]. Although the offset is consistently greater for VirT&R, the spread of the measurements is less than 10 cm for all marks, suggesting that VirT&R's localization performs similarly to LT&R, allowing for the robot to repeat in its tracks—a finding we observed qualitatively, as faint tracks were visible after going through snow and puddles.

A summary of the lateral RMSE and maximum error values for VirT&R and LT&R, determined through physical measurement, can be seen in Table 5.2, which also lists the T&R-estimated lateral and maximum errors obtained from the signed distances between every vertex in the teach and repeat pose graphs.

Table 5.2: Estimated and measured RMSE and maximum path-tracking errors for VirT&R and LT&R across two different environments.

Modality	**Path**	**T&R-Estimated Lateral RMSE [m]**	**T&R-Estimated Max Lateral Error [m]**	**Measured Lateral RMSE [m]**	**Measured Max Lateral Error [m]**
LT&R	UTIAS Parking	0.076	0.207	0.089	0.241
	Mars Dome	0.056	0.267	0.109	0.279
VirT&R (Ours)	UTIAS Parking	0.152	0.434	0.184	0.476
	Mars Dome	0.189	0.581	0.195	0.394

The difference between the LT&R estimated and measured values is within the uncertainty associated with this method and indicates that physically measuring lateral path-tracking error is a reliable assessment. Additionally, LT&R estimated performance also aligns with a previous benchmark [11]. The measured lateral RMSE for VirT&R was 18.4 cm on the UTIAS Parking Loop and 19.5 cm on the Mars Dome Loop, which both fall within the range of LT&R maximum errors. The maximum estimated VirT&R error for the Mars Dome Loop is likely higher than the measured value because the internal T&R estimate considers all vertices in the pose graph of a repeat, and the spray paint mark was likely not at the specific vertex with the highest offset, particularly as they were intentionally driven close to. The maximum VirT&R measured lateral path-tracking errors are 47.6 cm and 39.4 cm for the UTIAS Parking Loop and Mars Dome Loop, respectively, signifying there is less accuracy with VirT&R than with LT&R, as expected. Overall, these results highlight that localization via NeRF-generated point clouds is a comparable method.

5.3.2 Relative Repeat Path Deviation Error

Across the four routes in Fig. 5.4, the T&R estimated relative repeat deviance increased from nearly zero to 55.4 cm, with the UTIAS Parking Loop exhibiting the best repeatability and the Mars Dome Loop having the worst. The portion of the Mars Dome Loop where the deviation is higher is also the area of the path with trees, slightly less structure, and where intentionally rapid heading changes were driven as a performance test. The T&R estimated maximum relative error for all routes driven is found in this swerving section of the Mars Dome, near spray paint mark #4, which also has the highest measured lateral path-tracking error for this loop of 39.4 cm, seen in Table 5.2.

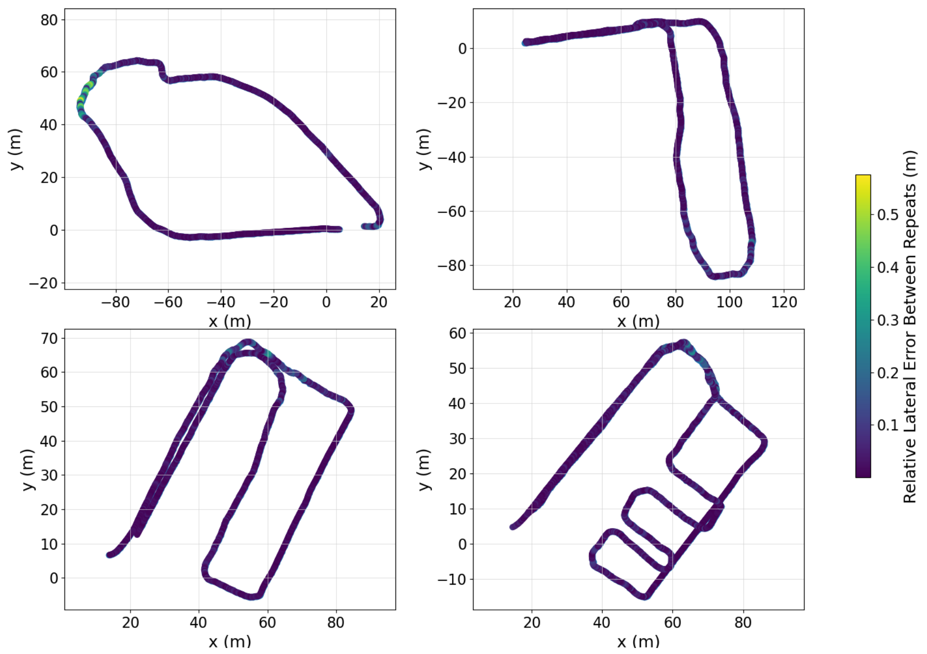


Figure 5.4: A comparison of two example repeat pose graphs for all four routes. The relative lateral path-tracking error between the trials is highlighted using a color scale and is observed to increase during tight turns or quick, precise manoeuvring.

5.4 Discussion and Future Work

Enabling autonomous control of a UGV via the T&R framework using NeRF-generated point cloud submaps and a virtually defined path as the sole inputs proved challenging in multiple respects. A significant observation was the difficulty in generating point clouds and meshes with minimal noise and accurate geometry under challenging snowy conditions, as NeRF model training noticeably suffered in environments with large white swaths of untouched snow compared to those with no snow. Additionally, although VirT&R was capable of running without issue in all the environments considered in this experiment, attempts to use VirT&R on the Grassy Loop—another commonly used route for T&R testing [11]—resulted in unpredictable localization likely due to the highly unstructured environment and reconstruction inaccuracies in the virtually generated point cloud. This highlights a potential pitfall of the method and will be investigated further by exploring alternate reconstruction options and additions to the T&R pipeline to aid in reducing localization error. Moreover, the methodology used to evaluate the performance of VirT&R relative to the virtual teach path is subject to inherent errors, both from how the Warthog is paused to take measurements and from positioning the jig to measure the marks on the ground. It is likely that if these errors can be reduced, the evaluated performance would more closely approach that of LT&R. As the current experimental structure does not lend itself to being fully evaluated using GPS measurements, improvements to creating GPS data for the teach path are also a future possibility.

5.5 Conclusions

The path-tracking performance provided by aligning NeRF-generated submaps to live Lidar scans on four different routes and over 12 km of uninterrupted autonomous driving showed VirT&R performed comparably to LT&R, achieving measured RMSE values of 19.5 cm and 18.4 cm and maximum errors of 39.4 cm and 47.6 cm. Relative repeat path analysis confirms the qualitative observation that VirT&R smoothly and repeatably follows its taught path, regardless of environmental changes or the presence of moving objects. We intend to improve the methodology for assessing VirT&R's accuracy and the pipeline in general, for subsequent testing and use, to ensure its functionality remains robust in increasingly unstructured environments.

5.6 Novel Contributions

This chapter presents the initial experimental results for the path-tracking error and performance of Virtual Teach and Repeat using NeRF-derived point cloud submaps for UGV localization.

- We are the first to achieve zero-shot autonomous UGV navigation in a variety of GPS-denied, outdoor, untraversed environments using localization layer data obtained from NeRF volumetric rendering.

Chapter 6

Expanded Field Testing of Virtual Teach and Repeat

This expanded field testing experiment includes capturing different scenes with more varied geometries and longer routes around the UTIAS campus, to produce virtual teach maps for both Radar and Lidar T&R, while assessing NeRF and Pix4D-derived localization layer performance. The autonomous repeating done using VirT&R, detailed in Chapter 5, showed that smooth and reliable path tracking with an RMSE range slightly under one tire width (24 cm) was possible using the NeRF-based reconstruction method in snow-covered environments with varying levels of distinct geometry. The primary goal of the testing conducted in this chapter is to perform several comparisons on key variables in the pipeline and operational environments to further elucidate the strengths and weaknesses of the method, improve its performance, and gain a more nuanced understanding of deploying the algorithm.

As mentioned in the beginning of Chapter 5, updates were made to the LT&R pipeline to enable continuous ICP and include high-frequency gyroscope measurements after the first experiment. Details regarding these updates can be found in Sections 2.1 and 6.1.1. Additionally, the ability to create virtual Radar point clouds using the point cloud from scene reconstructions was explored and tested so that VirT&R could be used with the existing Radar T&R pipeline [11]. This feature is detailed alongside the Lidar implementation in Chapter 3.2.4 and enables even more cross-modality within our system.

Furthermore, the Pix4D photogrammetry reconstruction tool was implemented as an alternative method to the NeRF-based reconstruction. We found that the photogrammetric reconstruction from Pix4D produced more crisp and geometrically accurate meshes and point clouds. Evaluating the differences between a learned method and a classical method in this chapter, each with distinct strengths, may reveal whether one method is more suitable for reconstructing a specific environment type. The process for using Pix4D for photogrammetry-based reconstruction is also described in Chapter 3.2. We also modified the submap association process slightly between initial validation and expanded field testing to enable creating pose graphs of larger scenes that could not be entirely captured in a single NeRF or Pix4D reconstruction due to increased training time or image upload limits. Details about how longer paths are created by modifying the submap association process can be found in Chapter 6.1.1.

Together with the integration of virtual Radar submaps, photogrammetry, and larger scene capture capabilities, this chapter compares the performance of VirT&R across a wider range of environment types, different sensing modalities, and different reconstruction tools. The comparisons covered in this chapter, enabled by the upgrades made to the pipeline, are as follows: Radar vs Lidar Virtual Teach and Repeat, NeRF vs Pix4D scene reconstruction, and off-road vs structured environments.

6.1 Testing Routes

The testing routes selected for the expanded field testing of the VirT&R algorithm cover a wider variety of environments, as the upgrades and new tools added to the VirT&R pipeline after its initial validation (see Chapter 3 and 6.1.1) enabled zero-shot autonomous route repeating in more challenging scenarios. The four routes considered in this experiment are the Mars Dome Loop, the UTIAS Parking Loop, the UTIAS Big Loop, and the Grassy Loop, which can be seen in Figure 6.1 overlaid on a satellite view of the UTIAS campus to showcase the respective terrain features.

Both the Mars Dome and the UTIAS Parking Loop were included in the initial validation experiments, and the routes through these environments remain largely unchanged, aside from seasonal differences that result in the absence of large snow build-up patches.

The Grassy Loop is another common route used for T&R testing at UTIAS. It consists of an entirely off-road path through a field of approximately 0.5 m tall grass surrounded by trees, with the only distinct geometry being a fence, a satellite antenna, and the corner of a distant building. This route was captured in the same way as the others; however, because of the tall grass in the scene, the reconstructed mesh and point cloud are not as accurate to the environment as they would be if the grass were trimmed. The mesh and point cloud do not incorporate interactions with the environment, such as being able to push through tall grass, which is what the Warthog does when it traverses this path. This route is the main new addition that poses further challenge for the algorithm, as during the initial testing of VirT&R with NeRF-based reconstructions, we found we were unable to operate with VirT&R on the Grassy Loop which we suspected was due to the NeRF model's inability to reconstruct the geometrically devoid environment.

The UTIAS Big Loop is a path that traverses the three aforementioned environments, showcasing the new ability to create virtual teach maps from multiple smaller reconstructions to enable much more vast areas to be covered by VirT&R. The scenes that make up the UTIAS Big Loop are the Mars Dome Loop, the UTIAS Parking Loop, and the Grassy Loop, meaning a huge range of environments from a structured parking lot to a tall grass-filled field are covered in this one route.

Due to the known challenge of using NeRF-based reconstructions on the Grassy Loop, the NeRF vs Pix4D comparison will only be evaluated on the Mars Dome and UTIAS Parking Loops, while

Pix4D reconstructions will be evaluated on all routes. In both reconstructions, detailed visual cues such as pavement cracks, prior Warthog tracks through grass, puddles, and shadows are present, enabling high-fidelity terrain assessment by the pilot in the Gazebo simulation. It should be noted that due to the tall grass in the Grassy Loop, it is not evaluated with spray paint marks to obtain path-tracking offsets, and neither is the portion of the UTIAS Big Loop that covers the Grassy Loop, as it is not possible to place marks there. The Grassy Loop will be evaluated solely using the internal T&R estimated PTE and the relative repeat deviation metric.

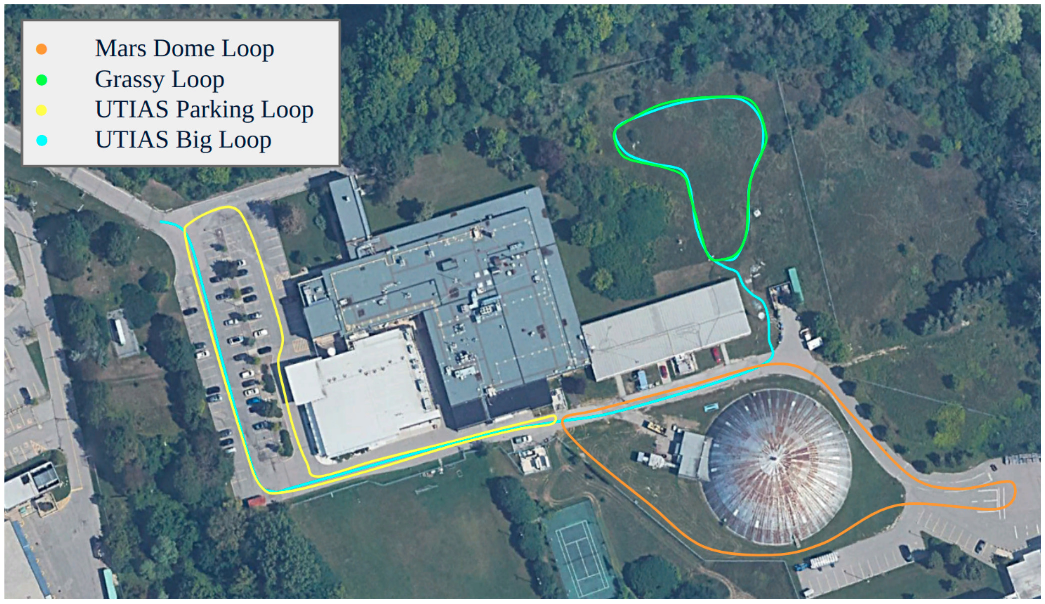


Figure 6.1: Visualizations of all four loops on which VirT&R was evaluated, consisting of urban settings with buildings, cars and paved paths, as well as more off-road settings with vegetation, trees, fencing, and grassy paths.

A total of 20.63 km of autonomous driving was carried out for the expanded field testing of VirT&R. All routes with both reconstructions will be evaluated with measured lateral path-tracking offsets using spray paint markings where possible, as in the initial testing of VirT&R, and we will rely on the internal T&R estimate for PTE where this is not possible. We find this to be acceptable given the correlation of the internal T&R estimate and the repeat deviation with the results from the marking offset measurements seen in the initial validation of VirT&R and in other T&R experiments [11]. Each route was repeated five times using each T&R variant we consider here: LT&R, RT&R, VirLT&R and VirRT&R. Additionally, the Mars Dome and UTIAS Parking Loops were repeated five times using both Pix4D and NeRF-generated localization layers where applicable. Table 6.1 summarizes the route lengths and distances driven with the experimental algorithms. An equal amount of driving (5 repeats) was performed with the baseline algorithms, resulting in an overall total of 27.26 km of autonomous driving.

Table 6.1: Breakdown of the Teaches and Repeats done as a part of the VirT&R expanded field testing at UTIAS VirT&R Repeats have been delineated to show 5 repeats were done for some number of methods tested on a given route.

Path	**Teach Distance**	**# and Distance of T&R Teach Distance (Repeat)**	**# and Distance VirT&R Repeats**
Mars Dome Loop	336 m	5 (1.68) km	4 x 5 (6.72 km)
UTIAS Parking Loop	401 m	5 (2.05) km	4 x 5 (8.02 km)
UTIAS Big Loop	451 m	5 (2.25) km	2 x 5 (4.51 km)
Grassy Loop	138 m	5 (0.69) km	2 x 5 (1.38 km)
Total	1326 m	20 (6.63) km	40 (20.63 km)

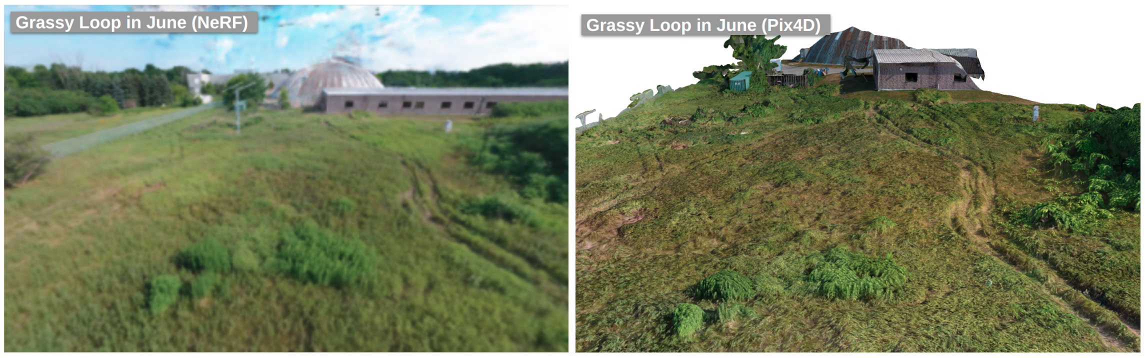
6.1.1 Covering Larger Environments with Sparse Geometric Features

As mentioned, the experiment in this chapter covers a larger environment (the UTIAS Big Loop) and a new environment with more sparse geometric features (the Grassy Loop). These capabilities stem from the described additions made to VirT&R in the time between the initial validation experiment, largely motivated by the results of that experiment and the corresponding conclusions and future work detailed in Chapter 5.5.

The ability to create pose graphs for longer paths was achieved by effectively stitching together the meshes from several separate but adjacent scene reconstructions to create one seamless mesh of a large environment. Larger scenes are effectively pieced together by strategically capturing aerial footage of areas around 200 square meters and ensuring there is approximately a 20% overlap between the scenes. This overlap is what allows the merging of the adjacent meshes, which is necessary for smooth virtual driving.

An important factor in the ability to make the larger environments is ensuring the meshes are seamlessly stitched together so the path driven in simulation is smooth and accurate to the reconstructed environment in a continuous way, so there are no abrupt rotations or translations required to drive the desired path across mesh boundaries that would not be in the real-world scene. This seamless mesh merging was aided by the spray paint marks placed in the scene before UAV flights, as they are reliable and easy points to use for correspondences between scenes in overlapping sections, allowing for the transformation and merging of meshes to create a continuous surface for an environment that is too vast to cover and process in a single aerial flight and reconstruction.

The ability to successfully localize and traverse an environment with more sparse geometry was achieved by using Pix4D for scene reconstruction, as was hypothesized. The increased geometric accuracy in the Pix4D reconstruction captured slight elevation changes, ditches, and thin or small features, such as fences, more accurately and with less noise, which enabled traversal in these challenging environments. Notably, the Grassy Loop was omitted from the initial VirT&R testing with NeRF-generated meshes and point clouds due to unreliable localization; however, it became immediately traversable when Pix4D reconstruction data products were used. The difference between NeRF and Pix4D reconstructions can be seen in Figure 6.2 for the Grassy Loop, which was very sparse in both reconstructions but not traversable with NeRF-generated data products.



(a) NeRF Volumetric Rendering of the Grassy Loop

(b) Pix4D Reconstruction of the Grassy Loop

Figure 6.2: Images taken from Nerfstudio and Pix4D, respectively. (a) Shows the volumetric rendering of the Grassy Loop created using aerial data and Nerfstudio; visually, it is blurrier and has less accurate recovery of geometry in the scene, which is translated to the extracted mesh and point cloud. (b) Shows the Pix4D reconstruction of the Grassy Loop. As Pix4D is a photogrammetry software, it does not provide a volumetric rendering; instead it visualizes the mesh, which is what is shown in the image. It is noticeably more crisp than the NeRF rendering.

Lastly, the LT&R pipeline was upgraded to incorporate the continuous time Lidar ICP formulation, which enables the inclusion of high-frequency single-axis gyroscope measurements, as the current RT&R pipeline does. This addition grows the state being optimized from 6 dimensions to 12 as it now includes angular velocities in addition to translation and rotation components. Including this robot heading information in the ICP formulation improves the initial localization and further constrains alignment throughout path traversal, resulting in more reliable localization in sparse environments. See Chapter 2.1 for a detailed description of this update.

6.2 Experimental Platform

The experiment done in this chapter utilizes the same UGV, a Clearpath Warthog [14], and the exact same sensor rig as in the initial VirT&R experiment. The same drone, a DJI Phantom 4 Pro [15], was also used for the aerial survey flights. A breakdown of specifications, along with pictures, can be found in Chapter 5.2.

In this experiment, however, we opted to collect still-image frames to enable quicker processing (as every captured image comes with EXIF data and GPS position if it is enabled), higher-resolution images, and reduced storage capacity. In the initial UTIAS experiment, images were extracted from video capture, and had GPS locations assigned based on a recorded log and starting location. This was done to allow us to easily vary the number of images used, ensuring we found the optimal number of images for a given scene size without needing to recapture it. We used the Phantom 4 drone to capture geo-tagged 4000 x 2250 resolution images at the highest available rate, 0.5 Hz (one every 2 s), mirroring the data density found to be optimal with the Phantom 4 in previous experiments to achieve the desired approximate 0.5 m spacing and 80% image overlap.

6.3 Results

As with the initial VirT&R experiment, we observed smooth, predictable, and reliable path-tracking from the Warthog using virtual Lidar and Radar teach maps, with both reconstruction tools on nearly all the routes. Tight hairpin turns, narrow openings between thin fixed obstacles, ground divets, uneven terrain, and patches of mud and gravel were successfully traversed by VirT&R. We also observed a similar level of robustness to traditional T&R when our algorithm was faced with dynamic or moved objects like cars, pedestrians, tall grass being blown in the wind, and other scene changes.

The only notable challenge resulting in very poor performance came from using virtual Radar teach maps on the Grassy Loop. Despite using the higher-quality Pix4D-generated point cloud as the localization layer for VirRT&R, traversal of the Grassy Loop was very jerky, unpredictable, and did not follow the path well, swerving in many places. Although the Warthog was able to complete the loop, the behaviour observed was not consistent with what we know to be a successful autonomous repeat, and was thus not included. This challenging environment also rendered the UTIAS Big Loop unusable with VirRT&R, as part of this loop covers the Grassy Loop's field, although it was able to operate in the other two more structured environments (the Mars Dome Loop and UTIAS Parking Loop) that make up the UTIAS Big Loop. This finding confirms that the Grassy Loop is a challenging scene, and that only when using Pix4D and VirLT&R, can it be navigated with virtual teach maps.

Measured and internally estimated RMSE and maximum errors for all permutations of route, algorithm, and reconstruction tool are presented in Table 6.2, obtained from the five repeats done for each configuration. Also included in Table 6.2 are the RMSE and maximum errors calculated using the internally estimated PTE data only at the locations along the path where spray paint marks were placed. This result provides important context for interpreting our results, as the internal estimate considers the entire tranjectory driven by the UGV and the measured marking offsets only capture the RMSE and maximum error in discrete locations, which often leads the internal estimate to present a higher maximum error despite it being an underestimated metric that only considers vehicle control because the actual maximum lateral PTE may not be at the location of a spray paint mark. Thus, given the explanation of the reported metrics in Chapter 4, we expect to see that the T&R estimated maximum error be greater than the estimated maximum error at the mark locations, and possibly near but never less than the measured maximum error due to the internal T&R estimate capturing the PTE along the whole path whereas the marks only capture the PTE at discrete locations.

These metrics, deemed to be valuable and accurate in the initial experiments by comparing their assessment of baseline methods to previous evaluation of the baselines in other T&R tests [11] involving GPS, present the same quality and trends in this experiment. Thus, our initial use of these metrics is further validated, providing confirmation that they offer a meaningful way to assess the performance of VirT&R in lieu of GPS data.

Table 6.2: Expanded field testing internally estimated and measured RMSE and maximum error for T&R and VirT&R across different environments and reconstruction tools. In this table a '-' signifies data that is not reported because it was not possible for the algorithm to run and an 'N/A' denotes data not being applicable due to markings not be able to be used.

Modality	Path	T&R-Estimated Lateral RMSE [m]	T&R-Estimated Max Lateral Error [m]	T&R-Estimated Lateral RMSE at Mark [m]	T&R-Estimated Max Lateral Error at Mark [m]	Measured Lateral RMSE at Mark [m]	Measured Max Lateral Error at Mark [m]
LT&R	Mars Dome Loop	0.035	0.203	0.032	0.052	0.042	0.078
	UTIAS Parking Loop	0.034	0.184	0.051	0.128	0.047	0.114
	UTIAS Big Loop	0.039	0.249	0.055	0.226	0.047	0.152
	Grassy Loop	0.042	0.150	N/A	N/A	N/A	N/A
RT&R	Mars Dome Loop	0.056	0.229	0.068	0.158	0.063	0.133
	UTIAS Parking Loop	0.059	0.228	0.064	0.194	0.067	0.200
	UTIAS Big Loop	0.061	0.449	0.050	0.120	0.065	0.219
	Grassy Loop	0.079	0.354	N/A	N/A	N/A	N/A
VirLT&R Pix4D (Ours)	Mars Dome Loop	0.044	0.198	0.039	0.072	0.054	0.122
	UTIAS Parking Loop	0.058	0.225	0.059	0.130	0.054	0.121
	UTIAS Big Loop	0.165	0.567	0.055	0.144	0.054	0.149
	Grassy Loop	0.248	0.534	N/A	N/A	N/A	N/A
VirRT&R Pix4D (Ours)	Mars Dome Loop	0.070	0.248	0.067	0.175	0.109	0.238
	UTIAS Parking Loop	0.078	0.364	0.079	0.229	0.092	0.227
	UTIAS Big Loop	-	-	-	-	-	-
	Grassy Loop	-	-	N/A	N/A	N/A	N/A
VirLT&R NeRF (Ours)	Mars Dome Loop	0.102	0.367	0.078	0.160	0.057	0.141
	UTIAS Parking Loop	0.112	0.272	0.107	0.231	0.068	0.136
VirRT&R NeRF (Ours)	Mars Dome Loop	0.120	0.632	0.100	0.221	0.189	0.464
	UTIAS Parking Loop	0.099	0.569	0.127	0.481	0.130	0.380

As in the initial experiment, an uncertainty of ± 4 cm has been associated with the readings, given the nature of physically measuring offsets to the spray paint marks includes human error in pausing the Warthog, placing the jig, reading the measuring tape, and the precision of the measuring tape itself. Thus, the baseline LT&R and RT&R methods in Table 6.2 should align with the baseline LT&R and presented RT&R results presented in [11] within this uncertainty.

In analyzing the overall results of our experiment, we found that our physical marking assessment of the baseline methods agrees with previously published RMSE and maximum errors for all paths in common within the ± 4 cm uncertainty. In general, the path-tracking performance of LT&R is between 4.2 and 4.7 cm with a maximum error between 7.8 and 15.2 cm, and the measured path-tracking performance of RT&R is between 6.3 and 6.7 cm with a maximum error between 13.3 and 21.9 cm, depending on the environment. We observed that LT&R outperforms RT&R in terms of RMSE and maximum error 100% of the time, regardless of the path, as expected, for all reported metrics as well. The same result of better path-tracking performance with LT&R was found to be true when comparing VirLT&R to VirRT&R, regardless of the reconstruction tool used, which can be seen in Figure 6.3. We also found that using Pix4D reconstructions as the localization layer source instead of NeRF reconstructions almost always yielded better path-tracking performance on all routes for both virtual Lidar and Radar T&R, as seen in Figure 6.3 as well.

Additionally, we found the T&R internally estimated RMSE to be less than or nearly equal to the measured RMSE, and the internally estimated RMSE only at the mark locations was generally well within the $\pm$ 4 cm uncertainty assigned to the measurements due to the error from the jerky stopping of the robot and the systematic measurement error from the measuring tape and jig placement for all configurations. The only exception to the internally estimated RMSE being lower or equal to the measured RMSE was found on the UTIAS Big Loop with VirLT&R (Pix4D), where the internally estimated RMSE was 16.5 cm compared to the measured RMSE of 5.4 cm, although the internally estimated RMSE calculated only at the mark locations closely aligns with the measured RMSE. This is thought to be due to the very tight corridor the robot must go through on a hill to go from the Mars Dome Loop map to the Grassy Loop map, this section is inbetween marks and only occurs on the UTIAS Big Loop, and was difficult to drive in simulation due to the robot needing to be piloted between support cables for a satellite tower in this narrow corridor.

The only exception to the internally estimated RMSE being within $\pm$ 4 cm of the measured RMSE occurs on the Mars Dome Loop with VirRT&R and NeRF-derived submaps (the most difficult configuration), where the estimated RMSE at mark locations is 10 cm and the measured RMSE is 18.9 cm. This outlier is likely outside the 4 cm uncertainty envelope due to the combination of using the worst performing method (VirRT&R with NeRF-derived submaps) on the Mars Dome Loop that has a mark placed just after the robot exits a hairpin turn, which can be seen in Figure 6.12, which shows the continuous relative repeat path deviation for VirRT&R with Pix4D.

Lastly, as expected, the internally estimated maximum error is always greater than the internally estimated maximum error at the mark locations.

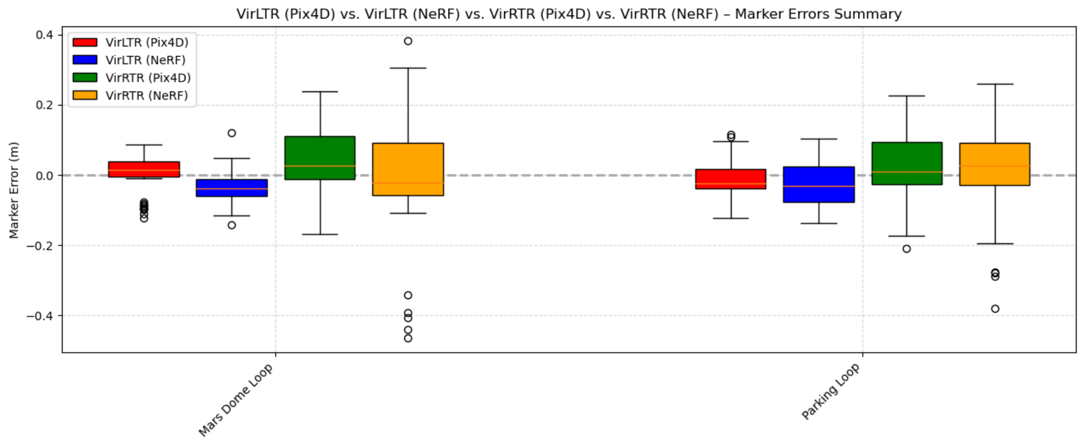
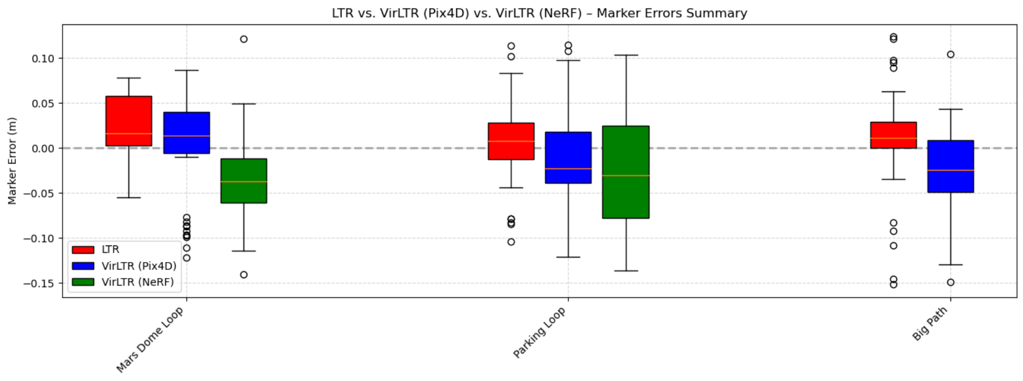


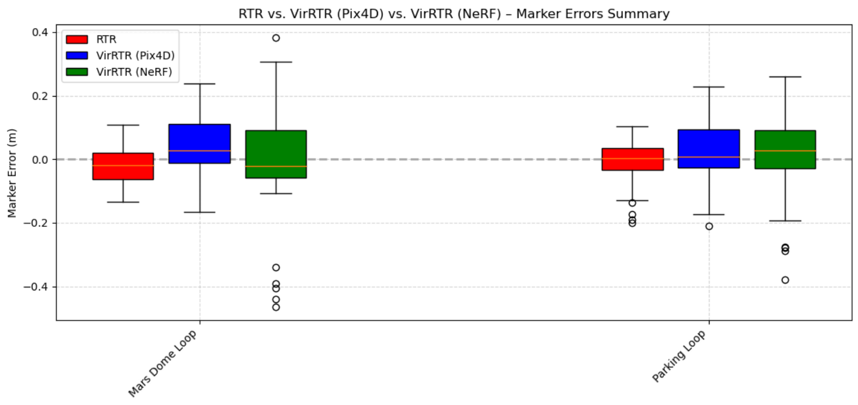
Figure 6.3: Box plot summary of measured marking offsets for Lidar and Radar-based Virtual T&R using both Pix4D and NeRF reconstruction tools on the Mars Dome and UTIAS Parking Loops.

As mentioned, the internally estimated T&R RMSE and maximum error come from an assessment of the signed path-tracking distance over the entire route, we expect that the internally estimated maximum error may sometimes be higher than the maximum error obtained from measuring offsets to the physical markings, as the markings may not necessarily be placed in the areas where the highest error occurs. Additionally, the markings are intentionally piloted alongside of, meaning it is even less likely that they capture the actual maximum error, as this requires extra careful driving. In the case of the baseline methods, the markings were placed during the real-world teaches and thus should provide an even more accurate assessment of the baselines. However, it is still expected that the internally estimated values should generally be lower than the measured values, as outlined in Chapter 4, due to the relationship between control and localization. Although the internally estimated metric quantifies error for the entire trajectory, the metric we choose to quote as the algorithm's performance is the measured lateral PTE obtained from the spray paint marks due to the aforementioned shortcomings of the internal estimated PTE in addition to actually being able to physically observe tangiable proof of the marking-based metric.

We found the overall best configuration to be VirLT&R using Pix4D for the reconstruction tool, both through qualitative observation and by analysing the data in Figure 6.3. A total method comparison showing marking measured offsets for all routes can be seen in Figure 6.4 as well, where Figure 6.4a showcases all Lidar-based repeats and Figure 6.4b displays all Radar-based repeats. Across the four routes tested, VirLT&R obtained a measured RMSE of 5.4 cm on all routes with marks,and the measured maximum errors were 12.2, 12.1 and 14.9 when using Pix4D point clouds as the localization layer for the Mars Dome Loop, UTIAS Parking Loop, and UTIAS Big Loop, respectively. As the Grassy Loop has no markings, the internal estimate provides an RMSE of 24.8 cm (approximately one tire width), and a maximum error of 53.4 cm.



(a) LT&R, VirLT&R (Pix4D), VirLT&R (NeRF) Measured Marking Offsets Per Path



(b) RT&R, VirRT&R (Pix4D), VirRT&R (NeRF) Measured Marking Offsets Per Path

Figure 6.4: Box plot summary of measured marking offsets for Baseline, Lidar and Radar-based Virtual T&R Methods using both Pix4D and NeRF reconstruction tools on all applicable routes.

It is notable that the Mars Dome, Parking, and UTIAS Big Loops all have the same measured

RMSE, although this is not necessarily erroneous, as the Mars Dome and UTIAS Parking Loops generally contain similar or shared structures, and these two environments compose the part of the UTIAS Big Loop that has marks. The baseline performance of the Grassy Loop is not drastically elevated compared to the Mars Dome and UTIAS Parking Loops, meaning it may be possible that the RMSE of the UTIAS Big Loop's Grassy section is elevating an otherwise lower RMSE on the Mars Dome and Parking Loop sections. The values are likely the same to within the associated uncertainty and rounding precision used in the calculation. Furthermore, as shown in the relative repeat analysis below in Chapter 6.3.1, the RMSE of the deviations for VirLT&R across all repeats is very low, corroborating the similar measured RMSE values and likely stemming from the inclusion of heading information and the improved quality reconstruction provided by Pix4D. The Grassy Loop's internally estimated 24.8 cm RMSE and 53.4 cm maximum error are likely elevated compared to the internally estimated RMSE and maximum error of the other VirLT&R routes and the LT&R baseline, due to the complexity and fine detail required to accurately reconstruct a scene composed mostly of moving vegetation. While Pix4D allowed us to traverse this route, it likely does not do a good enough job to provide performance similar to LT&R. Despite this, the performance approaches the measured RMSE and maximum errors found with VirLT&R in our initial experiment on other routes, and as the performance was still smooth and predictable, it does not discount the use of VirLT&R with Pix4D for this environment.

In the following sections, in-depth analysis of virtual Lidar and Radar T&R is presented with path-tracking performance plots that include marking offset measurements for all repeats and marks on all routes included in this experiment with both reconstruction tools, where applicable. We show individual marking offset measurements at the location of the mark along the route, which differs slightly across the three methods as each method has a unique but very similar path (manually driven in the real world for T&R, and in simulation for Pix4D and NeRF VirT&R methods). The corresponding coloured lines represent the internally estimated PTE for all five repeats, averaged to create one line for visual comparison, with the overall RMSE and maximum errors reported.

6.3.1 VirLT&R

When compared to the initial experimental results, there is a general improvement to the LT&R baseline RMSE and maximum error (of roughly 2-6 cm on the RMSE and 0-6 cm on the maximum), likely due to the incorperation of continuous time gyroscope measurements. In the figures below, we observe that the PTE increases from the LT&R baseline, to VirLT&R using Pix4D, to VirLT&R using NeRF, but the spread of measured marking offsets is rarely greater than 6 cm, which is an improvement over the 10 cm observed in our initial experiment, again suggesting that VirLT&R's localization performs similarly to LT&R even more so now with the added reconstruction accuracy in the Pix4D method and the integration of heading gyroscope information. VirLT&R with Pix4D-derived submaps improves upon the initial results when considering either the internal estimate or the measured results. Notably, the maximum measured error is significantly improved. PTE in show in Figure 6.5for the Mars Dome Loop, Figure 6.6 for the UTIAS Parking Loop, Figure 6.7 for the

UTIAS Big Loop, and Figure 6.8 for the Grassy Loop.

Directly comparing results, we found that VirLT&R with NeRF-derived submaps performed roughly 5 to 7 cm better in terms of RMSE and 6-22 cm better in terms of the maximum error than in the initial experiments, when considering the interally estimated PTE. The measured RMSE is also improved by roughly 13 cm and the maximum by over 20 cm.

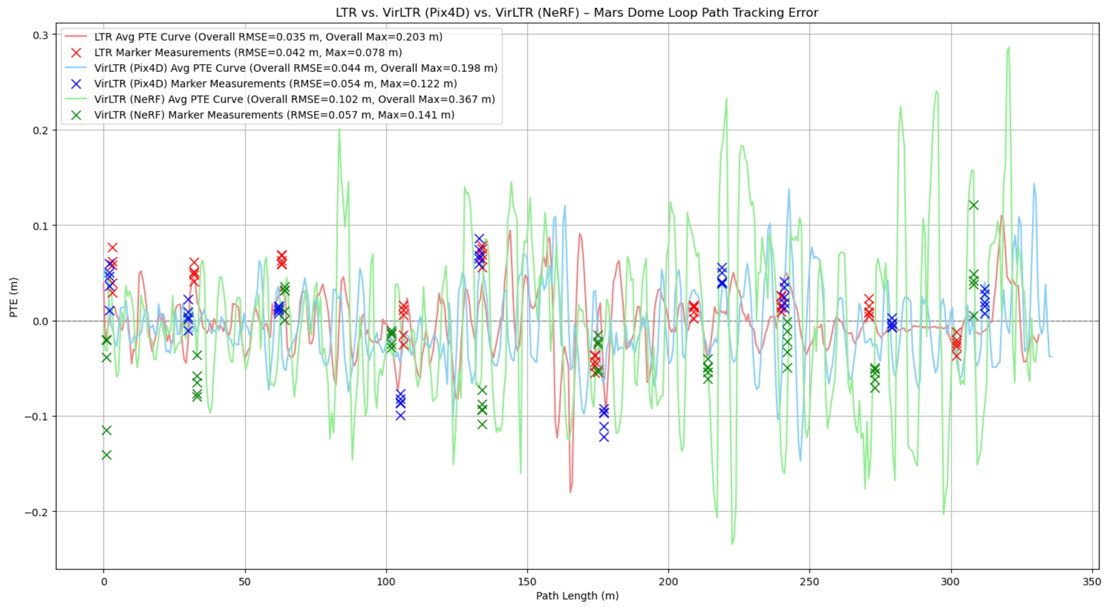


Figure 6.5: A comparison of Lidar-based T&R algorithms on the Mars Dome Loop: LT&R (baseline), VirLT&R using Pix4D (most optimal virtual teach configuration), and VirLT&R using NeRF. Five measurements of the 10 markings are shown with the internally estimated PTE for the continuous route.

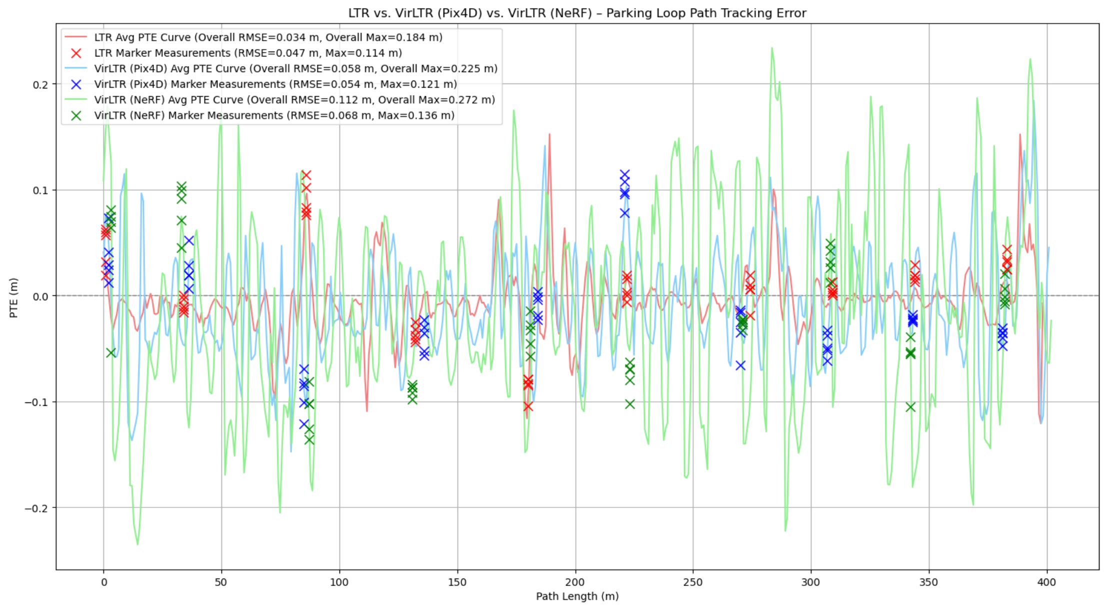


Figure 6.6: A comparison of Lidar-based T&R algorithms on the Parking Dome Loop: LT&R (baseline), VirLT&R using Pix4D (most optimal virtual teach configuration), and VirLT&R using NeRF. Five measurements of the 10 markings are shown with the internally estimated PTE for the continuous route.

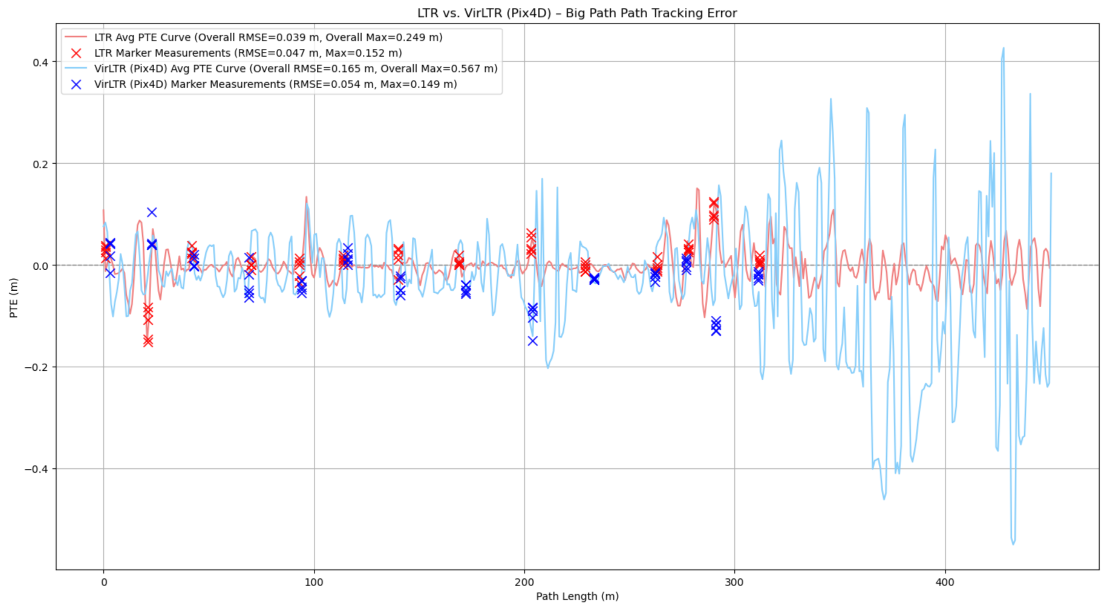


Figure 6.7: A comparison of Lidar-based T&R algorithms on the UTIAS Big Loop: LT&R (baseline), VirLT&R using Pix4D (most optimal virtual teach configuration), and VirLT&R using NeRF. Five measurements of the 14 markings are shown with the internally estimated PTE for the continuous route. Notably, the Avg PTE curves for LT&R and VirLT&R grow to be more varied after 300 m in the path, which is when the UGV enters the Grassy Loop's field.

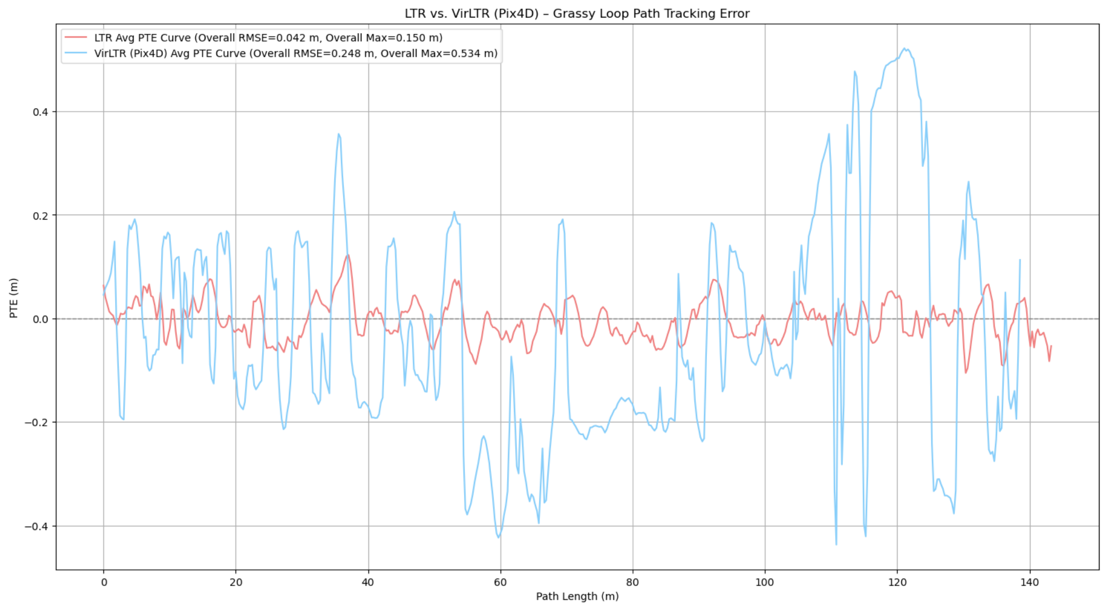


Figure 6.8: A comparison of Lidar-based T&R algorithms on the Parking Dome Loop: LT&R (baseline), VirLT&R using Pix4D (most optimal virtual teach configuration), and VirLT&R using NeRF. As no markings could be placed on this route, only the internally estimated PTE for the continuous route is shown.

Few challenges were faced when using VirLT&R, with either Pix4D or NeRF reconstructions, where the most significant hurdle was computation time. The addition of continuous ICP and gyroscope information to LT&R was observed to increase the time required to solve the optimization problem. It was also found to be unusable on some laptops, meaning a different machine was required to run LT&R for the baseline and the virtual map repeats. The majority of the repeats conducted in this experiment were done in the early morning, evening, or overnight, to prevent overheating in direct sunlight, and we ensured adequate ventilation to the machine by elevating it when repeating during the day. In the Mars Dome Loop LT&R baseline, the maximum error of 20.3 cm exceeds the VirLT&R maximum error of 19.8 cm. This is likely because of the extended computation time combined with the hairpin turn on the baseline's path being driven slightly tighter than the Pix4D path was driven in simulation.

We found substantial improvement in the performance of VirLT&R compared to what was observed and measured in our initial experiment with NeRF-based VirLT&R, suggesting the improvements made to the LT&R algorithm and the ablation study in Chapter 3.2.2 helped reach a more robust and accurate version of the algorithm capable of successfully functioning in larger and/or more geometrically sparse environments. The improved Pix4D reconstruction also enhanced the measured PTE such that it now closely trails behind LT&R by approximately 1 cm in the RMSE and approximately 0 to 5 cm in the maximum error, occasionally performing better in some locations, but drastically diverging on the Grassy Loop and the grassy section of the UTIAS Big Loop when considering the internal estimate. On the Mars Dome Loop, VirLT&R with Pix4D obtained a measured 5.4 cm RMSE and 12.2 cm maximum error, while VirLT&R with NeRF obtained a measured 5.7 cm RMSE and 14.1 cm maximum error, which is an improvement of 14.1 cm and 13.8 cm in the RMSE and of 27.2 cm and 25.3 cm in maximum error when compared to the initial experimental results in Chapter 5.

The spread of the marking measurements and similar RMSE values across the different routes is further supported by the relative repeat deviation analysis seen in Figure 6.9. By comparing the continuous internally estimated path-tracking error obtained from the lateral signed distances between the vertices in the repeat paths, a maximum deviation of 18.7 cm can be observed in the Grassy Loop. The maximum deviation occurring in the least accurately reconstructed route with the most uneven terrain is logical and substantiated by the Grassy Loop also having the highest internally estimated RMSE and maximum error.

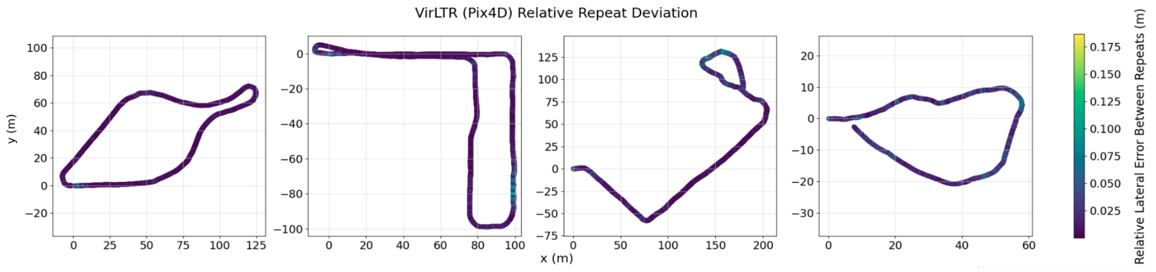


Figure 6.9: A comparison of all repeat pose graphs for all four routes. The relative lateral path-tracking error between the repeats is highlighted using a colour scale with yellow showing the highest error that is observed on the Grassy Loop.

The relative repeat deviation when using Pix4D-derived teach maps ranges from a near consistent 2.5 cm, occasionally reaching nearly 10 cm, with a maximum of 17.5 cm. This indicates that Pix4D reconstructions accurately recover the environment–ensuring that scene changes, moving objects, and people do not result in localization differences over time–allowing the robot to consistently repeat its tracks, traversing uneven terrain, patches of mud or gravel, and through narrow gaps.

6.3.2 VirRT&R

The PTE of Radar-based T&R methods also increases from the RT&R baseline, to VirRT&R using Pix4D, to VirRT&R using NeRF in the figures below, similarly to Lidar-based methods. However, the spread of measurements is closer to 10 cm, which approaches our initial VirLT&R benchmarked performance, suggesting similar localization quality. PTE plots can be seen in Figure 6.10 and Figure 6.11 for the Mars Dome and UTIAS Parking Loops.

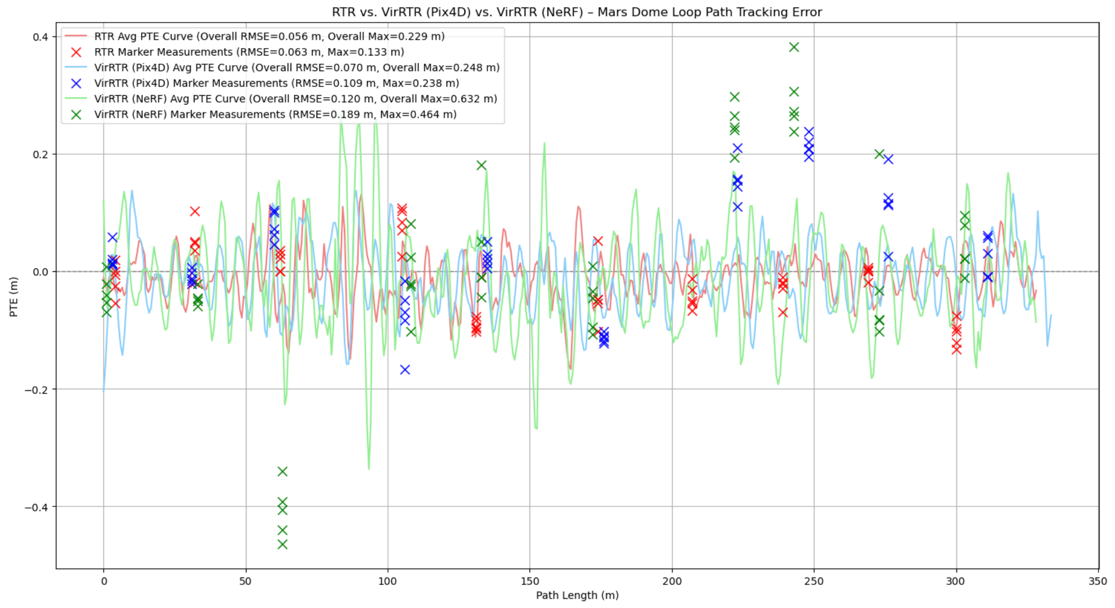


Figure 6.10: A comparison of Radar-based T&R algorithms on the Mars Dome Loop: RT&R (baseline), VirRT&R using Pix4D, and VirRT&R using NeRF (the overall worst configuration). Five measurements of the 10 markings are shown with the internally estimated PTE for the continuous route.

VirRT&R using either Pix4D or NeRF reconstructions was not as computationally intensive as VirLT&R, likely due to virtual Radar scans having far fewer points than virtual Lidar scans, as with the real sensor scans. Fewer points in a submap result in a faster solution to the optimization problem via continuous ICP, which meant the algorithm could be run on a broader range of laptops and did not struggle with overheating.

Qualitative analysis of the new virtual Radar modality revealed a very similar level of robustness to environmental changes, instantaneously and over time, establishing that it is feasible to conduct an aerial survey with a UAV to collect images of a scene for reconstruction, to then use the resultant point cloud as a localization layer for smooth and reliable Radar-based ground vehicle navigation. However, we did find that VirRT&R required more specific tuning to become functional. Specifically, a higher number of averaging steps in the ICP localization was required compared to Lidar ICP localization. Additionally, we use the K-Strongest method [24] for point extraction from raw Radar scans and found $k = 4$ to provide the best performance. Despite this tuning, we were unable

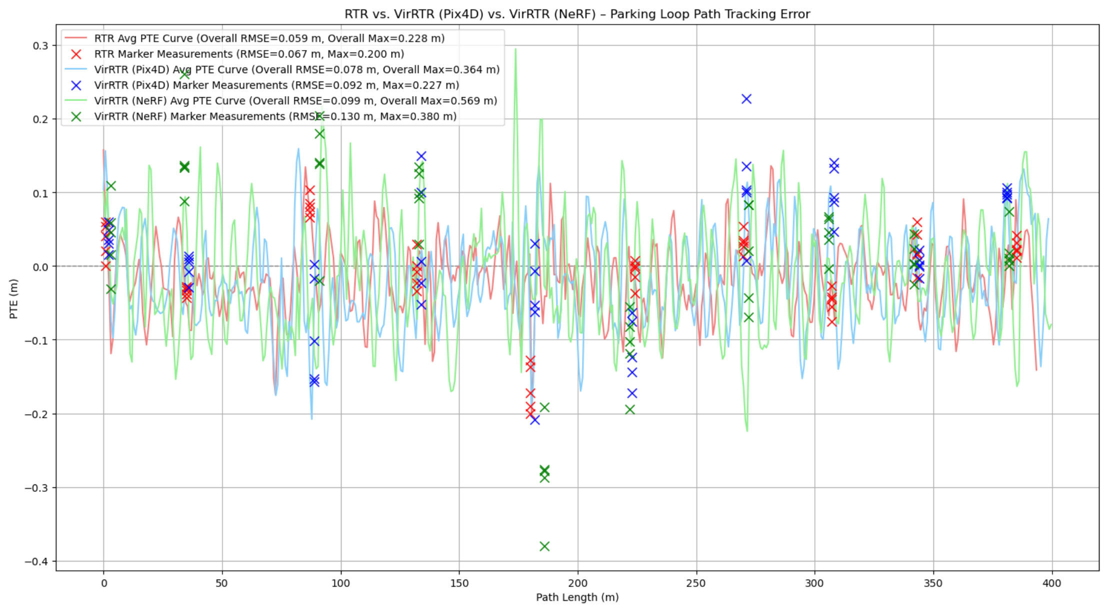


Figure 6.11: A comparison of Radar-based T&R algorithms on the Parking Dome Loop: RT&R (baseline), VirRT&R using Pix4D, and VirRT&R using NeRF (the overall worst configuration). Five measurements of the 10 markings are shown with the internally estimated PTE for the continuous route.

to get VirRT&R to function on the Grassy Loop, even with the Pix4D point cloud being used to create the submaps. Upon inspecting the differences between live and virtual scans, it appeared that the virtual Radar scan was not accurately modelling the ground strikes from the Radar well enough. The Grassy Loop features very uneven terrain, obscured by tall grass, that pitches the Warthog frequently and unpredictably, that was evident in the internally estimated PTE. These factors are probable causes of the reconstruction's inability to model the environment well enough. Thus, the Grassy Loop and UTIAS Big Loop were omitted, although VirRT&R had no trouble with the larger environment of the UTIAS Big Loop, VirLT&R methods must be used if traversing highly vegetated environments with sparse geometry is desired.

We found VirRT&R using Pix4D on the Mars Dome and UTIAS Parking Loops to have a measured RMSE of 10.9 cm and 9.2 cm, with maximum errors of 23.8 cm and 22.7 cm, respectively. When using NeRF-based VirRT&R, we measured an RMSE of 18.9 cm and 13.0 cm, with maximum errors of 46.4 cm and 38.0 cm on the Mars Dome and UTIAS Parking Loops, respectively. The use of VirRT&R with NeRF reconstructions is the poorest performing configuration we tested overall, likely due to the modality's less detailed scan of the world and the less accurate scene reconstruction from NeRF. Notably, the Mars Dome Loop exhibits a significantly higher RMSE and maximum error compared to the UTIAS Parking Loop, possibly due to the large, symmetrical Dome in the center of the scene posing a challenge for aligning virtual and live submaps. The other possible factor is the combination of the worst performing configuration being run on Mars Dome Loop which has the hairpin turn that was seen to cause an elevated PTE in Lidar data as well. However, the measured RMSE and maximum error do reasonably align with the internally estimated PTE seen in Figure 6.10, although a greater mismatch occurs after the third marker where the Warthog passes through a very narrow opening between the Mars Dome and a chain link fence. It is also worth noting that stopping and starting the Warthog while using the VirRT&R method with NeRF maps did have a more noticeable affect on performance, as runs where the UGV was not paused to take measurements, it qualitatively repeated more smoothly and seemingly closer to the path.

In Figure 6.12 we compare the continuous internally estimated PTE obtained from the lateral signed distances between the vertices in the repeat paths. A maximum deviation of 21.0 cm can be observed in the UTIAS Parking Loop. There are several sections in the UTIAS Parking Loop that have a higher relative deviation, which seem to be loosely correlated with marking locations, suggesting the pausing of the Warthog to take measurements is what inflates the error. Combined with the generally higher PTE from using virtual Radar scans for localization, the sometimes jerky stopping of the Warthog to take a measurement is the probable cause of the higher relative deviation over all repeats. The same behaviour exists when using Lidar-based virtual T&R as well.

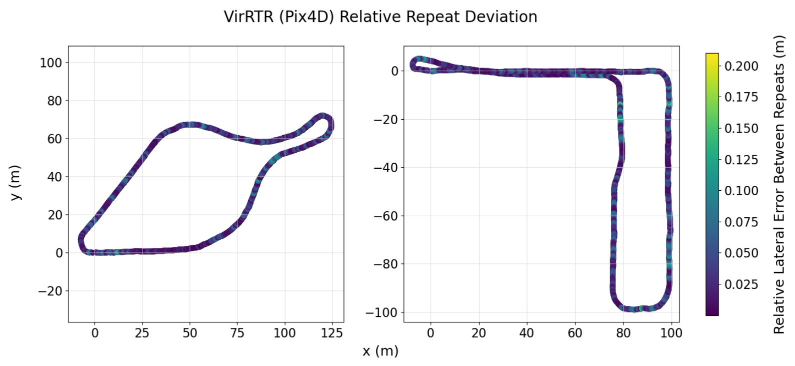


Figure 6.12: A comparison of all repeat pose graphs for both routes. The relative lateral path-tracking error between the repeats is highlighted using a colour scalethe with yellow showing the highest error that is observed on the UTIAS Parking Loop.

Overall, the relative deviation of repeats to a virtual teach map created with a Pix4D-generated point cloud ranges from 2.5 to 5.0 cm, occasionally reaching nearly 15 cm, with a 21 cm maximum. From this, we can conclude that using Pix4D-generated point clouds and cropping them into submaps according to the projection of the Radar beam from the Navtech Radar [17] accurately recovers the environment well enough that moving objects like large construction trucks or people do not result in meaningful localization differences over time, allowing for the robot to consistently repeat in its tracks in urban settings.

6.4 Discussion and Future Work

We were able to recreate and improve upon the findings in our initial validation of the VirT&R algorithm, expanding its operational window to include larger environments, more sparsely populated environments, and a different sensing modality. VirT&R has now also been tested extensively in both summer and winter conditions, allowing us to verify that seasonal differences, such as snow build-up or trees being full of leaves/having none at all, do not affect performance.

As mentioned above, we found difficulty using VirRT&R in the highly vegetated and sparsely populated Grassy Loop, meaning it would be necessary to use VirLT&R or real-world RT&R in that environment. Thankfully, both Lidar and Radar modalities are enabled with a single UAV scene capture, further highlighting the benefits of VirT&R and the added cross-modality it brings to T&R. Beyond the extra tuning required for VirRT&R, the only other challenge was the increased computational load that the continuous time Lidar ICP formulation creates when it is being used for localization, regardless if real or virtual scans are being used for submaps, due to the significantly higher number of points than in a Radar point cloud.

Future work will involve further tuning to enable ViRT&R functionality on the Grassy Loop, conducting tests in radically different environments, such as those found at remote testing sites in Alberta, and deploying VirT&R on various UGV platforms to further validate its robustness and capacity for generalization. Additionally, the use of GPS information via ground control points placed in the captured scenes is also being explored to potentially facilitate higher-quality pseudo-GPS for virtual teach paths to supplement the measured marking offsets currently used to obtain lateral path-tracking RMSE and maximum error.

6.5 Conclusions

The expanded field testing of VirT&R resulted in the successful incorporation of Radar-based virtual teaching, as well as the introduction of the ability to virtually pilot a UGV through larger and/or more sparsely populated environments. The comparisons outlined in the beginning of this chapter were evaluated and we found that VirLT&R outperformed VirRT&R, just as traditional LT&R has better path-tracking performance than traditional RT&R. Additionally, we found that a classical photogrammetry technique did improve PTE over a learned volumetric rendering method, with no distinct environmental case where photogrammetry was not superior. Through making various scene reconstructions, we found that NeRF volumetric rendering consistently produces more usable scenes with fewer images than Pix4D did, alluding to a potential niche for volumetric rendering methods being in specific instances where UAV survey flights are difficult to conduct, lighting conditions are slightly less stable, and fewer good images can be obtained.

Aligning both virtual Lidar and Radar submaps to the corresponding live sensor scans yielded comparable path-tracking performance to baseline methods, which we validated over 20 km of uninterrupted autonomous driving across four routes and various virtual method configurations. The best configuration was VirLT&R with Pix4D, that had measured RMSEs of 5.4, 5.4, 5.4m and 24.8 cm with maximum errors of 12.2, 12.1, 14.9, 53.4 cm for the Mars Dome Loop, UTIAS Parking Loop, UTIAS Big Loop and Grassy Loop, respectively.

6.6 Novel Contributions

This chapter presents the expanded field testing experimental results for the path-tracking error and performance of Virtual Teach and Repeat, where more sparsely populated environments are navigated, Radar localization is incorperated, and different reconstruction tools are tested.

- We are the first to achieve zero-shot autonomous UGV navigation using aerial imagery to virtually generate localization data across the following cases:
    a. a wide variety of urban to rural GPS-denied environments;
    b. across winter to summer seasonal conditions;
    c. using either a Lidar or Radar sensor for localization to virtually generated submaps derived from aerial reconstructions.
- We present a detailed comparison of learned and classical 3D reconstruction tools and the resulting performance of Virtual Teach and Repeat when using either method to generate pilot simulation and localization layer data.

Chapter 7

Conclusion and Future Work

The work presented in this thesis details the creation of a new method for teaching routes that seamlessly integrates into the existing VT&R3 framework [7]. We demonstrate the feasibility and performance of our novel VirT&R addition in the context of autonomous ground vehicle navigation on a grand total of 33.03 km of autonomous driving across two experiments that took place in opposite seasons, with different reconstruction tools, Radar and Lidar sensor modalities, and in large and/or sparsely geometrically populated environments. This feat was achieved by leveraging the agnostic nature of T&R to use virtual 3D reconstructions from digital twins generated using UAV imagery taken during survey flights done offline prior to UGV navigation.

Throughout our experiments, we observed smooth, predictable, and repeatable driving consistent with what is expected from the baseline T&R Lidar and Radar methods, with elevated but comparable measured PTE less than one tire width (24 cm) on the Clearpath Warthog used for testing. Improvements made to the algorithm between experiments led the PTE to converge to nearly the same GPS-measured values of the baseline methods, depending on the virtual method configuration used; although we do not assess PTE in this work with GPS, instead using physical markings in the environment to provide a sim-to-real path-tracking assessment.

7.1 Thesis Summary

The primary objective of this research was to determine if virtually generated point clouds could reliably function as the localization layer of T&R, replacing the existing Lidar and Radar geometric world representations, to enable a higher level of autonomy in the system by removing the requirement of having to do a physical teach pass with a human operator and UGV in the desired environment. This led us to investigate whether using only monocular UAV footage from survey flights could be used to create point clouds that accurately resemble Lidar and Radar point clouds.

To contextualize the problem, Chapter 1 discusses the limitations of conventional autonomous navigation methods, particularly their dependence on GPS and susceptibility to environmental degradation over time. These challenges frame and motivate the use of T&R in pursuit of using digital twin reconstructions of desired environments for the piloting of paths to be repeated in the real world. Core technical background on the relevant state estimation techniques used, existing T&R infrastructure and methodology, as well as related literature, is reviewed in Chapter 2. We make sure to explore a wide breadth of pertinent work in UGV navigation architectures, structure-from-motion photogrammetry pipelines, and volumetric rendering scene representations to fully understand potential tools in addition to discovered successes and pitfalls in the examined fields. The architecture of the proposed VirT&R pipeline is outlined in Chapter 3, which highlights its three main phases of offline mapping, offline pose graph generation, and online navigation. Chapter 3 also includes details on scene capture, the extraction of camera poses and depth information from reconstructions generated using Pix4D or NeRF, and the process of making compatible T&R teach pose graphs.

Details of the experimental methodology used for quantitative assessment are presented in Chapter 4, where the existing method is examined and a new process capable of capturing a sim-to-real PTE is developed. Our experiments and findings are detailed in Chapter 5 and Chapter 6, which describe the routes, experimental platforms, algorithm specifics, and virtual teach method configurations used throughout the 33 km of autonomous driving.

Overall, this thesis expands the operational boundaries of the existing T&R framework, offering an enhanced level of autonomy while maintaining a near state-of-the-art PTE. By transforming visual reconstructions into actionable navigation maps and by validating their use on a real autonomous vehicle, we provide a foundation for further integration of offline and virtually generated scene representation in autonomous robotic navigation pipelines.

7.2 Lessons Learned and Future Work

Throughout this thesis and its experiments, creating our pipeline, developing new evaluation methods, and running large-scale field tests yielded lessons and insights that expanded our understanding and broadened our expectations for what is possible.

Initially, using NeRF models to render a scene with UAV footage volumetrically was explored for integration into the vision-based T&R variant, which would utilize stereo images in its localization pipeline. After finding difficulty using feature and descriptor methods such as SIFT to match live images to NeRF-rendered images due to the slight blurriness in the foreground of NeRF images, we identified a path forward by utilizing point clouds extracted from trained NeRF models, which led to the current VirT&R pipeline. The consistent and reliable success of various virtual teaching configurations spanning Lidar and Radar sensors as well as Pix4D and NeRF reconstructions was unexpected, given that the only input to the system is monocular aerial imagery.

In creating our pipeline, we discovered that while photorealistic renderings, such as those produced by NeRF, can be visually convincing, they do not inherently provide the precise geometric integrity required for localization in challenging environments. In contrast, photogrammetry techniques like Pix4D offered better reconstruction quality and high-fidelity geometric representation. The experiments also demonstrated that relying on fundamental concepts, such as physical markings for PTE assessment when GPS is not available, as is often done in mining environments, holds up over time and is comparable to newer, more advanced or efficient assessment methods. To that end, many other facets regarding the logistics of carrying out many repeats with physical measurements, cross-comparisons, and multiple baselines were also learned.

7.2.1 Future VirT&R Testing

Looking forward, there are several directions to take the VirT&R framework. Primarily, we are focusing on deploying the algorithm on another UGV at a different testing facility by partnering with the Defence Research Department of Canada (DRDC). This testing will further validate VirT&R's robustness and capacity for generalization. The use of ground control points in Pix4D scene reconstructions will also be added during this testing to further anchor the generated mesh and point cloud in a more similar coordinate frame to the GPS recorded during autonomous repeats, facilitating higher-quality pseudo-GPS for virtual teach paths. Supplementing the measured marking offsets is of great interest to concretely determine the PTE that VirT&R provides. Future work will also include more in-depth tuning and exploration of the failure modes of VirRT&R on the Grassy Loop environment, as that was the only environment that posed a challenge to Radar-based VirT&R.

7.2.2 Algorithm Additions and Improvements

An inherent limitation of our algorithm is that it does not provide instantaneous repeating capability. In traditional T&R, an operator teaches a path and can then repeat to the generated map on the spot. With VirT&R, this is not possible as the digital twin environment must be made offline before operation, in addition to the path having to be piloted in simulation and needing submaps to be associated to it. The simulated driving and submap association process does not take long and can be done live in the field, taking only as long as the desired path takes to drive, with a couple of additional minutes for code compilation. The benefit of teaching multiple routes once a world reconstruction is made is a valuable asset, though, and is viewed to somewhat offset this limitation. A solution may be found in exploring the incorporation of existing 3D world models such as Google Earth to provide the localization layer and simulated driving environment.

Another significant opportunity lies in automating the creation of a desired teach path. Currently, virtually defined paths must be driven in simulation, just as traditional teach paths are driven in the real world. The requirement for a human to pilot the UGV in a digital twin environment was somewhat eliminated by creating the waypoint path definition method, but given that photogrammetric and neural reconstructions produce dense meshes and point clouds, existing motion planning methods—such as RRT [76], A*[77], or Artificial Potential Fields [78]—could be employed to generate optimal paths through the environment automatically. This would allow users to specify only a goal position or coarse waypoints, for automatic computation of a geometry-aware teach path that avoids obstacles and maximizes environmental observability for localization.

Finally, formalizing the evaluation of reconstruction quality in terms of navigation utility, rather than solely relying on visual fidelity, could help define metrics for predicting system performance prior to deployment and enable the intelligent selection or fusion of multiple reconstructions.

Bibliography

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, “Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges,” *IEEE Access*, vol. 7, pp. 48 572–48 634, 2019.

[2] C. Torresan, A. Berton, F. Carotenuto, S. F. Di Gennaro, B. Gioli, A. Matese, F. Miglietta, C. Vagnoli, A. Zaldei, and L. Wallace, “Forestry applications of uavs in europe: a review,” *International Journal of Remote Sensing*, vol. 38, no. 8-10, pp. 2427–2447, 2017.

[3] M. A. R. Estrada and A. Ndoma, “The uses of unmanned aerial vehicles –uav’s- (or drones) in social logistic: Natural disasters response and humanitarian relief aid,” *Procedia Computer Science*, vol. 149, pp. 375–383, 2019, iCTE in Transportation and Logistics 2018 (ICTE 2018). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050919301589

[4] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint kalman filter for vision-aided inertial navigation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 2007, pp. 3565–3572.

[5] S. Leutenegger, P. T. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, “Keyframe-based visual–inertial slam using nonlinear optimization,” in *Proceedings of Robotics: Science and Systems (RSS)*, Berlin, Germany, June 2013.

[6] A. E. Johnson, A. Ansar, L. H. Matthies, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis, “A general approach to terrain-relative navigation for planetary landing,” in *Proceedings of the AIAA Infotech{@}Aerospace Conference*, Rohnert Park, CA, USA, May 2007, AIAA 2007-2854.

[7] P. Furgale and T. D. Barfoot, “Visual teach and repeat for long-range rover autonomy,” *Journal of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.

[8] J. Sehn, Y. Wu, and T. D. Barfoot, “Along similar lines: Local obstacle avoidance for long-term autonomous path following,” in *2023 20th Conference on Robots and Vision (CRV)*. IEEE, 2023, pp. 81–88.

[9] K. Burnett, Y. Wu, D. J. Yoon, A. P. Schoellig, and T. D. Barfoot, “Are we ready for radar

to replace lidar in all-weather mapping and localization?" *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 328–10 335, 2022.

[10] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments," *Journal of Field Robotics*, vol. 34, no. 5, pp. 940–984, 2017.

[11] X. Qiao, A. Krawciw, S. Lilge, and T. D. Barfoot, "Radar teach and repeat: Architecture and initial field testing," *arXiv preprint arXiv:2409.10491*, 2024.

[12] M. Bianchi and T. D. Barfoot, "UAV localization using autoencoded satellite images," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1761–1768, 2021.

[13] B. Patel, T. D. Barfoot, and A. P. Schoellig, "Visual localization with google earth images for robust global pose estimation of UAVs," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 6491–6497.

[14] Clearpath Robotics. (2020) Warthog Unmanned Ground Vehicle. [Online]. Available: https://clearpathrobotics.com/%20warthog-unmanned-ground-vehicle-robot/

[15] DJI, "Support for Phantom 4 Pro." [Online]. Available: https://www.dji.com/ca/support/product/phantom-4-pro

[16] "Ouster OS1 LiDAR," Available Online [https://ouster.com/products/scanning-lidar/os1-sensor].

[17] Navtech RAS3 Radar. (2023). [Online]. Available: https://navtechradar.atlassian.net/wiki/spaces/IA/pages/2573172737/RAS3

[18] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[19] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[20] D. Lisus, J. Laconte, K. Burnett, Z. Zhang, and T. D. Barfoot, "Pointing the way: Refining radar-lidar localization using learned icp weights," *arXiv preprint arXiv:2309.08731*, 2023.

[21] Y. Wu, "VT&R3: Generalizing the teach and repeat navigation framework," MASc Thesis, University of Toronto Institute for Aerospace Studies, Toronto, Canada, September 2022. [Online]. Available: http://asrl.utias.utoronto.ca/~tdb/bib/wu_masc22.pdf

[22] ASRL. (2023) VT&R3 wiki. Autonomous Space Robotics Laboratory, University of Toronto. Accessed 8 July 2025. [Online]. Available: https://github.com/utiasASRL/vtr3/wiki

[23] H. Rohling, "Radar cfar thresholding in clutter and multiple target situations," *IEEE transactions on aerospace and electronic systems*, no. 4, pp. 608–621, 1983.

[24] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6433–6438.

[25] D. Adolfsson, M. Magnusson, A. Alhashimi, A. J. Lilienthal, and H. Andreasson, "CFEAR radarodometry-conservative filtering for efficient and accurate radar odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 5462–5469.

[26] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," in *ACM SIGGRAPH 2006 Papers*, ser. SIGGRAPH '06. Boston, MA, USA: ACM, 2006, pp. 835–846.

[27] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.

[28] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

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

[30] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," 2018. [Online]. Available: https://arxiv.org/abs/1712.07629

[31] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[32] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision*. Springer, 2016, pp. 501–518.

[33] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 1–15, 2022.

[34] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, *et al.*, "Nerfstudio: A Modular Framework for Neural Radiance Field Development," in *ACM SIGGRAPH 2023 conference proceedings*, 2023, pp. 1–12.

[35] Polycam, Inc., "Polycam: Photogrammetry software," https://poly.cam, 2024.

[36] Pix4D, "PIX4Dcloud Advanced," 2025, cloud-based photogrammetry and site-documentation platform. [Online]. Available: https://www.pix4d.com/product/pix4dcloud

[37] Pix4D Documentation. (2025) Best practices for image acquisition and photogrammetry. Pix4D. Photogrammetry knowledge article. [Online]. Available: https://support.pix4d.com/hc/best-practices-for-image-acquisition-and-photogrammetry

[38] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," arXiv preprint arXiv:2003.08934, 2020.

[39] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone, "Kimera: From slam to spatial perception with 3d dynamic scene graphs," *International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1510–1546, 2021, arXiv preprint arXiv:2101.06894.

[40] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "NICE-SLAM: Neural Implicit Scalable Encoding for SLAM," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 12 786–12 796.

[41] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, "iNeRF: Inverting neural radiance fields for pose estimation," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[42] M. Dronova, V. Cheremnykh, A. Kotcov, A. Fedoseev, and D. Tsetserukou, "FlyNeRF: NeRF-Based Aerial Mapping for High-Quality 3D Scene Reconstruction," Apr. 2024, arXiv:2404.12970 [cs] version: 1. [Online]. Available: http://arxiv.org/abs/2404.12970

[43] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-NeRF: Scalable Large Scene Neural View Synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8248–8258.

[44] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-NeRF: Scalable Construction of Large-Scale NeRFs for Virtual Fly-Throughs," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 12 922–12 931.

[45] A. Rosinol, J. J. Leonard, and L. Carlone, "NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023, pp. 3437–3444.

[46] Z. Zhu, S. Peng, V. Larsson, Z. Cui, M. R. Oswald, A. Geiger, and M. Pollefeys, "NICER-SLAM: Neural Implicit Scene Encoding for RGB SLAM," in *International Conference on 3D Vision (3DV)*. IEEE, 2024, pp. 42–52.

[47] D. Lisus, C. Holmes, and S. Waslander, "Towards Open World NeRF-Based SLAM," in *2023 20th Conference on Robots and Vision*. IEEE, 2023, pp. 37–44.

[48] D. Maggio, M. Abate, J. Shi, C. Mario, and L. Carlone, "Loc-NeRF: Monte Carlo Localization using Neural Radiance Fields," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 4018–4025.

[49] J. Deng, Q. Wu, X. Chen, S. Xia, Z. Sun, G. Liu, W. Yu, and L. Pei, "NeRF-LOAM: Neural Implicit Representation for Large-Scale Incremental LiDAR Odometry and Mapping," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 8218–8227.

[50] S. Katragadda, W. Lee, Y. Peng, P. Geneva, C. Chen, C. Guo, M. Li, and G. Huang, "NeRF-VINS: A Real-time Neural Radiance Field Map-based Visual-Inertial Navigation System," in *IEEE International Conference on Robotics and Automation*, 2024, pp. 10 230–10 237.

[51] J. Han, L. L. Beyer, G. V. Cavalheiro, and S. Karaman, "NVINS: Robust Visual Inertial Navigation Fused with NeRF-augmented Camera Pose Regressor and Uncertainty Quantification," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 12 601–12 608.

[52] H. Zhang, Y. Zou, Z. Yan, and H. Cheng, "Rapid-Mapping: LiDAR-Visual Implicit Neural Representations for Real-Time Dense Mapping," *IEEE Robotics and Automation Letters*, vol. 9, no. 9, pp. 8154–8161, 2024.

[53] L. Wiesmann, T. Guadagnino, I. Vizzo, N. Zimmerman, Y. Pan, H. Kuang, J. Behley, and C. Stachniss, "LocNDF: Neural Distance Field Mapping for Robot Localization," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4999–5006, 2023.

[54] A. Tonderski, C. Lindström, G. Hess, W. Ljungbergh, L. Svensson, and C. Petersson, "NeuRAD: Neural Rendering for Autonomous Driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[55] Y. Lin, T. Müller, J. Tremblay, B. Wen, S. Tyree, A. Evans, P. A. Vela, and S. Birchfield, "Parallel Inversion of Neural Radiance Fields for Robust Pose Estimation," *arXiv preprint arXiv:2210.10108v2*, Mar. 2023, `arXiv:2210.10108v2 [cs.CV]`. [Online]. Available: https://arxiv.org/abs/2210.10108v2

[56] Z. Li, K. Fu, H. Wang, and M. Wang, "Pi-nerf: A partial-invertible neural radiance fields for pose estimation," in *Proceedings of the 31st ACM International Conference on Multimedia (MM '23)*. New York, NY, USA: Association for Computing Machinery, 2023, pp. 7826–7836. [Online]. Available: https://doi.org/10.1145/3581783.3612590

[57] J. Hausberg, R. Ishikawa, M. Roxas, and T. Oishi, "Relative drone-ground vehicle localization using lidar and fisheye cameras through direct and indirect observations," *arXiv preprint arXiv:2011.07008*, 2020.

[58] C. Forster, M. Pizzoli, and D. Scaramuzza, "Air-ground localization and map augmentation using monocular dense reconstruction," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3971–3978.

[59] S. Qian, L. Cheng, X. Xu, and M. Ren, "Localization of UGV Guided by UAV Using Visual Inertia Sensors and UWB," in *IEEE 18th International Conference on Control & Automation*, 2024, pp. 906–911.

[60] H. Qin, Z. Meng, W. Meng, X. Chen, H. Sun, F. Lin, and M. H. Ang, "Autonomous Exploration and Mapping System Using Heterogeneous UAVs and UGVs in GPS-Denied Environments," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1339–1350, 2019.

[61] S. Hood, K. Benson, P. Hamod, D. Madison, J. M. O'Kane, and I. Rekleitis, "Bird's eye view: Cooperative exploration by UGV and UAV," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017, pp. 247–255.

[62] J. H. Kim, J.-W. Kwon, and J. Seo, "Multi-UAV-based stereo vision system without GPS for ground obstacle mapping to assist path planning of UGV," *Elec Letters*, vol. 50, no. 20, pp. 1431–1432, 2014.

[63] A. Lakas, B. Belkhouche, O. Benkraouda, A. Shuaib, and H. J. Alasmawi, "A Framework for a Cooperative UAV-UGV System for Path Discovery and Planning," in *International Conference on Innovations in Information Technology*, 2018, pp. 42–46.

[64] C. Potena, R. Khanna, J. I. Nieto, R. Siegwart, D. Nardi, and A. Pretto, "Agricolmap: Aerial–ground collaborative 3d mapping for precision farming," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1085–1092, 2019.

[65] B. Gilhuly and S. L. Smith, "Robotic Coverage for Continuous Mapping Ahead of a Moving Vehicle," in *IEEE 58th Conference on Decision and Control*, 2019, pp. 8224–8229.

[66] R. Wang, K. Wang, W. Song, and M. Fu, "Aerial-Ground Collaborative Continuous Risk Mapping for Autonomous Driving of Unmanned Ground Vehicle in Off-Road Environments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 6, pp. 9026–9041, 2023.

[67] P. Yang, Z. Li, H. Yan, and K. Rao, "Guidance Drone: Navigating Perception-Failure UGV with UAV Assistance in Cluttered Environments," in *14th Asian Control Conference*, 2024, pp. 332–337.

[68] R. Käslin, P. Fankhauser, E. Stumm, Z. Taylor, E. Mueggler, J. Delmerico, D. Scaramuzza, R. Siegwart, and M. Hutter, "Collaborative localization of aerial and ground robots through elevation maps," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2016.

[69] A. Dai, S. Gupta, and G. Gao, "Neural Radiance Maps for Extraterrestrial Navigation and Path Planning," in *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation*, 2023, pp. 1606–1620.

[70] N. Chebrolu, P. Lottes, T. Läbe, and C. Stachniss, "Robot localization based on aerial images for precision agriculture tasks in crop fields," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1787–1793.

[71] X. Zhang, Y. Qiu, Z. Sun, and Q. Liu, "Aerial-nerf: Adaptive spatial partitioning and sampling for large-scale aerial rendering," 2024. [Online]. Available: https://arxiv.org/abs/2405.06214

[72] Gazebo, "Gazebo Classic Simulation," 2020. [Online]. Available: https://classic.gazebosim.org/

[73] "NovAtel SMART6 GPS," Available Online [https://novatel.com/support/previous-generation-products-drop-down/previous-generation-products/smart6-smart-antenna].

[74] Y. Chen and T. D. Barfoot, "Self-Supervised Feature Learning for Long-Term Metric Visual Localization," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 472–479, 2022.

[75] M. Gridseth and T. D. Barfoot, "Keeping an Eye on Things: Deep Learned Features for Long-Term Visual Localization," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1016–1023, 2021.

[76] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Department of Computer Science, Iowa State University, Tech. Rep. TR 98-11, 1998.

[77] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[78] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.