3D Visual-Inertial Odometry and Autonomous Mobile Robot Exploration with Learned Map Prediction

by

Rakesh Shrestha

B.E., Tribhuvan University, 2013

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Science

SIMON FRASER UNIVERSITY
Fall 2018

Approval

Name:	**Rakesh Shrestha**
Degree:	**Master of Science (Computing Science)**
Title:	**3D Visual-Inertial Odometry and Autonomous Mobile Robot Exploration with Learned Map Prediction**
Examining Committee:	**Chair:** Mo Chen Assistant Professor
	Ping Tan Co-Senior Supervisor Associate Professor
	Richard Vaughan Co-Senior Supervisor Professor
	Yasutaka Furukawa External Examiner Assistant Professor
Date Defended:	**December 12, 2018**

Abstract

2D and 3D scene reconstruction are important topics in the field of robotics and computer vision. Mobile robots require a model of the environment to perform navigational tasks, and model acquisition is a useful application in itself . This thesis presents a) A 3D odometry and mapping system producing metric scale map and pose estimates using a minimal sensor-suite b) An autonomous ground robot for 2D mapping of an unknown environment using learned map prediction.

The first application proposes a direct visual-inertial odometry method working with a monocular camera. This system builds upon the state-of-the-art in direct vision-only odometry. It demonstrates superior system robustness and camera tracking accuracy compared to the original method. Furthermore, the system is able to produce a 3D map in metric scale, addressing the well known scale ambiguity inherent in monocular SLAM systems.

The second application demonstrates an autonomous ground robot capable of exploring unknown indoor environments for reconstructing their 2D maps. This method combines the strengths of traditional information-theoretic approaches towards solving this problem and more recent deep learning techniques. Specifically, it employs a state-of-the-art generative neural network to predict unknown regions of a partially explored map, and uses the prediction to enhance the exploration in an information-theoretic manner. The system is evaluated against traditional methods in simulation using floor plans of real buildings and demonstrates advantage in terms of exploration efficiency. We retain an advantage over end-to-end learned exploration methods in that the robot's behavior is easily explicable in terms of the predicted map.

Keywords: mobile robotics; computer vision; visual-inertial odometry; robotic exploration; machine learning

Dedication

For my loving parents and brother

Acknowledgements

I feel immense gratitude towards my co-senior supervisors, Professor Ping Tan and Professor Richard Vaughan for giving me the opportunity to work under their tutelage. It was my privilege to be mentored by experts in the fields of computer vision and robotics. Without their support, guidance and patience, I would not have been able to complete this work.

I am indebted to my family: my father, mother and brother for their unconditional love throughout my life. I would also like to thank the newest addition to our family, my sister-in-law, for taking care of the family while I had to be thousands of miles away from home.

I am grateful to my friends Payam Nikdel, Sicong Tang and Fei-Peng Tian for their collaborations in my graduate work. I thank Geoff Nagy and Jack Thomas for proofreading my thesis and giving their invaluable feedback. I would also like to thank all of my friends in the SFU Autonomy Lab and GrUVi Lab, including, but not limited to, Amir, Bita, Chenzhou, Faraz, Fei-peng, Feitong, Geoff, Jack, Jia, Luwei, Payam, Pratik, Renjiao, Sicong, for sharing this journey with me.

Table of Contents

List of Figures

Chapter 1

Introduction

Recent advancements in mobile robot technologies have brought robots into our day-to-day lives. We see them vacuuming our homes, guiding us in museums, managing warehouse inventories, doing aerial photography and more. One of the most crucial capabilities of a mobile robot is being able to perceive and reason about the environment it inhabits via its sensory inputs. A fundamental problem in robotic perception is building a world model, henceforth known as a 'map', for planning/executing robot trajectories, manipulating objects in the environment, and so forth.

Building maps in 3D is a well-studied problem in robotics and computer vision. Camera sensors are suitable for this application as they are light, inexpensive, have small form factors and yet provide a sizeable amount of information. A monocular camera is an especially appealing sensing modality for robots with limited payload capacity and computational resources. Direct Methods for Visual Odometry (VO) [2, 3, 4] allow for the recovery of 3D scene geometry from a single camera directly using image intensities. However, the absolute scale cannot be recovered unless we make strict assumptions about the environment. Another limitation of monocular 3D reconstruction techniques is their inability to handle quick rotations.

One way to address this problem is by augmenting monocular camera with an Inertial Measurement Unit (IMU). The key challenge in incorporating IMU information is the inherently noisy and biased readings from these sensors. This thesis explores methods that address these issues, providing a framework for integrating IMU information with visual information from a camera.

The second part of this thesis introduces a novel approach for autonomous exploration using a ground robot for 2D mapping of an unknown environment. Autonomous exploration has been widely studied using traditional geometric heuristics and information-theoretic reasoning. Recent advances in deep learning have shown its potential applications to this problem as well [5, 6, 7].

Planning for autonomous exploration entails reasoning about parts of the environment that are yet unexplored. Traditional information-theoretic methods and geometric heuristics-based methods do not have any prior (or have a flat/non-informative prior) on the unknown regions. End-to-end learnt robot behavior using deep learning does not explicitly model this reasoning about unknown regions for planning exploration despite the enormous potential of generative deep learning networks to infer large amounts of unexplored regions. This thesis aims to integrate these two methods,

exploiting the well-established and structured approach of traditional methods and the predictive power of deep learning-based methods for superior exploration performance. This is the first work to do so and is under review at the 2019 International Conference on Robotics and Automation (ICRA).

Chapter 2

Direct Visual-Inertial Odometry

Visual Odometry (VO) and Simultaneous Localization and Mapping (SLAM) are prominent research topics in computer vision. VO refers to the process of incremental sensor motion estimation and map building using a temporally local set of measurements. SLAM adds map reuse and loop closure capability to prevent accumulation of error in the trajectory when the sensor is moving in the same environment. They are indispensable components of many important applications like robotics, autonomous vehicles, virtual/augmented reality, and 3D modeling.

Most VO/SLAM algorithms [8, 9] are based on feature detection and matching. In comparison, direct methods [2, 3, 4] do not rely on hand-crafted features. By minimizing a photometric error defined directly on the raw pixel intensities, direct methods utilize the majority of pixels and demonstrate advantages in scenes that are sparsely textured. Furthermore, direct methods recover a dense/semi-dense map, which provides rich information about the environment for object detection and manipulation.

On the other hand, it has been demonstrated that vision-only VO/SLAM is limited especially with quick rotations, textureless environments, or sudden illumination changes. Sensor fusion with an on-board Inertial Measurement Unit (IMU) has been proven to be effective in these scenarios [10, 8, 11]. An IMU module provides measurements of angular velocity (from the gyroscope sensor) and linear acceleration (from the accelerometer), which despite being noisy and biased can be critical to overcome short term failures of visual tracking. This integration has enormous practical value as IMUs and cameras are part of most robot platforms (e.g. drones), smart phones and autonomous vehicles.

Sensor fusion is particularly useful for direct methods as they require a good initial estimate of the camera pose of the incoming frame. This is because they do not have explicit data-association between consecutive frames, but aim to minimize a highly non-convex photometric error between two frames to solve for camera pose and image point correspondences simultaneously. The widely adopted constant velocity model [2, 3, 12] may not provide reliable initialization during aggressive motions. Hence, the kinematic information provided by an IMU can significantly improve performance.

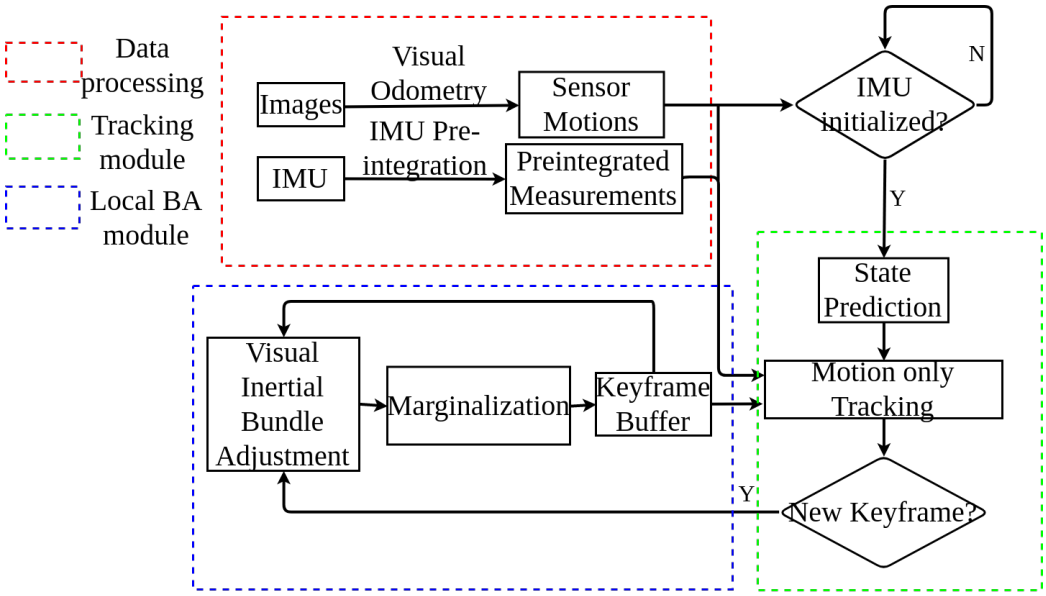

Figure 2.1: The pipeline of the DSVIO.

Another advantage of integrating the inertial information is that we can recover the absolute scale of the scene and motion of the sensor suite using a single camera in addition to an IMU. This is particularly useful for robotics applications where the map is required for path planning or manipulating objects in the environment.

Despite its usefulness, there are only a few works that integrate direct VO/SLAM algorithms with IMU information. Usenko et al. [13] implement visual-inertial odometry using a stereo camera, while only a monocular camera is available in many applications. Concha et al. [14] present a direct monocular visual-inertial SLAM system. Their system does not account for many photometric properties such as exposure time, lens vignetting, and non-linear response function of the camera. Our system is built upon state-of-the-art direct method, the Direct Sparse Odometry (DSO) [2] algorithm, which considers these photometric calibrations and has an online estimation of exposure time. These features are important for direct methods as the tracking is based on direct photometric measurements from the camera. [14] is also not evaluated on benchmark datasets for comparison.

This work integrates DSO [2], with inertial data using the IMU preintegration technique [15]. First, we initialize the visual-inertial system by aligning the tracking results from vision and IMU respectively to obtain IMU biases, velocity and gravity direction. During the optimization-based tracking, we formulate an objective function that combines the conventional photometric error in DSO and a kinematic error, which enforces constraints from the IMU between nearby frames.

Figure 2.1 shows the pipeline of our system. After IMU initialization, our system integrates all the IMU measurements between a newly arrived frame and the previous frame to get the prediction of the sensor state. The prediction is used as an initial value for nonlinear optimization that minimizes the photometric and kinematic error. When a new keyframe is created, the Bundle Ad-

justment (BA) thread performs visual-inertial optimization over a sliding window of keyframes to jointly optimize the vision and IMU errors.

Our experiment results demonstrate the robustness and accuracy of our system when compared with the original vision-only odometry. To the best of our knowledge, our work is the first direct monocular visual-inertial odometry system with online photometric calibration and extensive evaluations on publicly available benchmark sequences.

We note that following this work, VI-DSO [16] has been published with a similar system architecture as ours. There are notable differences: VI-DSO does not have a strict IMU initialization step but rather uses a rough initial estimate of scale, gravity direction and IMU biases which are jointly optimized during the run-time with other state variables. Another difference is their dynamic marginalization scheme which maintains several marginalization priors at the same time and resets the prior to the one computed using only the recent constraints when a state moves too far from the linearization point in the marginalization prior. Due to these differences, the work is able to estimate trajectories with higher accuracy than our system.

2.1 Related work

A comprehensive review of visual SLAM algorithms is beyond the scope of this thesis. We refer to a recent survey [17] for the interested audience. Here we briefly discuss some of the most relevant works.

Feature-based vs Direct methods Feature-based methods track camera pose by minimizing the reprojection error of point correspondences between manually designed features. Reprojection error is computed by projecting a 3D point to the current frame and taking the Euclidean distance between the projection and 2D point feature corresponding to the 3D point in the image. This has been a very active research topic with some notable works like PTAM [18] and ORB-SLAM [9]. Because of the sparse map representation, these methods are fast, and the explicit feature correspondence enhances robustness to outliers. However, these methods discard most of the image details, which makes their maps sparse and less visually appealing. Also, for textureless regions (like a uniformly colored wall), there might not be enough feature points to maintain good tracking.

In contrast, direct methods like LSD-SLAM [3], DTAM [19] track camera pose by minimizing the photometric error over a higher number of image pixels. This idea has also been applied to depth cameras in DVO [20] and Elasticfusion [21] with impressive results. However, the objective function in direct methods is highly non-convex and the convergence is often poor especially for aggressive motion. Furthermore, since the data association is solved implicitly, direct methods are more vulnerable to false associations caused by changing illumination, and photometric effects such as lens vignetting and exposure time. Our work is built upon the state-of-the-art direct method that take these issues into account and we increase its robustness by fusing it with IMU sensing.

Filtering-based vs optimization-based method Earlier work on visual/visual-inertial state estimation has been dominated by filtering-based methods [22, 23, 24] where only the latest state of the system is solved using the *Extended Kalman Filter* or its variants. MSCKF [22, 23] is a popular EKF-based VIO system. More recent work that uses filtering approach are [24] and [25]. Optimization-based methods [18, 8, 9] formulate the SLAM problem as an energy minimization, allowing re-linearization of the objective function after each step of the optimization, which avoids the error from committing to a sub-optimal linearization point. The computation cost of the optimization can be bounded by using a sliding window of frames whereby the states of the older frames are marginalized out while still retaining information from them.

While the EKF has been shown to be equivalent to the commonly used Gauss-Newton algorithm for non-linear optimization [26], the rigorous analysis by Strasdat et. al. [27] shows that optimization-based methods give "the most accuracy per unit of computing time". Our method belongs to the optimization-based approach with both photometric and kinematic terms derived from the visual and inertial sensor respectively.

Tightly-coupled vs loosely-coupled method Because of the complementary nature and ubiquity of cameras and IMU sensors, much work [15, 10, 14] fuses IMU and vision data. The loosely-coupled methods [28, 29] solve the vision and inertial modules independently and fuse their results using a Kalman Filter. In comparison, tightly coupled methods like OKVIS [8], VINS [10], VI-ORB [11] jointly estimate the sensors' states, and demonstrate better robustness and accuracy. Our method belongs to the tightly coupled category which integrate the visual and IMU constraints in a single energy function for joint optimization.

2.2 State representation

We parameterize the state of each camera frame i by a vector $\mathbf{x} = \{R_i, p_i, v_i, a_i, b_i, b_i^a, b_i^g\}$, where R_i, p_i and v_i are the orientation, translation, and velocity of the IMU sensor with respect to the world frame (or inertial frame, since here we ignore the self-rotation of Earth) at timestamp i. a_i, b_i are photometric parameters which correct the affine illumination changes between frames. b_i^a and b_i^g are the slowly varying accelerometer and gyroscope biases. The relative pose between camera and IMU is known *a priori*. For a landmark l in the map , we parameterize it by its inverse depth and pixel coordinate with respect to its unique host frame i (the frame where the landmark is first detected) , which is denoted as d_i^l . Assuming the camera intrinsics are known, we can easily compute the 3D position of each landmark from its inverse depth and pixel coordinates.

2.3 IMU Preintegration

The IMU sensor works at much higher frequency than a typical camera, producing multiple measurements between two camera frames. For all IMU measurements at Δt intervals between two

consecutive frames at times i and j, we integrate all the IMU data to get the rough sensor states:

$$
\begin{aligned}
R_j &= R_i \prod_{k=i}^{j-1} Exp((\omega_k) - b_i^g)\Delta t \\
v_j &= v_i + g\Delta t_{ij} + \sum_{k=i}^{j-1} \Delta R_k (a_k - b_i^a)\Delta t \\
p_j &= p_i + \sum_{k=1}^{j-1} v_k \Delta t + \frac{1}{2} g \Delta {t_{ij}}^2 + \frac{1}{2} \sum_{k=i}^{j-1} R_k (a_k - b_k^a) \ddot{\imath} \cdot \dot{Z} \Delta t \ddot{\imath} \cdot \dot{I}^2
\end{aligned}
\tag{2.1}
$$

where a_k and ω_k are accelerometer and gyroscope measurements at timestamp k respectively; b_a, b_g are their respective biases. g is the gravity vector obtained during initialization (in Section 2.4).

Applying first order approximation for the biases and rearranging Equation (2.1) such that the IMU measurements are separated from the pose and velocity of the sensor:

$$
\begin{aligned}
R_j &= R_i \Delta R_{ij} Exp(J_{\Delta R}^g b_i^g) \\
v_j &= v_i + g\Delta t_{ij} + R_i \Big(\Delta v_{ij} + J_{\Delta v}^g b_i^g + J_{\Delta v}^a b_i^a \Big) \\
p_j &= p_i + v_i \Delta t_{ij} + \frac{1}{2} g \Delta t_{ij}^2 + R_i \Big(\Delta p_{ij} + J_{\Delta p}^g b_i^g + J_{\Delta p}^a b_i^a \Big)
\end{aligned}
\tag{2.2}
$$

$J_{(.)}^a$ and $J_{(.)}^g$ are Jacobians of the corresponding preintegrated measurement $(.) = \{\Delta R, \Delta v, \Delta p\}$ [15] with respect to the biases. ΔR_{ij}, Δv_{ij} and Δp_{ij} are given by

$$
\begin{aligned}
\Delta R_{ij} &= \prod_{k=i}^{j-1} Exp((\omega_k) - b_i^g)\Delta t \\
\Delta v_{ij} &= \sum_{k=i}^{j-1} \Delta R_{ik} (a_k - b_i^a)\Delta t \\
\Delta p_{ij} &= \sum_{k=1}^{j-1} \Big[\Delta v_{ik} \Delta t + \frac{1}{2} \Delta R_{ik} (a_k - b_k^a) \Big]
\end{aligned}
\tag{2.3}
$$

Note that the preintegration measurements except ΔR are not the actual physical increment (i.e $\Delta v_{ij} \neq v_j - v_i, \Delta p_{ij} \neq p_j - p_i$).

Equation (2.1) and Equation (2.2) show that computing the physical increment in the velocity $v_j - v_i$, rotation $R_j R_i^T$ and position $p_j - p_i$ (and hence the constraints provided by the IMU between time i and j) depends on R_i and v_i. During the optimization, the estimated states are updated often but preintegration measurements remain the same since they only depend on the integrated IMU data. This avoids the need for reintegrating the IMU measurements after the state update in each iteration of optimization to obtain new constraints on rotation, translation and velocity.

The IMU preintegration measurements and their Jacobians are obtained using the GTSAM library[1] which implements the preintegrated technique proposed in [15].

2.4 IMU Initialization

Equations 2.1- 2.3 assume that the translations and velocities are in metric scale. Also, the gravity vector and the IMU biases should be known. To initialize these parameters we run monocular visual odometry for a short time and use the state estimates as ground truth with arbitrary scale (here we assume that the state estimation in a short period of time is accurate enough and is drift-free). Then we align these estimated up-to-scale states to metric-scale kinematic data from the IMU to obtain the scale along with the gravity vector and the IMU biases.

We follow the IMU initialization process proposed in [11]. The initialization is done in four steps as described below:

2.4.1 Gyroscope Bias

The gyroscope bias is estimated by minimizing relative rotations between two keyframes estimated by DSO and by integration of angular velocity measurements from the gyroscope. We assume that the biases do not change during the period of IMU initialization. This can be formulated as an optimization with the error function:

$$r_{\Delta R_{ij}} = Log\left(\left(\Delta R_{ij}(b_i^g) Exp\Big(\frac{\partial \Delta R_{ij}}{\partial b^g}\delta b^g\Big)\right)^T R_i^T R_j\right) \tag{2.4}$$

where Log maps $R \in SO(3)$ rotation matrix group to minimal representation in $so(3)$ and Exp is the inverse mapping from $so(3)$ to $SO(3)$ [30]. For simplicity, we define $so(3)$ as the 3D vector group opposed to the actual 3x3 skew symmetric matrix representation. R_i and R_j are rotations of keyframe at time i and j computed by DSO. The preintegrated measurement at current bias estimate $\Delta R_{ij}(b_i^g)$ (Equation (2.3)) and its Jacobian with respect to the bias $\frac{\partial \Delta R_{ij}}{\partial b^g}$ are defined in [15].

We solve for the gyroscope bias by minimizing Equation (2.4) using the Gauss-Newton algorithm.

2.4.2 Approximate scale and gravity direction estimation

Unlike the rotation, the positions obtained from vision-only DSO have an arbitrary scale. Moreover, to obtain relative positions and velocities using IMU measurements, the direction of gravity with respect to the current frame of reference (which in general is not the inertial frame) is required. The

[1] https://research.cc.gatech.edu/borg/gtsam/

relation between the position estimates of the camera by DSO $\mathbf{p}_{WC}$ and the positions of the body (IMU) frame $\mathbf{p}_{WB}$ in the arbitrary world frame W is:

$$\mathbf{p}_{WB} = s\mathbf{p}_{WC} + \mathbf{R}_{WC}\mathbf{p}_{CB} \tag{2.5}$$

where $\mathbf{R}_{WC}$ is the camera orientation obtained from DSO and $\mathbf{p}_{CB}$ is the known position of the IMU with respect to the camera.

From Equation (2.5) and Equation (2.1), assuming zero accelerometer bias, we obtain:

$$\begin{aligned} s\mathbf{p}^{i+1}_{WC} = s\mathbf{p}^{i}_{WC} + \mathbf{v}^{i}_{WB}\Delta t_{i,i+1} + \mathbf{g}_W {\Delta t_{i,i+1}}^2 \\ + \mathbf{R}^{i}_{WB}\Delta\mathbf{p}_{i,i+1} + (\mathbf{R}^{i}_{WC} - \mathbf{R}^{i+1}_{WC})\mathbf{p}_{CB} \end{aligned} \tag{2.6}$$

where $\mathbf{g}_W$ is the gravity vector in the world frame.

By using the relation in Equation (2.6) for three consecutive keyframes i, $i+1$ and $i+2$, we can avoid solving for velocity of the keyframes and obtain s and $\mathbf{g}_W$ given at-least 4 keyframes. We refer our reader to [11] for the exact implementation of this system of linear equations.

2.4.3 Scale and gravity vector refinement, and accelerometer bias estimation

By enforcing the magnitude of the gravity vector $\mathbf{g}_W$ to be G (standard value $9.8ms^{-2}$), the scale and gravity direction can be refined. At this phase, the effect of accelerometer bias is also considered. Using first order approximation and assuming bias to be close to zero, the preintegrated measurement is obtained as:

$$\Delta\mathbf{p}_{i,i+1}(\mathbf{b}^a) = \Delta\mathbf{p}_{i,i+1} + \mathbf{J}^{a}_{\Delta\mathbf{p}_{ij}}\mathbf{b}^a \tag{2.7}$$

where $\mathbf{J}^{a}_{\Delta\mathbf{p}_{ij}}$ is the Jacobian of the preintegrated measurement with respect to the accelerometer bias.

Replacing the preintegrated measurement $\Delta\mathbf{p}_{ij}$ in Equation (2.6) with the bias corrected one in Equation (2.7) and assuming only a small perturbation in the gravity direction $\mathbf{g}_W$, we can obtain refined scale/gravity direction and solve for accelerometer bias. We again refer our reader to [11] for more details.

2.4.4 Keyframe Velocity Estimate

Substituting the computed accelerometer bias, scale and gravity direction in Equation (2.6), the velocities of the keyframes $\mathbf{v}^{i}_{WB}$ can be obtained.

After initialization, we update the scale of all camera poses and map points. The world frame W is transformed such that the gravity direction lies along the vector $[0, 0, -1]^T$ (i.e. the world frame is transformed to inertial frame).

2.5 Model Formulation

2.5.1 Tracking

When a new frame is received, the tracking module is responsible for tracking its pose, velocity, affine photometric parameters and IMU biases. Since we have already aligned the IMU states and camera states in the IMU initialization, we use the increments obtained from Equation (2.2) to propagate the sensor state from time i to $i+1$.

The difference between our method and earlier work on a direct monocular visual-inertial SLAM system [14] is that we optimize both the state of the current frame and the previous frame for tracking a new frame. This is required as the IMU energy depends on the previous frame's state. [14] optimizes only the current sensor state on account of computation cost, but we argue that for better use of IMU information optimizing both states is necessary.

To eliminate the gauge freedom in the two pose optimization, we introduce a prior energy term that penalizes deviation from the previous sensor state [11, 10]. The information matrix of the prior error is obtained by marginalizing the state of previous frame. For consistency of error terms, we use the "First Estimate Jacobians" [31, 8] by fixing the linearization point of the state of previous frame.

Since we do not want the tracking thread to affect the running of the Local Bundle Adjustment module Section 2.5.2, inspired by [11], the system uses two methods to perform tracking according to the state of map update.

When the map has just been updated, then for the next incoming frame, we will only optimize the state of the current frame. Given the set of pixels $\{P\}$ on last keyframe F_k with inverse depth $D_k = \{d_k^1, d_k^2, ..., d_k^l\}$, we have the following objective function:

$$E(\mathbf{x_i}|D_k) = E_{Photometric}(\mathbf{x_i}|D_k) + E_{Kinematic}(\mathbf{x_i}) \tag{2.8}$$

Here, $E_{Photometric}(\mathbf{x_i}|D_k)$ is the photometric error of the pixel set $\{P\}$ with respect to the current frame. For more detail, we refer our readers to [2]. The kinematic error term $E_{Kinematic}(\mathbf{x_i})$ is defined as:

$$\begin{aligned} E_{Kinematic}(\mathbf{x_i}) = \left\| {r_{\Delta R}}^T, {r_{\Delta v}}^T, {r_{\Delta p}}^T \right\|^2_{\Sigma_{imu}} + \\ \left\| b_i^g - b_j^g \right\|^2_{\Sigma bgd} + \left\| b_i^a - b_j^a \right\|^2_{\Sigma bad} \end{aligned} \tag{2.9}$$

where the residuals $r_{\cdot}$ are defined as

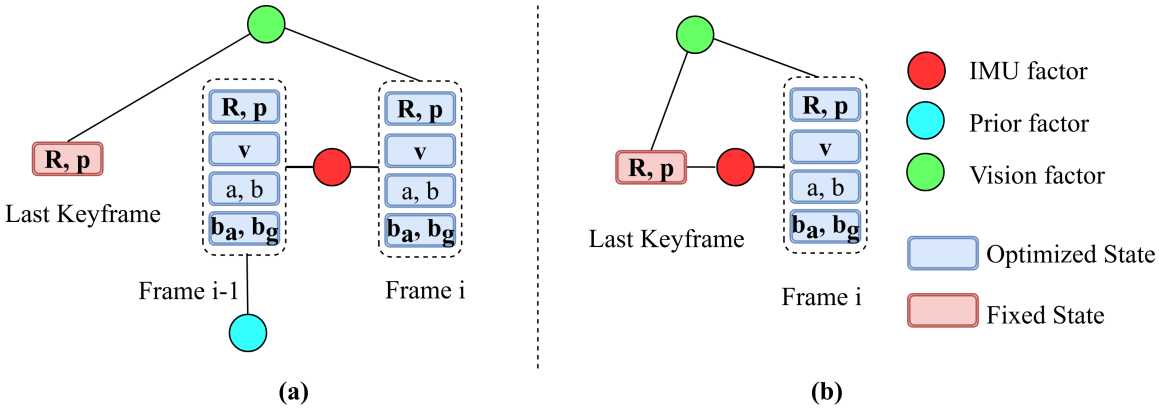

Figure 2.2: Illustration of factor graph of the tracking module. (a) When the map has not been updated recently (i.e. last frame is not the last keyframe) (b) When the map has been updated recently.

$$
\begin{aligned}
r_{\Delta R} &= Log((\Delta R_{ij} Exp(J^g_{\Delta R} b^g_j)^T R_i{}^T R_j)) \\
r_{\Delta v} &= R_i(v_j - v_i - g_w \Delta t_{ij}) - (\Delta v_{ij} + J^g_{\Delta v} b^g_j + J^a_{\Delta v} b^a_j) \\
r_{\Delta p} &= R_i(p_j - p_i - v_i \Delta t_{ij} - \frac{1}{2} g_w \Delta t_{ij}{}^2) \\
&\quad - (\Delta p_{ij} + J^g_{\Delta p} b^g_j + J^a_{\Delta p} b^a_j)
\end{aligned}
\tag{2.10}
$$

The information matrices $\Sigma_{imu}, \Sigma_{bgd}, \Sigma_{bad}$ are obtained by the propagation of covariances of the IMU measurements [15].

When there is no recent map update, we jointly optimize the incoming frame F_i and previous frame F_{i-1}, similar to the one pose optimization above. We define the objective function as:

$$
\begin{aligned}
E(\mathbf{x_i}, \mathbf{x_{i-1}} | D_k) = & E_{Photometric}(\mathbf{x_i} | D_k) \\
& + E_{Kinematic}(\mathbf{x_i}, \mathbf{x_{i-1}}) + E_{prior}(\mathbf{x_{i-1}})
\end{aligned}
\tag{2.11}
$$

The definitions of the photometric term $E_{photometric}$ and the Kinematic term $E_{Kinematic}$ are the same as the previous case; the only difference is that we optimize both $\mathbf{x_i}$ and $\mathbf{x_{i-1}}$. The prior term, E_{prior}, is the difference of the current estimated state of the frame F_{i-1} with its optimized state during previous tracking. The information matrix of the prior term is computed by marginalizing the frame F_{i-2}.

The factor graph of the tracking module in both of the cases are illustrated in Figure 2.2.

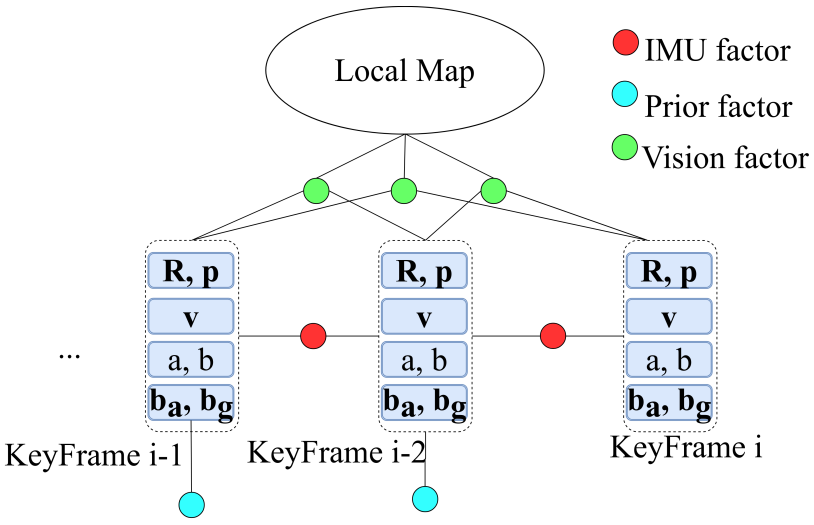

Figure 2.3: The factor graph of Local Bundle Adjustment

Marginalization

Let $\mathbf{x_m}$ denote the states to be marginalized and $\mathbf{x_r}$ the remaining states. The joint distribution $P(\mathbf{x_r}, \mathbf{x_m})$ is marginalized to obtain $P(\mathbf{x_r}|\mathbf{x_m})$ which can be used as a proxy for $P(\mathbf{x_r})$. Then we can eliminate $\mathbf{x_m}$ in later optimizations while still keeping the information from it in the form of $P(\mathbf{x_r}|\mathbf{x_m})$. The distribution of our states is a Multivariate Gaussian expressed by the mean (the current state estimate) and covariance matrices (inverse of Fisher information matrix/Hessian matrix).

Marginalization is performed using Schur Complement and is performed in a similar way to [2, 8].

2.5.2 Local Bundle Adjustment with IMU constraint

When inserting a new keyframe, all the keyframe states in the local window are optimized simultaneously by minimizing the photometric, kinematic and prior energy term (Equation (2.11)). Like the Local BA in work [11], we perform BA with 17 states (rotation, translation, velocity, biases and affine photometric parameters) after a new key frame has been created. Figure 2.3 shows our factor graph in the local BA thread. Since the number of states in the local window should be limited to bound the complexity of Bundle Adjustment, marginalization is used to eliminate the old states and save the information from those states in the form of conditional distribution. All the remaining states which have connection with the eliminated states will be constrained by this conditional distribution. We call this constraint prior factor.

The keyframe is chosen for marginalization based on the visibility of map points and a distance score with other keyframes in the window (refer to [2] for additional details). Hence the marginalized keyframe is not always the oldest one and the temporal difference between two consecutive

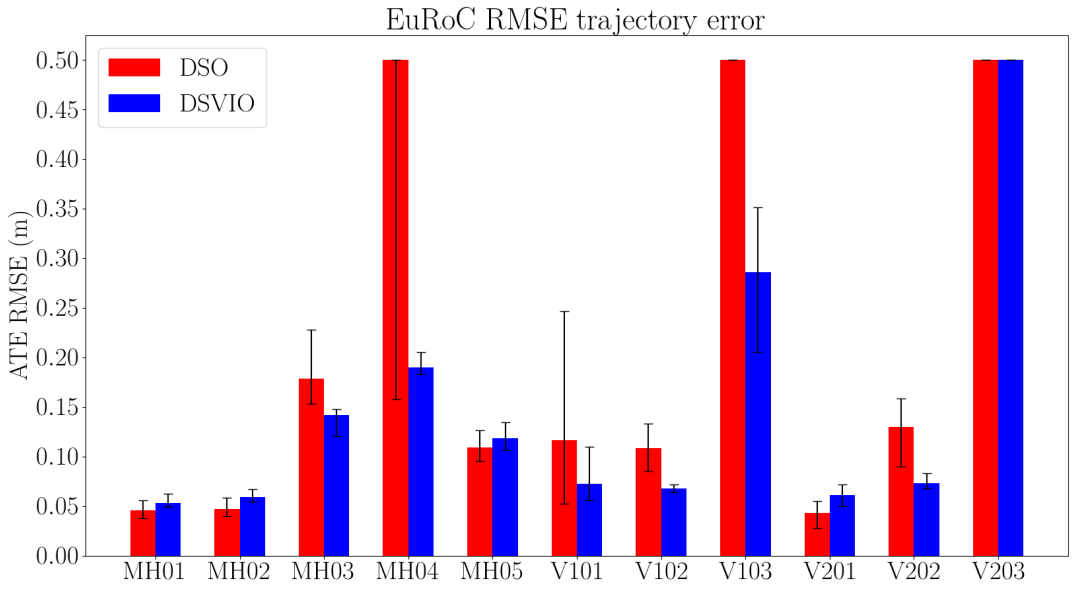

Figure 2.4: Mean error with minimum and maximum error bars

keyframes in the local window might be very large. However, a large temporal difference between consecutive keyframes weakens the IMU constraint because of accumulated noise in the time period. Hence, we only add the IMU constraint for the pair of consecutive keyframes whose time difference is no more than 0.5 seconds, and update the validity of the IMU factors in the local window after each new keyframe. The constraint from IMU preintegrated measurement will also be used for the prior factor computation when the corresponding keyframe is marginalized.

2.6 Experiment

We evaluate our system on the EuRoC MAV dataset [32]. It contains 11 sequences with stereo and inertial data recorded in 3 indoor environments: Machine Hall (MH), Vicon Room 1 (V1) and Vicon Room 2 (V2). The evaluation metric is the Root Mean Squared Error (RMSE) of trajectory. The scale of trajectory of the original DSO algorithm was aligned with ground truth using a 7 DOF Similarity Transform ($Sim(3)$) alignment using the technique proposed in [33]. The trajectory computed by our system was aligned with ground truth using a Euclidean alignment, i.e. rotation and translation only, without the scale alignment. The parameter settings of the visual odometry module in our system are the same as DSO.

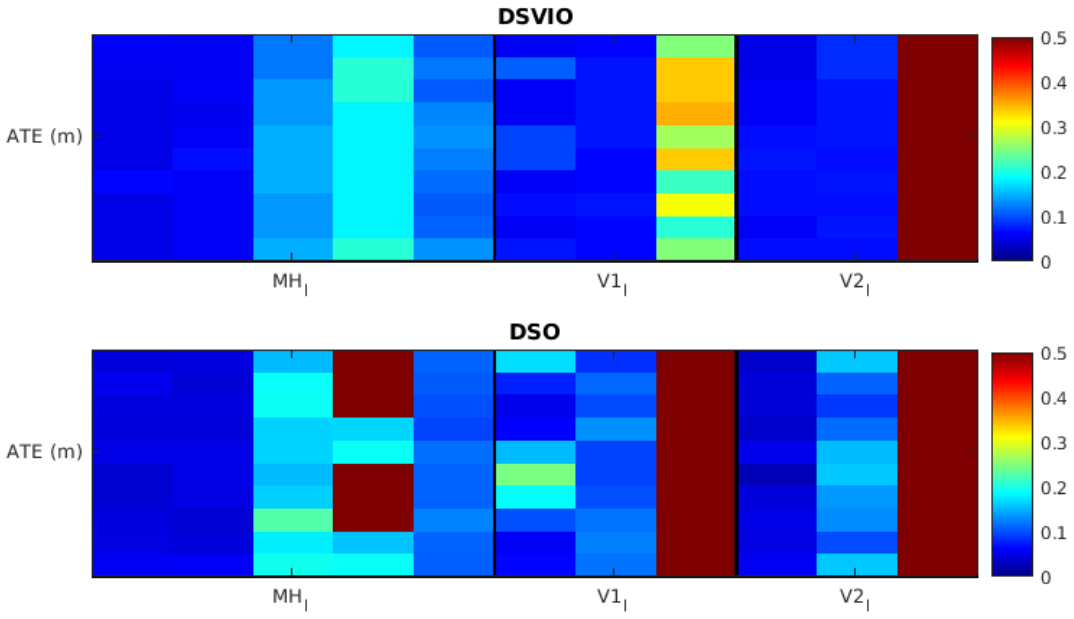

Figure 2.5: Error Heat Map

2.6.1 Methodology

We run our visual-inertial odometry system on each sequence 10 times. In our experiment we only use the sequences captured by the left camera, giving a total of 110 trials. In order to maintain invariance to the CPU load and machine specifications, we do not enforce real-time constraints on our system or the original DSO. We use the experimental results of the DSO that were made available by the authors as supplementary material. For the evaluation of our system, we exclude the beginning and end parts of each sequence (by the same amount as in DSO) on account of shaky motion that is detrimental to the visual tracking.

2.6.2 Results

The mean RMSE trajectory errors for different environments can be seen in Figure 2.4 along with error bars depicting the minimum and maximum errors among the 10 runs for each sequence. Additionally, Figure 2.5 shows the error of all 110 trials in the form of a heat map.

Our system is more robust in the difficult sequences V1_03 and MH_04 where the original implementation of DSO consistently failed. The time vs error plot in Figure 2.6 for the sequence V1_03 shows that during sudden fast motion, our system is able to maintain tracking while the original DSO implementation suffered significant drift. A sample image frame around the time where DSO trajectory drifted is given in Figure 2.7. The image shows considerable motion blur due to a sudden movement which makes the direct image alignment to solve relative camera motion unreliable. Also, the motion prediction based on constant velocity assumption will not be a good

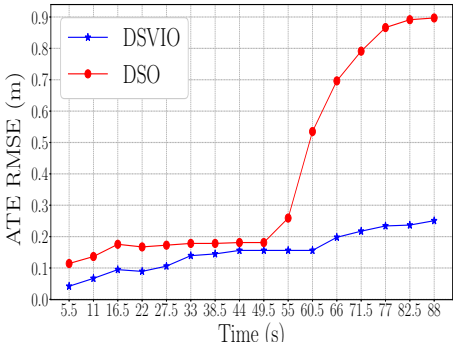

Figure 2.6: Cumulative error curve of DSO and our DSVIO

Figure 2.7: Image frame at the time around 50s in Figure 2.6 where the original DSO suffers from significant drift while our system is able to maintain tracking

initial seed for optimization due to which it will fail to converge to a good solution. Such motions, to a certain degree, can be handled using inertial sensing as it gives better initialization and additional constraints for the optimization.

Figure 2.8 and Figure 2.9 show the trajectory estimates and 3D reconstructions of the sequence V1_03 by DSO and DSVIO respectively. Because the camera motion is very aggressive at certain instances, the scale of the DSO trajectory is not consistent throughout the sequence while DSVIO can usually handle these situation.

2.7 Conclusion and Future work

To summarize, we combine sparse direct visual odometry with IMU preintegration to create a reliable VIO system. By automatically aligning the IMU factor with vision structure, our system can recover and preserve the metric scale of the scene very well. The kinematic information from IMU we add for prediction and state optimization helps the system attain robustness to aggressive motion.

The addition of loop closure constraints can help maintain consistent pose estimates over longer periods of time, which can be considered for future work. Also, since the IMU initialization result highly depends on the accuracy of the trajectory from vision-only DSO, a more robust visual initialization framework is still required. Another possibility is to have an initialization-free system which optimizes the scale and gravity vector as part of its state variable as done by [16].

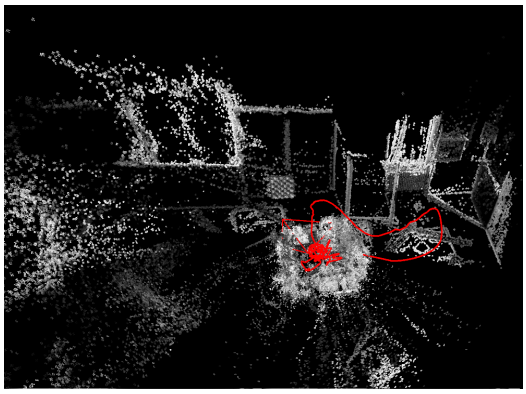

Figure 2.8: Reconstructed map by DSO for the sequence EuRoC MAV V1_03. Note the change in scale of the map

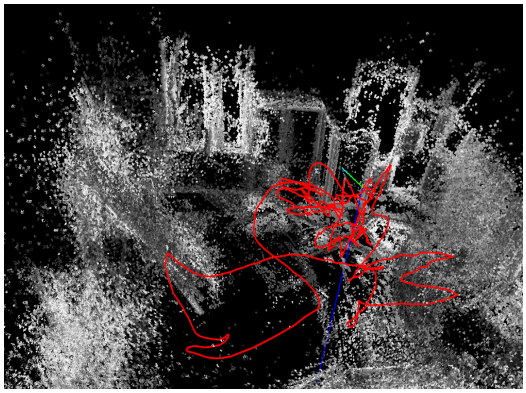

Figure 2.9: Reconstructed map by DSVIO for the sequence EuRoC MAV V1_03. The scale of the map is consistent throughout the trajectory

Chapter 3

Learned Map Prediction for Enhanced Mobile Robot Exploration

Modeling a previously-unknown environment is a canonical task in mobile robotics. The task of planning and executing robot trajectories to create a world model (i.e. 'map'), is known as 'exploration'. Autonomous exploration is a component of many real-world robotic applications, including search and rescue [1], planetary exploration [34], visual inspection [35], 2D/3D reconstruction [36] and mining [37]. The completeness and efficiency of exploration is important to facilitate these applications. Here we describe a novel exploration method that exploits learned priors over maps to explore more efficiently than current methods.

Exploration is closely related to some well-known NP-complete problems, such as the art gallery problem or the traveling salesman problem, with the important difference that the polygon to be covered or graph to be traversed is discovered dynamically and incrementally during run time. Previous methods for autonomous exploration can be broadly classified into two categories: frontier-based methods and information-theoretic methods. The frontier approach maximizes map coverage by moving to new frontiers - regions on the boundary between free space and unexplored space [38]. Path cost and expected information gain (i.e. utility) are commonly used to determine the next frontier to visit. The information-theoretic methods formulate the problem as map entropy minimization, i.e., information gain with probabilistic map representation [39]. Hence, a good information gain prediction plays a key role in both approaches.

Estimating information gain amounts to predicting sensor measurements in unseen regions of the map. The general approach is to assume that the robot is faced with a map that is either somewhat typical for its application domain, or somewhat regular in itself, or both, so that partial views of the map can afford useful predictions of the unseen parts. Methods for achieving this vary in their advantages and limitations in map prediction and hence information gain estimation. While this problem is not new, there has been recent interest and new ideas. Pimentel et al. [40] proposed an elegantly simple heuristic whereby wall segments at frontiers are assumed to extend into the unseen area when computing expected information gain. Cost-utility based exploration [41, 42] simply estimates information gain by counting unknown cells within a predefined radius of a fron-

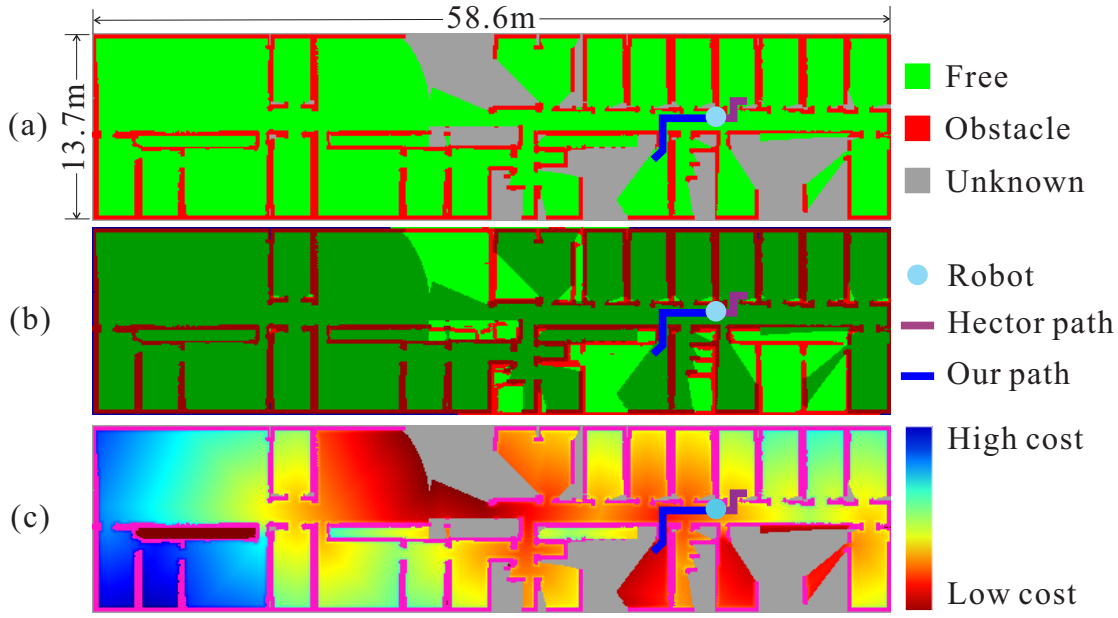

Figure 3.1: **Deep learning based map prediction for autonomous exploration.** (a) Current incomplete map. (b) Predicted map using the network trained over many previous maps. (c) Cost transform for explored cells. Compared to Hector exploration [1], our method chose a more rewarding direction to explore due to a better estimate of information gain.

tier point. While these methods may work locally on simple floor-plans, they are unlikely to perform well on more complicated floor-plans. Most recent information-theoretic methods [43, 44] predict the information gain based on Gaussian Mixture Models [45] or Gaussian processes with Bayesian inference, by one-step sensor look-ahead measurements using sensor likelihood prediction models [46, 47], which tacitly assume the unknown areas are always free. However, this assumption does not hold in general, thus the final information gain estimation is often inaccurate.

We propose a data-driven approach that does not rely on explicit assumptions about the environment, but instead learns regularities from examples. Specifically, we employ Variational Autoencoders (VAE) [48] to predict unknown map regions beyond frontiers. In our chosen indoor setting this allows us to learn generalizations over many building floor plans to make informed decisions for faster exploration of novel floor plans. As illustrated in Figure 3.1, using our VAE-based map prediction, we estimate the potential areas that can be mapped from each frontier, which we demonstrate can be more accurate and efficient than a single or multi-step lookahead in the sensor measurements.

Our system addresses the exploration problem in two steps: 1) predicting unknown regions to identify navigation goals that maximize map coverage; 2) navigating to those goal positions using well-established techniques. We employ deep learning to tackle the former in isolation, given the recent success of deep generative neural network models [49, 48]. Navigation is a well-studied problem in robotics where the traditional methods still outperform deep learning based methods [5, 6, 7]. By separating the problem into these two steps, we are able to utilize the strengths of both

deep learning and traditional methods while also maintaining the comprehensibility of the overall system pipeline.

3.1 Related Work

Frontier-based exploration methods are easy to understand and implement and work fairly well in practice. These methods maintain a list of boundaries between explored and unexplored regions in the partial map, known as 'frontiers'. The robot iterates over choosing the currently most promising frontier, planning and executing a path that drives through it while extending the map to eliminate the frontier, until no frontiers remain. Yamauchi et al. [38] proposed a well-known Nearest-Frontier Exploration method which always chooses the closest frontier as the next goal. Bourgault et al. [41] proposed to use a cost-utility function based on expected information gain and path cost. There are several extensions to this idea in [50, 42, 51, 52]. For efficient information gain computation, Umari et al. [42] used an RRT tree for frontier detection and counted unknown cells surrounding a frontier point within a predefined radius to estimate information gain. Ström et al. [51] presented a method to match the area beyond the frontiers with the most similar map in a database, and compute the expected information gain based on the retrieved match. Pimentel et al. [40] proposed a simple heuristic map prediction by linearly extending the walls or turning the walls by 90 degrees to compute expected information gain. While such simple local heuristic map completion and information gain computations can work well on simple floor-plans, we could hope to do better using more sophisticated priors over maps.

The information-theoretic methods use information theory to minimize uncertainty of the map. Several authors like [53, 54], suggest the use of information gain (also called mutual information in some contexts) as a measure of the reduction of map uncertainty. In recent work, Vallvé et al. [55] proposed to compute map and path information gain densely for the entire configuration space and apply a grid-step gradient on the potential fields to directly optimize a path. Jadidi [43] proposed a Gaussian Process (GP) based method to build an information field for the entire configuration space, but the final decision-making is still based on a utility function that chooses a goal which balances the path cost and expected information gain from frontiers extracted from the GP map. Bai et al. [44] present a method to predict the information gain in the surrounding partial map of the current robot pose based on a Gaussian process and Bayesian inference. Their method chooses the point with the largest expected information gain as the goal for the next step. These methods, while providing a new perspective on information gain estimation, often suppose unknown space as free or suppose Gaussian distribution of occupancy probabilities which does not hold in real floor-plans, thus the accuracy of the final information gain is limited.

Researchers have also proposed to use deep learning and reinforcement learning techniques to solve the autonomous exploration problem. Bai [56] presented a supervised learning method that chooses the next step which has a fixed distance to the robot and maximizes the expected

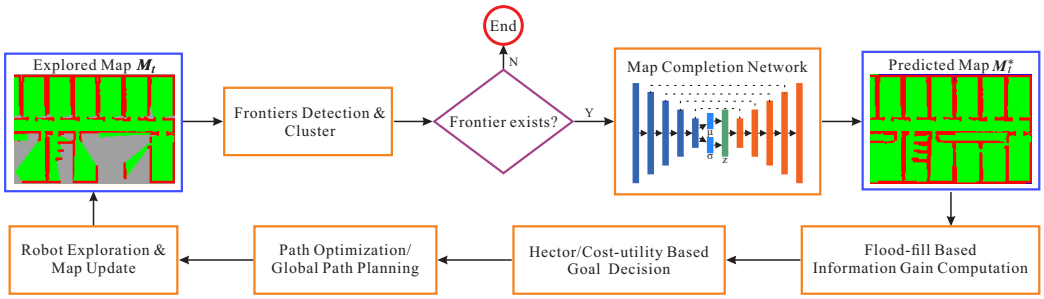

Figure 3.2: System workflow.

information gain from 36 candidate actions. Lei et al. [57] proposed a reinforcement learning based method aiming to avoid obstacles in exploration.

In contrast to these methods, we propose using deep learning to auto-complete unseen map regions in geometric detail and use this filled-in map to measure expected information gain. We then combine this prediction with different exploration strategies to improve exploration efficiency.

3.2 Problem Statement

Consider a nonholonomic wheeled robot equipped with a limited visibility laser range scanner, which is operating in a bounded 2D environment $E \subset \mathbb{R}^2$. It incrementally builds an occupancy grid representation $\boldsymbol{M}_t$ from the exploration of environment E, where t is the time step. The goal of autonomous exploration is to plan a sequence of actions $\mathcal{A}$ by which the area of $\boldsymbol{M}_t$ is maximized in every time step t. Here each action $a \in \mathcal{A}$ belongs to Lie Algebra [30] of special Euclidean group $SE(2)$. Due to the NP-hardness of this problem, we cannot obtain the optimal solution in polynomial time, thus we relax it as a minimization of an immediate cost $\boldsymbol{C}$ which is a function of path cost $\boldsymbol{C}_p$, possibility of colliding with obstacles $\boldsymbol{C}_o$ and expected information gain $\boldsymbol{I}$ to determine the actions $a \in \mathcal{A}$.

3.3 System Overview

Figure 3.2 shows the work-flow of the proposed method. Based on the current explored map $\boldsymbol{M}_t$, we first detect all the frontier points $\mathcal{F}_t$, then classify them into clusters $\boldsymbol{F}_t^i \subseteq \mathcal{F}_t$ using the DB-SCAN algorithm [58], where $i = 1, \ldots, N$. N is the number of clusters. Then we use the VAE network to predict the occupancy of cells in the unseen regions beyond each frontier cluster $\boldsymbol{F}_t^i$, to obtain a predicted map $\boldsymbol{M}_t^*$. We then compute the information gain $\boldsymbol{I}_t^i$ for each frontier cluster $\boldsymbol{F}_t^i$ using the predicted map $\boldsymbol{M}_t^*$, as input to the otherwise-standard Hector exploration [1] or cost-utility exploration [42, 59] which generates a feasible path for the robot's exploration.

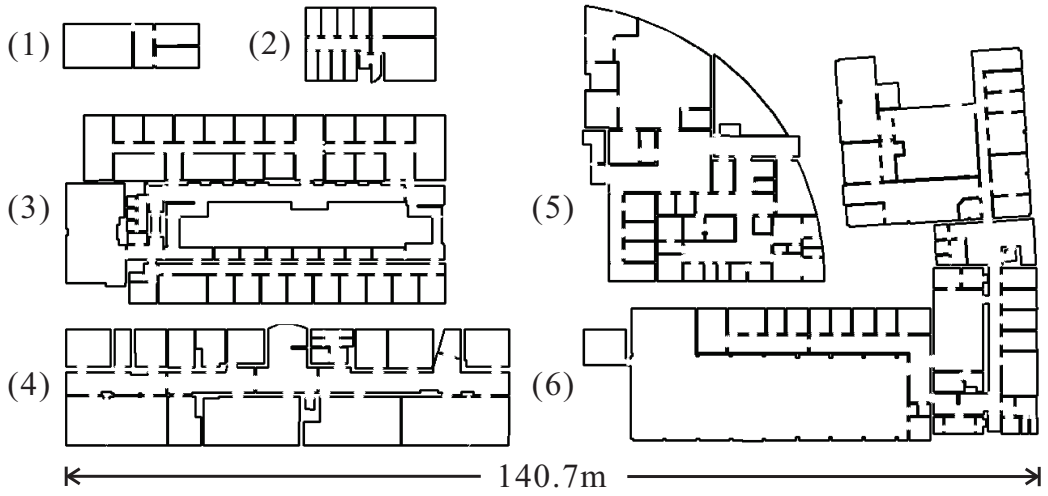

Figure 3.3: Some floor-plan examples in KTH dataset.

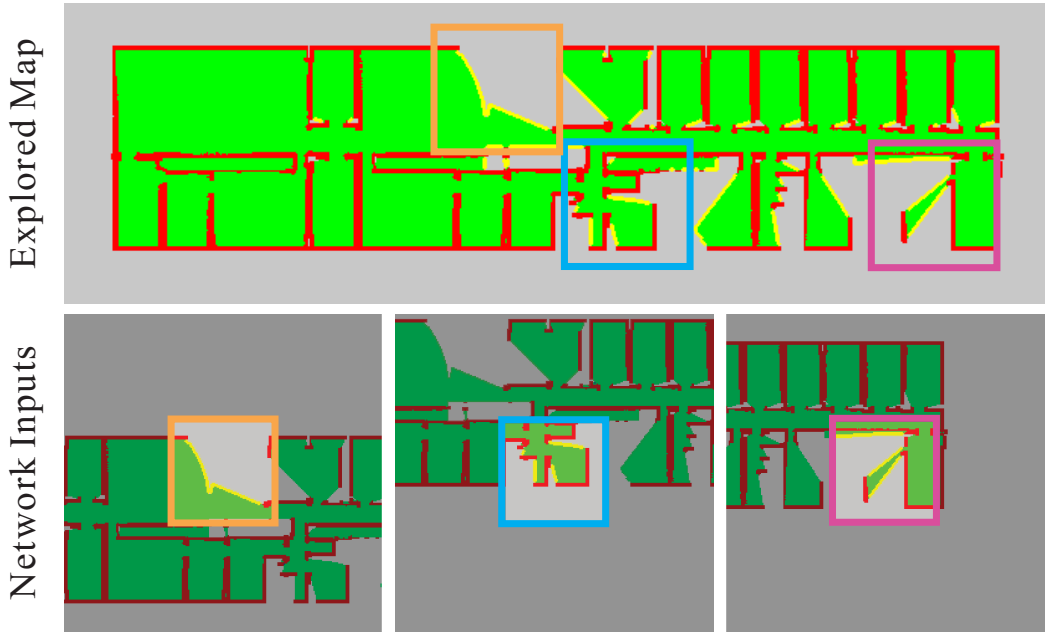

Figure 3.4: **Network inputs generation.** The upper image shows the current explored map with frontiers marked in yellow. The lower row shows three inputs for the map completion network, whose sizes are all 256×256 with a mask (prediction area) size of 80×80, centered in frontier cluster centers. The network is trained to generate the true contents of the masked center region given the input.

3.4 Map Completion Network

Recently, generative networks have shown impressive performance in semantic inpainting and inferring large missing regions in images with high accuracy [60, 61, 62]. Similar ideas have been applied to predict missing parts of partial 2D maps [63, 64] where a local set of robot-centric sensor measurements are used to infer a single missing observation.

We use the VAE network [48] for map completion because of its training/testing speed and the ability to predict large unknown regions. Alternatives like Generative Adversarial Networks [49] are too computationally demanding for online planning. [63, 64] use Deep Sum-Product Networks [65] to get performance similar to GANs, but the prediction is limited to a narrow Field of View (FOV).

3.4.1 Dataset

We use the KTH dataset [66] (see Figure 3.3 for some examples) to generate partial maps and ground truth for training the map completion network. The KTH dataset provides 165 floor plans with 6,248 rooms on the KTH campus. We manually cleaned some repeated floor plans and randomly split them into training and testing sets with a proportion of 3:1. We use Hector exploration [1] to explore the map and record a 256×256 region centered around each frontier cluster center. We encode obstacles, free space, and unknown space into different color channels and append an 80×80 mask to specify the region we want to predict. Figure 3.4 shows an example of a generated dataset for the proposed network.

3.4.2 Network Structure

Figure 3.5 shows the architecture of our map prediction network. The encoder part of the network learns to output a latent representation z in a lower dimensional space and the decoder reconstructs the missing parts of the map using the compact latent representation.

Encoder: Our encoder is based on the ResNet architecture [67]. We use the ResNet architecture on account of its demonstrated performance in the task of image classification on benchmark datasets and short training time.

The encoder outputs a mean (μ) and a log variance $log(\sigma)$ of the encoding of size 8×8×512. The final encoding fed to the decoder network is sampled from the Gaussian distribution $\mathcal{N}(\mu, \sigma)$. We do not use fully connected layers as the number of parameters in the network explodes and they did not yield a significant improvement in our results.

Decoder: The decoder network is essentially a mirror image of the encoder. While the encoder reduces the size of the feature plane while increasing their number, the decoder does the opposite. This is achieved by transposed convolution layers (also known as deconvolution layers) [68]. The output from the decoder network is a probability map of obstacles, with pixel values in the range 0 to 1. We then apply a threshold of 0.5 to determine obstacles and free spaces.

The network weights are randomly initialized without pre-training, because our training data is significantly different from standard benchmark datasets such as ImageNet [69].

3.4.3 Network Loss Functions

Prediction Loss

The prediction task in our case can be viewed as a pixel-wise binary classification: for each pixel we have to predict whether it is an obstacle (label 1) or free space (label 0). Hence, we choose Binary Cross Entropy loss between the network output x_n and its true label y_n for pixel n. The loss for a single pixel n is given as:

$$l_n = -[y_n \cdot log x_n + (1 - y_n) \cdot log(1 - x_n)] \tag{3.1}$$

The total loss of the prediction is the sum of all the pixel-wise losses for the masked 80×80 region,

$$L_{prediction} = \sum l_n. \tag{3.2}$$

Kullback-Leibler (KL) Divergence Loss

The KL-divergence on the latent encoding is used as a loss function to penalize the divergence of the encoding from a Standard Normal distribution The loss is given as:

$$L_{kld} = \sigma^2 + \mu^2 - log(\sigma) - 1 \tag{3.3}$$

The KL-divergence loss acts as a regularizer for the encoder by constraining the latent space. Please refer to [48] for its formal justification.

The final loss is the weighted sum of the two losses $L_{prediction}$ and L_{kld} from Equation (3.2) and Equation (3.3).

$$L_{final} = \gamma L_{prediction} + (1 - \gamma) L_{kld} \tag{3.4}$$

where $\gamma \in (0, 1)$ controls the relative importance of two losses. We use $\gamma = 0.99$ for training our network.

3.5 Map completion augmented exploration

3.5.1 Information Gain Computation

For estimating the information gain we need to find the regions that can be immediately explored from a frontier cluster, which is achieved by using the flood-fill algorithm. Starting from a frontier point, a Depth First Search is performed to find all the connected pixels that are unknown in the current map $\boldsymbol{M}_t$, but are free according to the predicted map $\boldsymbol{M}_t^*$. Figure 3.6 illustrate the flood-fill regions with and without map prediction.

The canonical definition of information gain for each frontier cluster $\boldsymbol{F}_t^i$ is:

$$\boldsymbol{I}_t^i(\boldsymbol{M}, x_i) = \boldsymbol{H}(\boldsymbol{M}) - \boldsymbol{H}(\boldsymbol{M}|x_i), \tag{3.5}$$

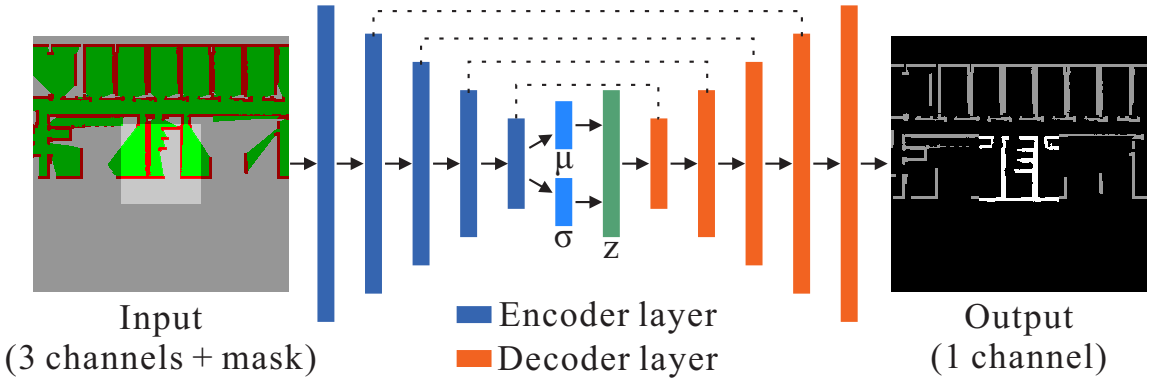

Figure 3.5: **The architecture of our map prediction network.** The input is a four-channel image containing obstacles (red), free space (green), unknown space (gray) and a mask for prediction region. The output is a single-channel image representing the probability of obstacles. We use a threshold of 0.5 to binarize output into obstacle and free space.

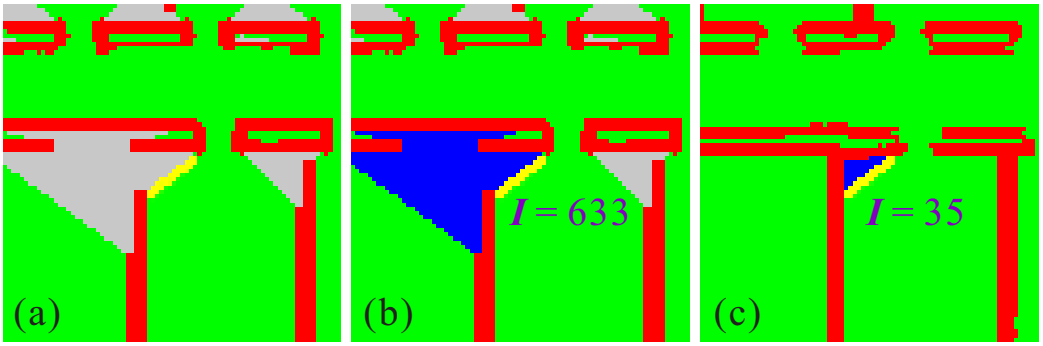

Figure 3.6: Comparison of flood-filled region without and with map prediction. Frontiers and flood-filled regions are marked as yellow and blue respectively. (a) Current explored map. (b) and (c) are flood-fill regions without and with map prediction. The information gain of the frontiers can be truthfully reflected from the map prediction in (c).

where $\boldsymbol{H}(\boldsymbol{M})$ is the *current entropy* of the map at time t and $\boldsymbol{H}(\boldsymbol{M}|x_i)$ is the *posterior entropy* of the predicted map with the new flood-filled map information x_i. The Shannon's entropy [70] over an occupancy grid map $\boldsymbol{M}$ is defined as

$$\boldsymbol{H}(\boldsymbol{M}) = -\sum_j p(m_j)\log p(m_j) + (1-p(m_j))\log(1-p(m_j)), \tag{3.6}$$

where $p(m_j)$ is the probability of the cell m_j being occupied. An unknown cell m_j in $\boldsymbol{M}$ decreases the entropy by 1, since $p(m_j) = 0.5$. In comparison, the known cells do not contribute to the information gain. The information gain $\boldsymbol{I}_t^i$ is then equal to the number of unknown cells that will become known which is in keeping with our flood-fill based method.

3.5.2 Augmenting Exploration Planners

The expected information gain $\boldsymbol{I}_t^i$ from the proposed map prediction can be used to augment different kinds of goal planners or path optimizers instead of just a specific method. Here we choose an exploration transform [71] based planner known as Hector exploration planner [1] and cost-utility function based planner [42, 41] as base methods, which are two representative methods for path optimization and navigation goal planning respectively.

Augmented Hector Planner

Original Hector exploration [1] builds a costmap known as an exploration transform [71] which assigns a cost to every explored free cell in the current map $\boldsymbol{M}_t$. The cost has two parts, i.e. the path cost to the nearest frontier point, and an obstacle cost to the nearest obstacle. Inspired by this idea, we built our cost transform $\boldsymbol{C}(m)$ for the free cell m by additionally considering our information gain estimate as follows:

$$\boldsymbol{C}(m) = \min_{\mathbf{F}_t^i \subseteq \mathcal{F}_t} \left\{ \min_{f \in \mathbf{F}_t^i} \boldsymbol{C}_p(f) - \alpha \sqrt{\boldsymbol{I}_t^i} \right\} + \beta \boldsymbol{C}_o, \tag{3.7}$$

where $\boldsymbol{C}_p(f)$ is the path cost from cell m to frontier cell f, $\boldsymbol{I}_t^i$ is the information gain for frontier cluster $\boldsymbol{F}_t^i$, and $\boldsymbol{C}_o$ is the collision cost of the cell m formulated as a thresholded distance to nearest obstacle. α and β are the weights for information gain and collision cost respectively. Since information gain is proportional to expected unmapped free areas (m^2) while path cost is related to length (m), we use the square root for information gain $\boldsymbol{I}_t^i$ to keep their units consistent. The square root response function for information gain cost also helps to dampen the high information gain estimates due to potential mispredictions. Note that the cost transform $\boldsymbol{C}(m)$ for each cell m can be computed efficiently in an incremental way by propagating the costs starting from the frontier points.

After evaluating our cost transform, we can plan a path from the current robot pose by following the steepest gradient of the transform until it reaches a frontier point.

Augmented Cost-Utility Planner

The cost-utility exploration scheme [42, 41, 72] assigns a cost to each frontier point based on its travel cost $\boldsymbol{C}_p(f)$: length of the shortest path to the frontier f, and utility (information gain), which is $\sqrt{\boldsymbol{I}(f)}$ in our case.

$$\boldsymbol{C}(f) = \boldsymbol{C}_p(f) - \lambda \sqrt{\boldsymbol{I}(f)} \tag{3.8}$$

where $\boldsymbol{I}(f) = \boldsymbol{I}_t^i$ for $f \in \boldsymbol{F}_t^i$ and λ is a weight to adjust relative importance between the cost and the utility. The frontier with the minimum cost is then chosen to be the next navigation goal and a global path planner (Dijkstra algorithm in our implementation) is used to generate a path to the goal.

3.6 Experiments

3.6.1 Experimental Setup

Simulation

The simulations for both dataset generation and evaluations were performed using the Stage simulator [73] Pioneer P2-DX mobile robot and SICK LMS scanning laser rangefinder (LRF) models. The FOV and range of the LRF were limited to 270° and $5m$ respectively with 512 samples. The sensor update rate was 10 Hz in simulation time. The system is implemented in ROS [74].

The environment map is a 2D occupancy grid with a resolution of 0.1m per pixel. Map updates are done by ray-casting range measurements for each laser range reading to label free and obstacle cells. To allow fast dataset generation and evaluations, we use ground truth localization from the simulation rather than running SLAM. Since we assume good localization/mapping and do not aim to optimize the path for lower pose uncertainty [75], the use of ground truth localization is justified.

Baselines for Comparison

The comparison baselines we use for our evaluations are: Nearest Frontier Exploration ("nearest_frontier") [38], Hector exploration planner ("original_hector") [1] and cost-utility exploration ("original_cost_utility") [41].

The Hector exploration planner augmented with our information gain is referred to as "ig_hector" and the cost-utility planner is "ig_cost_utility". As an upper bound we also compare the performance of the Hector planner augmented by information gain from ground truth map ("ig_hector_gt"): the impossible perfect oracle for map prediction. Note that the use of ground truth information gain does not guarantee a perfect exploration as it is still a greedy heuristic that in general will not lead to a globally optimal solution for the coverage of whole map. Thus, this provides a tight upper bound for a well-informed local exploration method.

Configuration Settings

For fair comparison, we set the weights for information gain in "ig_hector" and "ig_cost_utility" the same, viz. $\alpha = 3$ (Equation (3.7)) and $\lambda = 3$ (Equation (3.8)), and we set $\beta = 5$ in Equation (3.7).

3.6.2 Map Completion Network Evaluation

Evaluation Metrics

Following prior authors [40, 44], we measure the average percentage of area coverage against exploration time to evaluate exploration efficiency. We also report the average time and travel distances taken to achieve 85% map coverage. This approach is similar to the one proposed in [40] where the exploration is stopped after a fixed duration of time. This strategy allows us to ignore some minor

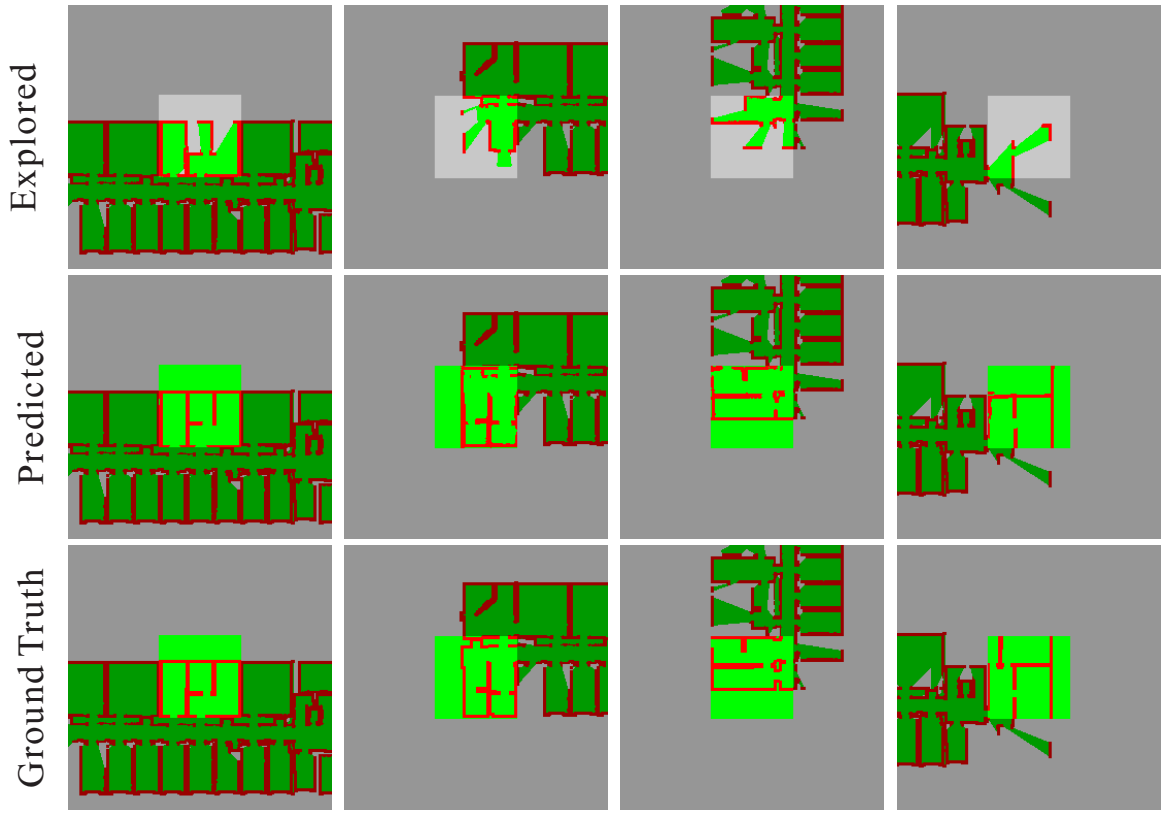

Figure 3.7: Four map prediction results by the VAE network. In the first three columns, the topology of the map is correctly predicted while the fourth column shows an incorrect but plausible prediction.

unmapped corners to better reflect the exploration efficiency. Since the size of our maps vary, selecting a constant time threshold will not lead to a fair comparison. Hence, we choose percentage of area covered instead.

We present the results for 12 different maps from the testing dataset of the KTH floor-plans. For each map and each method, we run 20 tests from different initial robot poses chosen at random and evaluate their performance.

For training the map completion network, we first run 50 explorations using the original Hector planner for each map in the training dataset and crop a 256×256 region around each frontier cluster. This results in a training dataset of 1.5 million partial maps. Then we test the map prediction performance on the testing maps with about 140,000 inputs. The network takes on average 5 milliseconds with 1 millisecond standard deviation to predict one batch of frontiers (32 inputs) on an Nvidia GTX980 GPU. We obtain overall map cell contents prediction accuracy of 92.5%; with precision and recall for free space prediction of 95.0% and 96.3% respectively. The precision/recall for obstacle predictions are 77.0% and 70.6% respectively. Some examples of our network's predictions are

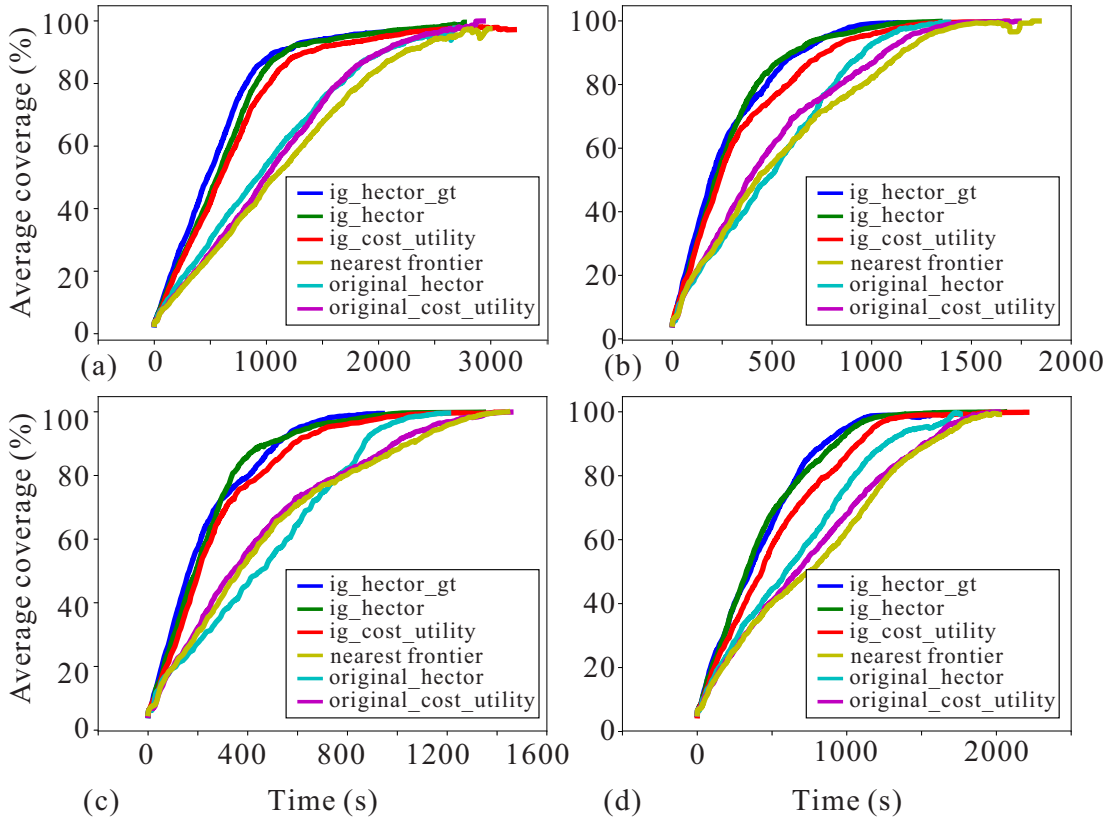

Figure 3.8: Percentage map coverage against time for 4 floor plans. Results are averaged over 20 experiments.

shown in Figure 3.7. From these evaluations and Figure 3.7, we can see the proposed VAE network can complete maps with high accuracy.

3.6.3 Analysis of Exploration Efficiency

Area Coverage Over Time

Figure 3.8 shows the percentage of total map coverage over exploration time. We compare our "ig_hector" and "ig_cost_utility" methods with baselines as well as the Hector planner [1] augmented by oracular information gain from ground truth maps ("ig_hector_gt"). The curves of Figure 3.8 show that the map-prediction augmented methods explore the map more quickly than the baselines in the exploration process. All methods eventually converge to near-perfect coverage, but the augmented methods do so earlier than the baselines. We notice that the nearest frontier planner and original cost-utility methods show similar performance while the original Hector is slightly better than the other two. In contrast, the proposed "ig_hector" perform much better, reaching nearly the same exploration efficiency of the upper bound oracle-informed method.

The results suggest the VAE network has learned to predict unseen map areas well enough to usefully improve exploration efficiency.

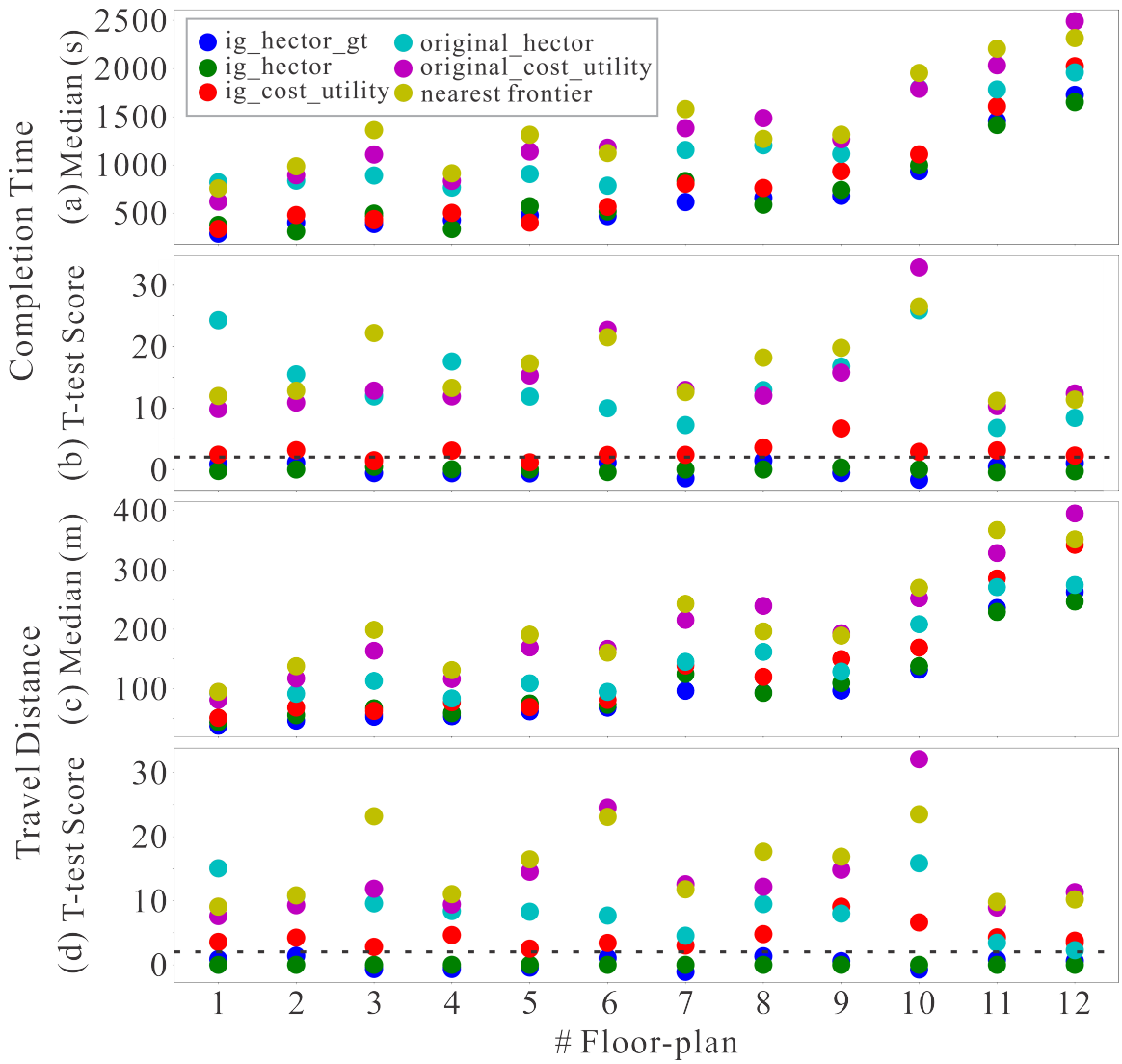

Figure 3.9: For each method, we show the median task completion time and trajectory length for 12 different floor-plans, over 20 trials each with different initial robot pose. Smaller values indicate better performance.

Travel Distance and Time

Figures 3.9(a) and (c) show the median time taken and distance traveled to cover 85% of the total area using different methods for the 12 test floor plans.

To assess the statistical significance of the difference between map completion time and distance traveled for the baseline methods against our methods, we use Welch's t-test [76]. We choose the proposed "ig_hector" as our basis for comparison. A higher t-value implies a higher confidence that baselines' map completion times or traveled distances are longer than the proposed "ig_hector" method. Figures 3.9(b) and (d) show the t-values for map completion times and travel distances of all the tests in 12 floor-plans. The t-value corresponding to 95% confidence level is shown in the graph as a dotted line.

Figures 3.9(a)–(d) show that the proposed methods "ig_hector' and "ig_cost_utility" always have lower map completion time and distance traveled compared to the baseline methods they were based on viz. "original_hector" and "original_cost_utility" respectively. The measured performance distribution of the enhanced methods are better, and differs with a large confidence interval, compared to their baselines.

Our method "ig_cost_utility" has similar travel distance to "original_hector" in some of the maps. This is likely explained by the superior performance of hector exploration compared to cost-utility exploration canceling out the advantage of better predictions.

Overall, these experiments show that the map prediction from the proposed VAE network results in a performance increase in travel time and distance traveled compared to previous methods in the maps we tested.

3.7 Conclusion and Future Work

We described a VAE deep neural network that learns to predict unseen regions of building floor plans. This model is used to enhance exploration performance by improving estimates used in path planning, leading to increased efficiency. We suggest that the navigation behavior of our system is more intelligible than that of end-to-end deep learning techniques as it splits the system into a learned generative model with a map output that is easily interpreted by humans, within a well known information-theoretic framework.

In future work, we aim to implement our system on real robots and evaluate its performance. The work can be extended to 3D reconstruction rather than 2D, using ground or aerial robots. Furthermore, we assume accurate localization and mapping so far. Removing this assumption and actively planning for efficient exploration and with good quality localization and mapping would be an interesting extension.

Reproduction, code and data products

All code, data and models used in the experiments are available at `https://git.io/fAX7k`.

Chapter 4

Conclusion

Creating a map of an environment is an important ability in many robotic applications. Therefore, in this thesis we explored a) A 3D odometry and mapping system producing metric scale maps and pose estimates using a minimal sensor suite, and b) An autonomous ground robot for 2D mapping of an unknown environment using learned map prediction.

The visual-inertial odometry system demonstrates superior robustness and camera tracking accuracy compared with the original vision-only method. The system is able to produce 3D maps in metric scale and can handle quick rotations, addressing the well-known limitations of monocular odometry/SLAM systems

The autonomous exploration system successfully employs a state-of-the-art generative neural network to predict unknown regions of a partially explored map, and uses the prediction to enhance the exploration in an information-theoretic manner. Evaluation against traditional exploration methods in simulation using floor plans of real buildings demonstrates advantages in terms of exploration efficiency, while still maintaining the explicability of the overall system.

The visual-inertial odometry system was implemented jointly by me and Ph.D. student Sicong Tang from Simon Fraser University through pair programming. The work was closely supervised by Professor Ping Tan along with helpful guidance from Professor Richard Vaughan.

The autonomous exploration system was implemented in collaboration with Ph.D. student Fei-Peng Tian from Tianjin University, China while he was a visiting student at Simon Fraser University. The original exploration pipeline development (simulation interfaces, real-time mapping, integration with hector exploration planner) and neural network design/implementation were performed by me. Mr. Tian developed software modules for data collection, oversaw the dataset generation and network training process along with implementation of baselines by modifying the existing pipeline. This project was jointly supervised by Professor Richard Vaughan and Professor Ping Tan.

Bibliography

[1] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, and O. von Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots," in *RoboCup 2013: Robot World Cup XVII*, Berlin, Heidelberg, 2014, pp. 624–631.

[2] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3. IEEE, 2018, pp. 611–625.

[3] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[4] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1935–1942.

[5] J. Oh, V. Chockalingam, S. Singh, and H. Lee, "Control of Memory, Active Perception, and Action in Minecraft," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. JMLR.org, 2016, pp. 2790–2799.

[6] T. D. Kulkarni, A. Saeedi, S. Gautam, and S. J. Gershman, "Deep successor reinforcement learning," *arXiv preprint arXiv:1606.02396*, 2016.

[7] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu *et al.*, "Learning to navigate in complex environments," *arXiv preprint arXiv:1611.03673*, 2016.

[8] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," in *The International Journal of Robotics Research*, vol. 34, no. 3. SAGE Publications Sage UK: London, England, 2015, pp. 314–334.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," in *IEEE Transactions on Robotics*, vol. 31, no. 5. IEEE Robotics and Automation Society, 2015, pp. 1147–1163.

[10] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," in *IEEE Transactions on Robotics*, vol. 34, no. 4. IEEE, 2018, pp. 1004–1020.

[11] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," in *IEEE Robotics and Automation Letters*, vol. 2, no. 2. IEEE, 2017, pp. 796–803.

[12] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 15–22.

[13] V. Usenko, J. Engel, J. Stückler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1885–1892.

[14] A. Concha, G. Loianno, V. Kumar, and J. Civera, "Visual-inertial direct SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016, pp. 1331–1338.

[15] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation," in *Proceedings of Robotics: Science and Systems*, 2015. [Online]. Available: http://doi.org/10.15607/RSS.2015.XI.006

[16] L. von Stumberg, V. Usenko, and D. Cremers, "Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2018.

[17] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," in *IEEE Transactions on Robotics*, vol. 32, no. 6. IEEE, 2016, pp. 1309–1332.

[18] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2007, pp. 225–234.

[19] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 2320–2327.

[20] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2013, pp. 2100–2106.

[21] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense SLAM and light source estimation," in *The International Journal of Robotics Research*, vol. 35, no. 14. SAGE Publications Sage UK: London, England, 2016, pp. 1697–1716.

[22] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE International conference on Robotics and automation*. IEEE, 2007, pp. 3565–3572.

[23] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," in *The International Journal of Robotics Research*, vol. 32, no. 6. SAGE Publications Sage UK: London, England, 2013, pp. 690–711.

[24] K. Wu, A. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, "A Square Root Inverse Filter for Efficient Vision-aided Inertial Navigation on Mobile Devices." in *Robotics: Science and Systems*. MIT Press Journals, 2015.

[25] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 298–304.

[26] B. M. Bell and F. W. Cathey, "The iterated Kalman filter update as a Gauss-Newton method," in *IEEE Transactions on Automatic Control*, vol. 38, no. 2. IEEE, 1993, pp. 294–297.

[27] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?" in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 2657–2664.

[28] K. Konolige, M. Agrawal, and J. Sola, "Large-scale visual odometry for rough terrain," in *Robotics research*. Springer, 2010, pp. 201–212.

[29] P. Gemeiner, P. Einramhof, and M. Vincze, "Simultaneous motion and structure estimation by fusion of inertial and vision data," in *The International Journal of Robotics Research*, vol. 26, no. 6. Sage Publications Sage UK: London, England, 2007, pp. 591–605.

[30] W. M. Boothby, *An introduction to differentiable manifolds and Riemannian geometry*, ser. Pure Appl. Math. Academic Press, 1986, vol. 120. [Online]. Available: https://cds.cern.ch/record/107707

[31] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A first-estimates jacobian ekf for improving slam consistency," in *Experimental Robotics*, O. Khatib, V. Kumar, and G. J. Pappas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 373–382.

[32] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," in *The International Journal of Robotics Research*, vol. 35, no. 10. SAGE Publications Sage UK: London, England, 2016, pp. 1157–1163.

[33] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-Squares Fitting of Two 3-D Point Sets," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5. IEEE, 1987, pp. 698–700.

[34] K. Otsu, A.-A. Agha-Mohammadi, and M. Paton, "Where to Look? Predictive Perception with Applications to Planetary Exploration," in *IEEE Robotics and Automation Letters*, vol. 3, no. 2. IEEE, 2018, pp. 635–642.

[35] A. Ahrary, A. A. Nassiraei, and M. Ishikawa, "A study of an autonomous mobile robot for a sewer inspection system," in *Artificial Life and Robotics*, vol. 11, no. 1. Springer, 2007, pp. 23–27.

[36] D. Klimentjew, M. Arli, and J. Zhang, "3D scene reconstruction based on a moving 2D laser range finder for service-robots," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2009, pp. 1129–1134.

[37] T. Neumann, A. Ferrein, S. Kallweit, and I. Scholl, "Towards a mobile mapping robot for underground mines," in *Proceedings of the PRASA, RobMech and AfLaI Int. Joint Symposium, Cape Town, South Africa*, 2014. [Online]. Available: www.prasa.org/proceedings/2014/prasa2014-48.pdf

[38] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *IEEE International Symposium on Computational Intelligence in Robotics and Automation*. IEEE, 1997, pp. 146–151.

[39] A. Elfes, in *Using occupancy grids for mobile robot perception and navigation*, vol. 22, no. 6. IEEE, June 1989, pp. 46–57.

[40] J. M. Pimentel, M. S. Alvim, M. F. Campos, and D. G. Macharet, "Information-driven rapidly-exploring random tree for efficient environment exploration," in *J. Intell. Robotics Syst.*, vol. 91, no. 2. Norwell, MA, USA: Kluwer Academic Publishers, Aug. 2018, pp. 313–331. [Online]. Available: https://doi.org/10.1007/s10846-017-0709-0

[41] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1. IEEE, 2002, pp. 540–545.

[42] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept 2017, pp. 1396–1402.

[43] M. G. Jadidi, J. V. Miro, and G. Dissanayake, "Mutual information-based exploration on continuous occupancy maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6086–6092.

[44] S. Bai, J. Wang, F. Chen, and B. Englot, "Information-theoretic exploration with Bayesian optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1816–1822.

[45] P. Pfaff, C. Plagemann, and W. Burgard, "Gaussian mixture models for probabilistic localization," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2008, pp. 467–472.

[46] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of artificial intelligence research*, vol. 11, pp. 391–427, 1999. [Online]. Available: https://doi.org/10.1613/jair.616

[47] S. Thrun, "A probabilistic on-line mapping algorithm for teams of mobile robots," in *The International Journal of Robotics Research*, vol. 20, no. 5. SAGE Publications, 2001, pp. 335–363.

[48] D. P. Kingma and M. Welling, "Auto-encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[49] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[50] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using Rao-Blackwellized particle filters." in *Robotics: Science and Systems (RSS)*, vol. 2, 2005, pp. 65–72. [Online]. Available: http://www.informatik.uni-freiburg.de/~stachnis/pdf/burgard05snowbird.pdf

[51] D. P. Ström, I. Bogoslavskyi, and C. Stachniss, "Robust exploration and homing for autonomous robots," in *Robotics and Autonomous Systems*, vol. 90. North-Holland Publishing Co., 2017, pp. 125–135.

[52] D. P. Ström, F. Nenci, and C. Stachniss, "Predictive exploration considering previously mapped environments," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2761–2766.

[53] A. Elfes, "Robot navigation: Integrating perception, environmental constraints and task execution within a probabilistic framework," in *Reasoning with Uncertainty in Robotics*. Springer Berlin Heidelberg, 1996, pp. 91–130.

[54] P. Whaite and F. P. Ferrie, "Autonomous exploration: Driven by uncertainty," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3. IEEE, 1997, pp. 193–205.

[55] J. Vallvé and J. Andrade-Cetto, "Potential information fields for mobile robot exploration," in *Robotics and Autonomous Systems*, vol. 69. Elsevier, 2015, pp. 68–79.

[56] S. Bai, F. Chen, and B. Englot, "Toward autonomous mapping and exploration for mobile robots through deep supervised learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2379–2384.

[57] T. Lei and L. Ming, "A robot exploration strategy based on Q-learning network," in *IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2016, pp. 57–62.

[58] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "Density-based spatial clustering of applications with noise," in *International Conference on Knowledge Discovery and Data Mining*, vol. 240. AAAI Press, 1996.

[59] H. H. González-Baños and J.-C. Latombe, "Navigation strategies for exploring indoor environments," in *The International Journal of Robotics Research*, vol. 21, no. 10-11. Thousand Oaks, CA, USA: Sage Publications, Inc., 2002, pp. 829–848. [Online]. Available: https://doi.org/10.1177/0278364902021010834

[60] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," in *ACM Transactions on Graphics*, vol. 36, no. 4. ACM, 2017, p. 107.

[61] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative Image Inpainting with Contextual Attention," *arXiv preprint arXiv:1801.07892*, 2018.

[62] R. A. Yeh, C. Chen, T.-Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic Image Inpainting with Deep Generative Models." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, no. 3. IEEE, 2017, p. 4.

[63] A. Pronobis and R. P. Rao, "Learning deep generative spatial models for mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 755–762.

[64] A. Pronobis, F. Riccio, and R. P. Rao, "Deep spatial affordance hierarchy: Spatial knowledge representation for planning in large-scale environments," in *International Conference on Automated Planning and Scheduling Workshop*. AAAI Press, 2017.

[65] H. Poon and P. Domingos, "Sum-product networks: A new deep architecture," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*. IEEE, 2011, pp. 689–690.

[66] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38,000 rooms? Reasoning about unexplored space in indoor environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 4675–4682.

[67] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016.

[68] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010.

[69] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," in *International Journal of Computer Vision*, vol. 115, no. 3. Springer, 2015, pp. 211–252.

[70] C. E. Shannon, "A mathematical theory of communication," in *The Bell System Technical Journal*, vol. 27, no. 3. Nokia Bell Labs, July 1948, pp. 379–423.

[71] S. Wirth and J. Pellenz, "Exploration transform: A stable exploring algorithm for robots in rescue environments," in *IEEE International Workshop on Safety, Security and Rescue Robotics*. IEEE, 2007, pp. 1–5.

[72] M. Juliá, A. Gil, and O. Reinoso, "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments," in *Autonomous Robots*, vol. 33, no. 4. Springer, 2012, pp. 427–444.

[73] R. Vaughan, "Massively multi-robot simulation in stage," in *Swarm intelligence*, vol. 2, no. 2-4. Springer, 2008, pp. 189–208.

[74] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009. [Online]. Available: http://www.willowgarage.com/sites/default/files/icraoss09-ROS.pdf

[75] H. Carrillo, I. Reid, and J. A. Castellanos, "On the comparison of uncertainty criteria for active SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 2080–2087.

[76] B. L. Welch, "The generalization of student's' problem when several different population variances are involved," in *Biometrika*, vol. 34, no. 1/2. JSTOR, 1947, pp. 28–35.