Graph-Based Thermal–Inertial SLAM With Probabilistic Neural Networks

Graph-based Thermal-Inertial SLAM with Probabilistic Neural Networks

Muhamad Risqi U. Saputra, Chris Xiaoxuan Lu, Pedro Porto B. de Gusmao, Bing Wang, Andrew Markham, and Niki Trigoni

***Abstract*—Simultaneous Localization and Mapping (SLAM) system typically employ vision-based sensors to observe the surrounding environment. However, the performance of such systems highly depends on the ambient illumination conditions. In scenarios with adverse visibility or in the presence of airborne particulates (e.g. smoke, dust, etc.), alternative modalities such as those based on thermal imaging and inertial sensors are more promising. In this paper, we propose the first complete thermal-inertial SLAM system which combines neural abstraction in the SLAM front end with robust pose graph optimization in the SLAM back end. We model the sensor abstraction in the front end by employing probabilistic deep learning parameterized by Mixture Density Networks (MDN). Our key strategies to successfully model this encoding from thermal imagery are the usage of normalized 14-bit radiometric data, the incorporation of hallucinated visual (RGB) features, and the inclusion of feature selection to estimate the MDN parameters. To enable a full SLAM system, we also design an efficient global image descriptor which is able to detect loop closures from thermal embedding vectors. We performed extensive experiments and analysis using three datasets, namely self-collected ground robot and handheld data taken in indoor environment, and one public dataset (SubT-tunnel) collected in underground tunnel. Finally, we demonstrate that an accurate thermal-inertial SLAM system can be realized in conditions of both benign and adverse visibility.**

***Index Terms*—Thermal-inertial SLAM, loop closure detection, probabilistic deep neural networks, pose graph optimization.**

I. Introduction

SIMULTANEOUS Localization and Mapping (SLAM) is an important task in robotics and autonomous systems. It enables a mobile agent to explore an unknown environment by simultaneously estimating the position of the agent whilst constructing a representation of the environment, termed a *map*. This task is a precursor to many other robotic tasks such as navigation, exploration, or manipulation, making accurate SLAM estimation a fundamental need for autonomous systems.

A SLAM framework typically consists of a *front end* and a *back end*. The front end acquires sensor data and transforms it into an abstraction that is more amenable for inference, while the back end estimates the states of the agent given the abstracted data from the front-end. The back end is also responsible for optimizing the agent states and generating a globally consistent representation of the environment [1], [2].

Most front ends in SLAM systems utilize range (e.g. depth, Lidar) or vision (RGB) sensors to sense the surrounding environment. Notable examples include ORB-SLAM [3] and LOAM [4] which employ RGB and Lidar sensors respectively for their SLAM front end. While these range- and vision-based SLAM systems can generally work well in a wide range of applications, their performance largely depends on the benign visibility. When it comes to the adverse illumination conditions and/or in the presence of airborne particulates (e.g. dust, soot, smoke, etc.), using existing range and vision sensors for SLAM estimation is problematic. For instance, it is widely known that RGB cameras cannot operate in darkness while depth cameras are sensitive to glare and strong illumination [5]–[7]. The same visibility issues also applies to RGB, depth, and even Lidar sensors when operating in environments with airborne particulates [8] or thick fog/mist. In contrast, thermal imaging cameras are not affected by illumination conditions and the presence of most airborne particulates [9]. Instead, they capture the Long Wave Infrared (LWIR) data emitted from objects in the environment. These advantages make thermal imaging cameras a viable alternative modality for SLAM application in visually-denied environments.

However, realizing a full thermal SLAM system comes with a set of challenging tasks. One of the most fundamental is how to abstract or encode the thermal data so as to maximally aid the graph optimization process. This is an intrinsically challenging task as thermal cameras capture the temperature profile of the environment instead of environmental appearance and geometry. The problem is even more pronounced with the fact that the re-scaled 8-bit resolution of thermal data has lower contrast, making standard feature matching and data association difficult. Moreover, thermal cameras periodically require suspension of camera operation for approximately 0.5-1 second to perform Non-Uniformity Correction (NUC) (also

This work was supported by EPSRC grant entitled "ACE-OPS: From Autonomy to Cognitive assistance in Emergency OPerationS" (EP/S030832/1) and by the US National Institute of Standards and Technology (NIST) grant entitled "Pervasive, Accurate, and Reliable LBS for Emergency Responders" (No. 70NANB17H185). The authors would like to thank John G. Rogers III and Arthur Schang from CCDC Army Research Laboratory (ARL) USA for their assistance on using SubT-tunnel dataset. Most parts of this work conducted when M. R. U. Saputra was a postdoc at University of Oxford. *(Corresponding author: M. R. U. Saputra.)*

M. R. U. Saputra is with Monash University, Indonesia. Email: risqi.saputra@monash.edu.

C. X. Lu is with the School of Informatics at the University of Edinburgh, United Kingdom (UK). Email: xiaoxuan.lu@ed.ac.uk.

Pedro P. B. de Gusmao is with the Department of Computer Science and Technology at the University of Cambridge, United Kingdom (UK). Email: pp524@cam.ac.uk.

B. Wang, A. Markham, and N. Trigoni are with the Department of Computer Science, University of Oxford, United Kingdom (UK). Email: {bing.wang, andrew.markham, niki.trigoni}@cs.ox.ac.uk.

known as Flat Field Correction (FFC) in other literature [10], [11]) in which a uniform temperature is presented to the sensor to estimate the fixed-pattern noise correction parameters. Together, these issues mean that traditional methods developed for other optical sensors fall short in the typical front-end abstraction pipeline (e.g. feature extraction, data association, estimating an odometry prior, etc.).

The past decade has witnessed the rapid development of Deep Neural Networks (DNN) as a strong non-linear function approximator. It has been seen in the recent works that DNNs can be successfully used in visual odometry [12]–[15] and (re-) localization estimation [16]–[18]. We therefore hypothesize that one can model the abstraction or encoding of thermal data for a SLAM front end using DNN. In particular, by employing a type of probabilistic neural network, i.e. Mixture Density Networks (MDN), we can fully model the front end by constructing both odometry and loop closure constraints along with their covariance as a metric of uncertainty. In this way a more traditional back end graph-based optimizer can be used to generate a global trajectory. In a nutshell, our key and novel insights in building a reliable pose graph from thermal imagery include the usage of normalized radiometric (14-bit resolution) thermal data to avoid re-scaling, the incorporation of hallucination networks as complementary information [19], the inclusion of selective fusion module [20] which filters out reliable features, and the use of a probabilistic DNN. We also present a novel approach to neural loop closure estimation. Combined with outlier rejection in the back end to filter noisy loop closure constraints, we demonstrate that is possible to achieve a complete thermal-inertial SLAM system which produces globally consistent trajectory estimation, in spite of the above mentioned challenges.

The work described in this article builds on our previous work in [19] which presented the first system for deep thermal-inertial odometry. The new contributions here can be summarized as follows:

- We demonstrate the first complete thermal-inertial SLAM (TI-SLAM) system in the literature, which combines robust pose graph optimization in the back end with neural abstraction in the front end generated by probabilistic neural networks.
- We construct odometry and loop closure constraints in the pose graph by using a Mixture Density Network (MDN) parameterized through hallucination and feature selection network given normalized 14-bit radiometric thermal data as the input. We also combine the odometry network with IMU measurements to increase robustness in unknown scenes or when the thermal imaging is performing NUC calibration.
- We present an efficient global descriptor-based neural loop closure detection based on thermal embedding vectors output by a DNN.
- We perform extensive experiments and analysis under both benign and poorly-illuminated conditions on in-house ground robot and handheld data (self-collected), and on a public ground robot data (SubT-tunnel) taken in underground tunnel. The code and in-house datasets are released to the community.[1]

II. Related Work

A. Conventional Visual SLAM

Visual SLAM was originally solved by filtering algorithm [21]. Notable examples include MonoSLAM [21] and its variants [22]–[24], in which every frame is processed by Extended Kalman Filter (EKF) to jointly estimate the camera pose and landmark locations. However, due to the nature of EKF algorithm which accumulates linearization errors across multiple frames, keyframe-based Bundle Adjustment (BA) approach is more widely used in the past decade since it has been shown to be more accurate than filtering [25]. Prominent examples from this category include PTAM [26] and ORB-SLAM [3] which employ point-based features in the front end or PL-SLAM [27] which utilizes line segment as the front end abstraction. These keyframe-based BA methods typically integrate hardware and algorithmic advances in the past decade by incorporating parallel computing, statistical model selection, loop closures detection based on bag-of-words place recognition, local BA, or other graph optimization approaches. However, despite their great performances in particular scenarios, these model based approaches are very sensitive to outliers (e.g. spurious correspondences, dynamic objects, etc.) [28] and easily lose tracks when the environment has limited hand-engineered features [29].

B. Deep Networks in the Context of SLAM

In the last couple of years, there are many works that aim to replace the SLAM front end with learning-based approaches. The learning-based approaches, especially based on DNN, are typically more robust as it does not rely on point or line features, but directly learn to solve the task from abundant data. Among these approaches, some deals with feature correspondences [30], [31], some with odometry [13], [15], [32], [33], global re-localization [17], [34], and place recognition [35], [36] or loop closure detection [37]. Nevertheless, the developed system is secluded from each other and is not trivial to be combined together as a single SLAM system.

1) Odometry Estimation: The first work on DNN based approach for Visual Odometry (VO) is pioneered by DeepVO [12] which models the camera pose estimation as an end-to-end pose regression problem. This was then followed by incorporating inertial (e.g. VINet [38], SelectFusion [20]), training the network by self-supervision (e.g. SfMLearner [32], GANVO [33], etc.), or combining together model-based and deep learning-based approaches (e.g. SalientDSO [15]). However, none of them address thermal camera system except our work in [19].

2) Place Recognition: NetVLAD [35] is a prominent place recognition algorithm which aggregates the statistics of local descriptors by computing the sum of residuals for each visual word. This approach was then typically improved by making it more robust across different environmental conditions by learning condition- and viewpoint-invariant features [36] or

[1] https://github.com/risqiutama/ti-slam

learning geometric features through depth generation [39]. Existing work on loop closure detection or place recognition using thermal camera is typically performed by using standard feature-based approaches (e.g. FAB-MAP [40]) and is aided by other modalities such as RGB camera [41].

3) Global Relocalization: In the context of SLAM, global relocalization can be used to construct loop closure constraints. PoseNet [34] is a pioneer work in this category which regresses the global camera pose given a single image as the input. This was then followed by incorporating an attention mechanism [17], enforcing temporal information [16], or fusing it with an additional sensor through variational inference [42]. While deep global relocalization can be used as loop closure constraints, recent work [43] observes that it cannot generalize in unseen scenario as they implicitly save the 'map' of the environment within the network. Different from the previous absolute pose regression methods, relative pose regression methods identifies the nearest neighbours in database images to the query image and recovers the relative pose between the reference images and the query. Specifically, NN-Net [44] utilises a neural network to estimate the pairwise relative poses between the query and the top N-ranked references. A triangulation-based fusion algorithm coalesces the predicted N relative poses and the ground truth of 3D geometry poses, and the absolute query pose can be naturally calculated. Furthermore, RelocNet [45] additionally exploits a frustum overlap loss to assist the learning of global descriptors that are suitable for camera localization. Motivated by these, CamNet [46] applies a two-stage retrieval, image-based coarse retrieval and pose-based fine retrieval, to select the most similar reference images for the finally precise relative pose estimation. We take this approach to construct loop closure constraints by first finding similar images in the sequence and then extracting relative poses between detected loop pair.

4) SLAM: Recently, researchers have started to combine existing works on odometry, relocalization, and loop closure detection in a complete SLAM system. DeepSLAM [47], for example, combines self-supervised deep learning based monocular visual odometry with pose graph optimization. The system consists of three main modules in which each of them deals with odometry (Tracking-Net), mapping (Mapping-Net), and loop closure detection (Loop-Net). Despite their great performances in public visual odometry benchmark, there is no uncertainty estimation in the odometry and loop closure constraints, making it less flexible to balancing the constraints or inspecting failure modes. DeepFactors [48] is another example which tries to combine deep learning and factor graph optimization. The system was trained to learn a compact depth map representation for dense visual SLAM system. However, they only demonstrate the system in a small indoor environment (e.g. ScanNet dataset). Finally, despite some emerging works on combining model-based and deep learning-based approaches, none of them address thermal camera system.

C. Thermal Odometry and SLAM

Realizing odometry and SLAM estimation using thermal imaging system remains a challenging problem due to the nature of thermal cameras which capture the heat distribution from the observed environment instead of the appearance and geometry. Nevertheless, some efforts have been made to construct thermal odometry, although it has been used for relatively short distances or yields sub-optimal performance compared to RGB-based odometry. Mouats et al. [10] utilized a Fast-Hessian feature extractor to estimate stereo thermal odometry for UAV tracking. Borges and Vidas [49] designed a practical thermal odometry by employing an automatic procedure to determine the correct time to perform the NUC operation. Nevertheless, the system can only work in outdoor scenario as it requires road lane estimation to compute the scale of the prediction.

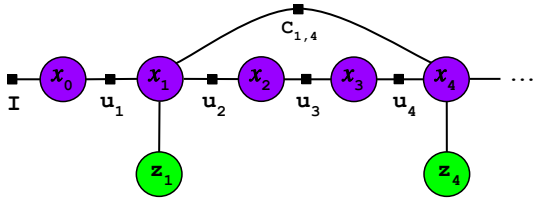


Fig. 1. Factor graph representation of TI-SLAM.

Recent work on thermal odometry typically fused together thermal imaging systems with other modalities. Delaune et al. [11] combined thermal and inertial sensors for UAV tracking by using an EKF algorithm. They showed that by employing FAST and KLT tracker, the thermal-inertial odometry can work well during day and night. Similarly, Khattak et al. [7], [50] also construct thermal-inertial odometry for UAV tracking by using keyframe-based direct approach which minimizes radiometric error between two adjacent frames. They used raw radiometric data instead of the normalized grayscale data to avoid difficult data association as the scene dynamically changes based on the environment temperature.

Despite some work on thermal-inertial odometry estimation, to the best of our knowledge, there is no published work on thermal-inertial SLAM to date. Vidas and Sridharan [51] realized a hand-held thermal SLAM by employing FAST-based feature tracking in the front end and bundle adjustment-based optimization in the back end. However, despite the claim of being a SLAM system, it is not a full SLAM system in a sense that there is no loop closure module which is used in state-of-the-art SLAM frameworks to generate a consistent trajectory and map (e.g. ORB-SLAM [3], LSD-SLAM [52]). Moreover, without an environment agnostic sensor like IMU, it is difficult to achieve robust estimation in an arbitrary environment. Shin and Kim [53] recently proposed feature-based lidar-thermal SLAM. They enhanced thermal data with sparse range measurement from lidar to improve the scale estimation of the system. However, they demonstrated their system for operation in an autonomous car which typically has more thermal gradients than in indoor scenario (e.g. corridor with planar walls). Furthermore, all these works utilize a hand-engineered feature extractor which may lose track in environments with limited thermal gradients.

III. SLAM Problem Formulation

A. Maximum a Posteriori (MAP) with Probabilistic Neural Networks

It is widely known in the robotic community that we can solve the SLAM problem using a graph-based formulation. In this formulation, the SLAM estimation problem is simplified by abstracting the raw sensor measurements into edges in the graph [54]. To solve a graph-based SLAM problem, a Maximum a Posteriori (MAP) approach is typically employed. Let $\mathcal{X} = \{x_i : i = 1, ..., m\}$ be an unknown variable (e.g. trajectory of the agent as discrete poses) that we want to estimate. Given a set of sensor measurements $Z = \{z_i : i = 1, ..., m\}$ such that z_i can be expressed as $z_i = h_i(x_i) + \epsilon_i$ where $h_i(.)$ and ϵ_i are measurement model and measurement noise respectively, we can compute $\mathcal{X}$ by estimating the assignment variables of $\mathcal{X}^*$ that yields the maximum of the posterior $\mathsf{p}(\mathcal{X}|Z)$ as in the following equation [1] when the probability of each measurement is the same

$$\mathcal{X}^* \doteq \underset{\mathcal{X}}{\operatorname{argmax}}\, \mathsf{p}(\mathcal{X}|Z) = \underset{\mathcal{X}}{\operatorname{argmax}}\, \mathsf{p}(Z|\mathcal{X})\mathsf{p}(\mathcal{X}). \tag{1}$$

Note that the equality in Eq. (1) follows the rule in the Bayes theorem. Eq. (1) can then be factorized into the following form by assuming that the measurement Z are independent

$$\mathcal{X}^* = \underset{\mathcal{X}}{\operatorname{argmax}}\, \mathsf{p}(\mathcal{X}) \prod_{i=1}^{m} \mathsf{p}(z_i|\mathcal{X}), \tag{2}$$

while both $\mathsf{p}(\mathcal{X})$ and $\mathsf{p}(z_i|\mathcal{X})$ are the *factors* in the factor graph representation which encodes the probabilistic constraints among the nodes. In order to make Eq. (2) more explicit, we can assume that the measurement follows a Gaussian distribution with a zero-mean ϵ and information matrix Ω (the inverse of covariance matrix). Then, the likelihood function $\mathsf{p}(z_i|\mathcal{X})$ will have the following form

$$\mathsf{p}(z_i|\mathcal{X}) \propto \exp\left(-\frac{1}{2} \left\| h_i(x_i) - z_i \right\|^2_{\Omega_i}\right), \tag{3}$$

where $\|e\|^2_\Omega = e^\intercal \Omega e$. Since maximizing the posterior is essentially the same as minimizing the negative log likelihood, then the MAP estimation in Eq. (2) becomes

$$\mathcal{X}^* = \underset{\mathcal{X}}{\operatorname{argmin}} -\log\left(\mathsf{p}(\mathcal{X}) \prod_{i=1}^{m} \mathsf{p}(z_i|\mathcal{X})\right) \tag{4}$$

$$= \underset{\mathcal{X}}{\operatorname{argmin}} \sum_{i=1}^{m} \left\| h_i(X_i) - z_i \right\|^2_{\Omega_i}, \tag{5}$$

where $h_i(.)$ represents a non-linear function. Eq. (5) is widely known as a non-linear least square optimization problem and can be solved by using Gauss-Newton or Levenberg-Marquardt algorithm. Note that we omit $\mathsf{p}(\mathcal{X})$ in Eq. (5) since it is usually uninformative (e.g. modeled as uniform distribution) or does not contribute in determining the optimized value.

To minimize Eq. (5), in our formulation, we encode $h_i(.)$ with a deep, probabilistic neural network that estimates both mean and covariance. Then, from the perspective of the MAP estimator, our approach can be viewed as replacing the likelihood estimation with an abstraction from a deep neural network. From the optimization perspective, our deep network can also be viewed as the initial guess for the optimization. In this sense, we could combine the well-established formulation of SLAM problem with recent advances in DNNs as a strong non-linear function estimator to model a better abstraction of the sensor measurement.

To model the likelihood estimation with a deep neural network, we employ a Mixture Density Network (MDN) [55] which has been shown to work well for the camera (RGB) re-localization problem [16]. MDN allows the network to estimate a multi-modal posterior distribution which maps well to our problem of estimating a SLAM posterior from multi-modal sensor data. In MDN, the output is composed of a Gaussian Mixture Model (GMM) and the networks predict the GMM parameters mean μ_k and variance σ_k where $k = 1, ..., K$ are indices of each Gaussian component $\mathcal{N}(\mu_k, \sigma_k^2)$. Then, given sensor measurement Z, the posterior $\mathsf{p}(\mathcal{X}|Z)$ becomes

$$\mathsf{p}(\mathcal{X}|Z) = \sum_{k=1}^{K} \alpha_k(Z)\mathcal{N}(\mathcal{X}|\mu_k(Z), \sigma_k^2(Z)), \tag{6}$$

where α_k are mixing coefficients constrained by $\sum_{k=1} \alpha_k = 1$ which is typically achieved by using a softmax function and learnt during training. Note that for training, we minimize the negative log-likehood of Eq. (6) such that $\mathcal{X}^* = \operatorname{argmin}_\mathcal{X} -\log\left(\mathsf{p}(\mathcal{X}|Z)\right)$. As we only estimate the variance instead of the full covariance, we assume that the output prediction (6-DoF poses) is independent of each other. This assumption has been used in [29], [56] as well.

B. SLAM Optimization Objectives

Eq. (5) can be interpreted as a general optimization objective for graph-based SLAM. In our implementation, we use a variant of this version which is called *pose-graph* SLAM. In pose-graph SLAM, the variables to be inferred are the agent poses (positions and orientations) and each factor in the factor graph imposes a constraint between two poses (e.g. relative estimate between a pair of poses). In our SLAM problem, we define two factors, i.e. an *odometry* factor and a *loop closure* factor, both of which are inferred from a DNN. Fig. 1 represents the factor graph representation of our problem. The odometry factor u_i imposes constraints between consecutive positions x_i and x_{i+1}, while the loop closure factor imposes constraints between two distant poses that have a large portion of image or feature correspondence (e.g. x_1 and x_4), but not necessarily obtained at the exact same location. These definitions refers those used by feature-based visual SLAM in a sense that as long as we have sufficient correspondence, although not exactly at the same location, we can estimate the relative pose between those two locations and use it as an additional constraint. These loop closure pairs are detected via observing similar measurements (e.g. z_1 and z_4) and also encoded with a further neural network. Then, given the odometry and the loop closure factors, our SLAM optimization

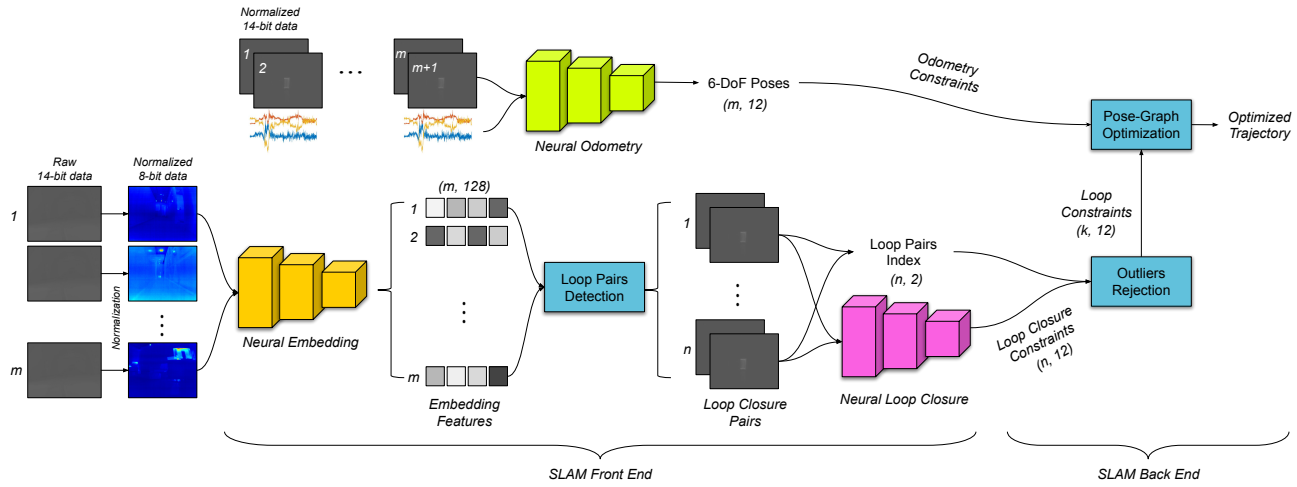


Fig. 2. The high-level architecture of TI-SLAM. The front end is abstracted by probabilistic neural networks while the back end employs robust second order-based graph optimization.

objective is described as follows

$$\mathcal{X}^* = \underset{x}{\operatorname{argmin}} \sum_i \underbrace{\|h_u(x_i, u_i) - x_{i+1}\|^2_{\hat{\Sigma}_i}}_{\text{odometry constraints}} + \sum_{<i,j>} \underbrace{\|h_c(x_i, c_{ij}) - x_j\|^2_{\hat{\Lambda}_{ij}}}_{\text{loop closure constraints}}, \tag{7}$$

where $h_u(x_i, u_i)$ represents the estimated agent position at $i+1$ after composing the previous position x_i with odometry estimation u_i from deep neural network. Note that we use the covariance matrix $\hat{\Sigma}_i = \varrho\Sigma_i$ to characterize the uncertainty of the odometry estimates where ϱ is a scale factor. $h_c(x_i, c_{ij})$ models the estimated position of the corresponding loop pair x_j after composing x_i with the relative poses between x_i and x_j $(c_i j)$ with $\hat{\Lambda}_{ij} = \rho\Lambda_{ij}$ as the covariance and ρ as the scale factor. Note that we use ϱ and ρ to balance the contribution of the covariance in odometry and loop closure constraints during optimization.

IV. Overview of The Thermal-Inertial SLAM System

Fig. 2 depicts the high-level architecture of our thermal-inertial SLAM system. As can be seen, the SLAM front end consists of three neural network branches to generate odometry and loop closure constraints. Odometry constraints (6-DoF relative camera poses and its variances) are estimated by a *neural odometry* network given consecutive normalized 14-bit thermal images and IMU sequences. To generate loop closure constraints, we first extract an embedding feature for each normalized 8-bit thermal image via a *neural embedding* network. These embedding vectors summarize the salient features that best describes a thermal image. By comparing these embedding features against all other embeddings, we can detect an image pair with sufficient correspondence and identify it as a potential loop pair. Lastly, the relative poses between these loop pairs are estimated by a *neural loop closure* network - these are then regarded as the loop closure constraints. Note that as we do not have IMU data to generate robust loop closure constraints in the SLAM back end, we perform outlier rejection to discard noisy loop closure pose estimations and only keep the inliers. Finally, given both odometry and (inlier) loop closure constraints, the back end optimizer will optimize the entire pose-graph using Eq. (7) to generate an optimized trajectory.

V. SLAM Front End

The SLAM front end is responsible for constructing the pose graph by abstracting the input from the thermal and inertial data using an MDN. In this section, we will detail the network structure and the training procedure for each neural network.

A. Neural Thermal-Inertial Odometry

1) Network Architecture: Fig. 3 depicts the architecture of our neural thermal-inertial odometry subsystem. This consists of three main parts, namely the feature extractor, the feature selector, and the pose regressor. The first part is the feature extractor, which is designed to distill geometrically meaningful features for odometry estimation. For images, this is typically implemented as optical flow estimation which captures the movement of pixels e.g. from edges of an object. However, since thermal images are inherently lacking in sufficient features to estimate dense optical flow (e.g. they are textureless), we follow the practice of [19] which not only extract features from the thermal images, but also hallucinates visual features, simulating the ones extracted from a DNN-based visual odometry [12], [13]. Given thermal images as the input, the hallucinated visual features will act as auxiliary information for the thermal features such that an accurate odometry can be inferred from textureless thermal sequences.

The second part is the feature selection module, which aims to select the most useful feature combination for odometry estimation. To this end, we follow the deterministic soft fusion structure in [20] to construct our feature selection. This feature selection is necessary because not all features are equally important at all times for accurate odometry estimation, e.g. each sensor comes with intrinsic noise. In particular, thermal data are plagued by fixed-pattern noise [57], while white random noise and sensor bias affect IMU data. Given these

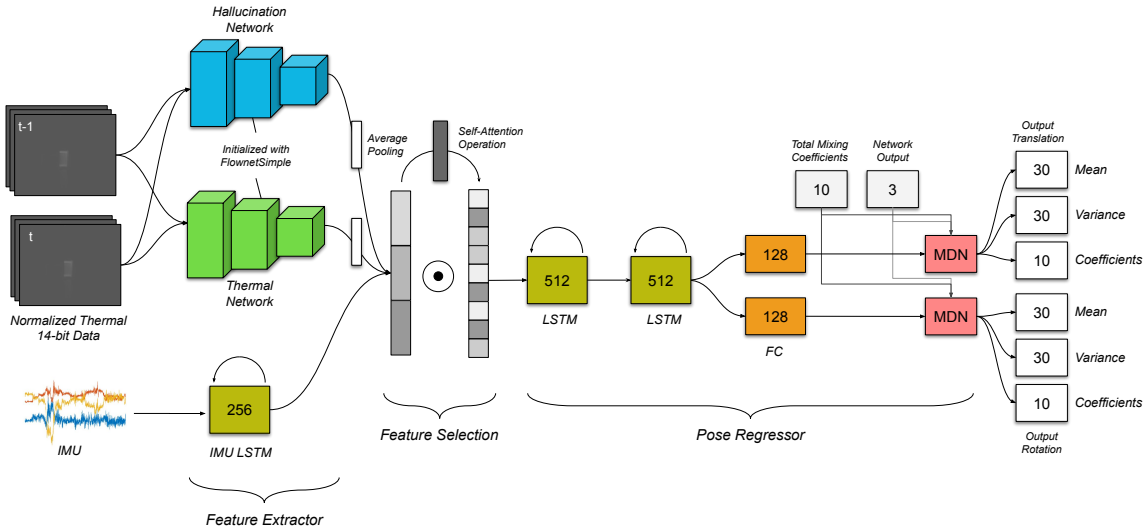


Fig. 3. The network structure of neural thermal-inertial odometry which consists of a feature extractor, feature selection, and pose regressor. The network is trained to estimate the parameters of Mixture Density Network (MDN) given normalized radiometric thermal data (14-bit) as the input. To extract the 6-DoF poses during testing, we need to sample from the mixture models.

noisy features, the feature selection module will generate masks to re-weigh each feature by conditioning these over all input channels. The concatenated, re-weighted features, are then fed to the pose regressor network to estimate the parameters of the MDN which will be described in detail in the following section.

2) Probabilistic Pose Regressor: Instead of directly predicting 6-DoF agent poses as in the previous work [12], [13], the probabilistic pose regressor estimates the parameters of MDN: mean (μ), variance (σ), and the mixing coefficients (α) as seen in Eq. (6). To this end, we construct our pose regressor by stacking two LSTM layers followed by Fully Connected (FC) layers that decouple MDN parameters for translation and rotation. We employed LSTM since odometry is considered as a sequential motion estimation problem, in which implicitly modelling the temporal dependencies within the network is important. The two stacked LSTM layers will encapsulate the dependencies between the current and the previous frame in the latent states as described in [12], [29]. For this purpose, we keep a one history of the previous hidden state in the LSTM although longer history is possible (yet it requires more computational time).

To derive both the camera poses (6-DoF) and its uncertainty estimation (covariance matrix) through MDN, we model each component of the poses (e.g. translation in x direction) as a mixture of Gaussian. The number of mixing coefficients (K) stated in Eq. (6) are determined empirically and will be discussed in Section VII-B4. The selection of K also determines the total paramaters of the MDN layer which are typically estimated via FC layers with $(3 * K)$ hidden units for the mean and the variances and (K) hidden units for the mixing coefficients. At test time, we can extract the 6-DoF poses together with the variances by sampling from the Gaussian mixture models.

3) Learning Mechanism: The neural odometry network is trained in two stages. In the first stage, we train the hallucination network and in the second stage, we train the remaining networks. As the visual hallucination network Ψ_H is intended to imitate the visual features $\mathbf{a}_V$ from real RGB images encoded by a visual encoder Ψ_V, we employ a deep Visual-Inertial Odometry (VIO) model as a pseudo ground truth to train Ψ_H. In particular, we use a VINet architecture [38] to generate $\mathbf{a}_V$ such that it can be used to train Ψ_H. Following [19], we employ the Huber loss [58] to train Ψ_H to avoid the catastrophic impact of outliers due to periodic NUC operation in thermal camera. Then, by trying to minimize the discrepancy ξ between the output activation from Ψ_H and Ψ_V, our objective function $\mathcal{L}_{\mathrm{H}}$ is defined as

$$\begin{aligned} \mathcal{L}_{\mathrm{H}} &= \frac{1}{n}\sum_{i=1}^{n}\mathcal{H}_i(\xi) \\ \mathcal{H}(\xi) &= \begin{cases} \frac{1}{2}\left\|\xi\right\|^2 & \text{for } \left\|\xi\right\| \leq \delta, \\ \delta(\left\|\xi\right\| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \\ \xi &= \Psi_H(\mathbf{X}_T;\mathbf{W}_H) - \Psi_V(\mathbf{X}_V;\mathbf{W}_V), \end{aligned} \tag{8}$$

where δ is a threshold, n is the batch size during training, $\mathbf{W}_V$ and $\mathbf{W}_H$ are the weights for Ψ_V and Ψ_H respectively. Note that during this training process, we have previously trained VINet and freeze its weights.

In the second stage, we train the remaining part of the networks to estimate the parameter of MDN. As this is a supervised learning, we will provide the ground truth poses for the training. Then, the objective function is defined as follows

$$\mathcal{L}_{MDN} = \sum_{i=1}^{n} \mathsf{p}(\mathbf{t}|Z)_i^- + \beta \sum_{i=1}^{n} \mathsf{p}(\mathbf{r}|Z)_i^-, \tag{9}$$

where $\mathsf{p}(\mathbf{t}|Z)_i^-$ and $\mathsf{p}(\mathbf{r}|Z)_i^-$ are the negative log likelihood of Eq. (6) for each translation and rotation component respectively, with $\mathbf{t} \in \mathbb{R}^3$ and $\mathbf{r} \in \mathbb{R}^3$ are predicted translation and rotation. Note that we use Euler angle to represent rotation $\mathbf{r}$ as it is free from constraints and easier to converge as described by [12]. Note that the odometry motion is usually also constrained (e.g. the ground robot only perform rotation in yaw axis) which makes the usage of Euler angle safe from

gimbal lock problem. We use β to balance the loss between translation and rotation component as seen in [12], [59]. Note that in this stage, we freeze the hallucination network Ψ_H to avoid altering the learnt hallucination weights that have been trained in the first stage.

B. Neural Embedding and Loop Closure Detection

In the context of SLAM, the aim of loop closure detection is to identify whether the mobile agent has revisited a place. This information can then be used to constrain the odometry estimation and optimize the overall pose graph.

Following the taxonomy described in [60], loop closure detection can be achieved through local or global image descriptors. Global descriptors represents an image in a holistic manner without the need to extract local features like SIFT [61] or SURF [62]. Typical example includes representing the image as a colour intensity histogram as described in [63] or other image statistics described in GIST [64]. On the other hand, local descriptor-based approaches extract keypoints (e.g. corners, blobs, or regions) and their corresponding descriptor vectors in which the measurements are typically taken from the vicinity of each keypoint. Then, aggregation methods such as those based on Bag-of-visual-words (BoW) [65], Vector of Locally Aggregated Descriptors (VLAD) [66], MAC [67], or NetVLAD [35] can be used to summarize the descriptors.

Our neural embedding model follows the global descriptor approach as we do not rely on local features extracted from hand-engineered keypoints or aggregation methods, but directly generate a global descriptor from a thermal image through a neural network. We employ a global descriptor based approach due to the textureless nature of thermal images, in which two different features from an equivalent RGB image might be merged in a thermal image as they may have the same temperature. Thus, instead of focusing on clustering these ambiguous features (as been done by BoW or NetVLAD), we instead rely on global image information extracted by the deep network to improve the generalization.

1) Network Structure: Fig. 4 depicts the structure of our neural embedding network. The network consists of Truncated ResNet, followed by global average pooling and fully connected layers. For the Truncated ResNet, we use ResNet50 structure [69] up to the 49th layer to remove the classification part of the network and obtain a large spatial output dimension. The global max pooling layer is then used on top of ResNet50 to filter the most important part of the output vectors. We project this output embedding to a lower dimensional space by applying 3 FC layers. The number of hidden states in FC layers follows this decreasing rule $\text{FC}_1(\gamma * 4)$, $\text{FC}_2(\gamma * 2)$, $\text{FC}_3(\gamma)$, where γ is the total output of the embedding vectors. In practice, we use $\gamma = 128$ to construct an efficient embedding vectors, following the practice in face recognition [70]. We presume that a small number of embedding vectors are sufficient to describe a thermal image since it has much less feature variation compared to the RGB images. Then, similar to the last layer in NetVLAD, we perform L2 normalization such that the entries of the embedding vectors will be sum to 1. To avoid training the entire network structure from scratch,

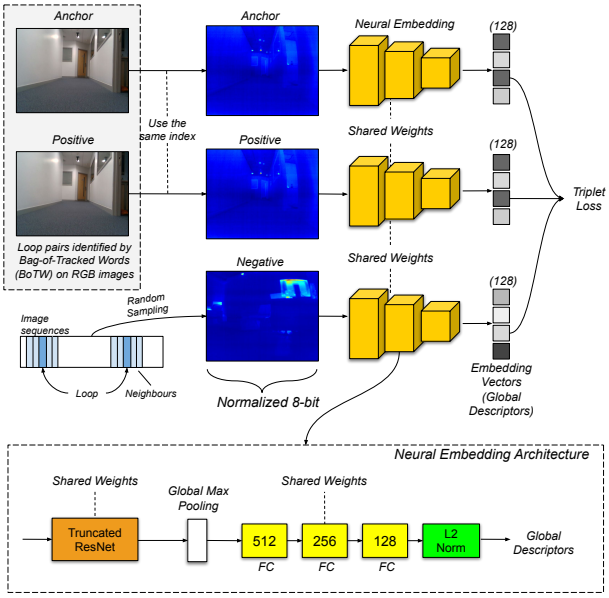


Fig. 4. The network structure of neural embedding during training, which generates 128-D global descriptor for a single thermal image. For training, we obtain the list of anchor and positive examples from BoTW [68] applied on simultaneously captured RGB images such that our network can emulate BoTW performances on RGB.

we initialize the ResNet50 weights from ImageNet model and fine-tune all layers when training the network using our thermal data.

2) Learning Procedure: To train the network, we follow the standard learning procedure to train a place recognition network based on triplet margin losses. A triplet $\{\mathbf{I}_T, \mathbf{I}_T^+, \mathbf{I}_T^-\}$ consists of an anchor image $\mathbf{I}_T$, a positive example $\mathbf{I}_T^+$, which represents a similar scene with sufficient correspondence with respect to the anchor (loop pair), and a negative example $\mathbf{I}_T^-$, which represents an unrelated scene with no or minimum image correspondences with respect to the anchor. Given the triplet information, our triplet loss is defined as

$$\mathcal{L}(\mathbf{I}_T, \mathbf{I}_T^+, \mathbf{I}_T^-) = \max(\lambda + \left\| d_{\mathbf{W}_T}(\mathbf{I}_T) - d_{\mathbf{W}_T}(\mathbf{I}_T^+) \right\|^2 - \left\| d_{\mathbf{W}_T}(\mathbf{I}_T) - d_{\mathbf{W}_T}(\mathbf{I}_T^-) \right\|^2, 0), \tag{10}$$

where $d_{\mathbf{W}_T}(.)$ is the neural embedding network, $d_{\mathbf{W}_T}(\mathbf{I}_T)$ is an embedding vector defining the global image descriptor of image $\mathbf{I}_T$, $\mathbf{W}_T$ is a shared trainable weights for the network, and λ is a hyper-parameter to control the margin between positive and negative examples. By training the neural embedding network using Eq. (10), the network is expected to produce similar embedding vectors when the mobile agent re-visits a place.

In order to provide the data for training the embedding network, we have to identify positive loop pairs amongst thermal image sequences. To avoid a manually laborious annotation task, we instead use the loop pair detected from a state-of-the-art place recognition algorithm applied on simultaneously captured RGB images as pseudo ground truth. In this sense, the embedding network can imitate how loop closures are formed from RGB correspondences, given thermal images as the input. Note that this is possible as both thermal and RGB cameras are placed in the same mobile agent with

sufficient spatial correspondence. Then, to detect loop pairs amongst RGB images, we employ the state-of-the-art Bag-of-Tracked-Words (BoTW) [68], an improved version of the standard Bag-of-Words (BoW) algorithm. The main difference between BoTW and Bow is that BoTW utilizes "Tracked Words" among successive images rather than the standard histogram-based visual words in a single image, yielding more robust recognition performance. Nevertheless, despite its great performances on RGB images, BoTW performs very poor on thermal images (see the results in Section VII). This is most likely because it relies heavily on point features, which are typically much scarcer on thermal images than in RGB images, especially when the images are captured in an uncluttered indoor environment. Hence, we use BoTW as our pseudo ground truth by applying it on RGB images rather than directly utilizing it to detect loops on thermal images. To improve the number of data during training, we interchange the anchor and the positive loop pair in a triplet and choose a random negative example that does not belong to the anchor and to the positive examples, nor is within adjacent frames with respect to the anchor and the positive examples (see illustration in Fig. 4). This was done with expectation that the network should produce a similar embedding vector for adjacent frames, although these will not be considered as a loop pair.

3) Loop Closure Detection: After correctly training the embedding network, we can generate the embedding vectors for each thermal image. To detect a loop pair, we compute the discrepancy between each pair of embedding vectors i and j using cosine distance as follows

$$\mathcal{S}_{ij} = \frac{d_{\mathbf{W}_T}(\mathbf{I}_T^i).d_{\mathbf{W}_T}(\mathbf{I}_T^j)}{\left\|d_{\mathbf{W}_T}(\mathbf{I}_T^i)\right\| \left\|d_{\mathbf{W}_T}(\mathbf{I}_T^j)\right\|}, \tag{11}$$

where $\|.\|$ is the magnitude of the embedding vectors. If $\mathcal{S}_{ij} < \zeta$, where ζ is a threshold, then the embedding pair is regarded as a loop closure pair. The selection of ζ is important as it determines the trade-off between the number of true positive and false positive pairs. In practice, we set $\zeta = 0.045$ to generate around 85% true positive loop pair although some false positive pair will also be detected (as depicted in ROC curve in Fig. 15). Nevertheless, in offline operation, ζ can be tuned individually for different sequences as necessary.

C. Neural Loop Closure

Given a loop pair, we need to extract the relative poses between them such that we can inject it into the SLAM back end as loop closure constraints. To this end, we construct another deep network, termed Neural Loop Closure, that can estimate relative poses (and uncertainty) using MDN only from a pair of thermal images. Fig. 5 depicts the architecture of the network which resembles the neural thermal-inertial odometry network. The main difference is that the input thermal pair does not come from consecutive images, but instead from a loop closure pair i and j. Note that we do not have IMU data between them, making the training process even more challenging.

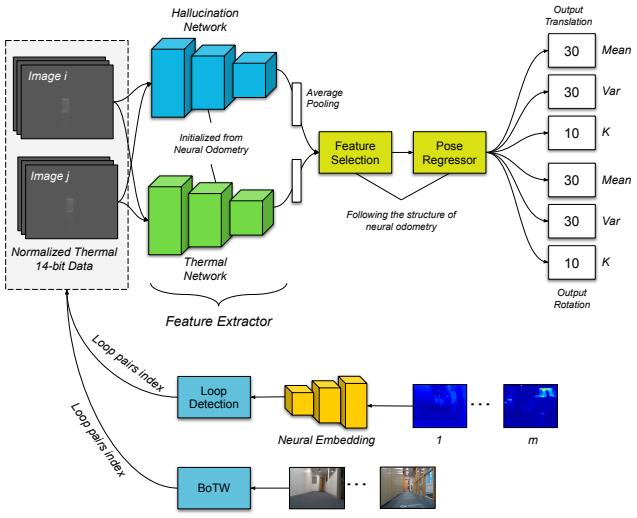


Fig. 5. The network structure of *neural loop closure* during training. This network is used to estimate the relative poses (and uncertainty) between a loop pair by using MDN. The structure resembles neural thermal-inertial odometry except that it does not have an IMU and the input from thermal images comes from frames i and j instead of successive frames. For training, the input loop pairs are obtained from both BoTW applied on RGB images and the neural embedding network applied on thermal images.

For training, we provide the network with the list of loop pairs together with the ground truth poses. We utilize the list of loop pair generated by both BoTW applied on RGB images and Neural Embedding network applied on thermal images to provide more data during training. We initialize both thermal and hallucination network with the weights from neural thermal-inertial odometry to ease the optimization process. For testing, we sample from the mixture models to obtain 6-DoF relative poses between loop pairs. Given both odometry and loop closure constraints, now we can finally construct a complete pose graph to be optimized by the SLAM back end.

VI. SLAM Back End

The SLAM back end is responsible for optimizing the whole trajectory given the pose graph constructed by the front end. However, the pose graph built by the front end is often subject to noise. For example, the odometry prior can be inaccurate and largely drift, or the loop closure constraints are erroneous as the poses are generated only from a pair of thermal images without the assistance of IMU. To mitigate such noise, we incorporate an outlier rejection module to filter and feed only the reliable ones to the subsequent graph-based optimization.

1) Outlier Rejection: The outlier rejection module extends GraphTinker [71] to our 6-DoF context, which is essentially based on the geometric consistency of loop closure constraints. Specifically, consider two loop closure proposals detected by the front end, if these two loop closure proposals are both true, then within any reference frame, the two trajectory segments defined by them will form a *sub-loop*. If the two segments are geometrically consistent (e.g., validated through the reverse odometry method as described in [71]), one can confirm that both loop closure proposals are true, and mark them as valid (ready to be used by the later graph optimization).

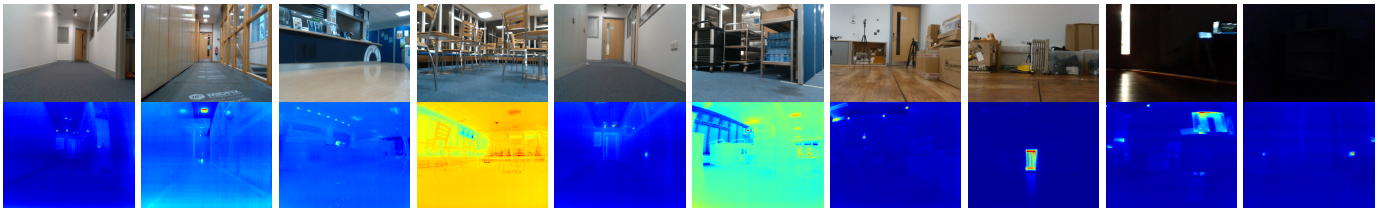


Fig. 6. Sample images from the in-house dataset. The top and bottom images are from RGB and thermal camera respectively. For clarity, we display the rescaled (normalized) radiometric data instead of the 14-bit raw radiometric data.

Fig. 7. Sample images from Subt-tunnel dataset [73].

Otherwise, at least one of the two loop closure proposals is false positive. In that case we will not push any of them to later optimization, but temporarily keep them in the proposal set for next validation. After all sampled loop closure proposals are examined, the closures with pass rate less than a threshold will be rejected as outliers. In practice, we typically set the threshold less than 1. However, the best setting are varied for different environments as it also highly depend on the accuracy of the individual loop closure constraints.

2) Optimization: Given the pose graph with odometry constraints and selected (filtered) loop closure constraints, the back end optimizer will minimize Eq. (7) using Levenberg-Marquardt algorithm (i.e., g^{2}o [72]). To generate optimal solution, the balancing coefficients between odometry and loop closure constraints have to be selected with care. From our experiments, the weight for loop closure constraints is usually set higher than the weight for odometry constraints. For online operation, we typically set $\varrho = 0.01$ and $\rho = 3$. However, the optimized solution might be obtained by setting different ϱ and ρ for different sequences and environment. This might be the right choice if online operation is not required, e.g. the pose graph optimization can be done in offline fashion as in the Structure from Motion (SfM) works.

VII. Experiments

A. Dataset

We use multiple datasets to test our model, including self-collected and public datasets. For the self-collected dataset, we conducted experiments on ground robot and handheld data in indoor environment with around 8 km total trajectory length. For the public dataset, we used SubT-tunnel dataset [73]. Each dataset is described as follows.

1) Indoor Ground Robot Data: We collected our data with Turtlebot 2. We use a Flir Boson 640 thermal camera to capture raw radiometric (14-bit) thermal data at 30 fps with 640×512 resolution. For the IMU, we utilize XSens MTI-1 Series running at 100 Hz. We also equip the robot with a Velodyne HDL-32E Lidar running at 10 Hz and an Intel Real Sense Depth camera with 680×480 RGB resolution (rolling shutter) running at 30 fps. These RGB data are used for training the hallucination network and assisting loop closure detection during training. Note that there are at least 2/3 spatial correspondences between both RGB and thermal images, enabling the training of hallucination network. In total, we have 36 sequences collected in different type of environments (e.g. hall, canteen, office, corridor, etc.), in which we use 23 sequences for training and 13 sequences for testing. For odometry training, we employ inertial-assisted wheel odometry provided by the Turtlebot 2 as the pseudo ground truth. For evaluation, following the practice in [19], we utilize VICON Motion Capture system and Lidar Gmapping[2] to generate the ground truth poses. Lidar is particularly used when we do not have VICON as we collected some of our data in public space. To examine system robustness against different visibility conditions, we collect the data with sufficient illumination (bright), dim, or in darkness. Fig. 6 shows the example images for both RGB and thermal cameras.

2) Indoor Handheld Data: For the handheld data, we built a 3D printed model that utilizes the same set of sensors as with the indoor ground robot data. The only difference is, instead of using Velodyne HDL-32E Lidar, we replaced it with a more lightweight Velodyne Ultra Puck. We collected data in ten distinct floors from 6 different multistorey buildings including hallways, canteen, common room, building junction, atrium, office, and cluttered store rooms. The smallest floor has an area around $205m^2$ while the largest one reaches $1500m^2$. For quantitative evaluation, we used lidar SLAM generated by ALOAM [3] as the (pseudo) ground truth. To test the system robustness in adverse lighting condition, we also collected data in a smoke-filled environment firefighter training facility, which is located at Washington DC Fire and EMS Training Academy, United States.

3) SubT-tunnel Data: SubT-tunnel dataset [73] is a public dataset collected by a participant of DARPA subterranean challenge from CCDC Army Research Laboratory (ARL). The dataset contains synchronized lidar, RGB (stereo), depth, thermal, and IMU, taken from a ground robot moving in a long trajectory in an underground tunnel. The dataset is divided into 2 categories, namely urban circuit and tunnel circuit dataset. Sample images can be seen in Fig. 7. For this experiment, we only used the tunnel circuit as this data contains usable 14-bit thermal data (10 Hz) captured from Flir Boson camera. In particular, we utilized `ex_B_route1` sequence (54 minutes) for testing while the remaining are used for fine-tuning (re-training with a lower learning rate). The (pseudo) ground truth for this experiment was provided by Lidar OmniMapper [74].

[2]https://openslam-org.github.io/gmapping.html
[3]https://github.com/HKUST-Aerial-Robotics/A-LOAM

TABLE I
RMS RELATIVE POSE ERRORS (RPE) IN INDOOR GROUND ROBOT DATA

Seq	Lighting	Length (m)	VINet [38] (Thermal 14-bit) **t** (m)	**r** (°)	TI odometry 8-bit (**ours**) **t** (m)	**r** (°)	TI odometry w/o hallu. (**ours**) **t** (m)	**r** (°)	TI odometry 14-bit (**ours**) **t** (m)	**r** (°)	VINet [38] (RGB) **t** (m)	**r** (°)	VINS-Mono (RGB) **t** (m)	**r** (°)	Inertial+Wheel **t** (m)	**r** (°)
32	Bright	31.4	0.049	2.543	0.047	2.496	0.048	2.514	**0.039**	**2.431**	2.613	0.044	-	-	0.029	2.426
33	Dim	22.5	0.027	1.440	0.038	1.366	**0.019**	1.218	**0.019**	**1.208**	1.341	0.029	-	-	0.017	1.294
34	Dim	20.7	0.023	1.406	0.044	1.358	0.022	1.344	**0.021**	**1.263**	1.653	0.028	-	-	0.017	1.531
37	Dark	71.1	0.031	1.374	0.042	1.284	0.023	1.306	**0.021**	**1.233**	1.809	0.033	-	-	0.017	1.386
39	Bright	16.8	0.029	1.696	0.044	1.586	0.029	1.645	**0.028**	**1.579**	1.679	0.034	-	-	0.022	1.550
42	Dim	65.1	0.029	1.714	0.049	1.755	**0.028**	1.618	**0.028**	**1.570**	1.773	0.035	0.144	0.816	0.026	1.676
43	Dim	66.5	0.029	1.713	0.039	1.727	0.027	1.614	**0.025**	**1.571**	1.670	0.033	0.166	0.799	0.023	1.651
44	Dim	71.1	0.035	1.727	0.048	1.685	0.031	1.763	**0.029**	**1.676**	1.839	0.038	0.190	0.859	0.026	1.758
45	Bright	61.3	0.034	1.674	0.046	1.633	0.030	1.655	**0.027**	**1.557**	1.870	0.037	0.173	0.848	0.024	1.641
46	Bright	21.7	0.032	1.517	0.052	1.516	**0.028**	1.430	**0.028**	**1.353**	1.518	0.038	0.150	0.695	0.038	1.452
47	Bright	22.2	0.033	1.779	0.045	1.753	0.028	1.644	**0.027**	**1.577**	1.756	0.035	0.173	0.859	0.018	1.808
48	Bright	42.1	0.032	1.557	0.047	1.605	**0.025**	1.525	**0.025**	**1.462**	1.655	0.036	0.205	0.787	0.022	1.623
49	Bright	81.1	0.025	0.766	0.043	0.705	**0.019**	0.695	0.022	**0.679**	0.737	0.032	0.166	0.474	0.025	0.745
Mean			0.034	1.608	0.044	1.575	0.028	1.536	**0.026**	**1.473**	0.031	1.686	0.171	0.767	0.023	1.580

*The bold indicates the most accurate method among algorithms with thermal and inertial data as the input.

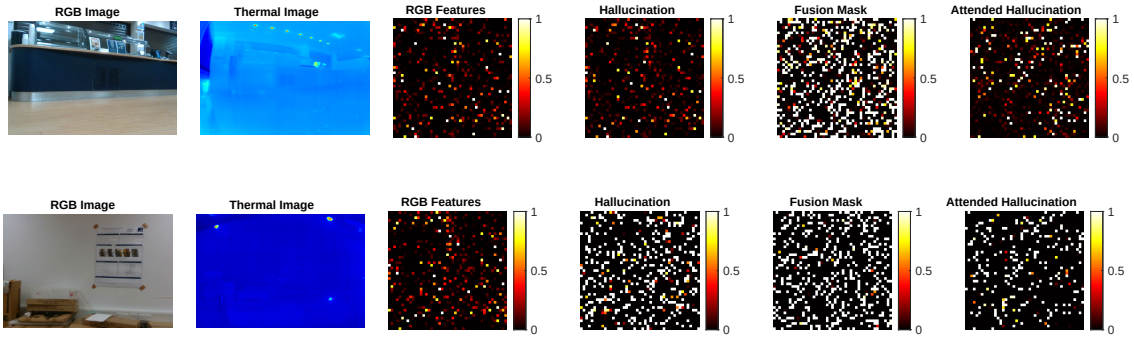


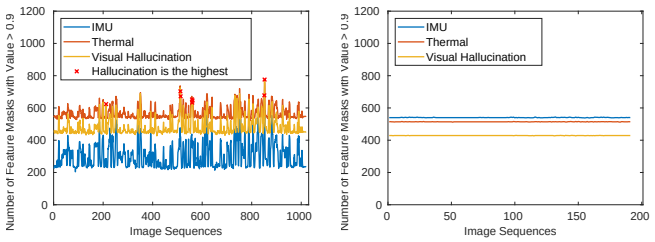
Fig. 8. Comparison between RGB features produced by VINet and fake RGB features produced by the hallucination network. **Top**: example of accurate hallucination, showing generalization in the test data. **Bottom**: example of erroneous hallucination due to limited thermal features. In this scenario, selective fusion produces less dense fusion mask, relying more on other modality like IMU. From left to right: RGB image, thermal image, original RGB features, hallucinated RGB features, fusion mask for the hallucination features, and the output attended hallucination features.

B. Odometry Evaluation in Ground Robot Data

Odometry constraints is an important factor to yield accurate thermal-inertial SLAM because it is utilized as the initial estimation in the pose graph optimization. In this section, we study the influence of thermal representation, measure the accuracy and the timing of the odometry factor, and validate the estimated variances. To measure the quality of the odometry estimation, we measure the Root Mean Square (RMS) of Relative Pose Errors (RPE) and Absolute Trajectory Errors (ATE) against ground truth provided by VICON and Lidar Gmapping.

1) The Influence of Thermal Representation: In this section, we investigate the choice of the thermal input by comparing the normalized 8-bit and 14-bit representation. As one can see in Table I, the 14-bit TI odometry produces more accurate results than the 8-bit TI odometry for all sequences. Hypothetically this is because our odometry network is devised based on the optical flow network (i.e. FlowNet [75]) which extracts pixel displacement features rather than image appearance features. By using the 14-bit representation, it is easier to retain similar gradient information between consecutive frames such that consistent pixel displacement distribution in a short period of time can be extracted, even when frames sub-sampling is performed. On the other hand, the re-scaling process in the 8-bit representation might (slightly) alter the weak gradient information, making it more difficult to extract consistent pixel displacement distribution for the network to learn. Nevertheless, the accuracy differences between 8-bit and 14-bit are marginal, indicating that the 8-bit representation is also usable for deep odometry estimation.

2) The Importance of Hallucination and Selective Fusion: To understand the importance of hallucination network and the selective fusion mechanism, we plot the output feature representation, comparing RGB features, hallucination features, fusion mask, and attended hallucination features as seen in Fig. 8. Fig. 8 (top) shows the hallucinated RGB features generated from the test data in the canteen sequence (Seq 39), which is the only canteen sequence we have (no other indoor structure that replicate this environment). It shows that the hallucination network produces accurate fake RGB features in this new environment, showing the generalization capability. We believe that as long as there are enough thermal features as the starting information, the hallucination network can



(a) Seq 39 (b) Poster data

Fig. 9. Selective fusion mask for mobile robot data in (a) Seq 39 and (b) poster data. We plot the total number of masks for each feature modality with value higher than 0.9, indicating the importance of the features.

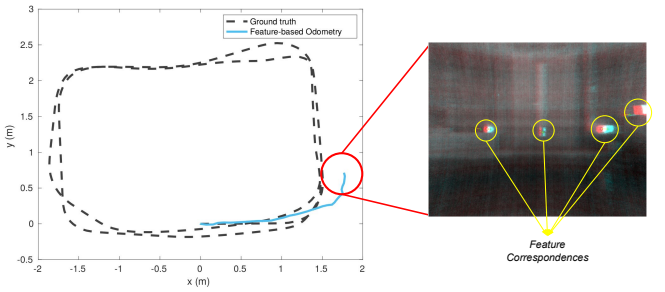


Fig. 10. Feature-based approach (SURF) using standard descriptors loses tracks on scenes with poor thermal gradients. In this example not enough correspondences are matched between consecutive images.

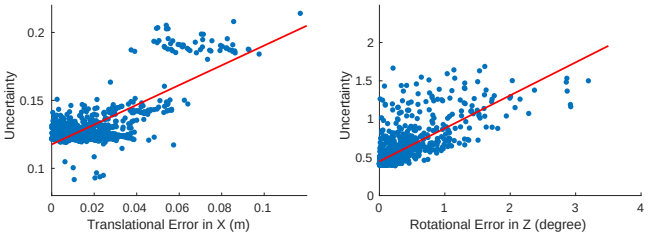


Fig. 11. Translational errors (in X axis) and rotational errors (w.r.t Z axis) against uncertainty (standard deviation).

perform meaningful hallucination by interpolating the RGB features. It can be seen as well from Fig. 9 (a) that in Seq 39, there are many cases when the selective fusion network used more hallucination features than other features (thermal and IMU), showing the importance of hallucination to improve the odometry accuracy. This is also supported by the quantitative results in terms of RPE as shown in Table I.

However, there is always a case that hallucination network produce erroneous results. The example is shown in Fig. 8 (bottom) when there is not enough thermal gradient information to hallucinate rich RGB features. This happens when the camera faces a poster that has a similar temperature with the wall. In this case, selective fusion module will rely on the other modalities, i.e. IMU, in order to produce meaningful odometry. As we can see from Fig. 9 (b), in poster data, selective fusion utilizes more IMU features than other modalities and places the hallucination as the least useful features.

3) Accuracy: To measure the quantitative performance of our odometry model, we report both RPE and ATE in Table I and Table IV respectively. We also compare our model with VINet [38] applied on RGB and thermal, VINS-Mono [76] applied on RGB, and IMU assisted wheel odometry. As shown in Table I, our neural odometry produces more accurate results compared to VINet. VINet applied on RGB falls short possibly due to the variation in lighting condition (e.g. dim and darkness), yielding sub-optimal performances. VINet applied on thermal generates less accurate estimation due to the difficulty of abstracting thermal data without the help from hallucination network. Our neural thermal-inertial odometry produces consistent results either from the perspective of RPE or ATE, and comparable to IMU assisted wheel odometry and VINS-Mono RGB (SLAM), showing the importance of hallucination network and feature selection in multi modal sensor fusion. Note that in scenarios with benign lighting, VINS-Mono typically yield more accurate rotation as it exploits the RGB images which have richer features for accurate rotation estimation. Nevertheless, VINS-Mono fails to initialize or loses tracks due to unstable frame rate, abrupt motion, and the presence of dynamic objects (people) in front of the RGB camera, particularly in Seq 32, 33, 34, 37 and 39. We also tried to run VINS-Mono with thermal camera (14-bit) but it lose tracks after couple of seconds to a minute due to abrupt motion and lack of features in the corridor. To investigate further the difficulty of tracking thermal features in our data, we tried to perform odometry estimation using strong feature matching algorithm like SURF [62] instead of using KLT tracker as demonstrated by VINS-Mono. The pose is then estimated by using five-point algorithm [77] and bundle adjustment based on Computer Vision Toolbox applied in Matlab. For this purpose, we normalized 14-bit images by re-scaling it into 256 intensity around the median of thermal value. As we can see from Fig. 10, SURF-based monocular thermal odometry loses tracks, specially following large changes in viewpoint and poor thermal gradients.

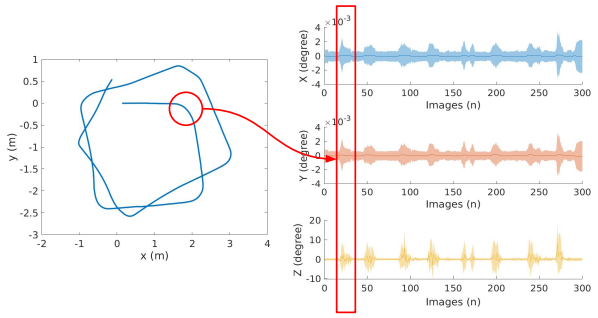
4) Probabilistic Estimates: To validate the proposed probabilistic approach, we first analyse the importance of modelling the odometry output through a mixture of Gaussian, followed by interpreting the output variances. Table II shows the influence of the number of Gaussian (indicated by the number mixing coefficients K) on accuracy. As it can be seen, by increasing the number of K, the accuracy also increases, showing that the network models the 6-DoF pose distribution more accurately by using a mixture of Gaussian instead of a single Gaussian ($K = 1$). Nevertheless, at some point, the accuracy saturates or even degrades.

Fig. 11 plots the uncertainty (variance) estimate against translational and rotational errors. We show the comparison for rotation in Z axis as the other axes (X and Y) mostly remain unchanged during operation. For translation, most changes happen in X and Y directions while translation in Z axis are almost zero since the robot moves in flat surface. The figure shows that the approximate uncertainty is correlated with the odometry error, validating our approach.

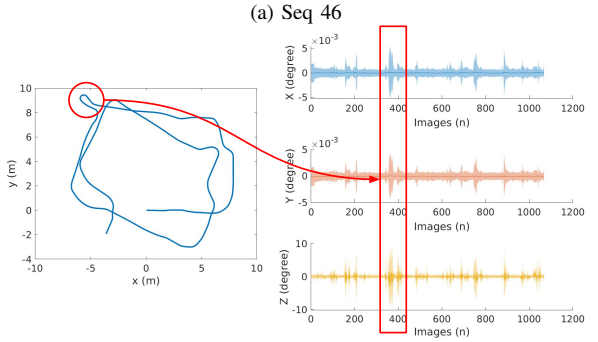
To have a better understanding in which condition the network produces larger uncertainty, we plot the rotational uncertainty in X, Y, and Z direction for Seq 46 and Seq 49 in Fig. 12. Following the practice in [29], for better visualization, we draw the rotational errors against 3σ variance interval. As one can see, the rotational errors are located within the variance intervals, verifying the meaningful of uncertainty

TABLE II
THE INFLUENCE OF THE NUMBER OF MIXING COEFFICIENTS (K) ON ACCURACY (ATE)

ATE	The Number of K				
	1	5	10	15	20
Mean (m)	0.793	0.725	**0.609**	1.051	0.862
Std (m)	0.449	0.435	**0.357**	0.441	0.606



(a) Seq 46



(b) Seq 49

Fig. 12. Relative pose errors from the neural odometry networks against 3σ variances interval.

estimation using MDN. In Seq 46, we can see that the network yields larger variance when the mobile agents perform large rotation (around 90 degree rotation in Z axis). In Seq 49, we can also observe that the largest uncertainty takes place when the robot performs U-turn, which typically generates the largest error in odometry estimation. This is interesting since this ability to approximate the variance is learnt during training without supervision on uncertainty. In this sense, we can validate that the uncertainty estimation can be used as a valid constraint for SLAM optimization.

5) Computation Cost: The neural odometry model was trained on an NVIDIA TITAN V GPU. It required approximately 6-18 hours for training the hallucination network and around 6-20 hours for training the remaining networks. The model contains 273 millions of parameters, requiring 547 MB of disk space. To generate a single prediction, the model requires approximately 0.5s in a standard CPU, which is typically slower than the real-time implementation of visual-inertial odometry (e.g., VINS-Mono [76]) which can produce the camera pose between 0.05-0.1s. Nevertheless, our model can be executed for up to 26 Hz (0.039s required for a single inference) in a powerful GPU like NVIDIA TITAN V.

C. Validating Loop Closure Detection and Loop Closure Constraints in Indoor Ground Robot Data

1) Qualitative and Quantitative Results for Loop Closure Detection: We perform qualitative validation of our loop closure detection by plotting the detected loop pair on top of the output odometry. For comparison, we also show the loop closure detection from state-of-the-art feature-based approach (i.e. BoTW [68]) and deep learning approach for place recognition (i.e. NetVLAD [35]). For NetVLAD, we generate the results from the pre-trained model from Pittsburgh dataset (RGB images) and also from the re-trained weights on our thermal images.

Before plotting the output loop closure pair, we inspect what the neural embedding network learn during training and compare it with NetVLAD, either applied on RGB (pre-trained) or applied on thermal (re-trained). Fig. 13 depicts the similarity matrix generated by measuring the cosine distance between embedding vectors on Seq 33. When we compare few images in the beginning of sequence with all other images in the whole sequence (red square area in Fig. 13), our neural embedding network identifies two areas with the highest similarity (excluding the adjacent frames). If we look at the ground truth trajectory, those two areas belong to the same place when the mobile agent re-visit the starting point. This indicates that our network can produce meaningful and distinct embedding vectors, identifying loop closure by clustering the same places into similar embedding. While NetVLAD can also identify two similar regions, they are less distinctive (fewer area with strong dissimilarity/yellow colored). We presume that this is because the performance of NetVLAD largely depends on the clustering algorithm. If the algorithm wrongly clusters local features due to the similar characteristic of particular thermal features (two different RGB patches might look similar on thermal as it lacks texture), it might classify two different scenes as an identical ones (with some degree of certainty).

Fig. 14 depicts the detected loop pair on the output odometry given by BoTW (on RGB and on thermal), NetVLAD (on thermal), and our loop closure detection. As it can be seen, BoTW clearly produces robust performance on RGB. However, it performs badly on thermal imagery, yielding a very small number of correctly detected loop. This emphasizes the difficulty of performing data association on thermal images due to the lack of robust features. Our model, as expected, can perform very well, detecting large number of positive loop pairs. Nevertheless, by carefully setting the threshold, NetVLAD (thermal) can also produce a similar number of positive loop pair. However, this is done with the cost of large space requirement and slow computation time as described in Section VII-C3.

For quantitative experiments, we compare our loop closure detection with respect to BoTW applied on RGB. We plot a Receiver Operating Characteristic (ROC) curve to measure the trade-off between sensitivity (true positive rate-TPR) and specificity (false positive rate-FPR) for every possible cut-off as it has been used by previous work on place recognition [78]–[80]. As it can be seen from Fig. 15, our model (nor-

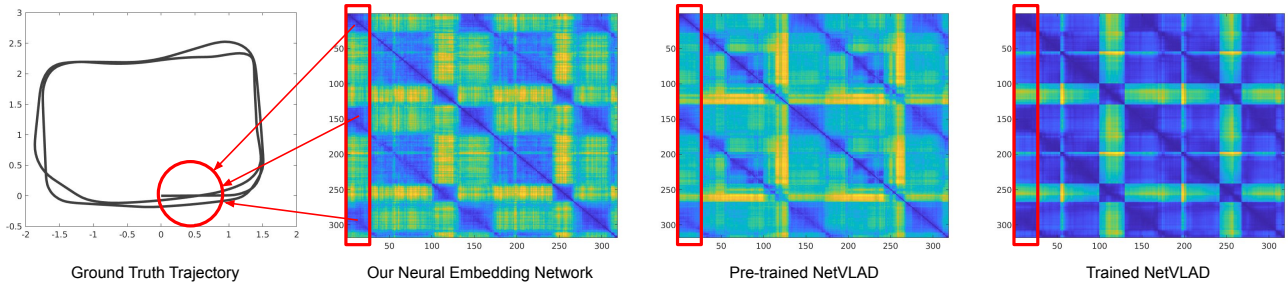
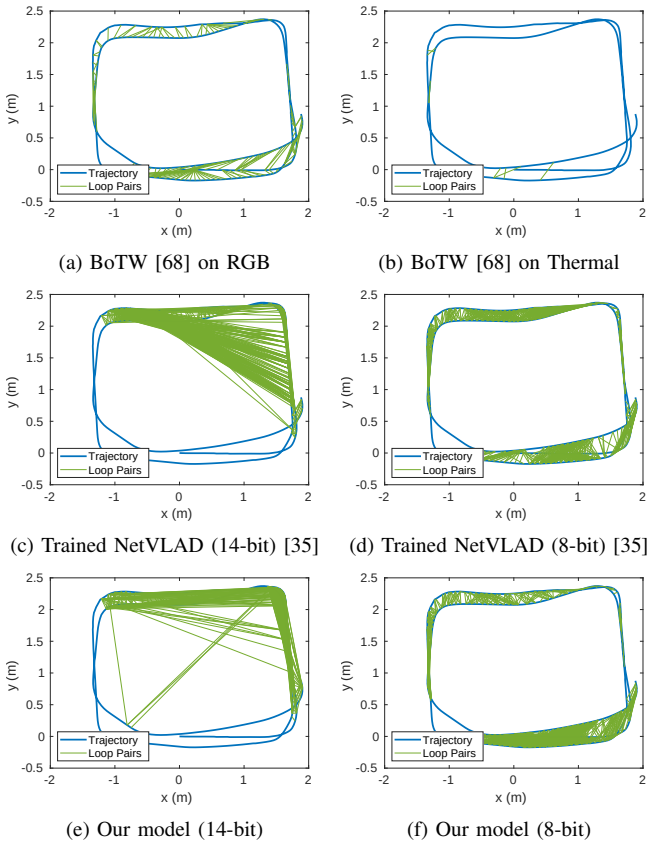


Fig. 13. Similarity matrix produced by our neural embedding network on Seq 2, compared to NetVLAD pre-trained on RGB and re-trained on thermal. Blue and yellow color indicates the most similar and the most dissimilar pair. Note that our model can produce distinct embedding to identify loop closure.



(a) BoTW [68] on RGB (b) BoTW [68] on Thermal

(c) Trained NetVLAD (14-bit) [35] (d) Trained NetVLAD (8-bit) [35]

(e) Our model (14-bit) (f) Our model (8-bit)

Fig. 14. Loop closure detection on Seq 33 from BoTW (applied on RGB and thermal imagery), trained NetVLAD (on thermal), and our neural embedding networks (either using 14-bit or 8-bit representation). Our model can produce similar performance to NetVLAD (8-bit) but with only 0.4% space requirement to store the embedding features.

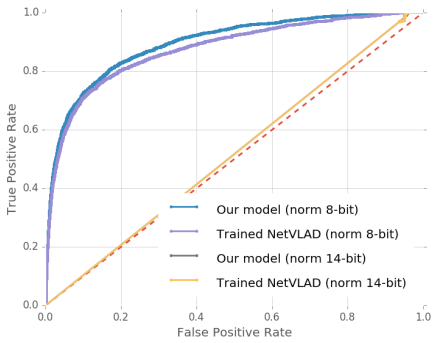


Fig. 15. ROC curves between Neural Embedding (our model) and trained NetVLAD applied on thermal data for all sequences in ground robot data.

malized 8-bit) produces slightly better performance than the NetVLAD (normalized 8-bit). Given 20% FPR, our model obtains around 82% TPR, showing a good trade-off between TPR and FPR.

2) The Impact of Thermal Representation for Loop Closure Detection: In Fig. 14 (c) and (e), we also display the loop pair detected from both NetVLAD and our embedding model by using normalized (between 0 and 1) 14-bit representation. For this purpose, we alter the discrepancy threshold $\mathcal{S}_{ij}$ such that we can obtain a similar amount of loop pair compared to the normalized 8-bit representation. As it can be seen, either by using NetVLAD or our embedding model, training loop closure detection by using normalized 14-bit thermal representation produces many wrong loop pair and becomes unusable. This happens most likely because both models utilize the pre-trained ResNet as the feature extractor which relies heavily on the image appearance properties depicted by 8-bit RGB images. Quantitative measures shown by ROC curves in Fig. 15 also indicate that 14-bit models produce very bad performance (close to random guess), further reinforcing the qualitative results in Fig. 14 (c) and (e).

3) Efficiency of Neural Embedding Network: Table III shows the disk space required (MB) to save the embedding vectors from NetVLAD compared to our model. Our model requires much less space as it can represents a thermal image by using only 128 vectors. On the other hand, NetVLAD needs to produce 32768-dimensional VLAD vectors, which requires almost 7 GB disk space to save embedding vectors for all sequences. On the other hand, our model only need 27.3 M, 0.4% from what NetVLAD requires. In terms of runtime efficiency, NetVLAD takes 2.12 second (in CPU) to generate the embedding vectors from a single image, while our model takes only 0.63 second (3.4× faster). However, this is again slower than the real time implementation of loop closure detection in VINS-Mono (15-25Hz), although our model can also reach real time performance (27Hz, 0.037s per single inference) when it was tested in powerful TITAN V GPU.

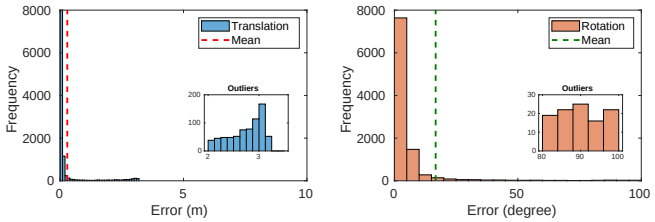
4) Validating Loop Closure Constraints: Being able to correctly detect loop pair is indeed an important stage in SLAM pipeline. However, estimating accurate poses between the loop closure constraints is actually more important. Even if we have large number of true positive loop pair, if the majority of relative poses among them are largely erroneous, it will badly impact the back end optimization, making the

TABLE III
SPACE REQUIREMENT (MB) TO SAVE EMBEDDING FEATURES IN GROUND ROBOT DATA

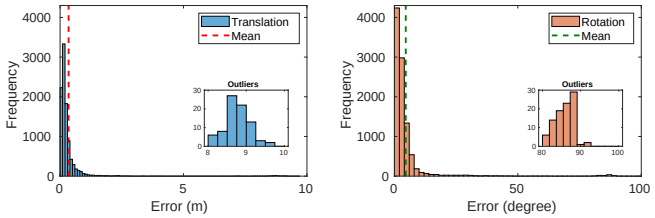
Seq	Files	NetVLAD	Our
32	2019-10-24-18-22-33	248	**1.1**
33	2019-11-23-15-54-25	266	**1**
34	2019-11-23-15-52-53	244	**0.9**
37	2019-11-23-15-59-12	362	**1.4**
39	2019-11-04-20-29-51	855	**3.3**
42	2019-11-22-10-10-00	789	**3.1**
43	2019-11-22-10-14-01	782	**3.2**
44	2019-11-22-10-22-48	829	**3.2**
45	2019-11-22-10-26-42	728	**2.8**
46	2019-11-22-10-34-57	252	**1**
47	2019-11-22-10-37-42	263	**1**
48	2019-11-22-10-38-47	472	**1.8**
49	2019-11-28-15-40-10	895	**3.5**
Total		6985	**27.3**

trajectory estimation even worse. To this end, we will validate the accuracy of our neural loop closure network. To have a better perspective on the accuracy, we compare our result with the standard feature-based pose estimation. Note that we have to re-scale the pose estimation generated by feature-based approach by using ground truth pose, since the estimation is correct only up to a scale. On the other hand, our model learns to implicitly estimate the scale by learning it from the ground truth during training.

Fig. 16 (a) and (b) describes the error distribution for translation and rotation component for all ground robot sequences generated by feature-based approach (SURF and 5-point algorithm) and neural loop closure respectively. Note that the translation estimation for SURF-based pose estimation is scaled with the ground truth. As one can see, our model produces robust and accurate results compared to the feature-based approach with around 0.344 m and 4.581 degree translation and rotation error respectively. However, there are conditions when the network produces large error possibly when it faces with NUC, large baseline scenario, or heavily featureless scene, in which hallucination network cannot even help. This is why outlier rejection in the back end becomes important component to generate accurate SLAM estimation.



(a) Feature-based approach



(b) Neural loop closure

Fig. 16. Comparison of error distribution between neural loop closure and SURF-based pose estimation approach for all ground robot sequences. Note that the translation estimation of SURF-based approach are scaled using the ground truth and only 89% of error data are displayed as the remaining fail to obtain sufficient correspondences.

TABLE IV
RMS ABSOLUTE TRAJECTORY ERRORS (M) IN INDOOR GROUND ROBOT DATA

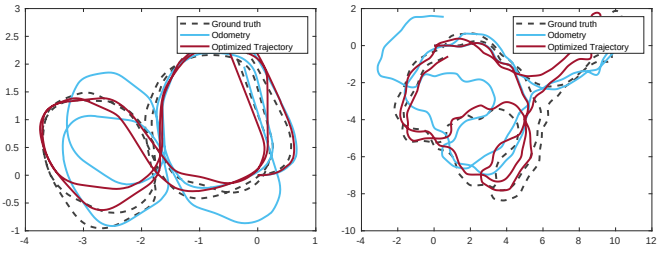
Seq	VINet (RGB)	VINet (Thermal)	TI odometry	TI-SLAM	Gain	VINS* (RGB)	IMU+ Wheel
32	1.453	2.195	0.308	**0.240**	22.1%	-	0.123
33	0.565	2.032	0.289	**0.182**	37%	-	0.067
34	1.583	0.804	0.364	**0.275**	24.5%	-	0.073
37	1.931	2.309	0.249	**0.164**	33.9%	-	0.076
39	5.309	5.975	0.916	**0.448**	51%	-	0.546
42	2.670	1.880	1.257	**0.141**	88.8%	0.351	0.270
43	1.543	2.819	0.592	**0.121**	79.5%	0.531	0.109
44	2.478	2.498	0.813	**0.419**	48.5%	0.691	0.188
45	2.022	2.329	0.526	**0.352**	33.1%	0.620	0.328
46	1.424	0.713	0.478	**0.143**	70.1%	0.240	0.160
47	1.182	1.348	0.393	**0.246**	37.5%	0.260	0.238
48	1.542	1.925	0.423	**0.369**	12.5%	0.783	0.505
49	3.218	10.47	1.311	**0.550**	58%	1.324	0.428
Mean	2.071	2.869	0.609	**0.281**	53.9%	0.600	0.239

*The trajectory is aligned with GT using [81].

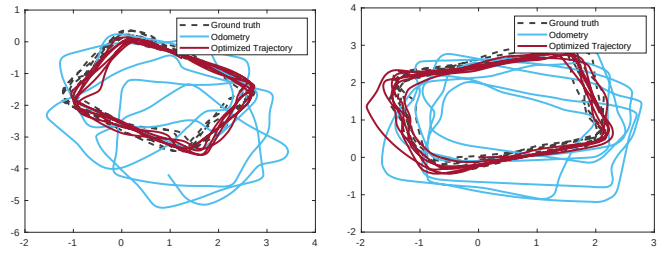
D. SLAM Performance in Ground Robot Data

1) Accuracy: Table IV lists the ATE of TI-SLAM for all test sequences. It can be seen that the complete SLAM system produces much better accuracy than the output trajectory from neural thermal-inertial odometry, showing an increase of 53.9%. In some sequences, the loop closure constraints can even improve the accuracy more than 70%, especially for a long trajectory that contains many loop pair (Seq 42, Seq 43, and Seq 46). On average, it yields 0.281 m errors, an order of magnitude smaller than VINet applied on thermal and better than VINS-Mono (RGB). Note that we even align VINS-Mono trajectory with the ground truth using [81]. In some sequences, TI-SLAM even generates more accurate performance than the IMU assisted wheel odometry (e.g. Seq 39, Seq 42, Seq 46 and Seq 48), showing the efficacy of our thermal-inertial SLAM system. To have better qualitative perspective, Fig. 17 shows some output trajectories.

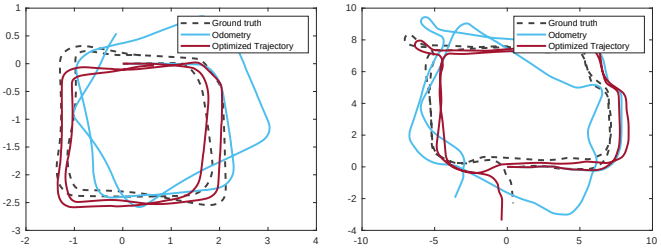
2) The importance of uncertainty estimates: In this section, we inspect the importance of incorporating uncertainty using our MDN model. We take Seq 43 for instance and replace the uncertainty estimation on that sequence with a simple identity matrix, either applied on the odometry or the loop closure constraints. Note that all other setting remain the same. Table V shows the output from this experiment. As expected, employing MDN covariance on both odometry and loop closure constraints can significantly increase the performance gain. This happens as our model estimates the uncertainty according to the input images and motion dynamics, which better reflects the actual condition compared to fix (identity) covariance. We can also observe that injecting covariance on the odometry produces better performance than injecting it on the loop constraints. This is possibly because odometry



(a) Seq 37 (b) Seq 39



(c) Seq 42 (d) Seq 43



(e) Seq 46 (f) Seq 49

Fig. 17. Qualitative result of TI-SLAM system in some ground robot data. Both odometry and optimized odometry are generated from thermal-inertial data.

TABLE V
THE IMPACT OF INCORPORATING COVARIANCE FROM MDN

Applying MDN covariance on Odometry	Applying MDN covariance on Loop	Odometry ATE	SLAM ATE	Gain
-	-	0.592	0.348	41.3%
✓	-	0.592	0.172	71.1%
-	✓	0.592	0.221	62.2%
✓	✓	0.592	0.121	79.5%

constraints are much denser than the loop closure constraints, in which a better uncertainty estimation on the denser data might lead to easier optimization.

3) The impact of uncertainty weight and scale: To understand the importance of balancing the weight between odometry and loop closure constraints by scaling the covariance, we measure the ATE of our model while changing the covariance scale from loop closure constraints ρ and fixing other parameters. Fig. 18 shows the result of this study. As we have discussed in Section VI, in general, we have to set larger weight in loop closure constraints to make the pose graph optimization work and improve the accuracy of odometry estimation. This reflects in Fig. 18 that the error typically lower when we set $\rho > 2$ and larger when we set $\rho < 2$. However, if we set ρ too large, this also generates sub-optimal performance as the effect of loop closure constraints becomes

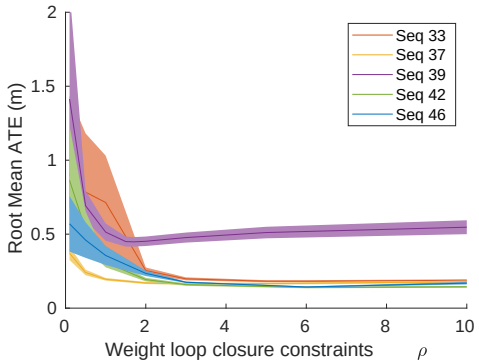


Fig. 18. The impact of uncertainty weights on Seq 33, 37, 39, 42, and 46. In this experiment, we fix every other parameters except the covariance weight ρ in loop closure constraints.

TABLE VI
RMS ABSOLUTE TRAJECTORY ERRORS (M) IN HANDHELD DATA

Seq	Length (m)	VINet (RGB)	VINet (Thermal)	TI odometry	TI-SLAM	Gain (%)
35	142	10.964	6.999	3.272	**1.033**	68.43
36	62	2.866	2.727	1.076	**0.661**	38.64
37	37	7.602	4.166	1.527	**0.538**	64.76
38	104	7.788	4.754	1.641	**0.444**	72.90
39	76	5.062	1.933	2.075	**0.921**	55.57
40	182	64.195	26.725	17.544	**4.175**	76.20
42	115	12.815	17.696	5.472	**1.859**	66.03
43	314	9.355	11.522	2.569	**1.038**	59.58
Mean		15.081	9.565	4.397	**1.334**	62.76

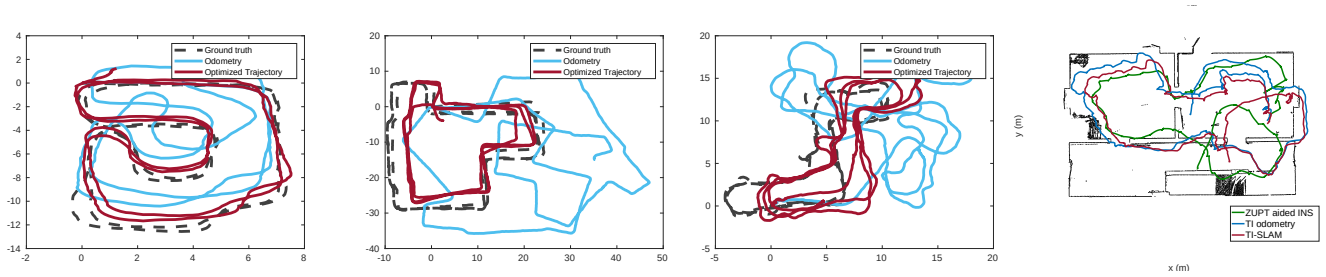
too dominant (as seen in Seq 39 in Fig. 18). Nevertheless, the impact on ATE is not as bad as setting small ρ.

E. SLAM Performance in Handheld Data

1) SLAM Accuracy: For the evaluation in the handheld data, we fine-tune our neural odometry and neural loop closure in the handheld data and keep the neural embedding network as it is. We need to fine-tune the neural odometry and neural loop closure as the dataset was collected with different sensor placement which impacts the pose estimation accuracy as the extrinsic parameters change. Table VI lists the ATE of TI-SLAM for all test sequences in handheld data. As we can see, our TI-SLAM produces the most accurate trajectories compared to VINet (RGB), VINet (Thermal), and TI-SLAM. Overall, by incorporating loop closure detection and robust pose graph optimization, we can improve the accuracy of TI odometry for up to 62.76%. Fig. 19 (a)-(c) show some qualitative results from the handheld evaluation.

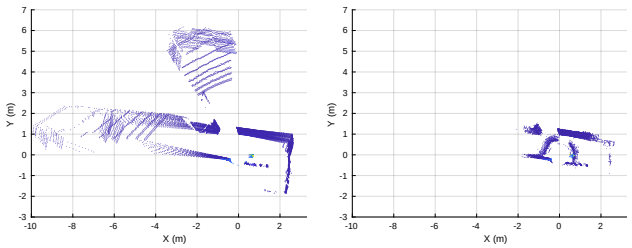
2) Test in Smoke-filled Environment: For the evaluation in smoke-filled environment, we compare our approach with a zero-velocity-aided (ZUPT) Inertial Navigation System (INS) [82] as none of RGB- and Lidar-based odometry/SLAM can work [8]. As can be seen from Fig. 20 (a) and (b), the smoke blocks the lidar signal, creating a half-sphere barrier in front of the device and degrading the lidar odometry/SLAM algorithm[4]. In that sense, without the availability of (pseudo) ground truth, only qualitative results are provided. Nevertheless, we can see from Fig. 19 (d) that our TI-SLAM produces

[4]https://www.youtube.com/watch?v=EZ1gpetEN8c



(a) Seq 38 (b) Seq 40 (c) Seq 42 (d) Firefighter Facility

Fig. 19. (a)-(c) Qualitative result of TI-SLAM system in the large scale handheld data. Both odometry and optimized odometry are generated from thermal-inertial data. (d) Test in real emergency scenario with smoke-filled environment. We qualitatively compared TI-SLAM with ZUPT aided INS as RGB, depth, or lidar-based odometry/SLAM system does not work. Note that the floor plan was generated with Lidar SLAM prior to the testing.



(a) Before smoke (b) After smoke

Fig. 20. Lidar point cloud (a) before and (b) after the environment is filled with smoke. As can be observed from (b) that the smoke blocks the lidar signal, creating barrier in front of the device.

similar trajectory with ZUPT aided INS. It is good to note that the loop closure detection plays important roles of correcting the drift of TI odometry.

F. SLAM Performance in SubT-tunnel Data

For the experiment in SubT-tunnel dataset, we use our in-house ground robot as the base model and fine tune the SLAM front end in sequences `sr_B_route1` and `sr_B_route2`. The output trajectory and the ATE can be seen from Fig. 21 and Table VII. For comparison, we provide the result from IMU assisted wheel odometry, VINet (thermal), VINS-Mono (RGB) [76], and our TI odometry. Despite the fact that in some areas the tunnel has no illumination (complete darkness), we still can utilize VINS-Mono (RGB) as the robot is equipped with four LED illuminators. On the other hand, VINS-Mono (thermal), either using 8-bit or 14-bit representation, loses tracks after running for about a minute or two due to abrupt motion, the lack of thermal features, and low frame rate (10Hz).

As you can see in Table VII, compared to the state-of-the-art visual-inertial odometry and SLAM algorithms (e.g., VINet, VINS-Mono), TI-SLAM produces more consistent trajectory for a long mission (54 minutes) in poorly illuminated scene with diverse motion types (e.g., U-turn, stop motion, etc.). TI-SLAM is even much better than IMU assisted wheel odometry which typically performs better than our model in shorter indoor mission as can be seen in Table IV. Our loop closure detection and loop closure constraints estimation again provide an important role as TI odometry drift significantly in the long

TABLE VII
RMS ABSOLUTE TRAJECTORY ERRORS (M) IN A LONG MISSION IN SUBT-TUNNEL DATA

	VINet (thermal)	IMU+ Wheel	VINS-Mono (RGB)	TI Odometry	TI-SLAM
Mean	156.41	42.276	30.242	32.048	**19.277**
Std. Dev.	88.948	48.315	20.723	17.184	**10.909**

mission. By closing the loop, the accuracy can be improved by around 40% in `ex_B_route1` sequence. Nevertheless, it is good to note that our approach is running offline while VINS-Mono has to respect realtime constraints.

VIII. LESSON LEARNT, LIMITATIONS, AND FUTURE WORK

Despite the fact that TI-SLAM can work well in our test scenarios, there are limitations and lessons learnt that can be used as a ground for future exploration. First, for thermal-inertial odometry, accurate scale estimation remains a challenge in some scenarios and pose graph SLAM cannot completely fix it. This problem becomes apparent especially in longer mission as depicted in hand-held (Fig. 19 (c)) and SubT-tunnel experiment (Fig. 21). Incorporating range sensor like millimeter wave radar [83] can be potentially used to alleviate this problem.

Second, the framework currently operates in offline fashion as real-time performance remain an open problem for deep networks, especially when it is executed in standard CPU. In real-time scenarios, deep network compression and acceleration [84], [85] can be used in the future investigation.

Third, as TI-SLAM brings together deep learning approach and conventional pose graph optimization, tuning hand-crafted parameters are required in the SLAM back end, e.g., scaling covariance between odometry and loop closure constraints, choosing a threshold to reject false positive loop constraints, etc. In this case, typically, there are no general hand-crafted parameters that can maximally perform for individual sequence. For online operation, parameters that generate the best average result are usually used while for offline operation, these parameters can be tuned individually for each sequence. In that sense, training a model that can automatically predict the back end parameters would be a viable future direction. Finally, domain adaptation also remains a challenge for deep odometry network. While the embedding network is usually

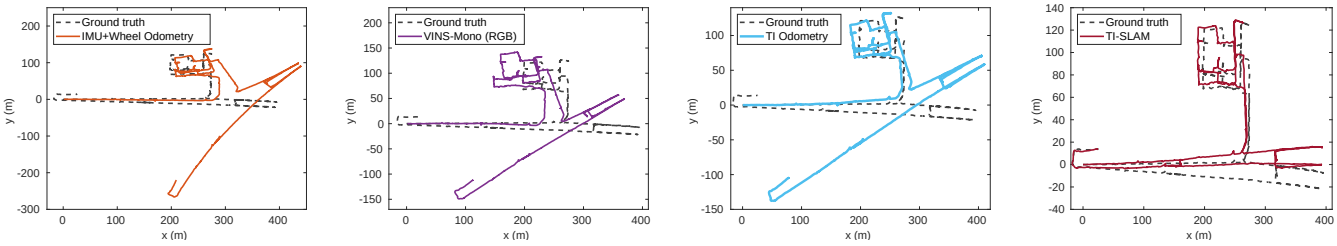


Fig. 21. Test in a long mission (54 minutes) in `ex_B_route1` sequence from SubT-tunnel dataset. The trajectory generated by inertial+wheel, VINS-Mono (RGB), TI-odometry (ours), and TI-SLAM (ours) are depicted.

more generalized in cross domain scenarios (e.g., the neural embedding network trained in ground robot data can be directly used for testing in handheld data), odometry and loop closure network need to be tuned for different domain as the intrinsic and extrinsic parameters between sensors might change. Designing a DNN model that can learn to estimate these intrinsic and extrinsic parameters during operation can be an interesting topic for future research direction.

IX. Conclusion

In this paper, we have demonstrated the first complete thermal-inertial SLAM system. Our key approach enabling full thermal-inertial SLAM is the usage of probabilistic neural networks to abstract noisy sensor data such that it will be more amenable for SLAM inference. By combining this neural abstraction in the SLAM front end with a robust graph-based optimization in the SLAM back end, we can generate an accurate trajectory estimation for different scenarios including handheld (firefighting) and ground robot motion in indoor and underground tunnel. Future research directions include designing online thermal-inertial SLAM system such that it can work within real time constraints and incorporating a range-based sensor to produce more accurate results in arbitrary environments.

Appendix
Training Details

A. Neural Thermal-Inertial Odometry

As we have mentioned in Sec. V-A3, to train neural thermal-inertial odometry, we use Eq. (8) as the objective function in the first stage of training. Adam optimizer with a 0.0001 learning rate is used for maximum of 200 epochs during this training process. Before training, we normalize the input 14-bit radiometric data (into between 0 and 1) by using the maximum and minimum radiometric value extracted from the training dataset. We then subtract it with the mean over the training dataset. We randomly cut the training sequence into small batches of consecutive pair ($n = 8$) to obtain better generalization. We also sub-sample the input such that we operate on around 5 fps to provide sufficient parallax between consecutive frames. In the second stage, we continue to train the network by using Eq. (9) for 200 epochs with RMSProp. We set 0.001 for the initial learning rate and then drop it by 25% after every 25 epochs. We also follow this procedure to train neural loop closure network.

B. Neural Embedding Network

For the neural embedding network, we train the network using Eq. 10 as the objective for a maximum of 200 epochs using Adam optimizer with 0.0001 initial learning rate. We set the threshold of adjacent frames empirically as 18. Before training, we normalize the raw 14-bit radiometric data (into between 0 and 1) using the maximum and minimum radiometric value extracted from the training dataset. We then convert back the normalized radiometric data into a grayscale image (8-bit) and copy the channels into three such that it can replicate the standard RGB images consumed by the ResNet50. We only use a batch size of 3 during training to fit it in our GPU.

References

[1] 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," *IEEE Trans. Robot. (T-RO)*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] J. Li, Z. Li, Y. Feng, Y. Liu, and G. Shi, "Development of a human–robot hybrid intelligent system based on brain teleoperation and deep learning slam," *IEEE Trans. Autom. Sci. Eng. (T-ASE)*, vol. 16, no. 4, pp. 1664–1674, 2019.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Trans. Robot. (T-RO)*, vol. 31, no. 5, pp. 1147–1163, 2015.

[4] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robot.: Sci. Syst. (RSS)*, vol. 2, no. 9, 2014.

[5] C. Debeunne and D. Vivet, "A review of visual-lidar fusion based simultaneous localization and mapping," *Sensors*, vol. 20, no. 7, p. 2068, 2020.

[6] J. Ruppelt and G. F. Trommer, "Stereo-camera visual odometry for outdoor areas and in dark indoor environments," *IEEE Aerospace Elect. Syst. Mag.*, vol. 31, no. 11, pp. 4–12, 2016.

[7] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based direct thermal–inertial odometry," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2019, pp. 3563–3569.

[8] M. Bijelic, F. Mannan, T. Gruber, W. Ritter, K. Dietmayer, and F. Heide, "Seeing through fog without seeing fog: Deep sensor fusion in the absence of labeled training data," *arXiv preprint arXiv:1902.08913*, 2019.

[9] C. Brunner and T. Peynot, "Perception quality evaluation with visual and infrared cameras in challenging environmental conditions," in *Experimental Robotics*. Springer, 2014, pp. 711–725.

[10] T. Mouats, N. Aouf, L. Chermak, and M. A. Richardson, "Thermal stereo odometry for uavs," *IEEE Sensors Journal*, vol. 15, no. 11, pp. 6335–6347, 2015.

[11] J. Delaune, R. Hewitt, L. Lytle, C. Sorice, R. Thakker, and L. Matthies, "Thermal-inertial odometry for autonomous flight throughout the night," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1122–1128.

[12] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2017, pp. 2043–2050.

[13] M. R. U. Saputra, P. P. de Gusmao, S. Wang, A. Markham, and N. Trigoni, "Learning monocular visual odometry through geometry-aware curriculum learning," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2019, pp. 3549–3555.

[14] Q. Li, S. Chen, C. Wang, X. Li, C. Wen, M. Cheng, and J. Li, "Lo-net: Deep real-time lidar odometry," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2019, pp. 8473–8482.

[15] H.-J. Liang, N. J. Sanket, C. Fermüller, and Y. Aloimonos, "Salientdso: Bringing attention to direct sparse odometry," *IEEE Trans. Autom. Sci. Eng. (T-ASE)*, vol. 16, no. 4, pp. 1619–1626, 2019.

[16] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, "Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2017, pp. 6856–6864.

[17] B. Wang, C. Chen, C. X. Lu, P. Zhao, N. Trigoni, and A. Markham, "Atloc: Attention guided camera localization," in *AAAI Conf. Art. Intell. (AAAI)*, 2020.

[18] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, "Learning to localize using a lidar intensity map." in *Conf. Robot. Learn. (CoRL)*, 2018, pp. 605–616.

[19] M. R. U. Saputra, P. P. de Gusmao, C. X. Lu, Y. Almalioglu, S. Rosa, C. Chen, J. Wahlström, W. Wang, A. Markham, and N. Trigoni, "Deeptio: A deep thermal-inertial odometry with visual hallucination," *IEEE Robot. Auto. Lett. (RA-L)*, vol. 5, no. 2, pp. 1672–1679, 2020.

[20] C. Chen, S. Rosa, Y. Miao, C. X. Lu, W. Wu, A. Markham, and N. Trigoni, "Selective sensor fusion for neural visual-inertial odometry," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2019, pp. 10 542–10 551.

[21] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI)*, vol. 29, no. 6, pp. 1052–1067, 2007.

[22] N. Sunderhauf, S. Lange, and P. Protzel, "Using the unscented kalman filter in mono-slam with inverse depth parametrization for autonomous airship control," in *IEEE Int. Work. Safety Security Resc. Robot.* IEEE, 2007, pp. 1–6.

[23] S. Holmes, G. Klein, and D. W. Murray, "A square root unscented kalman filter for visual monoslam," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2008, pp. 3710–3716.

[24] J. Civera, A. J. Davison, and J. M. Montiel, "Inverse depth parametrization for monocular slam," *IEEE Trans. Robot. (T-RO)*, vol. 24, no. 5, pp. 932–945, 2008.

[25] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual slam: why filter?" *Image Vis. Comp.*, vol. 30, no. 2, pp. 65–77, 2012.

[26] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *IEEE and ACM Int. Symp. Mixed Augmen. Real.* IEEE, 2007, pp. 225–234.

[27] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2017, pp. 4503–4508.

[28] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Comput. Surv. (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[29] S. Wang, R. Clark, H. Wen, and N. Trigoni, "End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks," *Int. Jour. Robot. Res. (IJRR)*, vol. 37, no. 4-5, pp. 513–542, 2018.

[30] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Toward geometric deep slam," *arXiv preprint arXiv:1707.07410*, 2017.

[31] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2020, pp. 4938–4947.

[32] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2017, pp. 1851–1858.

[33] Y. Almalioglu, M. R. U. Saputra, P. P. de Gusmao, A. Markham, and N. Trigoni, "Ganvo: Unsupervised deep monocular visual odometry and depth estimation with generative adversarial networks," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2019, pp. 5474–5480.

[34] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *IEEE Int. Conf. Comp. Vis. (ICCV)*, 2015, pp. 2938–2946.

[35] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *IEEE Conf. Comp. Vis. Patt. Recog, (CVPR)*, 2016, pp. 5297–5307.

[36] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2017, pp. 3223–3230.

[37] X. Zhang, Y. Su, and X. Zhu, "Loop closure detection for visual slam systems using convolutional neural network," in *Int. Conf. Autom. Comp. (ICAC)*. IEEE, 2017, pp. 1–6.

[38] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *AAAI Conf. Art. Intell. (AAAI)*, 2017.

[39] N. Piasco, D. Sidibé, V. Gouet-Brunet, and C. Demonceaux, "Learning scene geometry for visual localization in challenging conditions," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2019, pp. 9094–9100.

[40] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *Int. Jour. Robot. Res. (IJRR)*, vol. 27, no. 6, pp. 647–665, 2008.

[41] W. Maddern and S. Vidas, "Towards robust night and day place recognition using visible and thermal imaging," *RSS*, 2012.

[42] K. Zhou, C. Chen, B. Wang, M. R. U. Saputra, N. Trigoni, and A. Markham, "Vmloc: Variational fusion for learning-based multimodal camera localization," in *AAAI Conf. Artif. Intell. (AAAI)*, 2021, pp. 263–272.

[43] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe, "Understanding the limitations of cnn-based absolute camera pose regression," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2019, pp. 3302–3312.

[44] Z. Laskar, I. Melekhov, S. Kalia, and J. Kannala, "Camera relocalization by computing pairwise relative poses using convolutional neural network," in *IEEE Int. Conf. Comp. Vis. (ICCV Workshops)*, 2017, pp. 929–938.

[45] V. Balntas, S. Li, and V. Prisacariu, "Relocnet: Continuous metric learning relocalisation using neural nets," in *Europ. Conf. Comp. Vis. (ECCV)*, 2018, pp. 751–767.

[46] M. Ding, Z. Wang, J. Sun, J. Shi, and P. Luo, "Camnet: Coarse-to-fine retrieval for camera re-localization," in *IEEE Int. Conf. Comp. Vis. (ICCV)*, 2019, pp. 2871–2880.

[47] R. Li, S. Wang, and D. Gu, "Deepslam: A robust monocular slam system with unsupervised deep learning," *IEEE Trans. Ind. Electron. (T-IE)*, 2020.

[48] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robot. Autom. Lett. (RA-L)*, vol. 5, no. 2, pp. 721–728, 2020.

[49] P. V. K. Borges and S. Vidas, "Practical infrared visual odometry," *IEEE Trans. Intell. Transport. Syst. (T-ITS)*, vol. 17, no. 8, pp. 2205–2213, 2016.

[50] S. Khattak, C. Papachristos, and K. Alexis, "Keyframe-based thermal–inertial odometry," *Journal of Field Robotics*, vol. 37, no. 4, pp. 552–579, 2020.

[51] S. Vidas and S. Sridharan, "Hand-held monocular slam in thermal-infrared," in *12th International Conference on Control Automation Robotics & Vision (ICARCV)*. IEEE, 2012, pp. 859–864.

[52] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *Europ. Conf. Comp. Vis. (ECCV)*. Springer, 2014, pp. 834–849.

[53] Y.-S. Shin and A. Kim, "Sparse depth enhanced direct thermal-infrared slam beyond the visible spectrum," *IEEE Robot. Autom. Lett. (RA-L)*, vol. 4, no. 3, pp. 2918–2925, 2019.

[54] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intell. Transport. Syst. Mag.*, vol. 2, no. 4, pp. 31–43, 2010.

[55] C. M. Bishop, "Mixture density networks," 1994.

[56] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2016, pp. 4762–4769.

[57] R. Williams, W. J. Parrish, and J. Wolfe, "Fixed pattern noise mitigation for a thermal imaging system," Mar. 12 2019, uS Patent 10,230,912.

[58] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in statistics*. Springer, 1992, pp. 492–518.

[59] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2018, pp. 7286–7291.

[60] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robot. Autonom. Syst. (RAS)*, vol. 64, pp. 1–20, 2015.

[61] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. Jour. Comp. Vis. (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[62] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Europ. Conf. Comp. Vis. (ECCV)*. Springer, 2006, pp. 404–417.

[63] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2. Ieee, 2000, pp. 1023–1029.

[64] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. Jour. Comp. Vis. (IJCV)*, vol. 42, no. 3, pp. 145–175, 2001.

[65] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *IEEE Trans. Robot. (T-RO)*, vol. 24, no. 5, pp. 1027–1037, 2008.

[66] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*. IEEE, 2010, pp. 3304–3311.

[67] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning cnn image retrieval with no human annotation," *IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI)*, vol. 41, no. 7, pp. 1655–1668, 2018.

[68] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "Probabilistic appearance-based place recognition through bag of tracked words," *IEEE Robot. Autom. Lett. (RA-L)*, vol. 4, no. 2, pp. 1737–1744, 2019.

[69] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2016, pp. 770–778.

[70] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conf. Comp. Vis. Pattern Recog. (CVPR)*, 2015, pp. 815–823.

[71] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Graphtinker: Outlier rejection and inlier injection for pose graph slam," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2017.

[72] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2011, pp. 3607–3613.

[73] J. G. Rogers, J. M. Gregory, J. Fink, and E. Stump, "Test your slam! the subt-tunnel dataset and metric for mapping," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2020, pp. 955–961.

[74] A. J. Trevor, J. G. Rogers, and H. I. Christensen, "Omnimapper: A modular multimodal mapping framework," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2014, pp. 1983–1990.

[75] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *IEEE Int. Conf. Comp. Vis. (ICCV)*, 2015, pp. 2758–2766.

[76] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot. (T-RO)*, vol. 34, no. 4, pp. 1004–1020, 2018.

[77] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Mach. Intell. (T-PAMI)*, vol. 26, no. 6, pp. 756–770, 2004.

[78] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin, "Context-based vision system for place and object recognition," in *IEEE Int. Conf. Comp. Vis. (ICCV)*, vol. 2. IEEE Computer Society, 2003, pp. 273–273.

[79] H. Lategahn, J. Beck, B. Kitt, and C. Stiller, "How to learn an illumination robust image feature for place recognition," in *IEEE Intell. Vehic. Symp. (IV)*. IEEE, 2013, pp. 285–291.

[80] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3d point clouds," in *IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2017, pp. 5266–5272.

[81] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Comp. Arch. Lett.*, vol. 13, no. 04, pp. 376–380, 1991.

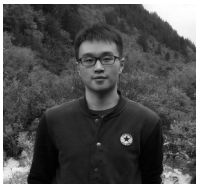
[82] J. Wahlström, I. Skog, F. Gustafsson, A. Markham, and N. Trigoni, "Zero-velocity detection – A Bayesian approach to adaptive thresholding," *IEEE Sensors Letters*, vol. 3, no. 6, 2019.

[83] C. X. Lu, M. R. U. Saputra, P. Zhao, Y. Almalioglu, P. P. de Gusmao, C. Chen, K. Sun, N. Trigoni, and A. Markham, "milliego: single-chip mmwave radar aided egomotion estimation via deep sensor fusion," in *ACM Conf. Embed. Net. Sensor Syst. (SenSys)*, 2020, pp. 109–122.

[84] M. R. U. Saputra, P. P. de Gusmao, Y. Almalioglu, A. Markham, and N. Trigoni, "Distilling knowledge from a deep pose regressor network," in *IEEE Int. Conf. Comp. Vis. (ICCV)*, 2019, pp. 263–272.

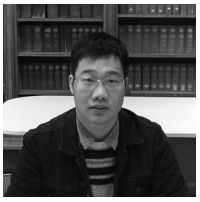
[85] T. Choudhary, V. Mishra, A. Goswami, and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artif. Intell. Rev.*, pp. 1–43, 2020.

Muhamad Risqi U. Saputra is currently an Assistant Professor in Data Science at Monash University, Indonesia. Previously, he was a postdoctoral research associate in Computer Science Department, University of Oxford, UK. He also obtained his DPhil degree from Oxford. Before coming to Oxford, he received his bachelor and master degrees from the Department of Electrical Engineering and Information Technology, Universtas Gadjah Mada, Indonesia. His main research interests revolves around machine learning, computer vision, and cyber-physical systems.



Chris Xiaoxuan Lu is currently an Assistant Professor (UK Lecturer) in the School of Informatics at University of Edinburgh. Before that he did both PhD study and post-doctoral research in the Department of Computer Science, University of Oxford. His research interest lies in Cyber Physical Systems, Robotics and Autonomous Systems and Artificial Intelligence of Things (AIoT).

Pedro Porto B. de Gusmao currently holds a senior research associate position at the Department of Computer Science and Technology, University of Cambridge, UK. Previously, he was a postdoctoral researcher at the Cyber-Physical Systems group, University of Oxford. He received bachelors degree in Telecommunication Engineering from the University of Sao Paulo and masters degree on the same field from the Politecnico di Torino in a double degree program. In 2017, he obtained his PhD from the same Politecnico. His research interests include computer vision, machine learning and signal processing.



Bing Wang is currently PhD student at Department of Computer Science, University of Oxford. Before that, he obtained his BEng Degree at Shenzhen University, China. His research interest lies in camera localization, feature detection, description and matching, and cross-domain representation learning.

Andrew Markham is an Associate Professor at the Department of Computer Science, University of Oxford. He obtained his BSc (2004) and PhD (2008) degrees from the University of Cape Town, South Africa. He is the Director of the MSc in Software Engineering. He works on resource-constrained systems, positioning systems, in particular magneto-inductive positioning and machine intelligence.

Niki Trigoni is a Professor at the Department of Computer Science, University of Oxford. She is currently the director of the EPSRC Centre for Doctoral Training on Autonomous Intelligent Machines and Systems, and leads the Cyber Physical Systems Group. Her research interests lie in intelligent and autonomous sensor systems with applications in positioning, healthcare, environmental monitoring and smart cities.