Article | Received 10 September 2024; Accepted 18 November 2024; Published Day Mon Year
https://doi.org/10.55092/rl20240003

FTI-SLAM: federated learning-enhanced thermal-inertial SLAM

Haochen Liu[1,†], Hantao Zhong[1,†] and Weiyong Si[2,*]

[1] Department of Computer Science and Technology, University of Cambridge, Cambridge, UK
[2] School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK

[†] Haochen Liu and Hantao Zhong contribute equally.

* Correspondence author; E-mail: w.si@essex.ac.uk.

Abstract: Utilising thermal imaging for simultaneous localisation and mapping has effectively improved the performance and robustness of robots and autonomous systems in unconventional environments. However, the transmission of large amounts of visual data from terminal devices to the central system for training not only results in high communication costs and pressure on bandwidth, but also induces concerns regarding privacy. Meanwhile, for applications in the real world, it is essential to expand the input domain to more practical scenarios rather than relying on experimental environments, and the terminal devices in-service can also benefit from further training with data collected in operations. To deal with these challenges, we investigated FTI-SLAM by applying federated learning to a thermal-inertial simultaneous localisation and mapping system. We conducted a series of experiments and showed that federated learning is feasible for the task and can improve overall performance.

Keywords: robotics; robot learning; SLAM; distributed artificial intelligence; distributed architectures

1. Introduction

1.1. Simultaneous localisation and mapping

Simultaneous localisation and mapping (SLAM) is an essential problem in robotics. For robots and other autonomous systems, it enables them to construct maps of surroundings and confirm their locations and moving trajectories in the environment at the same time, without prior knowledge of the environment, support from other positioning systems, or human interference [1]. It has broad application scenarios, including domestic robots [2], autonomous driving vehicles [3], and support in emergency and hazardous missions [4].

SLAM mainly relies on signals from visual sensors, such as panoramic cameras [5] and RGB-D cameras [6], sound navigation and ranging (Sonar) sensors [7], and light detection and ranging (LiDAR) sensors [8]. Other sources of signals, such as accelerations and angular velocities from inertial navigation systems (INS) [9], and radars [10], can also be involved to obtain enhanced performance. Currently, vision-based SLAM has been preferred over LiDAR-based due to the improvement in visual sensors and the significantly more information from various types of image and video data, making it suitable for tasks in various environments [1]. A SLAM system can be separated into a front-end and a back-end [1]. The front-end is responsible for processing and utilising data from sensors, obtaining abstractions such as extracted features from images, correspondence, loop closure, and more

[1,11]. The back-end optimises these abstractions and provides inferences about the states of the agent [8,11].

For visual-based SLAM, early attempts employed filter-based approaches. Kalman Filters (KF) has been actively applied and improved in SLAM. For example, Jensfelt *et al.* [12] utilised the Extended Kalman Filter (EKF), providing a solution based on data from a single camera. MonoSLAM proposed by Davison *et al.* [13] is also EKF- and single camera-based, outputting real-time 3-dimensional trajectories of the object. Jiang *et al.* [14] mitigated the problem of filter divergence that occurred in EKF methods with Adaptive Kalman Filter (AKF). However, these KF-based methods perform poorly in the aspect of correspondence, which is related to identifying previously visited places [1], and rely heavily on the hypothesis of Gaussian noise, limiting their performance in large and complex environments [1].

1.2. Deep learning and SLAM

In recent years, the rapid evolution in machine learning, especially deep neural networks (DNN), enables visual SLAM to become data-driven, performing more accurately and robustly in various environments [15,16]. These methods partially or fully adopt DNNs in the system. For the front-end, Wang *et al.* [17] proposed DeepVO, using deep recurrent convolutional networks (RCNN) to address the temporal dependencies and representations in raw data, which overcame the difficulties of hard-coding algorithm parameters for specific environments and performed well in new environments. Clark *et al.* [18] further involved using long short-term memory (LSTM) to incorporate inertial data with features learned by deep RCNN from RGM images, which relieves users from manual calibration and synchronisation between inertial sensors and cameras. For loop closure detection, Merrill and Huang [19] proposed a method based on the unsupervised auto-encoder, obtaining reliable features that were robust against changes in environments including brightness, weather, and other moving subjects. For the task of relocalisation, which aims to solve the kidnapped robot problem [20], Kendall *et al.* [21] propose PoseNet, which employs deep convolutional neural networks (CNN) and transfer learning for end-to-end relocalisation. In general, DNN models significantly reduce the need for hand-crafting features and show feasibility in multiple SLAM sub-tasks.

Researchers are also working on unifying the sub-tasks under deep learning to provide complete end-to-end SLAM approaches. Li *et al.* [22] proposed DeepSLAM, an unsupervised and deep learning SLAM system, using DNN architectures proven in related fields, including RCNN and encoder-decoder, for 3D map construction, robot pose estimation, and loop closure detection. The unsupervised system required no human annotations, making it capable of working on unlabelled video data independently. For applying multi-modal data on SLAM, An *et al.* [23] utilised the advantage of both visual and LiDAR data, providing a deep unsupervised SLAM system with better performance than methods with a single modality.

1.3. Thermal-based SLAM

Though widely used, visual SLAM has some obvious weaknesses. One of them is the high sensitivity to different illumination conditions [24]. In dark environments, RGB sensors cannot provide data with acceptable quality, which affects the performance of the back-end [1]. In comparison, LiDAR sensors are less prone to illumination changes, but in common non-ideal weather conditions, such as rain and fog, the quality of LiDAR sensing drops drastically [25]. Similarly, in complex environments, for example, in a scene of fire with lots of smoke, the moving particles will downgrade the LiDAR sensors' ability to recognise the environment.

Relying solely on RGB images overlooks other regions of the spectrum that can provide supplemental information, such as the long-wavelength infrared band [26]. Under scenarios with little or no visible lights, thermal sensors can still provide information through infrared radiations from the surrounding environment and objects, which improves the situational

perception and robustness of robots [26–28]. Inspired by these advantages, researchers have started to incorporate thermal sensing into SLAM tasks. For odometry, Saputra *et al.* [29] propose DeepTIO, the first deep learning-based thermal-inertial odometry model. DeepTIO performed selective feature fusion, adaptively selecting features across modalities including thermal images, inertial features from IMU, and supplementary features from thermal images using visual hallucination network [29–31]. On different devices in various environments including benign or smoke-filled rooms, DeepTIO outperformed existing visual-inertial odometry methods [29].

Other deep-learning-based designs, such as TP-TIO by Zhao *et al.* [32] that employed a CNN-based feature extraction tool, can outperform traditional visual odometry in more complex environments. In addition, there are also other explorations of thermal-inertial SLAM algorithms. Unlike TI-SLAM, which tackles the problem with learning-based approaches, Wang *et al.* [33] estimated odometry from thermal images and inertial measurements by focusing on the edges of objects where thermal radiations change significantly. The proposed Edge Thermal-Inertial Odometry (ETIO) algorithm adopts a distance-aided Kanade-Lucas-Tomasi tracker and adaptive feature tracking scheme, ensuring the robustness of the algorithm in environments with poor visual conditions.

1.4. Thermal-inertial SLAM

Based on the success of DeepTIO, by involving a graph-based approach and probabilistic neural networks, Saputra *et al.* [24] provided a full SLAM system based on thermal images and IMU sequences, referred to as TI-SLAM.

The front-end of TI-SLAM comprises three primary components: neural thermal-inertial odometry, neural embedding, and neural loop closure. Neural thermal-inertial odometry processes a series of thermal images to compute the 6-DoF poses over time. Similar to DeepTIO, features from thermal images are extracted with a thermal network and a hallucination network, and features from IMU sensors are extracted using long short-term memory. The architectures of the thermal network and hallucination network are based on Flownet [34]. The features are then selected using deterministic soft fusion [30], and based on the features, a pose regressor network built with LSTM and mixture density network [35] provides estimations of 6-Dof camera poses and uncertainty estimations for each timeframe. Neural embedding adapts a truncated section of the ResNet50 network to generate global descriptors, which serve as embeddings, from thermal images. These embeddings are subsequently utilised for loop pairs detection by evaluating the discrepancy among embedding using cosine distance. The Neural Loop Closure Network calculates loop closure constraints and ascertains whether the mobile agent has revisited a specific location.

The back-end of TI-SLAM contains an outlier rejection and an optimisation module. The outlier rejection module reduces the impact of noises and errors in the pose graph output by the front-end. The final trajectory is optimised using these loop closure constraints and 6-DoF poses through the Levenberg-Marquardt algorithm [36].

TI-SLAM is evaluated on multiple datasets of different environment conditions, including environments with smoke and rooms in different illumination conditions, against visual and/or inertial-based SLAM approaches and obtained much smaller root mean square (RMS) absolute trajectory errors, which shows the effectiveness and robustness of TI-SLAM [24].

1.5. Federated learning in robotics

The training of robots and autonomous systems requires large amounts of data from various sensors. For robot-related machine learning methods, training with data obtained from servicing can improve their performances, adjusting them correspondingly to the user scenarios [37, 38]. However, some data, such as images, videos, and audio recordings,

may contain sensitive information from the users, and transmitting these data from users to the central server causes concerns about privacy and security [39,40]. Recently, due to the concern of data leakage and inappropriate utilisation of user data, authorities across the world have enacted laws and regulations to restrict the transmission of sensitive data [41,42]. Meanwhile, transmitting large amounts of data leads to heavy pressure on the bandwidth, and the users' terminals are not always online for data transmission [37].

To efficiently utilise data while respecting concerns of privacy and provide equivalent or better support to tasks in robotics, internet of things (IoT), and other services involving autonomous systems powered by machine learning, federated learning [37, 43] provides solutions in a new perspective. Federated learning involves multiple terminal devices (clients) and a central server. The training of machine learning models is first conducted locally on the client's side using each client's specific data. After training, instead of sending training data, the parameters of local models are sent to the central server for aggregation, and the server will send back the combined model to clients for inference and further training. By doing this, federated learning not only secures privacy and reduces the abundant information transmission between users and service providers [37,40], but also allows more data from diverse sources to be involved in training, which improves the generalisation ability of obtained models and allows local fine-tuning to fit specific demands from users [44,45].

Many scenarios in robotics and autonomous systems can benefit from the introduction of federated learning. Except for privacy protection, for mobile robotics systems with various local datasets that are not independent and identically distributed, there are federated learning strategies that improve generalisation ability under such circumstances [46]. Federated learning methods have been introduced to self-driving cars [47], mobile IoT robots[48], and other specific tasks, including SLAM. For example, Li *et al.* [49] introduced federated learning to support the visual- and LiDAR-based SLAM system in feature extraction and place matching. Zhang *et al.* [50] proposed a distributed multi-vehicle SLAM system for autonomous driving, reducing the error in localisation through cooperative training and training time. There has not been any work focusing on federated learning, thermal images, and SLAM at the same time.

1.6. Federated TI-SLAM

The effectiveness of TI-SLAM has been shown in the original work [24]. However, the front-end of TI-SLAM contains multiple DNN models, which requires a large amount of time and computational resources for training. Meanwhile, transmitting a large amount of image and video data raises concerns for privacy and difficulties in communication. In practical user scenarios, the domain of data might vary from case to case, requiring a proper approach to efficiently utilise the data for better generalisation.

Inspired by the potential for improvement, we implemented and trained the federated learning version of TI-SLAM, referring as FTI-SLAM. We conducted experiments in different federated learning settings and data configurations. Our contributions can be summarised as follows:

(1) We provide a complete implementation of federated learning-adapted TI-SLAM, providing a framework for realistic experiments.
(2) We conduct a series of experiments on FTI-SLAM with limited computational resources and the same amount of training on the same data, showing that the introduction of federated learning can maintain similar performance levels and improve the system's overall performance when applying proper aggregation algorithms.
(3) We introduce alternations on FTI-SLAM, discuss the reasons for improvements and failures with detailed analysis, and provide possible guidelines for future works.

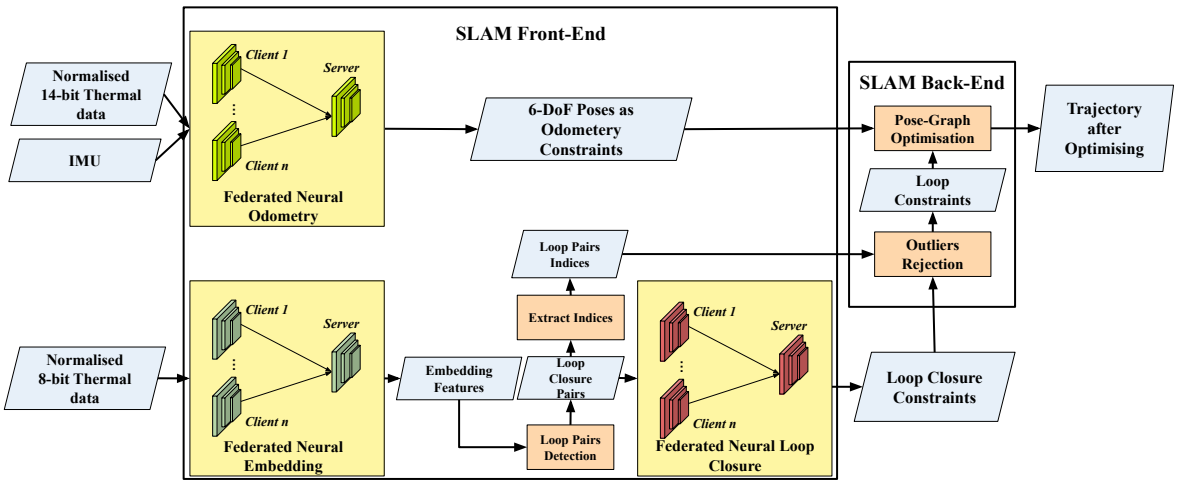


Figure 1. FTI-SLAM structure. Our framework introduced federated learning to the neural networks on the SLAM front-end. Adapted from the original TI-SLAM structure [24].

2. Methods

2.1. FTI-SLAM

To provide federated learning support for TI-SLAM, we adopt Flower framework [51] and implemented FTI-SLAM with it. Flower is a federated learning framework that is easy to use for simulating various federated learning settings. Although Flower supports TensorFlow 2.0 [52], the original TI-SLAM codebase was based on TensorFlow 1.0. Consequently, we have re-implemented and adapted the TI-SLAM codebase to TensorFlow 2.6 and Keras 2.6 [53]. The implementation of FTI-SLAM is available as a GitHub repository (https://github.com/MrTooOldDriver/ti-slam/). The structure of FTI-SLAM is displayed in Figure 1.

The training process of TI-SLAM front-end can be divided into the training of three parts: neural odometry, neural embedding, and neural loop closure. We establish three independent federated learning systems for each part. Each client is provided with a local dataset different from other clients, which closely reflects real-world scenarios as different clients might operate in various environments.

The federated neural embedding network training in FTI-SLAM follows the same process as in TI-SLAM, training from scratch with anchor $\mathbf{I}_T$, positive images $\mathbf{I}_T^+$, and negative images $\mathbf{I}_T^-$ for embedding training. The loss function for this part is a triplet loss. Given a triplet $\{\mathbf{I}_T, \mathbf{I}_T^+, \mathbf{I}_T^-\}$, the neural embedding network $d_{\mathbf{W}_T}$ where $\mathbf{W}_T$ is the trainable weights, and $d_{\mathbf{W}_T}(\mathbf{I}_T)$ denotes the embedding vector that defines the global image descriptor of an image $\mathbf{I}_T$, the triplet loss used for this part is defined as [24],

$$\mathscr{L}_{Triplet}(\mathbf{I}_T, \mathbf{I}_T^+, \mathbf{I}_T^-) = \max(\lambda + ||d_{\mathbf{W}_t}(\mathbf{I}_T) - d_{\mathbf{W}_t}(\mathbf{I}_T^+)||^2 - ||d_{\mathbf{W}_t}(\mathbf{I}_T) - d_{\mathbf{W}_t}(\mathbf{I}_T^-)||^2, 0), \quad (1)$$

where λ limits the margin between positive and negative images [24]. If the location is revisited, we expect the network to produce similar embeddings for those visits [24].

The training process for the federated neural odometry network and federated neural loop closure network differs from the corresponding parts of TI-SLAM. For the original neural odometry network, it is necessary first to initialise a hallucination network using Flownet [34], followed by hallucination training, before continuing to the thermal network training. Unfortunately, we were unable to locate a Flownet pre-trained weight compatible with TI-SLAM hallucination network, and the authors of TI-SLAM did not provide any details on that. To address and simplify this issue, we omit the hallucination network training and initialise it using the best checkpoint provided by the TI-SLAM authors. In this case, the

training only involves the second stage of neural odometry in the original implementation [24]. For each transition component $\mathbf{t} \in \mathbb{R}^3$ and rotation component $\mathbf{r} \in \mathbb{R}^3$ to estimate, and $Z = \{z_i : i = 1, 2, ..., m\}$ as the sensor measurements, the objective is to optimise the loss of mixture density networks (MDN) [35], *i.e.*,

$$\mathscr{L}_{MDN} = \sum_{i=1}^{m} \mathrm{p}(\mathbf{t}|Z)_i^- + \beta \sum_{i=1}^{m} \mathrm{p}(\mathbf{r}|Z)_i^-, \tag{2}$$

where $\mathrm{p}(\mathbf{t}|Z)_i^-$ and $\mathrm{p}(\mathbf{r}|Z)_i^-$ are the negative log-likelihoods for the posterior $\mathrm{p}(\mathbf{t}|Z)_i$ and $\mathrm{p}(\mathbf{r}|Z)_i$ [24]. In FTI-SLAM, the federated neural loop closure network begins training with a trained hallucination weight loaded into the global model, which is from the best checkpoint of TI-SLAM as well. It also uses MDN to estimate relative poses. However, unlike the neural odometry network, the neural loop closure network does not use IMU data as input; it only uses thermal images. The loss function is also the negative log-likelihood of MDN posterior.

For the neural loop closure network in TI-SLAM, the trained neural odometry network is used to initialise the feature extractor. This design shares the same feature extractor between the neural odometry network and the neural loop closure network. However, rewriting the training processes of both networks in FTI-SLAM with a federated learning update strategy for simultaneous feature extractor updating would be overly complex. Similar to our approach with the federated neural odometry network, we utilise the best checkpoint from the TI-SLAM codebase to initialise the feature extractor for the global model at the outset.

In TI-SLAM, data is read with float64 precision and cast to float32 precision to be used in the framework. Due to constraints in computational resources, we reduced the precision of image data from float32 to float16. This reduction conserves VRAM during the training and inference phases, and we verify that the reduction only affects the performance slightly.

2.2. *Federated aggregation algorithms*

For federated learning, a key component is the weight updating of the server model. It relies on what information is sent from clients to the server and how the update is conducted.

The first feasible federated aggregation algorithm is FederatedAveraging (FedAvg) [37]. FedAvg is improved upon the baseline of distributed training with synchronous stochastic gradient descent [54]. In the baseline, clients compute the gradient at each step and then upload the gradients to the server. The server updates global model weights with the aggregated gradients and then distributes the updated model weights to the clients. FedAvg provides an equivalent and efficient alternative by letting clients perform multiple local weight updates before sending local updated weights for aggregation [37]. For a system with K clients, the new weight after step t is given as

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{k,t+1}, \tag{3}$$

where

$$w_{k,t+1} \leftarrow w_t - \eta g_k, \tag{4}$$

with η being the learning rate, g_k being the gradient of client k, and $\frac{n_k}{n}$ denotes the proportion of weight contribution based on the local data proportion of client k [37]. Compared to the baseline, FedAvg reduces the communication cost between clients and the server by involving local updating, maintaining the performance on various tasks and DNN architectures [37].

In distributed systems including federated learning systems, abnormal behaviours in certain computing units, such as providing gradients or weights that diverge from the proper direction of optimisation, may cause performance degradation to the whole system [55,56]. Some serious and malicious attacks are even threatening the safety of user data and the

functionality of the entire system [57]. These attacks are categorised as Byzantine faults, sourcing from the Byzantine Generals Problem [58]. There has been active research in federated learning on mitigating the impact of a fraction of unidentifiable malfunctioning clients. For example, Yin *et al.* [56] provided algorithms that are Byzantine-robust and achieve optimal statistical performance, one of which is introducing trimmed mean to gradients in distributed gradient descent, removing two equal fractions of gradients that are too large or too small before computing the mean across remaining clients. Flower combined this approach with FedAvg and proposed FedTrimmedAvg, which trims the weighs of clients rather than gradients, *i.e.*, [51,56],

$$w_{t+1} \leftarrow \frac{1}{1-2\beta} \sum_{k \in \mathbf{K}'} \frac{n_k}{n} w_{k,t+1}, \tag{5}$$

where β is the trimming proportion and $\mathbf{K}'$ is the set of clients after trimming.

2.3. *Datasets and training configurations*

The dataset used in this study is collected by TI-SLAM authors [24]. This dataset consists of two parts. The first part is collected with a ground-moving robot (Turtlebot 2). This robot is equipped with a Flir Boson 640 thermal camera and an XSens MTI1 Series IMU sensor, and the data were collected in an indoor environment. The ground truth is based on Velodyne HDL-32E Lidar. The second part is collected with a handheld device customised by the TI-SLAM authors with the same set of sensors. Data for this part is collected from indoor environments with different floor areas and functions. The ground truth is based on Velodyne Ultra Puck LiDAR.

However, after carefully checking the available data, we noticed many data sequences are missing one or more necessary records for training and testing TI-SLAM and FTI-SLAM. For example, all sequences of data collected by the handheld device and sequences 1 to 11, 29, and 30 collected by the ground-moving robot do not have corresponding relative pose records, which makes the training of neural embedding and neural loop closure unable to be conducted. Meanwhile, data sequence 31 and its following sequences collected by the ground-moving robot do not have corresponding RGB features, preventing them from being used in experiments. This significantly limits the choices of data sequences available for training, validation, and testing of the FTI-SLAM framework.

Eventually, due to computational resource limitations and the inadequacies in data sequences, we selected all usable data sequences, with a total number of 16, and organised them in the experiment. These sequences are all collected by the ground-moving robot, with nine sequences (Seq 12-20) for training, three sequences (Seq 21-23) for validation, and four sequences (Seq 27-30) for testing. Meanwhile, to demonstrate the performances of FTI-SLAM in a more realistic environment, we exchanged Sequences 27 and 28 in the test set with Sequences 21 and 22 in the validation set. Sequences 29 and 30 cannot be used for training or validation due to missing necessary data entries. Given the relatively smaller size of the dataset, we reduce the training epochs to just 50. Except where specifically stated, all experimental settings are identical to the TI-SLAM paper [24]. For training FTI-SLAM, we utilised 16 NVIDIA 2080Ti 22GB graphics cards, where each client used a dedicated GPU. The server also hosted the centralised model on a dedicated GPU, allowing us to run all clients in parallel.

2.4. *Experiment setting*

We conduct a series of experiments on FTI-SLAM. We first focus on comparing our federated learning framework with the original form, TI-SLAM, which is completely centralised. In our experiment, the original TI-SLAM uses 9 sequences of training data and 3 validation sequences. In comparison, the FTI-SLAM involves 3 clients, and the 9 same sequences of data for training are separated evenly for the clients. It is the same for the 3 sequences for validation.

We also involve a 6-client experiment with clients using overlapped but not identical training sequences. Both 3-client and 6-client experiments adopt FedAvg and FedTrimmedAvg to compare how different federated aggregation algorithms affect the results. We also exchanged two sequences in the test set with two sequences in the validation set to allow sequences 21 and 22 to be used for testing as well.

3. Results

Table 1 shows the validation losses of FTI-SLAM frond-end components for all experiments conducted. We can see that two federated learning settings of FTI-SLAM slightly outperform the original setting, while in neural loop closure, non-federated learning achieves significantly lower loss than all federated learning ones.

Table 1. Validation losses of three parts of FTI-SLAM front-end in different experimental settings. Non-FL refers to the original TI-SLAM configuration. The lowest validation losses for each of the three parts are presented in **bold**.

Aggregation Algorithms	Clients	Train/Validation per Client	Neural Odometry	Neural Embedding	Neural Loop Closure
Non-FL	1	9 training, 3 validation	-28.4200	2.5310	**-4.4430**
FedAvg	3	3 training, 1 validation	-30.8130	**2.1438**	3.7723
FedAvg	6	3 training, 1 validation	-29.1692	2.1944	2.5621
FedTrimmedAvg	3	3 training, 1 validation	-30.9511	2.2600	2.9702
FedTrimmedAvg	6	3 training, 1 validation	**-31.0009**	2.1817	1.4772

In the following sections of discussion, we further compare the performances of different SLAM settings on the test sequences based on their estimations of odometry. The four metrics are:

- Root root mean square of the absolute trajectory errors of the thermal-inertial odometry without loop closure detection, denoted as RMSE, in meters (m).
- Root root mean square of the absolute trajectory errors of the thermal-inertial odometry after incorporating loop closure detection, denoted as RMSE_L, in meters (m).
- Variance of RMSE_L, denoted as $\text{Var}_{\text{RMSE}_L}$.
- The improvement after incorporating loop closure detection compared to using thermal-inertial odometry only, calculated as $\frac{\text{RMSE}-\text{RMSE}_L}{\text{RMSE}}$.

These metrics are also used for quantifying performances in the TI-SLAM paper [24]. We also present the ground truth trajectories, odometry trajectories, and trajectories after optimisation of test sequences for visually intuitive comparisons. As declared in [24], the data sequences involved in our experiment are collected by a ground robot moving on flat surfaces, resulting in negligible Z-axis translations. In this case, we visualise the ground truth trajectory, the odometry trajectory, and the optimised trajectory for each sequence in the X-Y plane for visual comparisons.

3.1. Verifying the applicability of data reduction

As described in Section 2.1., we reduce the data precision to float16 due to the strained computational resources. Before moving on to full-scale experiments, we compare the performance of TI-SLAM before and after applying data reduction.

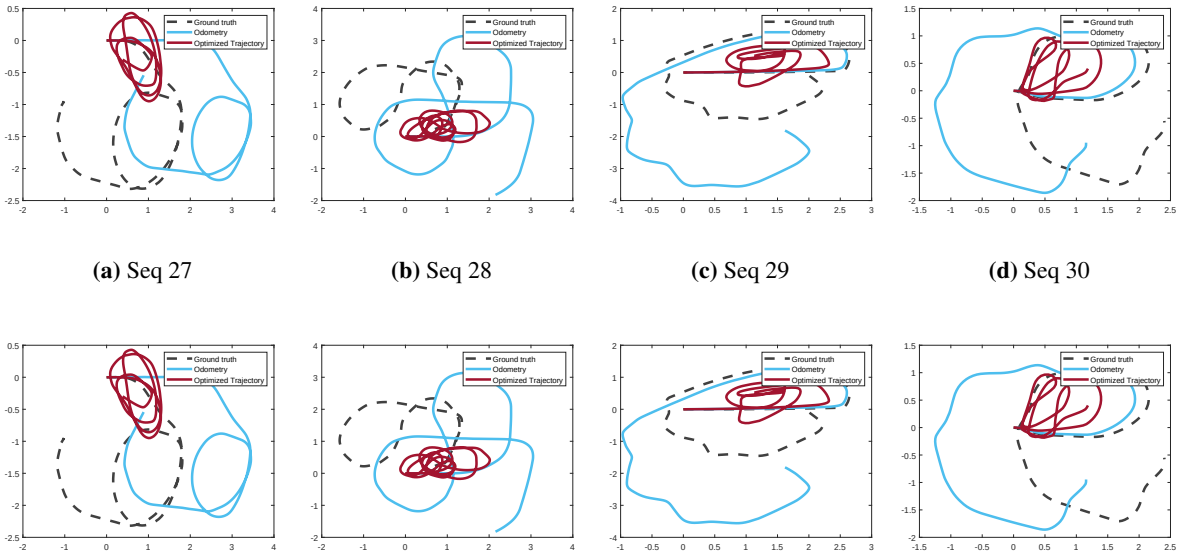
The two TI-SLAM frameworks with different data precisions are trained with a smaller number of sequences and tested on sequences 27 to 30. From the results in Table 2, it can be seen that data with reduced precision brings negligible impacts to the resulting metrics even after multiple steps of computation in the front-end neural network models and back-end algorithms.

Visually, as shown in Figure 2, it is difficult to tell the difference between the two settings.

For all four sequences, using data with reduced precision achieves almost identical trajectories compared to the original precision. In this case, we validate that using reduced precision here is feasible and does not bring a noticeable impact on the performance of the framework. We proceed with using reduced precision in the following experiments.

Table 2. RMSE, RMSE_L, $\text{Var}_{\text{RMSE}_L}$, and the percentage of improvement by optimisation of sequences obtained from TI-SLAM frameworks before and after data precision reduction to float16. All metrics are presented with 9 significant digits.

Test Sequence	Setting	RMSE (m)	RMSE_L (m)	$\text{Var}_{\text{RMSE}_L}$	Improvement
Seq 27	Before	0.936396639	0.907995997	0.158176793	3.03297141
	After	0.936396639	0.907996049	0.158176792	3.03296586
Seq 28	Before	0.856796745	0.889722414	0.228328572	-3.84287984
	After	0.856796745	0.889722410	0.228328568	-3.84287943
Seq 29	Before	0.599048563	0.666784891	0.176294525	-11.3073182
	After	0.599048563	0.666784879	0.176294498	-11.3073163
Seq 30	Before	0.525298776	0.671268465	0.197105876	-27.7879362
	After	0.525298776	0.671268492	0.197105878	-27.7879413



(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29 **(d)** Seq 30

(e) Seq 27 **(f)** Seq 28 **(g)** Seq 29 **(h)** Seq 30

Figure 2. Trajectory optimisation results of TI-SLAM in the original setting (a, b, c, d) and after data reduction to float16 precision (e, f, g, h). The grey dashed lines represent ground truth, the blue solid lines represent odometry trajectories, and the red solid lines represent the trajectories after optimisation.

3.2. Effect of federated learning (FedAvg)

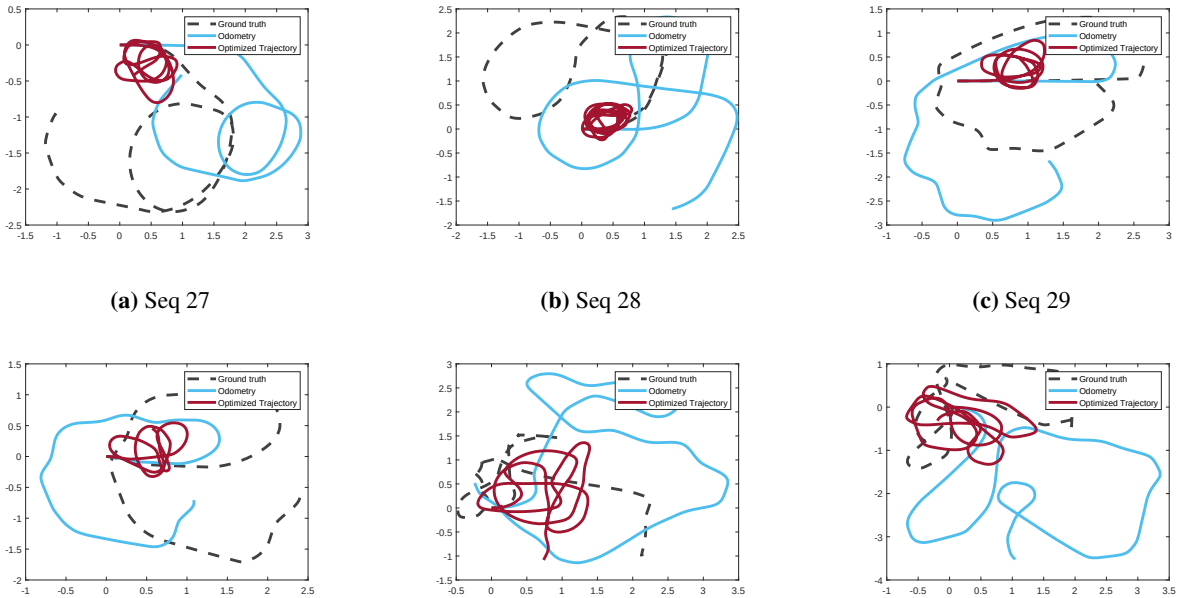
We first look at how federated learning influences the performance. When looking at the behavioural metrics as presented in Table 3, we can see that FTI-SLAM using FedAvg with 3 clients completely outperforms the original setting in Sequences 27, 29, and 22. For Sequences 30 and 21, though the optimisation does not bring larger improvements in the FedAvg setting, FedAvg still achieves a lower RMSE_L than the regular setting. However, this information cannot provide the full story of the performance. Figures 3 and 4 show how the optimised trajectories behave compared to the ground truth from LiDAR.

For the original setting, though the odometry trajectories are not very close to the ground truth, they can properly recover the pattern of ground truths. But it is clear that the optimised

trajectories for all sequences get tangled up, not even showing a proper pattern. This shows that the embedding and loop closure done by the front end might be overfitting, resulting in problems in optimisation.

Table 3. RMSE, RMSE_L, $\text{Var}_{\text{RMSE}_L}$, and the percentage of improvement by optimisation of sequences of the original non-federated learning TI-SLAM setting and FTI-SLAM with 3 clients and FedAvg for aggregation.

Test Sequence	Setting	RMSE (m)	RMSE_L (m)	$\text{Var}_{\text{RMSE}_L}$	Improvement(%)
Seq 27	Non-FL	0.7108	0.8482	0.1318	-19.3310
	FedAvg 3 Clients	**0.5892**	**0.5173**	**0.0783**	**12.2061**
Seq 28	Non-FL	0.7083	**0.8124**	**0.1360**	**-14.7035**
	FedAvg 3 Clients	**0.5162**	1.0668	0.5827	-106.6509
Seq 29	Non-FL	0.4862	0.6910	0.0952	-42.1315
	FedAvg 3 Clients	**0.3697**	**0.4817**	**0.0549**	**-30.2843**
Seq 30	Non-FL	0.5450	0.7021	**0.1090**	**-28.8233**
	FedAvg 3 Clients	**0.3363**	**0.5070**	0.5070	-50.7308
Seq 21	Non-FL	0.5682	0.5541	**0.0575**	**2.4755**
	FedAvg 3 Clients	**0.4936**	**0.4926**	0.0686	0.2072
Seq 22	Non-FL	0.5681	0.6041	0.0495	-6.3298
	FedAvg 3 Clients	**0.4652**	**0.4453**	**0.0245**	**4.2830**



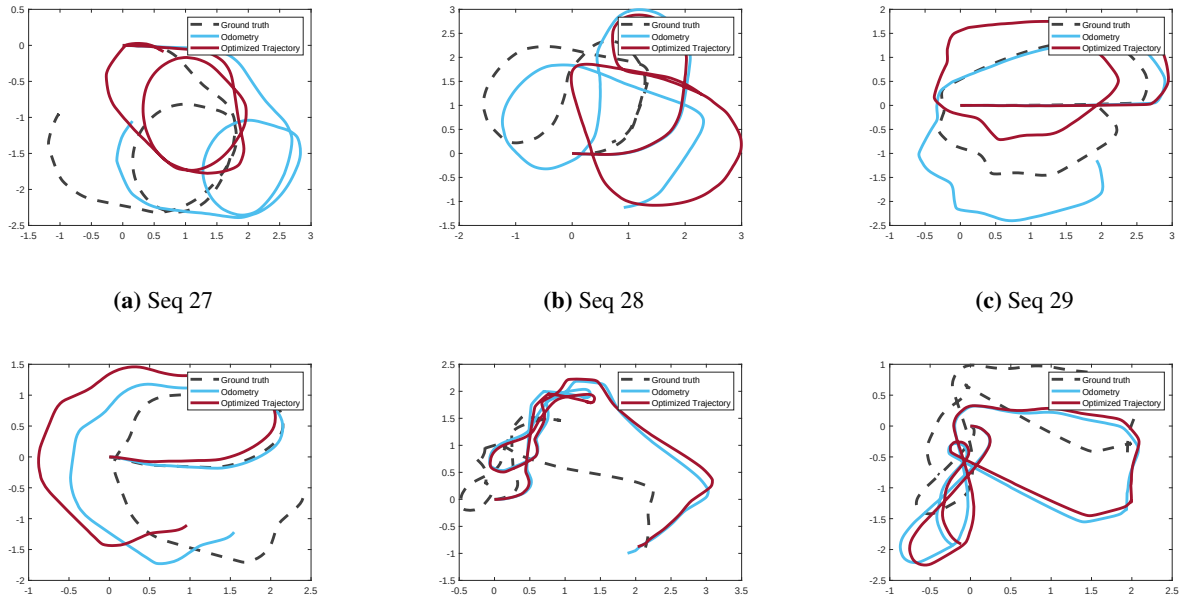
(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29

(d) Seq 30 **(e)** Seq 21 **(f)** Seq 22

Figure 3. Trajectory optimisation results of the non-federated learning setting.

On the contrary, for FedAvg with 3 clients, the optimised trajectories are not greatly diverged from the odometry trajectories, but they manage to recover some parts of the ground truth in Seq 29 and 30, as shown in Figures 4c and 4d. Figure 4a demonstrates that the optimisation is tending more towards the ground truth. With the same amount of training on the same training data, federated neural odometry performs similarly to the original neural odometry. Loop constraints from federated neural embedding and federated neural loop closure are not overfitting but would require more complex training data to make the optimised trajectory closer to the ground truth. In more scenarios of Sequences 21 and 22, their ground truth trajectories are more complicated than other sequences, and the optimisation results

based on loop closure detection almost overlap with trajectories from neural odometry. The results suggest that compared to the original fully centralised setting, federated learning can provide better generalisation ability for the front-end, especially for neural odometry. Still, to boost the performance of neural embedding and neural loop closure, we need to involve more complex cases in training.



(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29

(d) Seq 30 **(e)** Seq 21 **(f)** Seq 22

Figure 4. Trajectory optimisation results of FTI-SLAM with 3 clients using FedAvg for aggregation.

3.3. Aggregation algorithms

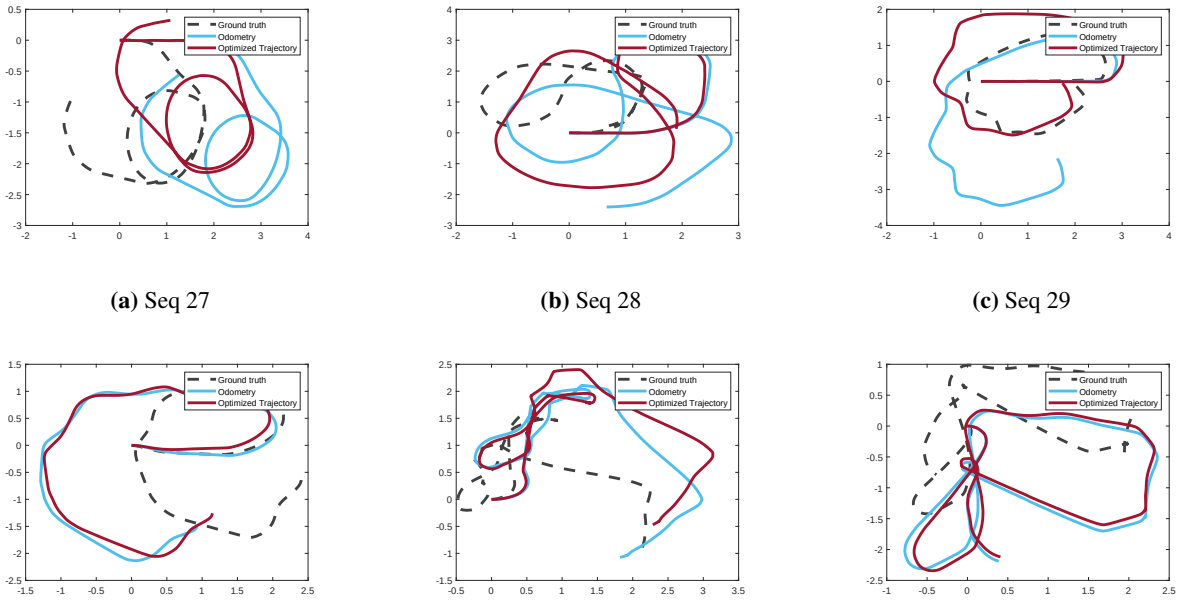
Within the scope of federated learning, we compare the performance of FTI-SLAM with 3 clients using different aggregation algorithms. In our experiment, the aggregation of weights in FedTrimmedAvg is done by excluding two fractions $\beta = 0.2$ of each corresponding weight in the clients' uploaded model. Since we have 3 clients, FedTrimmedAvg removes $round(3 \times 0.2) = round(0.6) = 1$ weight on both sides, and only 1 median weight is kept as the aggregated result.

Table 4 presents the quantitative metrics. For neural odometry, it is clear that FedAvg outperforms FedTrimmedAvg in 5 out of 6 sequences. Meanwhile, the improvement brought by applying optimisation benefits 3 out of 6 sequences for FedAvg, while only one sequence for FedTrimmedAvg. The reason for this great difference in performance is that FedAvg involves significantly more clients than FedTrimmedAvg. Though the starting point for FedTrimmedAvg is to exclude clients providing extreme gradients, using the model of only one client as the global model cannot be suitable for the situation for other clients, as the data involved is not representative enough.

Figure 5 shows the situation of trajectories. Compared to Figure 4, FedTrimmedAvg makes the optimised trajectory of Seq 29 moving from the odometry trajectory to the ground truth trajectory with a very similar shape, and the right part of Seq 30 also fits the ground truth better, as shown in Fig 5c and 5d respectively. An improvement in Seq 28 compared to FedAvg can also be witnessed, as the optimised trajectory tends closer to overlap more with the ground truth than the one in FedAvg. The situation in Seq 28, 29, and 30 reflects FedTrimmedAvg's advantage in improvements, as shown in 4. But for Sequences 21 and 22, the results for both odometry and odometry after optimisation are not fit for the ground truth,

Table 4. RMSE, RMSE_L, $\text{Var}_{\text{RMSE}_L}$, and the percentage of improvement by optimisation of sequences of FTI-SLAM with 3 clients using FedAvg and FedTrimmedAvg for aggregation.

Test Sequence	Setting	RMSE (m)	RMSE_L (m)	$\text{Var}_{\text{RMSE}_L}$	Improvement(%)
Seq 27	FedAvg 3 Clients	**0.5892**	**0.5173**	**0.0783**	**12.2061**
	FedTrimmedAvg 3 Clients	0.6727	0.8529	0.1898	-26.7886
Seq 28	FedAvg 3 Clients	**0.5162**	1.0668	0.5827	-106.6509
	FedTrimmedAvg 3 Clients	0.8188	**0.9608**	**0.2274**	**-17.3429**
Seq 29	FedAvg 3 Clients	**0.3697**	0.4817	**0.0549**	-30.2843
	FedTrimmedAvg 3 Clients	0.5749	**0.4477**	0.0804	**22.1394**
Seq 30	FedAvg 3 Clients	**0.3363**	**0.5070**	0.5070	-50.7308
	FedTrimmedAvg 3 Clients	0.5882	0.5484	**0.1009**	**6.7646**
Seq 21	FedAvg 3 Clients	0.4936	**0.4926**	**0.0686**	**0.2072**
	FedTrimmedAvg 3 Clients	**0.4256**	0.5379	0.2895	-26.3654
Seq 22	FedAvg 3 Clients	**0.4652**	**0.4453**	**0.0245**	**4.2830**
	FedTrimmedAvg 3 Clients	0.4956	0.6104	0.2363	-23.1533



(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29

(d) Seq 30 **(e)** Seq 21 **(f)** Seq 22

Figure 5. Trajectory optimisation results of FTI-SLAM with 3 clients using FedTrimmedAvg for aggregation.

which also reflects the impact of having too few clients in aggregation.

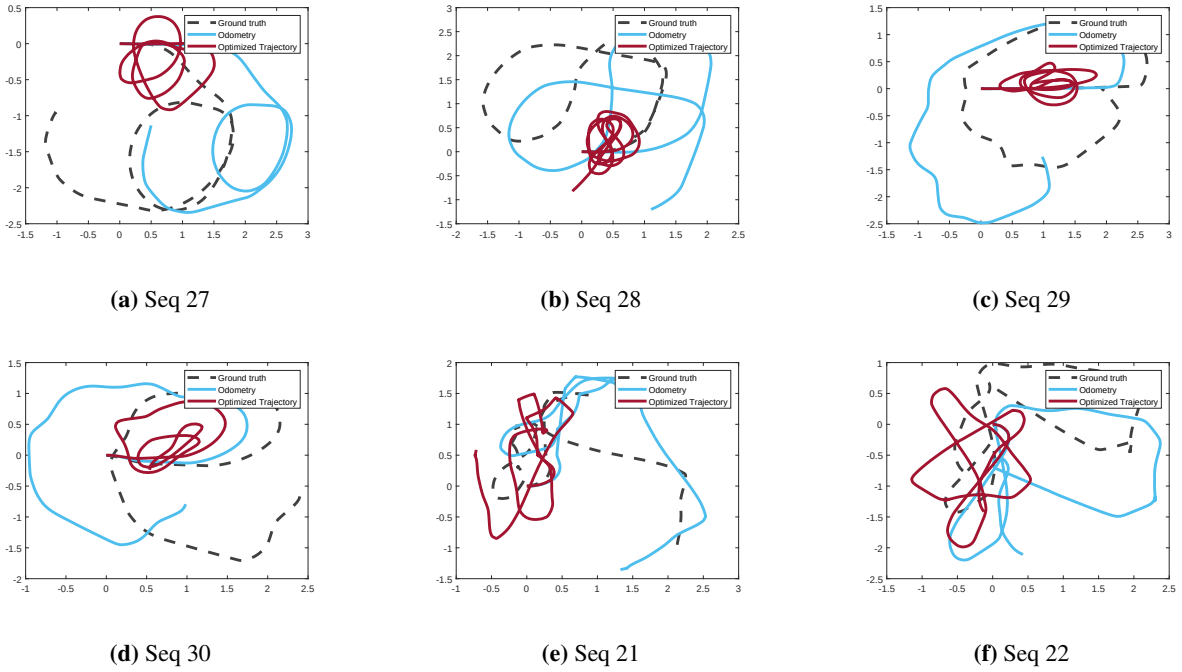
Based on these, the behaviours of FedAvg and FedTrimmedAvg on visualised results do not show a distinguishable gap. However, the quantitative metrics show the inferior position of FedTrimmedAvg in this case, especially in Sequences 21 and 22 which are more complicated. The advantage of FedTrimmedAvg is overridden by the lack of clients who can provide representative information about the actual dataset.

3.4. More clients with data overlapping

In this set of experiments, we increase the number of clients from 3 to 6 for both FTI-SLAM with FedAvg and with FedTrimmedAvg. By comparing Table 5 with Table 4 which is with 3 clients, it is clear that performance in terms of percentage of improvement drops significantly for both cases. Optimisation of odometry in these settings does not work properly.

Table 5. RMSE, RMSE_L, $\text{Var}_{\text{RMSE}_L}$, and the percentage of improvement by optimisation of sequences of FTI-SLAM with 6 clients using FedAvg and FedTrimmedAvg for aggregation.

Test Sequence	Setting	RMSE (m)	RMSE_L (m)	$\text{Var}_{\text{RMSE}_L}$	Improvement(%)
Seq 27	FedAvg 6 Clients	**0.6727**	0.1898	0.8529	-26.7886
	FedTrimmedAvg 6 Clients	0.9376	**0.1284**	**0.6704**	**28.5009**
Seq 28	FedAvg 6 Clients	**0.4751**	**0.1300**	**0.8480**	-78.4884
	FedTrimmedAvg 6 Clients	0.9035	0.4482	1.3535	**-49.8036**
Seq 29	FedAvg 6 Clients	**0.4328**	**0.0804**	**0.6652**	-53.6967
	FedTrimmedAvg 6 Clients	0.6298	0.2532	0.7572	**-20.2303**
Seq 30	FedAvg 6 Clients	0.5476	**0.1067**	0.6666	**-21.7351**
	FedTrimmedAvg 6 Clients	**0.4612**	0.1397	**0.6569**	-42.4469
Seq 21	FedAvg 6 Clients	**0.4944**	**0.5377**	**0.0677**	**-8.7723**
	FedTrimmedAvg 6 Clients	0.6224	0.9576	0.6603	-53.8650
Seq 22	FedAvg 6 Clients	0.5482	**0.5234**	0.4535	**0.0453**
	FedTrimmedAvg 6 Clients	**0.4628**	0.7495	**0.3475**	-61.9362



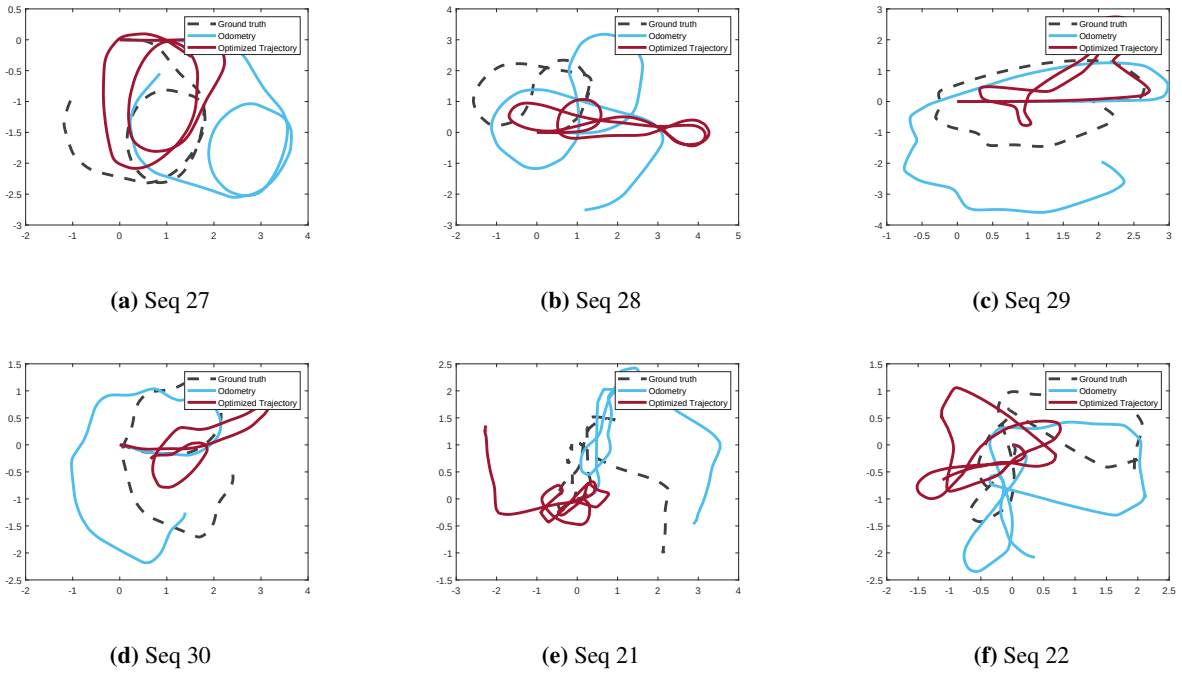
(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29

(d) Seq 30 **(e)** Seq 21 **(f)** Seq 22

Figure 6. Trajectory optimisation results of FTI-SLAM with 6 clients using FedAvg for aggregation.

For the optimised trajectories, the ones from FedAvg with 6 clients (Figure 6) show a similar situation as in Figure 3. The results of odometry alone are also not as satisfying. For FedTrimmedAvg with 6 clients, according to Figure 7, though the quality of optimised trajectories also degraded compared to the case with fewer clients, it is worth noticing that they are not showing the same pattern as FedAvg with 6 clients. First, for Seq 27, the performance actually improved compared to previous cases, and it is the best-optimised trajectory for Seq 27 across all experiments. Second, the ones for Seq 28, 29, and 30 are less strangled compared to that of FedAvg with 6 clients. In this setting, FedTrimmedAvg removes $round(6 \times 0.2) = round(1.2) = 1$ weight that is the largest and 1 smallest weights, keeping the remaining 4 for aggregation. For Sequences 21 and 22, with 6 clients involved in the framework, both FedAvg and FedTrimmedAvg present different behaviours compared to when there are only 3 clients, though quantitative metrics do not present a clear change trend. For other sequences, the optimised trajectories from FedTrimmedAvg present more reasonable patterns and more overlaps to the ground truth than those from FedAvg. The sub-optimal

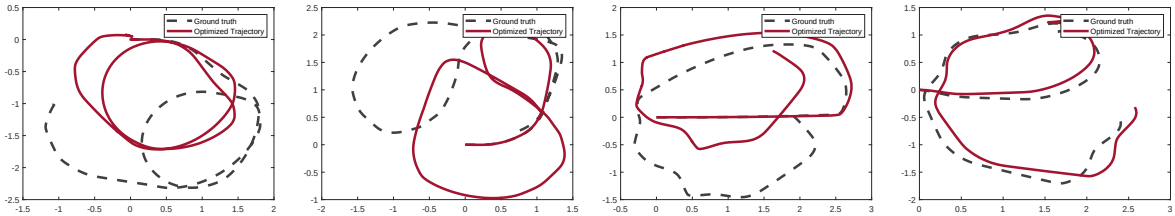
performances suggest that in situations when facing more complex attacks rather than just extreme values, FTI-SLAM would require more suitable aggregation algorithms to prevent malicious attempts and mitigate their negative impacts.



(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29

(d) Seq 30 **(e)** Seq 21 **(f)** Seq 22

Figure 7. Trajectory optimisation results of FTI-SLAM with 6 clients using FedTrimmedAvg for aggregation.



(a) Seq 27 **(b)** Seq 28 **(c)** Seq 29 **(d)** Seq 30

Figure 8. Trajectory optimisation results of FTI-SLAM with 3 clients with ground truth odometry and FedAvg for aggregation.

3.5. Impact of bias and noise in data

As discussed in TI-SLAM, both thermal data and IMU data are affected by noise [24]. For thermal imaging systems, the collected data is commonly affected by fixed-pattern noise [59]. For IMU data, the noises and biases can come from various sources, including but not limited to sensor configurations, white random noise, and pink noise [60]. In TI-SLAM, the researchers introduced deterministic the soft fusion structure for feature selection to mitigate the impact of bias and noises [24,30]. This is also used continuously in FTI-SLAM.

To demonstrate the impact of bias and noise in thermal and IMU data in FTI-SLAM, we compare the performance of the regular experiment of FTI-SLAM with three clients and FedAvg as the aggregation algorithm and use the odometry constraints and loop closure constraints from the front-end to obtain the optimised trajectories. For comparison, we use the ground truth data as odometry constraints and the loop closure constraints from FedAvg 3

clients setting to obtain the optimised trajectories without the affection noises from thermal images and IMU sensing.

The quantitative results are presented in Table 6, and the optimised trajectories based on ground truth odometry, which is not affected by the noises, are shown in Figure 8. It is clear that RMSE_L (m) and $\text{Var}_{\text{RMSE}_L}$ for optimised trajectories with ground truth trajectories are both lower, and for Sequence 30, the difference is significant. This is also reflected in the visualised trajectories. Compared to Figure 4, the obtained optimised trajectories are much closer to the ground truth trajectories, and Sequence 30 shows the best result. In this case, we can conclude that data noise and bias do have an impact on FTI-SLAM.

Table 6. RMSE_L and $\text{Var}_{\text{RMSE}_L}$ of the optimised odometry trajectories for FTI-SLAM aggregated with FedAvg with 3 clients, using estimated odometry and ground truth trajectory respectively.

Test Sequence	Setting	RMSE_L (m)	$\text{Var}_{\text{RMSE}_L}$
Seq 27	FedAvg 3 Client	0.5173	0.0783
	FedAvg 3 Client GT Odometry	**0.4337**	**0.0036**
Seq 28	FedAvg 3 Client	1.0668	0.5827
	FedAvg 3 Client GT Odometry	**0.6808**	**0.3369**
Seq 29	FedAvg 3 Client	0.4817	0.0549
	FedAvg 3 Client GT Odometry	**0.3537**	**0.0423**
Seq 30	FedAvg 3 Client	0.5070	0.5070
	FedAvg 3 Client GT Odometry	**0.1258**	**0.0035**

4. Conclusion

From the previous sets of experiments, we use limited amounts of data and computational resources to show the feasibility and effectiveness of FTI-SLAM. For robotics, the introduction of federated learning can provide efficient support for distributed training, allowing a local dataset to be properly utilised. In some federated learning settings, FTI-SLAM shows improved performance compared to the original system with the same training contents, which shows the better generalisation ability provided by federated learning. Future works on this perspective can focus on introducing local fine-tuning to make the device suitable for conducting tasks in its designated environment and maintaining robustness against less common situations and settings. We presented possible alternations to the initial federated learning setting, making the system not easily prone to potentially misleading training data. Future works can focus on applying and adjusting more robust aggregation algorithms, such as Bulyan [55], when conducting experiments with adversarial activities against global models. We also demonstrated the impact of data noise, and future works on FTI-SLAM could improve the current approach of feature selection or involve other techniques to further mitigate the negative impact of biases and noises. In addition, due to the limitation of our computational resources, our experiments involve up to 6 clients. Future experiments on this could involve more clients for simulations on a larger scale or more sources of data to explore the impact of data heterogeneity.

Conflicts of Interests

The authors declare that they have no conflict of interest.

Authors' contribution

Conceptualization, H.L. and H.Z.; Data Curation, H.Z. and H.L.; Formal Analysis, H.L. and H.Z.; Funding Acquisition, W.S.; Investigation, H.Z. and H.L.; Methodology, H.Z. and H.L.; Project Administration, H.L. and H.Z.; Resources, H.Z.; Software, H.Z. and H.L.; Supervision,

W.S.; Validation, H.L. and H.Z.; Visualization, H.L. and H.Z.; Writing – Original Draft, H.L. and H.Z.; Writing – Review & Editing, H.L., H.Z., and W.S. All authors have read and agreed to the published version of the manuscript.

References

[1] Taheri H, Xia ZC. SLAM; definition and evolution. *Eng. Appl. Artif. Intell.* 2021, 97:104032.

[2] Ekvall S, Jensfelt P, Kragic D. Integrating Active Mobile Robot Object Recognition and SLAM in Natural Environments. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2006 pp. 5792–5797. 10.1109/IROS.2006.282389.

[3] Cheng J, Zhang L, Chen Q, Hu X, Cai J. A review of visual SLAM methods for autonomous driving vehicles. *Engineering Applications of Artificial Intelligence* 2022 114:104992.

[4] Tseng PY, Lin JJ, Chan YC, Chen AY. Real-time indoor localization with visual SLAM for in-building emergency response. *Automation in Construction* 2022 140:104319.

[5] Ji S, Qin Z, Shan J, Lu M. Panoramic SLAM from a multiple fisheye camera rig. *ISPRS Journal of Photogrammetry and Remote Sensing* 2020 159:169–183.

[6] Zhang S, Zheng L, Tao W. Survey and evaluation of RGB-D SLAM. *IEEE Access* 2021 9:21367–21387.

[7] Franchi M, Ridolfi A, Pagliai M. A forward-looking SONAR and dynamic model-based AUV navigation strategy: Preliminary validation with FeelHippo AUV. *Ocean Engineering* 2020 196:106770.

[8] Huang L. Review on LiDAR-based SLAM techniques. In *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, IEEE2021 pp. 163–168.

[9] Mohamed SA, Haghbayan MH, Westerlund T, Heikkonen J, Tenhunen H, *et al.* A survey on odometry for autonomous navigation systems. *IEEE access* 2019 7:97466–97486.

[10] Hong Z, Petillot Y, Wallace A, Wang S. RadarSLAM: A robust simultaneous localization and mapping system for all weather conditions. *The international journal of robotics research* 2022 41(5):519–542.

[11] Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, *et al.* Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics* 2016 32(6):1309–1332.

[12] Jensfelt P, Kragic D, Folkesson J, Bjorkman M. A framework for vision based bearing only 3D SLAM. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, IEEE2006 pp. 1944–1950.

[13] Davison AJ, Reid ID, Molton ND, Stasse O. MonoSLAM: Real-time single camera SLAM. *IEEE transactions on pattern analysis and machine intelligence* 2007 29(6):1052–1067.

[14] Jiang X, Li T, Yu Y. A novel SLAM algorithm with Adaptive Kalman filter. In *2016 International Conference on Advanced Robotics and Mechatronics (ICARM)*, IEEE2016 pp. 107–111.

[15] Beghdadi A, Mallem M. A comprehensive overview of dynamic visual SLAM and deep learning: concepts, methods and challenges. *Machine Vision and Applications* 2022 33(4):54.

[16] Mokssit S, Licea DB, Guermah B, Ghogho M. Deep learning techniques for visual slam: A survey. *IEEE Access* 2023 11:20026–20050.

[17] Wang S, Clark R, Wen H, Trigoni N. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE international conference*

on robotics and automation (ICRA), IEEE2017 pp. 2043–2050.

[18] Clark R, Wang S, Wen H, Markham A, Trigoni N. Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem. In *Proceedings of the AAAI conference on artificial intelligence*. 2017, vol. 31 .

[19] Merrill N, Huang G. Lightweight unsupervised deep loop closure. *arXiv preprint arXiv:1805.07703* 2018 .

[20] Engelson SP, McDermott DV. Error correction in mobile robot map learning. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*, IEEE Computer Society1992 pp. 2555–2556.

[21] Kendall A, Grimes M, Cipolla R. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*. 2015 pp. 2938–2946.

[22] Li R, Wang S, Gu D. DeepSLAM: A robust monocular SLAM system with unsupervised deep learning. *IEEE Transactions on Industrial Electronics* 2020 68(4):3577–3587.

[23] An Y, Shi J, Gu D, Liu Q. Visual-LiDAR SLAM based on unsupervised multi-channel deep neural networks. *Cognitive Computation* 2022 14(4):1496–1508.

[24] Saputra MRU, Lu CX, de Gusmao PPB, Wang B, Markham A, *et al.* Graph-based thermal–inertial SLAM with probabilistic neural networks. *IEEE Transactions on Robotics* 2021 38(3):1875–1893.

[25] Kutila M, Pyykönen P, Holzhüter H, Colomb M, Duthon P. Automotive LiDAR performance verification in fog and rain. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018 pp. 1695–1701. 10.1109/ITSC.2018.8569624.

[26] Brenner M, Reyes NH, Susnjak T, Barczak AL. RGB-D and thermal sensor fusion: a systematic literature review. *IEEE Access* 2023 .

[27] Nguyen TXB, Rosser K, Chahl J. A review of modern thermal imaging sensor technology and applications for autonomous aerial navigation. *Journal of Imaging* 2021 7(10):217.

[28] Bavle H, Sanchez-Lopez JL, Cimarelli C, Tourani A, Voos H. From slam to situational awareness: Challenges and survey. *Sensors* 2023 23(10):4849.

[29] Saputra MRU, De Gusmao PP, Lu CX, Almalioglu Y, Rosa S, *et al.* Deeptio: A deep thermal-inertial odometry with visual hallucination. *IEEE Robotics and Automation Letters* 2020 5(2):1672–1679.

[30] Chen C, Rosa S, Miao Y, Lu CX, Wu W, *et al.* Selective sensor fusion for neural visual-inertial odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019 pp. 10542–10551.

[31] Hoffman J, Gupta S, Darrell T. Learning with side information through modality hallucination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016 pp. 826–834.

[32] Zhao S, Wang P, Zhang H, Fang Z, Scherer S. Tp-tio: A robust thermal-inertial odometry with deep thermalpoint. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE2020 pp. 4505–4512.

[33] Wang Y, Chen H, Liu Y, Zhang S. Edge-based monocular thermal-inertial odometry in visually degraded environments. *IEEE Robotics and Automation Letters* 2023 8(4):2078–2085.

[34] Dosovitskiy A, Fischer P, Ilg E, Hausser P, Hazirbas C, *et al.* Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 2015 pp. 2758–2766.

[35] Bishop CM. Mixture density networks, 1994.

[36] Ranganathan A. The levenberg-marquardt algorithm. *Tutoral on LM algorithm* 2004 11(1):101–110.

[37] McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, PMLR2017 pp. 1273–1282.

[38] Yin H, Qu L, Chen T, Yuan W, Zheng R, *et al.* On-device recommender systems: A comprehensive survey. *arXiv preprint arXiv:2401.11441* 2024 .

[39] Aledhari M, Razzak R, Parizi RM, Saeed F. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access* 2020 8:140699–140725.

[40] Zhang C, Xie Y, Bai H, Yu B, Li W, *et al.* A survey on federated learning. *Knowledge-Based Systems* 2021 216:106775.

[41] Chik WB. The Singapore Personal Data Protection Act and an assessment of future trends in data privacy reform. *Computer Law & Security Review* 2013 29(5):554–575.

[42] Albrecht JP. How the GDPR will change the world. *Eur. Data Prot. L. Rev.* 2016 2:287.

[43] Ma X, Zhu J, Lin Z, Chen S, Qin Y. A state-of-the-art survey on solving non-iid data in federated learning. *Future Generation Computer Systems* 2022 135:244–258.

[44] Collins L, Hassani H, Mokhtari A, Shakkottai S. Fedavg with fine tuning: Local updates lead to representation learning. *Advances in Neural Information Processing Systems* 2022 35:10572–10586.

[45] Zhang J, Hua Y, Wang H, Song T, Xue Z, *et al.* Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 2023, vol. 37 pp. 11237–11244.

[46] Yeganeh Y, Farshad A, Navab N, Albarqouni S. Inverse distance aggregation for federated learning with non-iid data. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning: Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4–8, 2020, Proceedings 2*, Springer2020 pp. 150–159.

[47] Nguyen A, Do T, Tran M, Nguyen BX, Duong C, *et al.* Deep federated learning for autonomous driving. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, IEEE2022 pp. 1824–1830.

[48] Imteaj A, Amini MH. Fedar: Activity and resource-aware federated learning model for distributed mobile robots. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE2020 pp. 1153–1160.

[49] Li Z, Wang L, Jiang L, Xu CZ. FC-SLAM: Federated learning enhanced distributed visual-LiDAR SLAM in cloud robotic system. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE2019 pp. 1995–2000.

[50] Zhang H, Yang Z, Tian Y, Zhang H, Di B, *et al.* Reconfigurable holographic surface aided collaborative wireless SLAM using federated learning for autonomous driving. *IEEE Transactions on Intelligent Vehicles* 2023 8(8):4031–4046.

[51] Beutel DJ, Topal T, Mathur A, Qiu X, Fernandez-Marques J, *et al.* Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390* 2020 .

[52] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Software available from tensorflow.org.

[53] Chollet F, *et al.* Keras. https://keras.io, 2015.

[54] Chen J, Pan X, Monga R, Bengio S, Jozefowicz R. Revisiting distributed synchronous SGD. *arXiv preprint arXiv:1604.00981* 2016 .

[55] Guerraoui R, Rouault S, *et al.* The hidden vulnerability of distributed learning

in byzantium. In *International Conference on Machine Learning*, PMLR2018 pp. 3521–3530.

[56] Yin D, Chen Y, Kannan R, Bartlett P. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International conference on machine learning*, Pmlr2018 pp. 5650–5659.

[57] Lyu L, Yu H, Yang Q. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133* 2020 .

[58] Lamport L, Shostak R, Pease M. The Byzantine generals problem. In *Concurrency: the works of leslie lamport*, Association for Computing Machinery2019, pp. 203–226.

[59] Williams R, Parrish WJ, Wolfe J. Fixed pattern noise mitigation for a thermal imaging system, 2019. US Patent 10,230,912.

[60] Nirmal K, Sreejith A, Mathew J, Sarpotdar M, Suresh A, *et al.* Noise modeling and analysis of an IMU-based attitude sensor: improvement of performance by filtering and sensor fusion. In *Advances in optical and mechanical technologies for telescopes and instrumentation II*, SPIE2016, vol. 9912 pp. 2138–2147.