MTC-SLAM: Multi-scale Tightly Coupled LiDAR-Inertial SLAM Method for Complex Environments with Dynamic Objects and Similar Features

Qifeng Wang, *Student Member, IEEE*, Weigang Li, *Member, IEEE*, Lei Nie, *Member, IEEE*, Wenping Liu, *Senior Member, IEEE*, Hongbo Jiang, *Senior Member, IEEE*

***Abstract*—imultaneous Localization and Mappingimultaneous Localization and MappingS (SLAM) is critical for autonomous driving. Compared with visual SLAM which suffers from variable lighting conditions and sparse features, Light Detection and Ranging (LiDAR) based solution is highly competitive since LiDAR can offer precise distance measurements and robust resistance to changes in illumination. However, current LiDAR SLAM techniques only estimate pose from a single scale, such that complex environments with dynamic objects and similar features can heavily affect the performance, leading to poor accuracy and robustness of pose estimation. To address these challenges, we present MTC-SLAM, a multi-scale and tightly coupled LiDAR-Inertial SLAM method. It begins with a coarse adjustment across six degrees of freedom at a larger scale, followed by adaptive step-size least squares updates for x, y, and yaw on a smaller scale, and dynamic interpolation updates for z, $pitch$, and $roll$. The updated poses are integrated as odometer factors, alongside Global Positioning System (GPS) and loop closure factors, to construct a factor graph for backend optimization. Additionally, a multi-scale loop closure detection method is proposed to identify loops using multi-scale descriptors for global optimization. Extensive experiments on the MulRan dataset demonstrate that MTC-SLAM outperforms existing mainstream methods in both accuracy and robustness.**

***Index Terms*—iDAR-InertialiDAR-InertialL SLAM, complex environments, multi-scale, loop closure, autonomous driving**

I. Introduction

In recent years, Simultaneous Localization and Mapping (SLAM) technology has emerged as a pivotal component in the field of robotics and automation, particularly in propelling autonomous vehicles and other mobile intelligent agents toward full autonomy [1]. The principal challenge of SLAM involves the self-localization of mobile agents in unfamiliar environments coupled with the simultaneous mapping of these surroundings [2]. This dual capability is crucial for the deployment of fully autonomous robotic systems, enabling robots and autonomous vehicles to navigate and recognize environments autonomously, without external input [3].

SLAM technology can be primarily categorized into two types based on the employed sensors, i.e., visual SLAM and Light Detection and Ranging (LiDAR) SLAM. Visual SLAM [4], [5] utilizes cameras to perceive the environment and compute the robot's motion trajectory by analyzing variations in feature points across successive image frames, and thus builds a three-dimensional representation of the underlying environment [6], [7]. The affordability, compact size, and ease of integration of cameras render visual SLAM advantageous for cost-sensitive applications [8]. Nevertheless, its accuracy is significantly influenced by lighting conditions and the sparsity of visual features, impacting the precision of localization and the fidelity of map creation. Conversely, LiDAR SLAM uses LiDAR scanners to gauge distances to nearby objects and leverages point cloud data for accurate localization and environmental mapping [9], [10]. LiDAR sensors, known for their precision in distance measurement and insensitivity to lighting variations, ensure high stability and reliability in LiDAR SLAM systems. As such, LiDAR SLAM has become integral to the operation of autonomous cars, drones, and various mobile robotic platforms[11].

Despite the effectiveness of current LiDAR SLAM methods in reducing errors and enhancing system robustness, their performance and reliability encounter limitations in complex environments with dynamic objects and similar features [12], [13]. During SLAM, inaccuracies in any state quantity and the accumulation of errors reduce the accuracy of pose estimation and degrade the quality of the map. Consequently, updating the six degrees of freedom state from a single scale inevitably introduces deviations, affecting the system's reliability in real autonomous driving scenarios.

To address these shortcomings, this paper introduces MTC-SLAM, a multi-scale, tightly coupled LiDAR-Inertial SLAM method tailored for complex environments with dynamic objects and similar features. MTC-SLAM initiates with a coarse adjustment of the six degrees of freedom at a larger scale, followed by adaptive step-size least squares iterative

This work was partially supported by the National Natural Science Foundation of China under Grant (No.51774219), and the key R&D Program of Hubei Province (No.2020BAB098). Numerical calculation is supported by High-Performance Computing Center of Wuhan University of Science and Technology. *(Corresponding author: Weigang Li.)*

Qifeng Wang and Weigang Li are with the School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan, China.

Lei Nie is with the School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, China.

Wenping Liu is with the School of Information Management and Institute of Big Data and Digital Economy, Hubei University of Economics, Wuhan, China.

Hongbo Jiang is with the College of Computer Science and Electroni Engineering, Hunan University, Changsha, China.

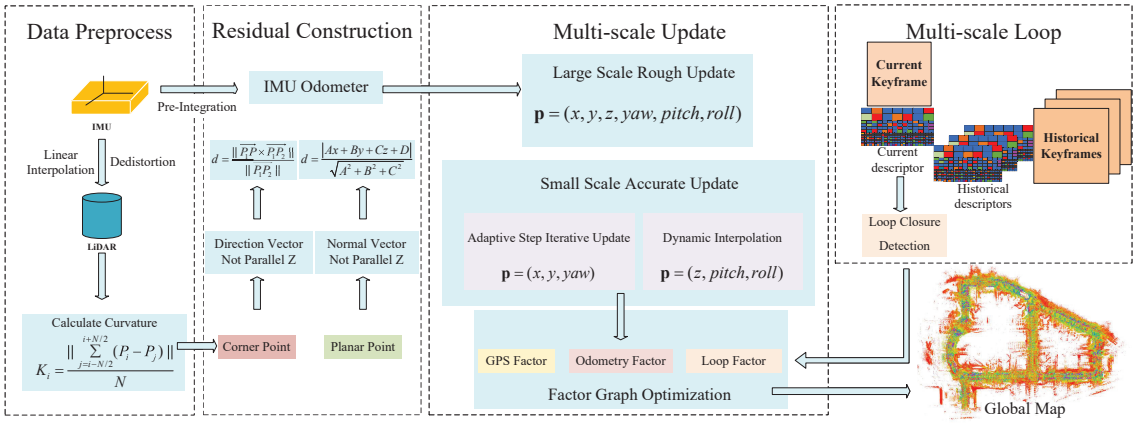

Fig. 1. Overview of MTC-SLAM. The diagram includes data preprocess, residual construction, multi-scale update, and multi-scale loop.

refinements for the x, y, and yaw at a smaller scale, and dynamic interpolation for the z, $pitch$, and $roll$. The updated poses are then integrated with Global Positioning System (GPS) and loop closure factors to construct a factor graph and complete backend optimization. In Additional, we propose a multi-scale loop detection method that reduces cumulative errors by employing multi-scale descriptors derived from point cloud geometry and intensity to identify loops during the SLAM process, and the global optimization can thus be achieved. The framework of MTC-SLAM is depicted in Fig. 1. Experimental results indicate that this method outperforms the existing mainstream LiDAR-Inertial SLAM methods in accuracy and robustness of pose estimation.

The main contributions of our work are summarized as follows:

- We propose a multi-scale tightly coupled LiDAR-Inertial SLAM method. It updates the six degrees of freedom at a larger scale and performs adaptive step-size least squares iterative updates for the x, y, and yaw at a smaller scale, as well as dynamic interpolation updates for the z, $pitch$, and $roll$, achieving more accurate odometer poses.
- We design a multi-scale loop detection method, which constructs multi-scale descriptors based on the geometric and intensity information of point clouds, detects loops during the SLAM process, and completes global optimization.
- We validate the effectiveness of our MTC-SLAM method in real-world scenarios, and compare with contemporary mainstream LiDAR-Inertial SLAM methods using the public MulRan dataset [14]. The experimental results demonstrate that MTC-SLAM outperforms existing mainstream methods in both accuracy and robustness.

II. Related Work

LiDAR technology plays a crucial role in autonomous vehicles, drones, and other robotic systems due to its ability to provide accurate distance measurements and high-resolution three-dimensional images, especially in visually limited environments [15], [16].

LiDAR Odometry (LO) generally involves some form of scan-to-scan or scan-to-local map registration, initially based on modified registration algorithms such as Iterative Closest Point (ICP) [17] and Normal Distributions Transform (NDT) [18]. A significant milestone in LiDAR SLAM research was the Lidar Odometry and Mapping (LOAM) algorithm proposed by Dr. Zhang from Carnegie Mellon University in 2014 [19]. The LOAM algorithm efficiently extracts linear and planar features from point cloud data for registration, significantly reducing computational complexity. Its innovative design decomposes motion estimation into high-frequency, low-accuracy odometry measurements and low-frequency, high-accuracy map matching, effectively balancing system accuracy and real-time performance. The LeGO-LOAM algorithm [20], introduced by Shan et al. in 2018, integrates ground segmentation and object clustering techniques to further enhance feature extraction accuracy and employs the iSAM2 [21] loop closure module to reduce long-term cumulative errors, significantly improving the stability and computational efficiency of the SLAM system [22]. While LO relies on point cloud registration algorithms to localize the current frame's global position accurately, challenges such as LiDAR data sparsity and motion interference, particularly in structurally sparse scenes, can lead to motion estimation degradation and system performance decline [23].

To address the limitations of single LiDAR systems in dynamic and sparse environments, researchers have explored LiDAR-Inertial SLAM systems that incorporate Inertial Measurement Unit (IMU) data, categorized into loosely-coupled and tightly-coupled forms [24], [25]. Loosely-coupled systems process LiDAR and IMU data separately for motion correction and estimation, featuring lower computational complexity and a simplified system architecture. Zhen and colleagues have effectively utilized error state Kalman filtering and Gaussian particle filtering methods to integrate IMU and LiDAR scanner data [26]. In 2021, Tagliabue et al. introduced the LION algorithm [27], which employs a fixed-lag smoother to fuse high-frequency IMU data with low-frequency LiDAR pose estimates, achieving precise high-frequency motion estimation. Although implementing loosely-coupled systems is relatively

straightforward and demands less computation, their performance can be significantly constrained by sensor costs and environmental conditions.

To enhance the robustness of SLAM systems, researchers have shifted their focus toward tightly-coupled methods for LiDAR-Inertial SLAM. Tightly-coupled systems offer more reliable pose estimation through the joint optimization of all sensor data within a unified framework. Following the LeGO-LOAM, Shan's subsequent study proposed the LIO-SAM algorithm [28], which uses factor graph optimization techniques to integrate data from multiple sensors for more accurate motion estimation. However, its feature extraction is dependent on geometric environments and may suffer from motion degradation in open scenes. Xu et al. proposed the FAST-LIO algorithm [29], a robust method that fuses LiDAR and IMU data using the Error State Iterative Kalman Filter (ESIKF) and corrects motion distortion in LiDAR point clouds via backward propagation. Subsequently, Xu and his team developed FAST-LIO2 [30], addressing computational growth issues with the new IKDTree data structure. This innovation allows for the direct processing of raw point cloud data without feature extraction, leading to faster processing speeds and enhanced accuracy. However, the system overlooks the impact of historical data on the current state, fails to perform global corrections, and struggles in complex environments due to its single-scale approach.

In addition to LiDAR-IMU integration, recent research has increasingly focused on fusing LiDAR and camera data to exploit their complementary strengths—precise geometric structure from LiDAR and rich texture from images. These multisensor approaches aim to enhance SLAM robustness, especially in complex environments. For instance, Zheng et al. proposed FAST-LIVO [8], a sparse-direct, tightly-coupled LiDAR-inertial-visual odometry system that improves real-time performance by directly utilizing image and point cloud data without feature extraction. Liu et al. [31] further developed a vision–LiDAR–inertial–GPS fusion framework with automatic calibration and factor graph optimization, achieving enhanced robustness and semantic mapping capabilities.

Loop closure detection has also received increasing attention as a key element for ensuring global consistency in SLAM. RING++ [32] introduces a rotation- and translation-invariant descriptor for global localization using sparse scan maps, with theoretical guarantees and scalability to multi-robot scenarios. Zou et al. proposed iBTC [33], a multimodal descriptor that fuses camera and LiDAR data via triangle-based geometric constraints, demonstrating strong performance under challenging conditions. Several effective and widely adopted LiDAR-based loop detection methods have also emerged. Notably, Scan Context (SC) [34], Intensity Scan Context (ISC) [35], and Cross-section Shape Context (CSSC) [36] represent the mainstream. SC encodes 3D spatial information into a compact 2D polar representation for efficient place recognition; ISC enhances robustness in appearance-changing environments by incorporating intensity cues; and CSSC projects point clouds onto cross-sectional views to build shape context descriptors, enabling accurate one-shot global localization.

Considering the dynamics and complexity of real-world autonomous driving environments, this paper proposes a multi-scale tightly-coupled LiDAR-Inertial SLAM method designed for complex environments with dynamic objects and similar features.

III. Our approach

We introduce a multi-scale, tightly coupled LiDAR-Inertial SLAM method designed for complex environments with dynamic objects and similar features. Initially, this method corrects LiDAR data distortion by using linear interpolation of IMU data, then obtains initial odometry estimates from IMU pre-integration. This is followed by feature extraction from edge and planar points to construct residuals. The method then updates the six degrees of freedom of the pose on a larger scale and iteratively refines the x, y, and yaw degrees using adaptive step-size least squares on a smaller scale, with dynamic interpolation for the z, $pitch$, and $roll$ degrees. The refined pose is integrated as an odometry factor along with GPS factors and loop closure factors to construct a factor graph and achieve backend optimization. Furthermore, we present a multi-scale loop detection method that uses multi-scale descriptors to identify loops during SLAM operations, effectively eliminating cumulative errors and enhancing global optimization.

A. Data Preprocess

In the SLAM systems of mobile robots and autonomous vehicles, LiDAR and IMU are two indispensable sensors. The IMU measures acceleration and angular velocity at high frequencies, aiding in the motion estimation of the device. However, directly integrating this high-frequency data to estimate the device's pose entails significant computational demands and management challenges. To mitigate this, IMU pre-integration is utilized to accumulate measurements over shorter intervals, effectively reducing the computational load for each pose estimation and providing initial IMU odometry [28].

Since LIDAR typically collects data at a lower frequency than the IMU, which provides data at a higher frequency, directly using LIDAR data obtained during rapid movement of the robot or vehicle can result in motion-induced distortions [29]. To correct these distortions, linear interpolation of the IMU data is used to rectify the LIDAR data.

After correcting distortions in the LiDAR data, the curvature of each point is calculated. Points are then classified as either corner points or plane points based on their curvature magnitudes. For each point P_i in the point cloud, curvature K_i is computed using its N nearest neighbors $(P_{i-N/2}, ...P_{i-1}, P_{i+1}, ..., P_{i+N/2})$. Points with a curvature exceeding threshold T_{corner} are classified as corner points, whereas those with a curvature below another threshold T_{plane} are identified as plane points.

$$K_i = \frac{\| \sum_{j=i-N/2}^{i+N/2} (P_i - P_j) \|}{N} \tag{1}$$

B. Residual Construction

After extracting corner points and plane points, these features are used to construct residuals. For a current corner point $P = (x, y, z)$, two points, $P_1 = (x_1, y_1, z_1)$ and $P_2 = (x_2, y_2, z_2)$, are located on a line located from neighboring corner points in the local map. The shortest distance from P to the line is calculated as the residual d.

$$d = \frac{\| \overrightarrow{P_1P} \times \overrightarrow{P_1P_2} \|}{\| \overrightarrow{P_1P_2} \|} \tag{2}$$

Similarly, for the current plane point $P = (x, y, z)$, the plane equation derived from the neighboring corner points in the local map can be expressed as $Ax + By + Cz + D = 0$, where A, B, C is the normal vector of the plane, and D is the constant term. The perpendicular distance from the point to the plane is calculated as the residual d.

$$d = \frac{|Ax + By + Cz + D|}{\sqrt{A^2 + B^2 + C^2}} \tag{3}$$

Given the low vertical resolution of LiDAR point clouds, which results in fewer feature points and increases susceptibility to errors, and considering that points at various heights produce consistent residuals relative to lines or planes parallel to the z-axis, special considerations are necessary. For corner points, residuals are not constructed if the direction vectors of the fitted lines have a nonzero z-component while their x- and y-components are significantly smaller than the z-component. For plane points, if the normal vector of the plane, fitted from these points, is predominantly horizontal, the plane fails to provide adequate vertical constraints. Consequently, such plane points are excluded from further consideration.

C. Multi-scale Update

The objective of the multiscale pose update is to estimate the pose $\mathbf{p} = (x, y, z, yaw, pitch, roll)$ that minimizes the overall point-to-feature residuals. The global optimization problem can be formulated as:

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \sum_i r_i^2(\mathbf{p}) \tag{4}$$

where $r_i(\mathbf{p})$ denotes the residual computed from geometric constraints (e.g., point-to-line or point-to-plane distances). Directly solving this full 6-DoF nonlinear least squares problem at each frame may introduce unnecessary computational cost and increase sensitivity to dynamic noise. Therefore, we adopt a hierarchical strategy that splits the update process into two stages. The first stage performs coarse full-DoF optimization to ensure global consistency, while the second stage focuses on refining dominant motion components using iterative least squares and estimating the remaining degrees via dynamic interpolation.

The pose update process consists of an initial coarse adjustment of the six degrees of freedom on a larger scale, followed by a fine-tuning of selected degrees of freedom on a smaller scale, as detailed in Algorithm 1. The primary steps include:

(1) In the large-scale pose update, the pose is updated by minimizing the sum of squared residuals $\min_{\mathbf{p}} \sum_i r_i(\mathbf{p})^2$. $r_i(\mathbf{p})$ is the residual of the pose $\mathbf{p} = (x, y, z, yaw, pitch, roll)$ of the ith point. The pose is updated using an iterative method similar to that used in the smaller scale updates for the x, y, and yaw degrees, progressively approximating the solution that minimizes the sum of squared residuals, thereby achieving a coarse update of the pose on a large scale.

Algorithm 1: Multiscale Update Algorithm

Input: Initial pose $\mathbf{p} = (x, y, z, yaw, pitch, roll)$
Output: Updated pose $\mathbf{p}^{new}$

// Larger scale :
for *each point* i **do**
  Calculate residual $r_i(\mathbf{p})$
  update $(x, y, z, yaw, pitch, roll)$
end
// Smaller scale:
for *selected (x, y, yaw)* **do**
  Compute Jacobian $\mathbf{J}_i(\mathbf{p}) = \left[\frac{\partial r_i}{\partial x}, \frac{\partial r_i}{\partial y}, \frac{\partial r_i}{\partial yaw}\right]$;
  Solve optimization for $\Delta\mathbf{p}$:

$$\left(\sum_i \mathbf{J}_i(\mathbf{p})^\top \mathbf{J}_i(\mathbf{p}) + \lambda I\right) \Delta\mathbf{p} = -\sum_i \mathbf{J}_i(\mathbf{p})^\top r_i(\mathbf{p})$$

  Update pose $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$
  Check residuals and adjust λ
  if *convergence* **then**
    break
  end
end
for *selected (pitch, roll, z)* **do**
  // Calculate the $pitch$ and $roll$ differences
  $\Delta pitch = pitch_{curr} - pitch_{prev}$
  $\Delta roll = roll_{curr} - roll_{prev}$
  // Adjust the interpolation factor
  $t_{pitch} = \max\left(t_{\min}, \min\left(t_{\max}, a_{pitch} - b_{pitch} \cdot \left|\frac{\Delta pitch}{\pi}\right|\right)\right)$
  $t_{roll} = \max\left(t_{\min}, \min\left(t_{\max}, a_{roll} - b_{roll} \cdot \left|\frac{\Delta roll}{\pi}\right|\right)\right)$
  $t_z = \max\left(t_{\min}, \min\left(t_{\max}, a_z - b_z \cdot \left|\frac{z_{\text{curr}} - z_{\text{prev}}}{Z_{\max}}\right|\right)\right)$
end
Return
$\mathbf{p}^{new} = (x^{new}, y^{new}, z^{new}, yaw^{new}, pitch^{new}, roll^{new})$

(2) We observe that in ground-based LiDAR SLAM scenarios, the x, y, and yaw components dominate the motion and contribute most to accumulated drift. These are more sensitive to optimization accuracy and thus are selected for iterative refinement in the fine-scale stage. In contrast, z, $pitch$, and $roll$ tend to vary more smoothly and are often less constrained due to the sparsity of vertical features. Optimizing these directly may lead to instability or overfitting in dynamic environments. To balance efficiency and robustness, we choose to interpolate these components using temporally adjacent frames. This design reduces the parameter space of the nonlinear system while preserving continuity and stability.

In the small-scale update stage, a fine adaptive step-size least squares iterative update is first carried out on the x, y, and yaw degrees. The pose vector is defined as $\mathbf{p} = (x, y, yaw)$. For each point's residual r_i, the Jacobian matrix $J_i(\mathbf{p})$ is calculated.

$$J_i(\mathbf{p}) = \left[\ \frac{\partial r_i}{\partial x}, \frac{\partial r_i}{\partial y}, \frac{\partial r_i}{\partial yaw}\ \right] \tag{5}$$

(3) After deriving the Jacobian matrix, the optimization equation is specified as Equation (5). To enhance convergence

speed and improve solution accuracy, we introduce a tuning factor λ, which can dynamically adjust the step size based on the current iteration, influencing the computation of the update vector $\Delta\mathbf{p}$, where I represents the identity matrix.

$$\left(\sum_i J_i(\mathbf{p})^\top J_i(\mathbf{p}) + \lambda I\right)\Delta\mathbf{p} = -\sum_i J_i(\mathbf{p})^\top r_i(\mathbf{p}) \tag{6}$$

(4)The solution of the optimization equation yields $\Delta\mathbf{p} = (\Delta x, \Delta y, \Delta yaw)$, representing the pose update amount for the current iteration. The current pose is then updated to $\mathbf{p}^{(\text{new})} = (x^{(\text{new})} + \Delta x, y^{(\text{new})} + \Delta y, yaw^{(\text{new})} + \Delta yaw)$ The total residual after the update is calculated and compared to the previous iteration. If the residual has decreased significantly, λ is reduced; if the residual has increased or the decrease is minimal, λ is increased. If the predetermined convergence criteria are reached, the iteration stops; otherwise, the iteration updates continue.

(5)For the $pitch$ and $roll$ degrees, we define $Q_{\text{prev}} = [q_{0\text{prev}}, \vec{q}_{\text{prev}}]$ and $Q_{\text{curr}} = [q_{0\text{curr}}, \vec{q}_{\text{curr}}]$ as the quaternions representing the poses of the previous and current frames, respectively. Here, q_0 is the real part of the quaternion, $\vec{q} = [q_1, q_2, q_3]$ constitutes the imaginary part. We then convert these quaternions to Euler angles, with $\text{roll} = \text{atan2}\left(2(q_0q_1 + q_2q_3), 1 - 2(q_1^2 + q_2^2)\right)$ and $\text{pitch} = \text{asin}\left(2(q_0q_2 - q_3q_1)\right)$. Subsequently, we calculate the differences in $pitch$ and $roll$ between the current and previous frames, denoted as Δpitch and Δroll.

$$\begin{aligned}\Delta pitch &= pitch_{curr} - pitch_{prev}\\ \Delta roll &= roll_{curr} - roll_{prev}\end{aligned} \tag{7}$$

(6) Based on the size of Δpitch and Δroll, we dynamically adjust the interpolation factor t, constrained within the limits $t_{\min}$ and $t_{\max}$, with a and b serving as tuning factors to modulate the interpolation speed. When the difference in pitch or roll between consecutive frames is large, the interpolation factor t is reduced. This results in a more cautious transition, preventing abrupt changes that could introduce instability. The parameter a represents the base interpolation factor in the absence of significant pitch or roll variations, while the parameter b determines how strongly the interpolation factor is affected by $\Delta pitch$ and $\Delta roll$. A larger b results in a more aggressive reduction of t when significant variations occur. In addition, the interpolation factor t is constrained within the bounds $t_{\min}$ and $t_{\max}$ to ensure numerical stability and maintain reasonable update behavior. The lower bound $t_{\min}$ prevents the interpolation from becoming excessively conservative, which could hinder system responsiveness, while the upper bound $t_{\max}$ avoids over-reacting to small pose changes and ensures smooth updates during steady motion. In our experiments, we empirically set: $a_{pitch} = a_{roll} = 0.9$, $b_{pitch} = b_{roll} = 0.5$, $t_{\min} = 0.3$, $t_{\max} = 1.0$.

$$\begin{aligned}t_{pitch} &= \max\left(t_{\min}, \min\left(t_{\max}, a_{pitch} - b_{pitch}\cdot\left|\frac{\Delta pitch}{\pi}\right|\right)\right)\\ t_{roll} &= \max\left(t_{\min}, \min\left(t_{\max}, a_{roll} - b_{roll}\cdot\left|\frac{\Delta roll}{\pi}\right|\right)\right)\end{aligned} \tag{8}$$

Finally, we utilize the adjusted interpolation factors t_{pitch} and t_{roll} to interpolate the $pitch$ and $roll$ angles accordingly.

$$\begin{aligned}pitch_{\text{interp}} &= pitch_{prev} + t_{pitch}\cdot\Delta pitch\\ roll_{\text{interp}} &= roll_{prev} + t_{roll}\cdot\Delta roll\end{aligned} \tag{9}$$

(7) For the z degree, dynamic linear interpolation is utilized, employing t_z, where z_{prev} and z_{curr} denote the z-values of the previous and current frames, respectively, $Z_{\max}$ represents the predefined maximum z-axis movement distance, which normalizes the difference. The range of t_z is constrained by $t_{\min}$ and $t_{\max}$, while a and b serve as adjustment factors to fine-tune the interpolation process.

$$t_z = \max\left(t_{\min}, \min\left(t_{\max}, a_z - b_z\cdot\left|\frac{z_{\text{curr}} - z_{\text{prev}}}{Z_{\max}}\right|\right)\right) \tag{10}$$

(8) After completing the iterative updates for the x, y, and yaw degrees on a smaller scale and the dynamic interpolations for the z, $pitch$, and $roll$ degrees, a more accurate pose is obtained. Then the updated attitude is used as odometer factor, and the factor map is constructed together with GPS factor and closed-loop factor, and the final attitude is obtained by completing back-end optimization. This hierarchical optimization-interpolation scheme ensures that the update process remains both efficient and robust, while maintaining accuracy in the most critical degrees of freedom.

D. Multi-scale Loop

As robots navigate their environments, errors in pose estimation tend to accumulate, resulting in map distortions. Loop closure detection, by matching current observations with historical data, effectively corrects these accumulated positional errors and redistributes them across the entire map, facilitating the creation of a globally consistent environmental map [37]. The stability and reliability of loop closure detection are especially critical in complex and dynamically changing environments [38][34]. Consequently, this paper introduces a multiscale loop closure detection method that utilizes both geometric and intensity information from point clouds to develop multiscale descriptors (illustrated in Fig. 2). Using these descriptors, the current frame is compared with historical frames, and the frame that best matches is identified as the loop closure candidate frame.

The main process of multi-scale loop closure detection is as follows:

(1) Each point $p_i = [x_i, y_i, z_i, i_i](0 < i \le n)$ from a single LiDAR scan into the polar coordinate system. Within this system, the point cloud is segmented into multiple annular regions. Each ring encapsulates spatial and intensity information specific to a certain distance range. Each ring is further divided into several sectors, with each sector containing point cloud data within a specific angular range.

$$(r_i, \theta_i) = \left(\sqrt{x_i^2 + y_i^2}, \text{atan2}(y_i, x_i)\right) \tag{11}$$

(2) Multi-scale height descriptor (MH) characterizes environmental features at various scales by computing the Scan Context at differing spatial resolutions. For each scale s, we define a Scan Context with a different number of sectors M_s and rings N_s, along with a corresponding maximum distance $R_{\max,s}$. Here, $r_{\min}^{n,s}$ and $r_{\max}^{n,s}$, as well as $\theta_{\min}^{m,s}$ and $\theta_{\max}^{m,s}$, denote the inner and outer radii of the nth ring and the angular range

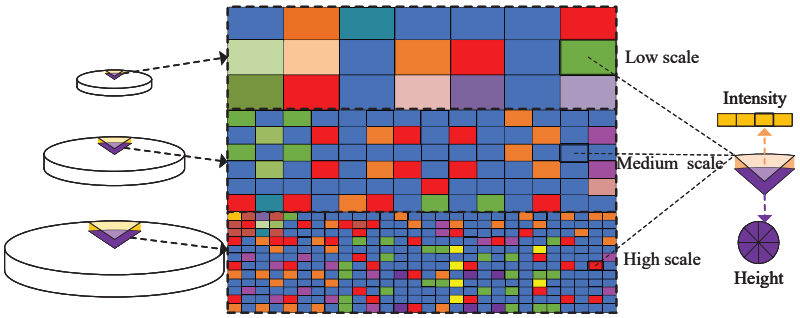

Fig. 2. Multi-scale descriptor

of the mth sector at scale s, while z_i indicates the height value of point p_i.

$$MH^s(m,n) = \max_{\substack{r_{\min}^{n,s} \le r_i < r_{\max}^{n,s} \\ \theta_{\min}^{m,s} \le \theta_i < \theta_{\max}^{m,s}}} z_i \tag{12}$$

Similarly, we develop a multi-scale intensity descriptor (MI) for capturing the intensity information of point clouds. For each scale s, the calculations are performed according to formula (12), where i_i represents the intensity value of point p_i.

$$MI^s(m,n) = \max_{\substack{r_{\min}^{n,s} \le r_i < r_{\max}^{n,s} \\ \theta_{\min}^{m,s} \le \theta_i < \theta_{\max}^{m,s}}} i_i \tag{13}$$

In the case of the multi-scale height descriptors MH_1^s and MH_2^s from two different point cloud frames, their difference, denoted as D_{MH} is determined by calculating the cumulative difference across all scales. Here, w_s is a weight coefficient for each scale s. In our implementation, all scales are treated equally with $w_s = 1$.

$$D_{MH}(MH_1^s, MH_2^s) = \sum_s w_s \cdot \sqrt{\sum_{m=1}^{M_s}\sum_{n=1}^{N_s}(MH_1^s(m,n) - MH_2^s(m,n))^2} \tag{14}$$

Likewise, the difference of the multi-scale intensity descriptors is represented by D_{MI}.

$$D_{MI}(MI_1^s, MI_2^s) = \sum_s w_s \cdot \sqrt{\sum_{m=1}^{M_s}\sum_{n=1}^{N_s}(MI_1^s(m,n) - MI_2^s(m,n))^2} \tag{15}$$

(3) For the current frame, we calculate its difference $D_{MH}(S, S_{map})$ and $D_{MI}(S, S_{map})$ with the existing multi-scale height descriptors (MH) and multi-scale intensity descriptors (MI) of historical frames, where S_{map} is one of the frames in the historical data.

To identify loop closure candidates, we first compute the combined descriptor distance $D = D_{MH} + D_{MI}$ between the current frame and all historical frames. The historical frames are ranked in ascending order of D, and the top-K candidates with the smallest distances are selected.

To ensure both appearance similarity and geometric consistency, we only accept a candidate frame if its D value is below a predefined threshold T_D. Additionally, to prevent selecting nearby frames that may belong to the same local trajectory segment, we define a minimum temporal separation between the candidate frame S_h and current frame S_i:

$$\Delta_{ID}(S_h, S_i) = |ID(S_h) - ID(S_i)| \tag{16}$$

Only candidate frames satisfying $D < T_D$ and $\Delta_{ID} > \Delta_{\min}$ are passed to the final ICP refinement stage.

(4) Finally, for the loop closure candidate frames that pass through the aforementioned filtering process, ICP algorithm is used to further optimize the pose alignment between them and calculate the final alignment error. If the error is less than the preset threshold $ICP_{\min}$, the loop closure detection is considered successful; otherwise, it is deemed a failure.

IV. Experiments

To assess the effectiveness and accuracy of the proposed MTC-SLAM method, we conducted detailed comparative experiments using the MulRan public dataset and contrasted our approach against current mainstream LiDAR-Inertial SLAM methods. We also performed ablation studies to evaluate the impact of the multi-scale tightly-coupled LiDAR-Inertial SLAM approach and the multi-scale loop closure detection method. All experiments were conducted on a computer equipped with an Intel i7-13700F CPU and 32GB of memory. In our implementation, we exclusively utilize the GTSAM library for nonlinear factor graph optimization, employing the iSAM2 incremental solver. To balance real-time performance and convergence quality, the optimization is configured with a maximum of 100 iterations, an initial damping factor of 1.0, and convergence thresholds of 10^{-5} for both relative error and gradient norm. After each loop closure, the factor graph is re-optimized for 5 additional steps to ensure global consistency.

A. Experiment setup

1) Datasets: We carried out the experiments on the MulRan dataset, choosing three types of complex scene datasets to validate the performance of our method. These included complex campus environments with dynamic objects (Kaist01, Kaist02), dynamic urban scenes with high-rise buildings (DCC01, DCC02), and complex scenarios with similar structural features like rivers and bridges (Riverside01, Riverside02). The numerals indicate that datasets were collected at the same locations but at different times, featuring non-identical trajectories.

TABLE I
ABSOLUTE POSE ERROR METRICS ON THE KAIST DATASET

Dataset	Distance(m)	Duration(s)	Error Type	SC-LEGO-LOAM	SC-LIO-SAM	SC-FASTLIO2	Ours
Kaist01	6123.117	808.69	Max(m)	15.71	9.29	10.82	**5.44**
			Mean(m)	4.53	3.22	4.48	**2.13**
			Median(m)	4.11	2.94	3.9	**2**
			Min(m)	0.41	0.18	0.1	**0.08**
			RMSE	5.27	3.55	4.99	**2.34**
			Std	2.69	1.51	2.21	**0.97**
Kaist02	5964.589	884.275	Max(m)	16.98	10.56	14.65	**5.22**
			Mean(m)	4.4	3.21	4.47	**2.02**
			Median(m)	3.61	2.78	4	**1.98**
			Min(m)	0.32	0.59	0.34	**0.16**
			RMSE	5.38	3.76	5.07	**2.17**
			Std	3.1	1.96	2.39	**0.81**

2) Evaluation criteria: Absolute Pose Error (APE) is the direct difference between estimated poses and true poses, which can intuitively reflect algorithm accuracy and global trajectory consistency. We used the following metrics to evaluate method accuracy: Max Absolute Error, Mean Absolute Error, Median Absolute Error, Min Absolute Error, Root Mean Square Error (RMSE) of Absolute Errors, and Standard Deviation (Std) of Absolute Errors.

B. Comparative Experiments

We compared the proposed MTC-SLAM method with existing methods such as SC-LEGO-LOAM, SC-LIO-SAM, and SC-FASTLIO2, which are enhancements of LEGO-LOAM [20], LIO-SAM [28], and FASTLIO2 [30] using Scan Context [34], respectively. To ensure a fair and unbiased comparison with other LiDAR-based SLAM methods that do not rely on GPS, we have conducted experiments by disabling the GPS in our system.

1) Kaist Dataset: The Kaist dataset represents a campus environment with dynamic objects. It includes multiple reverse revisits and lane shifts for loop closure testing. Table I displays various APE performance metrics on the Kaist dataset. Results indicate that on the Kaist01 dataset, our method outperforms others in all metrics, particularly in RMSE, achieving only 2.34 m, a reduction of 34.08% compared to the well-performing SC-LIO-SAM. SC-LEGO-LOAM showed the weakest performance on Kaist01. On the Kaist02 dataset, our method excelled in all metrics, with an RMSE of only 1.98 m, which is a 28.78% improvement compared to SC-LIO-SAM. These metrics demonstrate that in dynamic campus environments with multiple loops and lane shifts, our method not only maintains accuracy but also exhibits enhanced robustness.

Real-time APE comparisons for the Kaist01 and Kaist02 datasets are illustrated in Fig. 3. The graph clearly shows that our method surpasses other SLAM methods in the Kaist campus environment, particularly in scenarios involving dynamic objects, loop closures, and lane shifts. The APE curve of our method exhibits more stable fluctuations over time, characterized by lower and fewer peaks. This consistency suggests that our algorithm offers significant advantages in terms of accuracy, stability, and robustness. In dynamic and challenging environments, maintaining a low-error SLAM method is essential for effective autonomous navigation and accurate map construction.

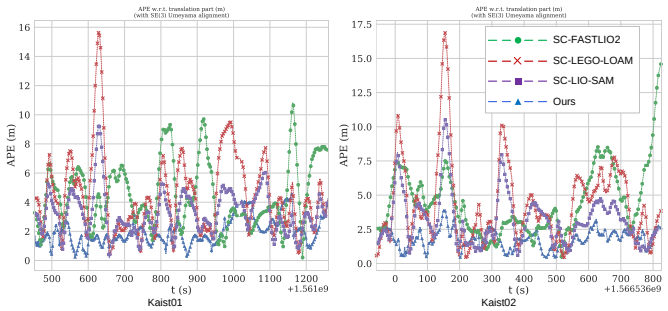

Fig. 3. Kaist real-time APE

Fig. 4 presents the box plots of the APE for various SLAM methods on the Kaist01 and Kaist02 datasets. For Kaist01, our method demonstrates the narrowest interquartile range, indicating a relatively tight error distribution. On Kaist02, while the interquartile range is broader, it consistently remains lower than that of the other methods. Our approach shows fewer outliers on Kaist01 and some on Kaist02, though these are less pronounced compared to other algorithms. These boxplots reveal that our method consistently achieves smaller and more concentrated absolute positional errors on both datasets, enhancing stability. This suggests that our approach is not only more accurate in handling SLAM challenges in complex campus environments but also more robust amid dynamic changes.

2) DCC Dataset: The DCC dataset features urban scenes with high-rise buildings and includes multiple reverse revisits and loop closures with lane shifts. Table II displays the APE performance metrics for the DCC dataset. For DCC01, our method surpasses all other metrics on both DCC01 and DCC02. Notably, the RMSE is only 4.81 m for DCC01 and 3.16 m for DCC02, compared to SC-LIO-SAM, which records 5.76 m and 3.55 m, respectively. SC-FASTLIO2 generally shows the poorest performance, with an RMSE reaching up to 9.50 m on DCC02. Our approach demonstrates overall superiority in the DCC dataset, providing more precise and consistent positioning results under both extreme conditions and on average.

TABLE II
ABSOLUTE POSE ERROR METRICS ON THE DCC DATASET

Dataset	Distance(m)	Duration(s)	Error Type	SC-LEGO-LOAM	SC-LIO-SAM	SC-FASTLIO2	Ours
DCC01	4911.759	533.598	Max(m)	15.76	10.5	15.84	**8.87**
			Mean(m)	6.3	5.42	6.9	**4.54**
			Median(m)	5.92	4.97	6.89	**4.46**
			Min(m)	1.96	2.02	0.79	**0.58**
			RMSE	6.69	5.76	7.4	**4.81**
			Std	2.26	1.94	2.67	**1.61**
DCC02	4272.795	742.284	Max(m)	9.32	6.74	25.56	**7.11**
			Mean(m)	4.13	3.3	8.52	**2.85**
			Median(m)	3.87	3.07	7.83	**2.62**
			Min(m)	0.91	0.95	0.98	**0.16**
			RMSE	4.44	3.55	9.5	**3.16**
			Std	1.63	1.32	4.18	**1.37**

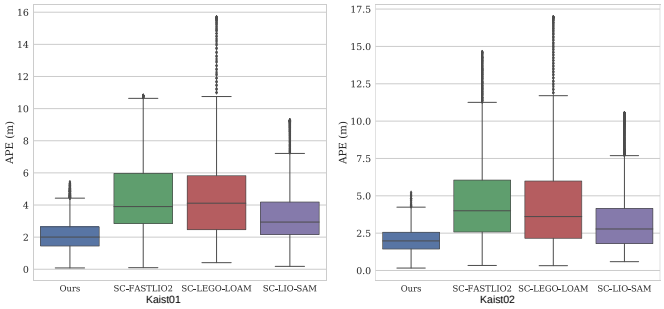

Fig. 4. Kaist APE box plot

Real-time APE comparisons on the DCC01 and DCC02 datasets are illustrated in Fig. 5. The figure shows that for the majority of time periods, our method maintains lower APE values, indicating its ability to consistently deliver precise localization within urban environments. In contrast, other methods experience spikes in error, significantly higher than those of our approach, especially SC-FASTLIO2, which displayed extremely high peaks on the DCC02 dataset. The performance of our method on both DCC datasets indicates lower error levels and enhanced robustness, essential for localization algorithms in urban settings, which are often affected by various interferences. These results suggest that our method is better suited for precise positioning and navigation in densely built-up urban areas.

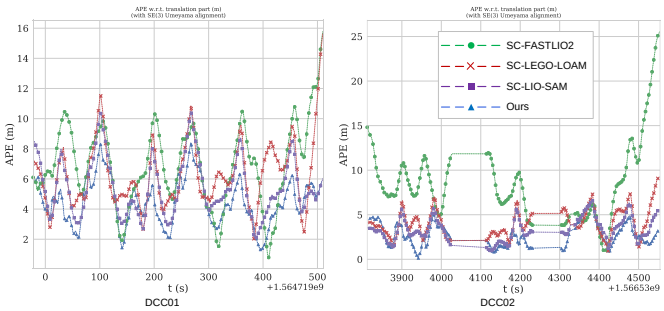

Fig. 5. DCC real-time APE

Fig. 6 presents the APE box plots for the DCC01 and DCC02 datasets. The results show that our method exhibits lower error margins and higher stability across both urban datasets. Although there are some outliers in the DCC01 dataset, overall, our method demonstrates greater reliability and robustness in managing absolute positional errors compared to other SLAM methods.

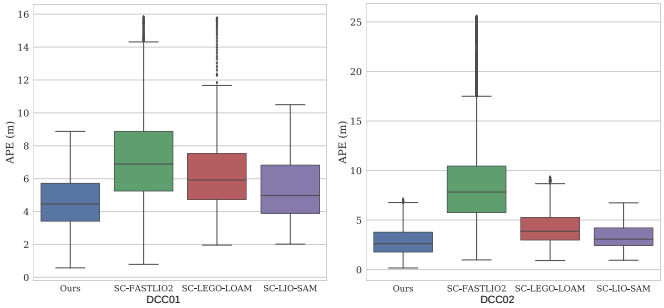

Fig. 6. DCC APE box plot

3) Riverside Dataset: The Riverside dataset features complex environments characterized by rivers and bridges, with similar structural elements. Table III outlines various APE performance metrics on this dataset. Our method consistently achieved the best performance across all metrics on both Riverside01 and Riverside02 datasets, with RMSE values of 4.83 m and 5.25 m, respectively. In comparison, SC-LIO-SAM recorded RMSE values of 9.63 m and 13.59 m. SC-LEGO-LOAM exhibited the weakest performance, especially on Riverside02, with an RMSE reaching up to 32.37 m. Our approach not only delivers highly accurate localization under typical conditions but also maintains lower errors in more challenging scenarios, demonstrating superior consistency and reliability. This suggests that our method is exceptionally well-suited to complex, feature-repetitive environments.

Real-time APE comparisons on the Riverside01 and Riverside02 datasets are illustrated in Fig. 7. The APE curve for our method is consistently lower than those of the other three algorithms throughout Riverside01, indicating more precise localization. Throughout most of the timeframe, our method maintained lower error levels, whereas other algorithms displayed greater error fluctuations, with SC-FASTLIO2 experiencing notably high error peaks at certain points. On the Riverside02 dataset, while errors for all methods increased significantly

TABLE III
ABSOLUTE POSE ERROR METRICS ON THE RIVERSIDE DATASET

Dataset	Distance(m)	Duration(s)	Error Type	SC-LEGO-LOAM	SC-LIO-SAM	SC-FASTLIO2	Ours
Riverside01	6426.614	540.6	Max(m)	82.21	32.94	36.04	**15.05**
			Mean(m)	21.38	8.54	18.57	**3.72**
			Median(m)	19.05	7.68	18.5	**2.7**
			Min(m)	0.58	1.07	1.74	**0.15**
			RMSE	24.57	9.63	20.88	**4.83**
			Std	12.11	4.45	9.56	**3.09**
Riverside02	6612.108	808.791	Max(m)	73.84	37.46	64.32	**23.83**
			Mean(m)	28.41	11.93	24.84	**4.19**
			Median(m)	29.94	12.39	22.34	**3.51**
			Min(m)	4.55	2.04	4.6	**0.31**
			RMSE	32.37	13.59	29.15	**5.25**
			Std	15.53	6.51	15.25	**3.17**

TABLE IV
RUNTIME AND CPU COMPARISON

Dataset	SC-LEGO-LOAM		SC-LIO-SAM		SC-FSTLIO2		Ours	
	Time (ms)	CPU (%)	Time (ms)	CPU (%)	Time (ms)	CPU (%)	Time (ms)	CPU (%)
Kaist01	85	46	78	43	**65**	**40**	67	41
Kaist02	91	48	83	45	73	41	**68**	**39**
DCC01	79	44	**76**	**42**	74	42	77	43
DCC02	88	47	85	45	72	41	**69**	**36**
Riverside01	108	49	101	46	90	**43**	**85**	44
Riverside02	105	48	98	45	89	42	**83**	**39**
Average	92.67	47.0	86.83	44.3	77.17	41.5	**74.83**	**40.3**

towards the end, our method's performance remained more stable over the entire series. Despite occasional elevated errors, the frequency and magnitude of these peaks were substantially lower compared to the others. This performance demonstrates that our method provides higher stability and accuracy in environments with complex features like rivers and bridges.

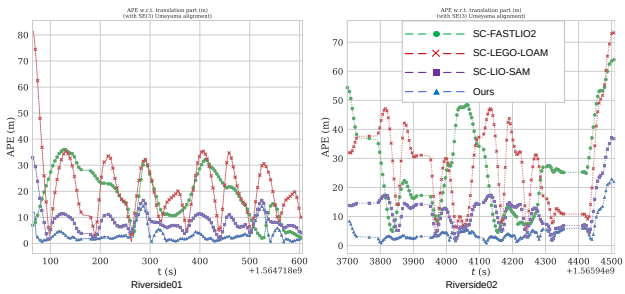

Fig. 7. Riverside real-time APE

Fig. 8 presents the APE box plots for the Riverside01 and Riverside02 datasets. Results show a very compact error distribution for our algorithm on Riverside01, with a median significantly lower than that of other algorithms, indicating very precise localization in most situations. On the Riverside02 dataset, the median and interquartile range for our method remained at lower levels. Although the data was more dispersed than in Riverside01, it still showed good accuracy and consistency. Overall, our method demonstrated robustness and reliability in complex scenarios involving rivers and bridges, maintaining excellent performance even under varying environmental conditions.

In addition, we compare the proposed method with SC-

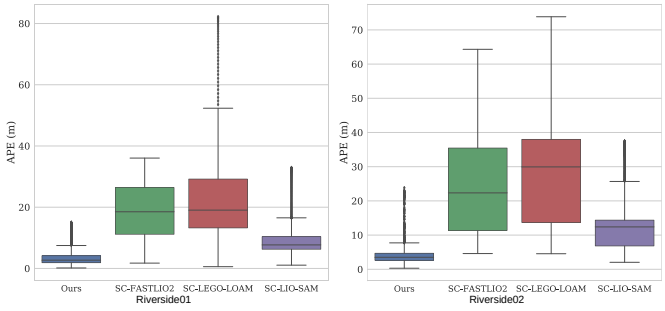

Fig. 8. Riverside APE box plot

LEGO-LOAM, SC-LIO-SAM, and SC-FASTLIO2 in terms of average running time and CPU usage per frame. As shown in Table IV, despite the added complexity introduced by multiscale map updates and loop detection, our method maintains a high level of efficiency. It achieves the lowest average runtime (74.83 ms) and CPU utilization (40.3%) among all compared methods, demonstrating that real-time performance remains achievable. This is primarily attributed to two factors: the multiscale pose update employs selective optimization, where only key degrees of freedom are iteratively refined while the others are efficiently interpolated, significantly reducing unnecessary computation; and the multiscale loop detection leverages hierarchical descriptors for early-stage filtering, which limits the number of frames passed to ICP refinement and reduces memory and runtime costs. These results validate the practicality of our design and confirm that the increased algorithmic complexity does not lead to significant computational overhead.

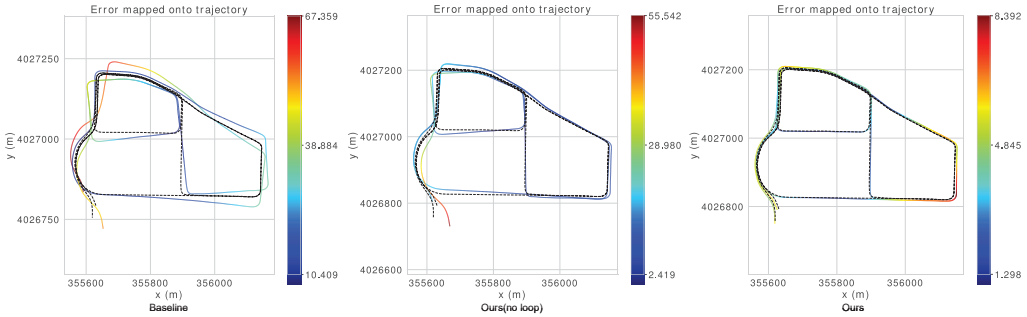

Fig. 9. DCC01 trajectory comparison

C. Ablation Experiments

We conducted ablation studies in complex urban scenarios with high-rise buildings using the DCC01 and DCC02 datasets. In these studies, the baseline algorithm was LIO-SAM, and Ours (no loop) refers to our method implemented without the multi-scale loop closure.

Fig. 9 shows the trajectory error comparison for the DCC01 dataset, and illustrates the impact of different algorithms on trajectory accuracy and error distribution. Errors are represented by the color intensity of the three-dimensional spatial trajectory, with darker colors indicating greater errors. The results reveal that the Baseline exhibits significant errors, particularly in the middle and end sections of the trajectory, as shown by darker colors, which denote lower localization precision. In contrast, Ours (no loop) shows a reduction in trajectory error, especially in the middle section, where lighter colors signify fewer errors. This suggests that multi-scale state updates have effectively improved trajectory accuracy. Compared to Ours (no loop), Ours significantly reduced the trajectory error, with lighter colors along the entire path and alignment close to the true trajectory, demonstrating the clear benefits of multi-scale loop closure.

As demonstrated in Fig. 10, the APE metric comparison for the DCC01 ablation study presents the trend of APE over time along with error statistical metrics such as Mean, Median, RMSE, and Std. The Baseline's APE displays several significant peaks, indicating large localization errors at specific moments. Moreover, higher values in Mean and Median, combined with a wide range for RMSE and Std, suggest greater variability and substantial performance fluctuations. In comparison, the APE curve of Ours (no loop) is generally lower than that of the Baseline, with fewer and less pronounced peaks, indicating more stable and smaller localization errors. This underscores the importance of multi-scale state updates in enhancing system robustness. The APE curve of Ours fluctuates within the lowest range, with the fewest peaks and smallest magnitude, showcasing the highest stability and accuracy. Significant improvements in Median and RMSE emphasize the advantages of multi-scale loop closure in improving localization precision and trajectory stability.

The trajectory error comparison for the DCC02 ablation study is illustrated in Fig. 11. The Baseline exhibits significant deviations in certain segments of the trajectory, particularly at turns, with errors ranging from a minimum of 7.972 m to a maximum of 55.786 m. This highlights the Baseline method's lack of stability in complex environments due to its reliance on a single scale. Ours (no loop), which incorporates multi-scale state updates, demonstrates a marked reduction in trajectory error, especially at the start and middle of the trajectory. The maximum error was reduced to 23.008 m, and the minimum error was brought down to 3.314 m, affirming the effectiveness of multi-scale state updates in enhancing trajectory estimation accuracy. The complete Ours method, featuring multi-scale loop closure, achieves an even more significant reduction in error. The entire trajectory closely aligns with the true trajectory, with a much narrower error range, where the maximum error is only 6.316 m, showcasing the clear benefits of multi-scale loop closure in minimizing errors and enhancing both trajectory accuracy and system robustness.

As depicted in Fig. 12, the APE metric comparison for the DCC02 ablation study shows that the Baseline's APE fluctuates significantly, with several pronounced error peaks illustrating its challenges in managing complex scenarios. In contrast, the APE fluctuations of Ours (no loop) are notably reduced, which underscores the efficacy of multi-scale state updates. Our complete method, Ours, exhibits the least error fluctuation and the lowest statistical metrics, demonstrating exceptional localization precision and stability across the trajectory. The marked reductions in both mean and median errors underscore the substantial improvements in the algorithm's robustness and precision brought about by multi-scale loop closure, affirming the superior stability and accuracy of our approach.

Moreover, we have compared the real-time trajectories of various methods on DCC01 and DCC02, as shown in Fig. 13. The Baseline method, while generally following the true trajectory, shows large deviations in certain parts. Compared to the Baseline, the gap between Ours(no loop) and the true trajectory is noticeably reduced, indicating higher localization accuracy of the multi-scale state update method. The trajectory of the Ours method is very close to the true trajectory for most of the time, exhibiting superior performance and indicating the significant role of multi-scale loop closure in improving trajectory precision.

To further validate the effectiveness of the proposed loop closure detection method, we conducted extensive evaluations on the KITTI dataset. We used precision and recall rates as

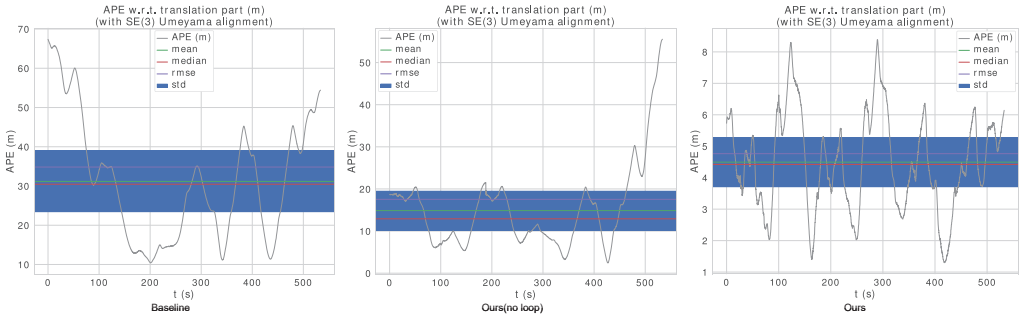

Fig. 10. DCC01 APE comparison

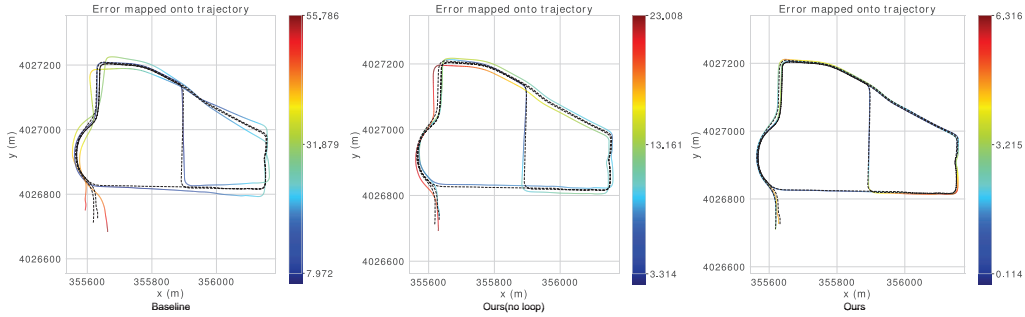

Fig. 11. DCC02 trajectory comparison

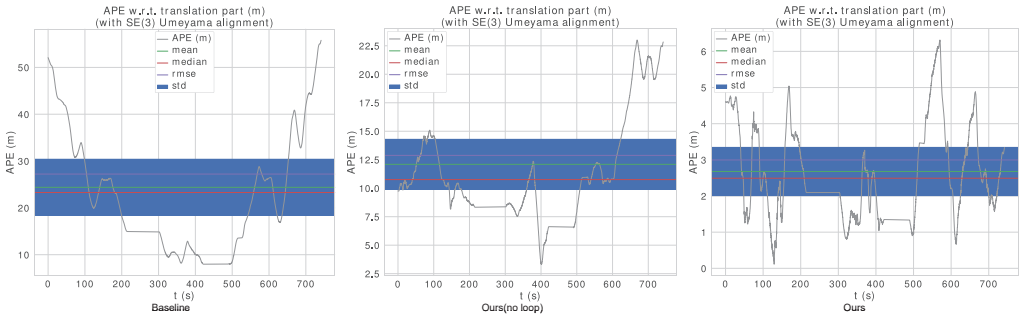

Fig. 12. DCC02 APE comparison

performance metrics and compared our approach with state-of-the-art methods, including SC [34], ISC [35], and CSSC [36].

The experimental results are summarized in Table V. As shown, our method consistently achieves high precision across all evaluated sequences while delivering a notable improvement in recall compared to the baseline methods.

Specifically, our approach maintains 100% precision across multiple sequences, effectively avoiding false positive loop closures. At the same time, it achieves the highest recall in all tested cases. For example, in Sequence 02, our method improves recall by 20% over SC and 4% over CSSC. These improvements highlight the effectiveness of the proposed multiscale height and intensity descriptors in capturing environmental structure and enabling accurate loop closure candidate identification.

The results demonstrate that while all methods exhibit strong precision, our multiscale descriptor design yields significantly better recall. This is primarily due to its ability to represent scene information at multiple spatial resolutions, which improves robustness to environmental variations and occlusions. Therefore, the proposed method enhances both the reliability and completeness of loop closure detection, especially in complex or dynamic environments.

TABLE V
COMPARISON OF PRECISION AND RECALL RATE

Dataset	Approach	Precision (%)	Recall Rate (%)
Sequence 00	SC	100	87
	ISC	100	90
	CSSC	98	90
	Ours	**100**	**92**
Sequence 02	SC	90	73
	ISC	98	91
	CSSC	97	89
	Ours	**98**	**93**
Sequence 05	SC	100	90
	ISC	100	91
	CSSC	100	89
	Ours	**100**	**92**

Overall, the experimental results demonstrate that the pro-

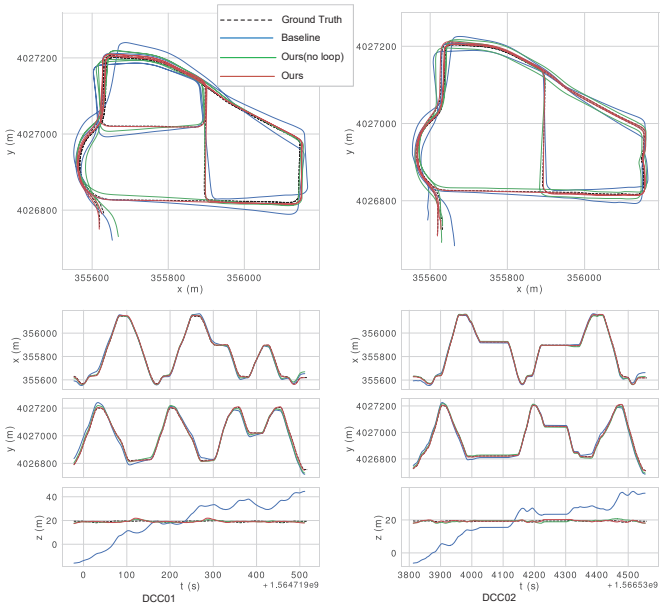

Fig. 13. DCC real-time trajectory

posed MTC-SLAM method exhibits superior robustness and accuracy in dynamic environments compared to existing LiDAR-inertial SLAM systems. This advantage stems from several key design choices. First, the selective multiscale optimization strategy refines only the dominant degrees of freedom (x, y, and yaw) while interpolating z, $pitch$, and $roll$, thereby avoiding overfitting to transient disturbances caused by moving objects and ensuring stable pose estimation in dynamic scenes. Second, the residual construction process incorporates dynamic filtering by excluding unreliable geometric structures such as lines with dominant z-components or planes lacking vertical constraints, which reduces the influence of dynamic points during optimization. Third, the hierarchical loop detection module uses multiscale descriptors to efficiently reject dynamic or inconsistent frames at an early stage, minimizing the risk of incorrect loop closures. These design elements collectively enhance the system's robustness under dynamic interference. Quantitative results on dynamic sequences from the DCC and Kaist datasets support these claims. For example, on DCC sequence 02, MTC-SLAM achieves an RMSE of 3.16 m, outperforming SC-LIO-SAM (3.55 m) and SC-FASTLIO2 (9.5 m), confirming the method's effectiveness in maintaining high localization accuracy under dynamic conditions.

V. Conclusion

Autonomous vehicles operating in dynamic and complex environments require SLAM systems that are both highly precise and robust, far beyond the capabilities needed in controlled settings. Traditional LiDAR-Inertial SLAM methods, which often focus on single-scale issues, struggle with dynamic objects and similar features, leading to poor accuracy and robustness of pose estimation.

In response to these real-world challenges, this paper introduces a multi-scale, tightly-coupled LiDAR-Inertial SLAM method. Initially, the method updates the pose coarsely on a larger scale for the six degrees of freedom. This is followed by adaptive step-size least squares iterations for the x, y, and yaw degrees on a smaller scale, complemented by dynamic interpolation updates for the z, $pitch$, and $roll$ degrees. The updated poses are then utilized to construct a factor graph, incorporating GPS factors and multi-scale loop closure factors to finalize the backend optimization.

However, in high-dynamic, long-distance driving scenarios, relying solely on LiDAR and IMU data may limit SLAM performance. In future we will explore integrating visual sensors to further enhance system precision and robustness.

References

[1] X. Liu, S. Wen, and H. Zhang, "A real-time stereo visual-inertial slam system based on point-and-line features," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 5, pp. 5747–5758, 2023.

[2] M. M. Azarbeik, H. Razavi, K. Merat, and H. Salarieh, "Augmenting inertial motion capture with slam using ekf and srukf data fusion algorithms," *Measurement*, vol. 222, p. 113690, 2023.

[3] Z. Shen, J. Wang, C. Pang, Z. Lan, and Z. Fang, "A lidar-imu-gnss fused mapping method for large-scale and high-speed scenarios," *Measurement*, vol. 225, p. 113961, 2024.

[4] J. Yu, Z. Xiang, and J. Su, "Hierarchical multi-level information fusion for robust and consistent visual slam," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 1, pp. 250–259, 2022.

[5] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[6] S. Wen, X. Li, X. Liu, J. Li, S. Tao, Y. Long, and T. Qiu, "Dynamic slam: A visual slam in outdoor dynamic scenes," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–11, 2023.

[7] M. Li and F. Rottensteiner, "Vision-based indoor localization via a visual slam approach," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 827–833, 2019.

[8] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "Fast-livo: Fast and tightly-coupled sparse-direct lidar-inertial-visual odometry," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 4003–4009.

[9] Y. Chang, Y. Tian, J. P. How, and L. Carlone, "Kimera-multi: a system for distributed multi-robot metric-semantic simultaneous localization and mapping," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 210–11 218.

[10] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "Door-slam: Distributed, online, and outlier resilient slam for robotic teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, 2020.

[11] C. Sun, J. Leng, B. Wang, W. Liang, B. Jia, Z. Huang, B. Lu, and J. Li, "Vmc-lio: Incorporating vehicle motion characteristics in lidar inertial odometry," *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2024.

[12] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-loam: Fast lidar odometry and mapping," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4390–4396.

[13] C. Qian, Z. Xiang, Z. Wu, and H. Sun, "Rf-lio: Removal-first tightly-coupled lidar inertial odometry in high dynamic environments," *arXiv preprint arXiv:2206.09463*, 2022.

[14] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "Mulran: Multimodal range dataset for urban place recognition," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6246–6253.

[15] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4530–4537.

[16] J. Chen, K. Wu, M. Hu, P. N. Suganthan, and A. Makur, "Lidar-based end-to-end active slam using deep reinforcement learning in large-scale environments," *IEEE Transactions on Vehicular Technology*, pp. 1–14, 2024.

[17] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606.

[18] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3d-ndt," *Journal of Field Robotics*, vol. 24, no. 10, pp. 803–827, 2007.

[19] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.
[20] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4758–4765.
[21] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
[22] Z. Liu, C. Suo, S. Zhou, F. Xu, H. Wei, W. Chen, H. Wang, X. Liang, and Y.-H. Liu, "Seqlpd: Sequence matching enhanced loop-closure detection based on large-scale point cloud description for self-driving vehicles," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1218–1223.
[23] Y. S. Park, H. Jang, and A. Kim, "I-loam: Intensity enhanced lidar odometry and mapping," in *2020 17th International Conference on Ubiquitous Robots (UR)*, 2020, pp. 455–458.
[24] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-lio: Robust high-bandwidth light detection and ranging inertial odometry," *Advanced Intelligent Systems*, p. 2200459, 2023.
[25] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Globally consistent and tightly coupled 3d lidar inertial mapping," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5622–5628.
[26] W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 6240–6245.
[27] A. Tagliabue, J. Tordesillas, X. Cai, A. Santamaria-Navarro, J. P. How, L. Carlone, and A.-a. Agha-mohammadi, "Lion: Lidar-inertial observability-aware navigator for vision-denied environments," in *Experimental Robotics: The 17th International Symposium*. Springer, 2021, pp. 380–390.
[28] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2020, pp. 5135–5142.
[29] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.
[30] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.
[31] X. Liu, S. Wen, Z. Jiang, W. Tian, T. Z. Qiu, and K. M. Othman, "A multisensor fusion with automatic vision–lidar calibration based on factor graph joint optimization for slam," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–9, 2023.
[32] X. Xu, S. Lu, J. Wu, H. Lu, Q. Zhu, Y. Liao, R. Xiong, and Y. Wang, "Ring++: Roto-translation invariant gram for global localization on a sparse scan map," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4616–4635, 2023.
[33] Z. Zou, C. Zheng, C. Yuan, S. Zhou, K. Xue, and F. Zhang, "ibtc: An image-assisting binary and triangle combined descriptor for place recognition by fusing lidar and camera measurements," *IEEE Robotics and Automation Letters*, vol. 9, no. 12, pp. 10 858–10 865, 2024.
[34] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4802–4809.
[35] H. Wang, C. Wang, and L. Xie, "Intensity scan context: Coding intensity and geometry relations for loop closure detection," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 2095–2101.
[36] D. Xu, J. Liu, Y. Liang, X. Lv, and J. Hyyppä, "A lidar-based single-shot global localization solution using a cross-section shape context descriptor," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 189, pp. 272–288, 2022.
[37] Y. Wang, Z. Sun, C.-Z. Xu, S. E. Sarma, J. Yang, and H. Kong, "Lidar iris for loop-closure detection," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5769–5775.
[38] L. He, X. Wang, and H. Zhang, "M2dp: A novel 3d point cloud descriptor and its application in loop closure detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 231–237.

Qifeng Wang (Student Member, IEEE) received his master's degree from Wuhan University of Science and Technology in Wuhan, Hubei Province, in 2023. He is currently pursuing a PhD in Control Science and Engineering at Wuhan University of Science and Technology. His research interests include SLAM and point cloud registration.

Weigang Li (Member, IEEE) is an professor and doctoral supervisor, as well as a specially appointed professor of Chu Tian Scholars, specializing in research on industrial artificial intelligence technology. He is the primary recipient of the first prize of the Hubei Province Science and Technology Progress Award. His research focuses on artificial intelligence and machine learning algorithms, as well as SLAM and 3D machine vision.

Lei Nie (Member, IEEE) received the B.S. degree in computer science and technology from Wuhan University of Science and Technology, Wuhan, China, in 2011, and the Ph.D. degree in computer architecture from Wuhan University, Wuhan, China, in 2017. He is currently a lecturer with the School of Computer Science and Technology, Wuhan University of Science and Technology. His main research interests include vehicular networks and intelligent transportation.

Wenping Liu (Senior Member, IEEE) is currently a Professor in Hubei University of Economics, China. He received the Ph.D. degree from Huazhong University of Science and Technology in 2012. His research interests include mobile computing, Internet of things, smart city and sensor networks.

Hongbo Jiang (Senior Member, IEEE) received the Ph.D. degree from Case Western Reserve University in 2008. He ever was a Professor with the Huazhong University of Science and Technology. He is currently a Full Professor with the College of Computer Science and Electronic Engineering, Hunan University. His research concerns computer networking, especially algorithms and protocols for wireless and mobile networks. He is serving as an Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING, an Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Technical Editor for the IEEE Communications Magazine.