Testing and Evaluation of Collaborative SLAM

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Siqi Li

Graduate Program in Civil Engineering

The Ohio State University

2017

Master's Examination Committee:

Dr. Dorota Grejner-Brzezinska, Advisor

Dr. Charles Toth, Co-Advisor

Dr. Alper Yilmaz
Abstract

Localization and navigation of a moving platform like a human or a mobile robot in an unknown environment is a challenging problem. Simultaneous localization and mapping (SLAM) algorithms have been developed to solve this issue. Compared with a single-user SLAM, where limited information and error accumulation may limit the availability of a solution and its accuracy, Collaborative SLAM has the potential to reuse map information and enhance localization accuracy. In this thesis, Collaborative Visual-inertial SLAM method is presented, tested and evaluated using a crowd-sourced dataset. System performance testing verifies that this approach currently offers a near real-time performance on a client side, effective information exchange between clients and the server, as well as the benefits of collaborative SLAM on the server in terms of accuracy and robustness. Finally, the performance of the system is analyzed as a function of varying blur, camera frame rate, and IMU data sampling rate.
Acknowledgments

First, I would like to thank my advisors, Dr. Grejner-Brzezinska and Dr. Toth. for their continuous advice of my master study, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of researching and writing this thesis.

Besides my advisors, I would like to thank Dr. Yilmaz for serving on the committee, reviewing my thesis and providing insightful comments.

My sincere thanks also go to Dr. Jianzhu Huai, who provided his Collaborative SLAM software implementation, and helped me through my research and patiently answered all of my questions regarding this project.

I would like to thank Dr. Zoltan Koppanyi, for holding discussions with me and providing advice on the organization of my thesis.

I would also like to thank my labmates, for creating an atmosphere conducive to open discussions about courses and research.

Finally, I must express my heartfelt gratitude to my parents and to my friend Xiao Ma for providing me with unlimited support and continuous encouragement through my years of study and through the process of writing this thesis. Without them, this accomplishment would not have been possible.
Vita

2011 - 2015 ...........................................B.S. Spatial Informatics & Digitalized Technology, Wuhan University

2015 to present ...............................Graduate Student, Department of Civil, Environmental, and Geodetic Engineering, The Ohio State University

Fields of Study

Major Field: Civil Engineering
## Table of Contents

Abstract ........................................................................................................................................... ii  

Acknowledgments ......................................................................................................................... iii  

Vita ................................................................................................................................................ iv  

List of Acronyms ............................................................................................................................ vii  

List of Tables .................................................................................................................................. ix  

List of Figures ................................................................................................................................. xii  

Chapter 1: Introduction .................................................................................................................. 1  

1.1 Background and Motivations ................................................................................................. 1  

1.2 Research Objectives ............................................................................................................. 3  

1.3 Methodology .......................................................................................................................... 4  

1.4 Outline ..................................................................................................................................... 5  

Chapter 2: Visual-Inertial SLAM ................................................................................................. 7  

2.1 IMU and Camera ..................................................................................................................... 7  

2.1.1 IMU and INS ..................................................................................................................... 7  

2.1.2 Camera and SfM ............................................................................................................... 9  

v
## List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATE</td>
<td>Absolute Trajectory Error</td>
</tr>
<tr>
<td>BA</td>
<td>Bundle Adjustment</td>
</tr>
<tr>
<td>CSLAM</td>
<td>Collaborative SLAM</td>
</tr>
<tr>
<td>DoF</td>
<td>Degrees of Freedom</td>
</tr>
<tr>
<td>ECEF</td>
<td>Earth-Center Earth-Fixed</td>
</tr>
<tr>
<td>EKF</td>
<td>Extended Kalman Filter</td>
</tr>
<tr>
<td>fps</td>
<td>Frame per Second</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial Measurement Unit</td>
</tr>
<tr>
<td>INS</td>
<td>Inertial Navigation System</td>
</tr>
<tr>
<td>MEMS</td>
<td>Micro-Electro-Mechanical System</td>
</tr>
<tr>
<td>monoSLAM</td>
<td>Monocular SLAM</td>
</tr>
<tr>
<td>MSCKF</td>
<td>Multi-State Constraint Kalman Filter</td>
</tr>
<tr>
<td>NED</td>
<td>North-East-Down</td>
</tr>
<tr>
<td>OKVIS</td>
<td>Open Keyframe-based Visual-Inertial SLAM</td>
</tr>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>PSD</td>
<td>Power Spectral Density</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>PF</td>
<td>Particle Filter</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Square Error</td>
</tr>
<tr>
<td>ROS</td>
<td>Robot Operating System</td>
</tr>
<tr>
<td>RTK</td>
<td>Real-Time Kinematic</td>
</tr>
<tr>
<td>SE (3)</td>
<td>Special Euclidean group of the three-dimensional Euclidean space</td>
</tr>
<tr>
<td>SfM</td>
<td>Structure from Motion</td>
</tr>
<tr>
<td>Sim (3)</td>
<td>Similarity transformation group of the three-dimensional Euclidean space</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous Localization and Mapping</td>
</tr>
<tr>
<td>SPIN Lab</td>
<td>Satellite Positioning and Inertial Navigation Laboratory</td>
</tr>
<tr>
<td>std. dev.</td>
<td>standard deviation</td>
</tr>
<tr>
<td>UAV</td>
<td>Unmanned Aerial Vehicle</td>
</tr>
<tr>
<td>UKF</td>
<td>Unscented Kalman Filter</td>
</tr>
<tr>
<td>VIO</td>
<td>Visual-Inertial Odometry</td>
</tr>
</tbody>
</table>
List of Tables

Table 1. Dataset information............................................................................................................. 30

Table 2. IMU Characteristics. Root PSD stands for the root power spectral density...... 30

Table 3. Standard Deviation of IMU Biases Initial Value............................................................... 30

Table 4. Alignment between Two Sets of Corresponding Points. $SPQ$ is the best-fitting rigid transformation that aligns two sets of corresponding points, $P = \{p_1, p_2, ..., p_n\}$ and $Q = \{q_1, q_2, ..., q_n\}$. $S'$ is the best-fitting similarity transformation that aligns these two point sets. $p$ and $q$ are the centroids of point sets $P$ and $Q$, respectively. $R$ is the optimal rotation and $t$ stands for the optimal translation. $\sigma$ represents the computed scale factor ................................................................. 34

Table 5. Characteristics of Data Segments. ..................................................................................... 35

Table 6. Error statistics of point-based comparison between the estimated positions and GPS reference solution. ............................................................................................................ 37

Table 7. Error statistics of point-based comparison between the estimated positions and GPS reference solution. ............................................................................................................ 40

Table 8. Error Metrics of the VIO Algorithm on Client. Note for the ATE, the ratio is the error divided by the trajectory length...................................................................................... 43

Table 9. Error Statistics for Estimated Trajectory on Client. 3D point distance refers to the distance between the estimated position (after a rigid transformation using
algorithm stated in Table 2) and the ground truth position. $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$ refers to errors between the $x, y, z$ components of the estimated trajectory with the $x, y, z$ components of the ground truth data. RMSE stands for root mean square error.

Table 10. Error statistics of point based comparison between the estimated positions and GPS reference solution.

Table 11. Error metrics of the Collaborative SLAM. Note for the ATE (absolute trajectory error), the ratio is the error divided by the trajectory length.

Table 12. Error Statistics for Estimated Trajectory on Sever. 3D point distance refers to the distance between the estimated position (after a rigid transformation using algorithm stated in Table 2) and the ground truth position. $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$ refers to the errors between the $x, y, z$ components of the estimated trajectory with the $x, y, z$ components of the ground truth data. RMSE stands for root mean square error.

Table 13. Error metrics of the Collaborative SLAM Trajectory under the impact of image blur, camera frame rate and IMU data sampling rate. The first two rows have IMU sampling rate of 200Hz. The last row has camera frame rate of 30fps. ATE is the absolute trajectory error. The scale is calculated using Equation (16).

Table 14. Error metrics of the Collaborative SLAM Trajectory under the impact of image blur, camera frame rate and IMU data sampling rate. The default camera frame rate is 30fps and default IMU sampling rate is 200Hz, if not mentioned. ATE is the absolute trajectory error described in Equation (8). The scale is calculated using Equation (16). The first element accepted by operator “+” is the data parameter of the first client, and the second element is the data parameter of the
second client. For example, G3+G9 (30FPS) stands for the case that the first client uses 30fps start-segments data filtered by a $3 \times 3$ Gaussian smoother, and the second client uses 30fps end-segment data filtered by a $9 \times 9$ Gaussian smoother, both clients adopted a 200Hz IMU data sampling rate.
List of Figures

Figure 1. Strapdown Mechanization Algorithm (Woodman, 2007). The orientation of an INS relative to the global frame is tracked by integrating the angular velocity signals obtained from gyroscopes. The computed orientation is used to project accelerations obtained from accelerometers into the global frame. Acceleration due to gravity is then subtracted and the remaining accelerations are integrated once to obtain velocity, and again to get displacement. The displacement is added to the initial position to get current position. ................................................................. 8

Figure 2. UKF Principle (Barfoot, 2017). One-dimensional Gaussian PDF p(x) is transformed through a deterministic nonlinear function $g(\cdot)$ using the basic sigmapoint transformation. Here only two deterministic samples ($\mu_x - \sigma x$ and $\mu x + \sigma x$) are selected to approximate the input density. $\mu x$ and $\sigma x$ are mean and standard deviation of variable x. $\mu y$ and $\sigma y$ are mean and standard deviation of variable y................................................................. 14

Figure 3. Taxonomy of the Different Filtering Methods and Relationships to the Bayes Filter (Barfoot, 2017). .................................................................................................................................................. 15

Figure 4. The Client-Server Structure Deployed in the Collaborative SLAM System (Huai, 2017). Each client processes their camera and inertial data using a visual-inertial odometry algorithm, and the keyframe messages are sent from each client
to the server containing information about keyframes. The server receives the keyframes message from n clients and sets up keyframe handler threads for each client together with a loop closing thread to build maps. The loop closing thread detects and closes the loop in maps from each keyframe handler thread using visual vocabulary (Gálvez-López & Tardos, 2012) and keyframe database which records the occurrence of visual words from the visual vocabulary in keyframes. The server also sends messages to each client with information used to update camera pose estimations.

Figure 5. Examples of Simulated Different Camera Rates. The highlighted frames are frames used as the visual data for the client unit, and t refers to time. Top: Original camera rate. Bottom: ×0.5 camera frame rate.

Figure 6. Examples of Simulated Different IMU Data Sampling Rate. Each stick stands for one piece of IMU data and t refers to time. The highlighted sticks are used as the inertial data for the client unit. Top: Original IMU data frequency (upper one). Middle: ×0.5 sampling rate. Bottom: ×0.25 sampling rate.

Figure 7. The trajectory of the data which contains several repeated loops.

Figure 8. Examples of Scenes from Collected Dataset.

Figure 9. GPS trajectories of the dataset collected at The Ohio State University. (a) Start-segment trajectory (blue) and End-segment trajectory (red). (b) Start-segment Trajectory. (c) End-segment trajectory.
Figure 10. (a) Top view of the estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. (b) 3D position error at each epoch. (c) Zoom in of the starting point surrounding.

Figure 11. (a): Top view of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. (b): 3D position distance error are calculated at each epoch (c): 3D view of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point.

Figure 12. (a): Zoom in between 160 second and 245 second of Figure 11 (b). The peak values and valley values are marked. (b): Corresponding portion of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. Peak points and valley points from (a) are noted in the two trajectories. Pink lines between the trajectories represent the error vectors at every 10 points.

Figure 13. Left: Top view of trajectory estimated by the VIO on Client aligned to the GPS reference solution with the same starting point. Right: Top view of trajectory estimated by the VIO on Client aligned to the GPS reference solution with a rigid transformation using algorithm stated in Table 2.

Figure 14. Error between client-side position estimates (after a rigid transformation) and GPS reference positions in x, y, and z directions.

Figure 15. Examples of corner information contained in keyframe messages. The green boxes are the detected corner in the keyframe.
Figure 16. Left: Point based comparison between the estimated positions and GPS reference solution in a local NED frame. Errors are calculated as 3D point distance at each epoch. Right: Top view of estimated trajectory and GPS reference trajectory in the local NED frame of the starting point. 46

Figure 17. Left: Top view of trajectory estimated by Collaborative SLAM aligned to the linear-interpolated GPS reference trajectory with the same starting point. Right: Top view of trajectory estimated by Collaborative SLAM aligned to the linear-interpolated GPS reference trajectory with a rigid transformation using algorithm stated in Table 2. The blue point density (represents the density of keyframe) in the figure is much less than the blue point density in Figure 10. 47

Figure 18. Error between Collaborative SLAM position estimates (after a rigid transformation) and GPS ground truth in x, y, and z directions. 48

Figure 19. Different image blur size and camera frame rate. Accumulated position error between GPS reference trajectory with estimated trajectory after a rigid transformation when changing the image blur size and camera frame rate. G1 stands for no image blur and G3, G5, G9 represent a 3×3, 5×5 or 9×9 Gaussian blur applied to the image frame, respectively. 50

Figure 20. Position error statistics of the image blur: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. G1 stands for no image blur and G3, G5, G9 represent a 3×3, 5×5 or 9×9 Gaussian blur applied to the image frame,
respectively. The camera frame rate is constant and equals to 30 fps in the figure.

Figure 21. Position error statistics of the image blur: median, 5th and 95th percentiles.
Position error refers to the distance between estimated position and its corresponding GPS reference position. The camera frame rate is constant and equals to 15 fps in the figure.

Figure 22. Different camera frame rate and IMU frequency. Accumulated position error when changing the camera frame rate and IMU frequency. G3 stands for a 3×3 Gaussian filter which applied to image frames.

Figure 23. Different image blur size and IMU frequency. Accumulated position error when changing the image blur size and IMU frequency. G1 and G3, G5 represent no blurring, 3×3 or 5×5 Gaussian blurring applied to the image frame, respectively.

Figure 24. Position error statistics of the IMU data sampling rate: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. G1 stands for no image blur and G3, G5 represent a 3×3, 5×5 Gaussian blurring applied to the image frame, respectively. The camera frame rate is constant and equals to 30 fps in the figure.

Figure 25. Position error statistics of two clients using different camera frame rate, image blur or IMU data sampling rate: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS
reference position. G3 represent a $3 \times 3$ Gaussian blurring applied to the image frame. 200Hz, 100Hz are the IMU data sampling rate.
Simultaneous localization and mapping (SLAM) problem was first introduced by Hugh Durrant-Whyte and John J. Leonard (1991). SLAM was designed to determine if it was possible for a moving platform such as a robot or an autonomous vehicle to build a consistent map of the unknown environment where it is placed while determining its location and navigating the environment using this map. The moving platform is placed at an unknown location in an unknown environment and uses only relative observations of its surroundings under the constraint that a priori topological knowledge is unavailable. A solution to the SLAM problem is an estimated pose graph, a relative moving trajectory of the object, and a generated map of its surrounding environment. The autonomous characteristic of a SLAM solution makes it notable in a broad range of applications, where the absolute position or precise map information is inaccessible. The applications include a hand-held mobile device, unmanned aerial vehicles (UAV), autonomous planetary exploration, underwater autonomous vehicles, and autonomous all-terrain vehicles in tasks such as mining and construction (Dissanayake et al., 2001).

As seen from the soaring number of publications in relevant areas, SLAM techniques have become one of the major topics in robotics. With the increasing interests in the
combination of the onboard cameras and inertial measurement unit (IMU) sensors on small-size and lightweight platforms such as hand-held mobile devices and UAVs, visual-inertial SLAM techniques attract a widespread attention of many researchers. There are two types of dominant approaches adopted in visual-inertial SLAM, which are the filtering-based (Mourikis and Roumeliotis, 2007; Kelly and Sukhatme, 2011; Montemerlo and Thrun, 2002) and the optimization-based approaches (Leutenegger and Lynen, 2015).

Collaborative SLAM problems ask if it is possible for multiple moving platforms to build one or several maps of their surrounding environments while simultaneously estimating the pose of each platform and navigating them in a single or multiple maps. The participants of multiple platforms bring the potential of using redundant and parallel mechanisms to achieve better accuracy and robustness (Forster, 2013). The collaborative SLAM is employed in several tasks, which can be shared among multiple platforms, such as the localization of multiple robots or crews in search and rescue missions, in addition to the workload of inspection of an unknown environment. The shared map between the platforms allows the reuse and improvement of map information. However, solving the Collaborative SLAM problem generally increases the computational and communicational loads. It is highly challenging to finding solutions to Collaborative SLAM problem when the accuracy, online performance, robustness and operation scale are considered.

A Collaborative SLAM solution (Huai, 2017) solves the problem of building maps of unknown environments using data collected from multiple users and estimating the pose of each user within these maps. The system uses a client-server framework, where the server
helps improve the state estimates on the client by the communication (information exchange) between them.

To validate and enhance this collaborative SLAM technique, test and evaluation on the system is an inevitable part of the system developing progress. Test and evaluation would provide the means to ensure that the system performs in an intended way and with the intended results. Moreover, we can gain further insight into the Collaborative SLAM by conducting tests on different simulated situations.

1.2 Research Objectives

As stated in section 1.1, several tasks which can be shared among multiple platforms require the collaborative localization and mapping, such as the environment exploration, navigating from point A to point B, or search and rescue. The platforms participating in these tasks can be robots, hand-held mobile devices or UAVs. The diversity of the platforms, and consequently, the location and type of the imaging sensors, makes achieving the consistency of the quality of the onboard sensors extremely hard. The quality of the visual sensors may vary in the image resolution and camera rates. The quality of the inertial sensors differs in the IMU data accuracy and sampling frequency.

The Collaborative SLAM approach developed by Huai (2017) aims to provide a cooperative SLAM solution based on crowd-sourced data collected by smartphones. To verify that this system meets the design requirements, we should consider the diversity of the quality of the crowd-sourced data.
The main objective of this thesis is to present a method for testing, evaluation as well as validation of the Collaborative SLAM system to assure that it meets the design requirements of the system. This process involves two types of testing, the system performance testing and testing the impact of sensor parameters on the performance. The first testing is a series of tests that examine the qualification and interactive relationship of each unit of the system. The second testing refers to the performance evaluation of the system under several different sensor parameters.

1.3 Methodology

Before starting to plan for system test and evaluation, a description of the SLAM problem and the techniques used is necessary to gain a comprehensive understanding of the problem and the system. This includes how the SLAM problem and the collaborative SLAM is defined, and the current techniques developed to solve these problems. The description of the system is considered as a preparation for the test and evaluation part.

The system performance testing focuses on verifying whether the initially specified requirements of the system are met or not. First, we identify the designed requirements of the Collaborative SLAM system by investigating the system framework and its components (client and server).

After the system requirements are clarified, a method must be established for testing and evaluation to answer whether these demands are met in this system. The question is, how are we able to determine if the system requirements are fulfilled, and what is the convenient test and evaluation approach that should be implemented for verification?
Evaluation metrics are specified for the Collaborative SLAM problem based on its functions.

Sometimes tests are conducted to obtain further insight in a given area. For this system, it is desirable to vary the accuracy, completeness or frequency of the system input data to determine their impacts on total system effectiveness, as well as on computational and communication cost.

To simulate the variation of the crowd-sourced visual-inertial data, we pre-process the original crowd-sourced data and obtain visual-inertial data with different camera frame rates, optical resolutions, and IMU data sampling rates.

1.4 Outline

The thesis is organized as follows:

Chapter 2 is an introduction to sensor models and visual-inertial SLAM problem with different solution approaches. Furthermore, the techniques often used in the visual-inertial SLAM problem are explained.

Chapter 3 involves a description of Collaborative SLAM problem and an in-depth analysis of the solution (Huai, 2017). This includes the investigation of the system’s framework and each of its components. Chapter 2 and 3 are the preparations for the test and evaluation of the Collaborative SLAM system.

Chapter 4 provides the set-up of the testing experiments and evaluation metrics. The experiment results are also presented. Tests are conducted on preprocessed data with various camera frame rates, optical resolution, and IMU data sampling rates.
Chapter 5 summarizes the work in this thesis by including the conclusions of the system test and evaluation.
Chapter 2: Visual-Inertial SLAM

2.1 IMU and Camera

2.1.1 IMU and INS

An inertial measurement unit (IMU) typically comprises sensors of an inertial navigation system (INS), which is defined as the real-time indication of position and velocity of a moving vehicle using sensors that react based on Newton’s laws of motion. There are two types of inertial sensor: the accelerometer that senses linear accelerations, and the gyroscope that senses the angular rate with respect to the inertial frame (Jekeli, 2011). These sensors are independent of any external references and ubiquitous gravity field.

The impact of significant developments in sensor technology is shown in the wide range of applications of accelerometers and gyroscopes in military navigation, guidance, and attitude determination, as well as commercial aviation and automotive industry. The advances in micromachining of mechanical systems have led to the design and manufacture of small, cheap inertial sensors for a variety of other applications that require any linear and angular motion detection (Jekeli, 2011).

Inertial Navigation System (INS) is a system consisting of IMU sensors, the platform where IMU sensors are attached, and the computer which calculates positions, velocities,
and attitudes from sensed accelerations and rotational rates. Position and orientation of a moving body can be obtained by integration of measured linear accelerations and angular rates using three orthogonal accelerometers and three orthogonal gyroscopes. This is the principle behind INS.

The mechanization of INS is the physical arrangement of inertial sensors relative to the vehicle and coordinates frames. There are two kinds of mechanization: stabilized platform and strapdown. In the strapdown mechanization, IMUs are attached to the vehicle frame so that we can ignore the transformations between the reference frame of IMU and the platform.

![Figure 1. Strapdown Mechanization Algorithm (Woodman, 2007).](image)

The orientation of an INS relative to the global frame is tracked by integrating the angular velocity signals obtained from gyroscopes. The computed orientation is used to project accelerations obtained from accelerometers into the global frame. Acceleration due to gravity is then subtracted, and the remaining accelerations are integrated once to obtain velocity, and again to get displacement. The displacement is added to the initial position to get current position.

As shown in Figure 1, the angular velocities are integrated to get the orientation of the INS. Therefore errors in the gyroscopes propagate through the integration to the calculated
orientation. Errors that arise in the accelerometers propagate through the double integration, as described in Figure 1, are the obvious cause of drift in the tracked position. Errors in the angular velocities also cause drift in the tracked position, since a drift error in the orientation of the platform causes an incorrect projection of the acceleration signals into the global frame. The accelerations are integrated into the wrong direction and acceleration due to gravity cannot be eliminated correctly.

2.1.2 Camera and Structure from Motion (SfM)

Imaging sensors record a spatially discrete picture or a continuous one (Ponce et al., 2011). An imaging sensor may record discrete or continuous signal at a point on its retina. Moreover, the signal may consist of a single number (as for a black-and-white camera), a few values (as for a color camera), many numbers (as for the response of hyperspectral sensors), or even a continuous function of wavelength (in the case for spectrometers).

A convex lens is equipped in a real camera and projects a 2D image of the world onto the image plane. In a mostly used pinhole camera model, a projection point $p_i = (u, v, 1)^T$ in the camera’s reference frame is related with a 3D point $P = (X, Y, Z)^T \in \mathbb{R}^3$ or $P = (X, Y, Z, 1)^T \in \mathbb{R}^4$ in some fixed world coordinate system by the perspective projection equations

$$u = k f \frac{X}{Z} + u_0, \quad v = l f \frac{Y}{Z} + v_0$$

(1)

Where $u$ and $v$ are expressed in pixel units, $(u_0, v_0)^T$ defines the position in pixel units of image center, $f$ is a distance expressed in meters, $k$ and $l$ are expressed in pixel × meter$^{-1}$. The parameter $k$, $l$ and $f$ are not independent, and they can be rewrite as $\alpha = kf$ and $\beta = lf$. 

9
expressed in pixel units. This can be written as a projective mapping, up to a scale factor $s$ as

$$sp_i = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & u_0 & 0 \\ 0 & 1 & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot [R \ t] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = CP$$ (2)

Where matrix $C$ is the projection matrix comprising intrinsic parameters $\alpha, \beta, u_0, v_0$ of the camera and the extrinsic parameters of the camera pose $[R \ t]$ with respect to the world. The scale factor is arbitrary and shows that only the projective ray for each image point is known.

While direct depth information is missing in a single perspective view, multiple views have the potential of recovering depth. This is called the structure from motion (SfM). The key issues in SfM are estimating the 3D position of points in the image scene with respect to some fixed coordinate frame, and the pose of the camera for each image frame. SfM and the simultaneous localization and mapping (SLAM) are very much similar in their principles. Visual SLAM is supposed to work in real-time on a time-ordered sequence of images acquired from a fixed camera setup (usually one or two cameras), whereas SfM investigates on an unordered set of images with barely time constraints which can be taken from different cameras.

2.1.3 Inertial and Visual Sensor Fusion

Inertial sensors and cameras can supplement each other due to their complementary sensor characteristics. Inertial sensors cannot work on separating a change in inclination from an acceleration of a body, according to Einstein’s equivalence principle (Corke et al.,
2007). However, cameras are not affected by this. Compared to slow motion, inertial sensors have lower measurement uncertainty at high velocities. In other words, IMU can measure very high velocities and accelerations, which is the weak part of tracking features on consecutive image frames. Although higher camera rate can be used up to a limit, the increasing bandwidth would be a big problem. In contrast, reliable feature tracking can be obtained at low velocities which can make up for the significant measurement uncertainty of IMU at slow motion. From a projective image, a slightly rotational motion cannot be distinguished from a gently translational motion. Also, a near object with low relative speed has the same appearance as an object with high-speed far away.

From these facts, we can see the complementary nature of inertial and visual sensors in the use of estimating the 3D position of points in the scene and 6 DoF motion of cameras.

2.2 Literature Review of VIO and Visual-Inertial SLAM

Simultaneous Localization and Mapping (SLAM) is a concept of building a map of an unknown environment (mapping) while navigating the environment using the constructed map and keeping track of the relative pose of a platform (localization), which in turn help the map growing.

SLAM techniques are widely used in robotics and augmented reality (Davison, 2003). There are many ways to implement SLAM where several types of sensors can be utilized. Most approaches of SLAM usually include using optical cameras, depth sensors, IMU and more. We focus on visual-inertial SLAM, where only a single camera is used to complement positioning and orientation information delivered by IMU.
Visual odometry is the process of solving for the pose of a robot by analyzing the associated camera images. Like visual odometry, visual-inertial odometry fuses information from cameras and inertial sensors to estimate the pose of a platform. Compared with visual odometry, visual SLAM is more than a simple algorithm. Visual odometry focuses on determining the ego-motion of an agent, while visual SLAM aims to estimate motion and to create maps at the same time. There are multiple parts in visual SLAM, extracting landmarks, such as lines (e.g. edges of a table or door) or corners, from each image; associating landmarks in successive frame with 3D objects in camera's environment; matching the observed landmarks to re-associate with objects; estimating the position and orientation of the camera and generating map information about the environment.

Combining visual and inertial measurements is a way to make a full use of the rich information in an image and accurate short-term motion estimation presented by gyroscopes and accelerometers, which are implemented in most smartphones.

Thanks to the complementary nature of cameras and IMU sensors and widespread implementation of these sensors in hand-held mobile devices, the fusion of visual and inertial measurements has attracted significant attentions in finding effective solutions to visual-inertial odometry (VIO) and visual-inertial SLAM. However, it is still a challenging problem to achieve an accurate, robust, continuous (or high sampling rate) real-time outcome while navigating and mapping an extended and complex scene.

The key issue in VIO problems is to estimate the 3D pose (translation and orientation) of a mobile platform relative to its starting position using visual and inertial features. Based
on different techniques used, VIO approaches can be classified into two categories, filtering (Mourikis and Roumeliotis, 2007; Kelly and Sukhatme, 2011; Montemerlo and Thrun, 2002; Crassidis, 2006) and optimization-based (Leutenegger and Lynen, 2015).

The filtering-based approaches are iterative processes, where the current estimation is related to current measurements and past estimations while future measurements do not contribute to the current estimation. IMU measurements are used for state propagation in prediction part, and optical measurements assist in the update part. In filtering-based VIO approaches, stochastic state estimation methods for dynamic systems, such as extended Kalman filter (EKF), Unscented Kalman filter (UKF) or particle filters (PF), are applied.

EKF is the extension of the Kalman filter to nonlinear systems linearizing about current state estimate mean, which is one of the most popular estimation tools in navigation systems and GPS. In the extended Kalman filter, the belief function for state and noise variables are constrained to be Gaussian. In EKF-based VIO, the dynamic model is driven by acceleration and rotational velocity measurements from IMU sensors in three axes, which contribute to motion prediction in prediction step. The key information provided by cameras, such as features extracted from images by feature detectors or direct light intensity with depth information, serves as the measurement model and updates the prediction results in the update step (Gui et al., 2015). Mourikis and Roumeliotis (2007) presented an extended Kalman filter (EKF)-based algorithm, called the Multi-State Constraint Kalman Filter (MSCKF), for real-time vision-aided inertial navigation in large-scale real-world environments and contributed to the derivation of a measurement model.
Figure 2. UKF Principle (Barfoot, 2017). One-dimensional Gaussian PDF $p(x)$ is transformed through a deterministic nonlinear function $g(\cdot)$ using the basic sigmapoint transformation. Here only two deterministic samples ($\mu_x - \sigma_x$ and $\mu_x + \sigma_x$) are selected to approximate the input density. $\mu_x$ and $\sigma_x$ are the mean and the standard deviation of the variable $x$. $\mu_y$ and $\sigma_y$ are the mean and the standard deviation of the variable $y$.

UKF uses the principle that a set of sample points (sigma points) can be used to parameterize mean and covariance of state and noise by a sigmapoint transformation shown in Figure 2, and is implemented without the linearization steps required by the EFK (Julier and Uhlmann, 1997). In UKF, each sample is passed through the nonlinear function and recombined to get an approximation of the transformation of a Probability Density Function (PDF). Two advantages of the UKF are that it is derivative-free and uses only basic linear algebra operations. UKF predicts the state of the system more accurately estimates than the EKF with an equivalent computational complexity. Kelly and Sukhatme (2011) described a UKF-based algorithm for self-calibration of the transform between a camera and an IMU in visual-inertial sensor fusion.

Particle filter implements a prediction-updating scheme using Monte Carlo methodologies to solve filtering problems. The FastSLAM algorithm proposed by
Montemerlo and Thrun (2002) is an efficient approach based on particle filtering. FastSLAM uses a particle filter to sample over robot paths. Each particle possesses $n$ low-dimensional EKFs, one for each of the $n$ landmarks. This algorithm is modified by Montemerlo and Thrun (2007) utilizing a different proposal distribution which makes more efficient use of the particles.

Figure 3. Taxonomy of the Different Filtering Methods and Relationships to the Bayes Filter (Barfoot, 2017).

Figure 3 shows the relationships between the various filtering methods we have discussed above. Each of the methods is considered having a place in a larger taxonomy. The EKF and UKF are related to the full Bayesian posterior by the participant of iteration (Barfoot, 2017).

EKF remains one of the most popular strategies for solving SLAM. However EKF-based SLAM can have inconsistency problem. It is caused by the violation of some
fundamental constraints controlling the relationships between various Jacobians (Huang and Dissanayake, 2007).

SLAM problems can be a pose graph optimization problem. Using nonlinear optimization algorithms can solve the problem efficiently, even when the initial estimate is of poor quality. Optimization approaches are based on optimization of the image alignment with inertial measurement served as prior, regularization terms. An optimization-based approach can be divided into two parts: mapping and tracking. In the mapping part, features are extracted from images using feature detectors and reprojection error is calculated between the two images for all available features. The reprojection errors are employed in the cost function to be optimized. The aim of optimization in the mapping part is to find the 3D coordinates of features or landmarks. In tracking part, the coordinates of features in the map built from mapping part are used to calculate the reprojection error between the two images. The nonlinear optimization algorithm is applied again to find the changes in position and orientation of the moving platform which minimizes the cost function. Leutenegger and Lynen (2015) show that it is possible to do the Gauss-Newton optimization in mapping and tracking part simultaneously.
Chapter 3: System Description and Test Strategy

This chapter gives a brief intuitive description of the collaborative SLAM method to be tested in the experiment. The method developed by Huai (2016) solves the problem of building maps of unknown environments using data collected from multiple users and estimating the pose of each user within these maps. Compared with offline SfM methods, collaborative SLAM has benefits of growing maps in real time, estimating camera pose efficiently, and localizing a user on maps generated by others. The collaborative SLAM method was proposed in a client-server framework where clients processing crowd-sourced data from each user and communicating with the server in a cloud computing network.

3.1 System Description

3.1.1 Overview of the Framework

The system is designed to have the following expected properties of a collaborative SLAM. The system uses a low acquisition bandwidth, represents map in a compact way and operates in real time. A visual-inertial odometry algorithm is deployed on each client to estimate the camera motion and keyframes are selected to be sent to the server. On the
server, maps are generated for each client, merged when there are overlaps between them, and updated when overlaps found in a single map using loop closure technique.

The system consists of multiple clients, a single server and bidirectional communication between clients and server. The client-server framework of the collaborative SLAM system is shown in Figure 4. There are \( n \) clients \((n \geq 1)\) working on the visual-inertial data assigned by users, and these data may or may not be collected with the same device. While reading in the image frames from a video or an image folder, a VIO algorithm is used in a client to keep track of the camera motion in a generated map. In the meanwhile, keyframes are selected to build the client map. Previous keyframes are removed from the client map and keyframe messages which contain keyframe poses, point features in keyframes and inertial data and relevant states, such as velocities and biases, are sent to server continuously.

In Figure 4, \( n+1 \) threads are working on the server, \( n \) of them are keyframe handler thread for each client, and the extra one is a loop closing thread in charge of visual vocabulary, keyframe database and updating SLAM maps. The server receives the keyframe messages from each client and processes them separately on corresponding keyframe handlers. Each handler sustains a map for its client and keeps sending chosen keyframes to the loop closing thread. Loop closure refers to the problem of recognizing a previously-visited location and updating the states and maps accordingly. This can be done in the loop closing thread by storing and comparing the visual words of features, which is from the visual vocabulary (Gálvez-López and Tardos, 2012) from each previously visited location in the keyframe database. Once a loop is detected, the loop closing thread would
update the related maps using nonlinear optimization techniques, such as pose graph optimization and global bundle adjustment. With the updated map information, the server send back messages to help the client get better pose estimations and map generations.

Figure 4. The Client-Server Structure Deployed in the Collaborative SLAM System (Huai, 2017). Each client processes their camera and inertial data using a visual-inertial odometry algorithm, and the keyframe messages are sent from each client to the server containing information about keyframes. The server receives the keyframes message from \( n \) clients and sets up keyframe handler threads for each client together with a loop closing thread to build maps. The loop closing thread detects and closes the loop in maps from each keyframe handler thread using visual vocabulary (Gámez-López & Tardos, 2012) and keyframe database which records the occurrence of visual words from the visual vocabulary in keyframes. The server also sends messages to each client with information used to update camera pose estimations.

### 3.1.2 Operations on a Client

There are two main problems for a client to handle, one is processing visual-inertial data using a VIO algorithm, and the other is selecting and sending keyframes to the server in the form of published keyframe messages. For the online use of the system, the
processing on the client is required to close to real time. For example, the VIO algorithm is needed to process images at 30Hz, and the long-run computation complexity is bounded.

A free open-source visual-inertial SLAM solution, OKVIS (open keyframe-based visual-inertial SLAM) (Leutenegger et al., 2015) is used in the client to solve these two problems. OKVIS is a tightly-coupled visual-inertial fusion method using nonlinear optimization. The loop closure detection in OKVIS is disabled when applying the algorithm to the client part of collaborative SLAM.

The purpose of nonlinear optimization in visual odometry or SLAM is to get the camera poses and landmark’s positions which minimize the reprojection errors of landmarks in camera frames. Temporal constraints between successive camera pose and IMU biases are introduced with the availability of inertial measurements. Except for the reprojection error \( e_r \), another temporal error term \( e_s \) from IMU needs to be added to the cost function \( J(x) \) and minimized in the nonlinear optimization.

\[
J(x) := \sum_{i=1}^{I} \sum_{k=1}^{K} \sum_{j \in I(i, k)} e_{r,i,j,k}W_{r,i,j,k}e_{r,i,j,k} + \sum_{k=1}^{K-1} e_{s,k}W_{s,k}e_{s,k},
\]

where \( i, j \) and \( k \) denote the camera index, landmark index, and camera frame index, respectively. The set \( I(i, k) \) contains the indices of landmarks which are visual in the \( i^{th} \) camera and \( k^{th} \) frame. \( W_{r,i,j,k} \) and \( W_{s,k} \) are the information matrix of the corresponding landmark measurements and IMU errors.

The reprojection error is defined following the standard formulation from Furgale (2011).

\[
e_{r,i,j,k} = z_{i,j,k} - h_{i}(T_{cIS}^{k}T_{SW}^{k}T_{j}^{j})
\]
where $h_i(\cdot)$ denotes the 3D-2D camera projection matrix, $z^{i,j,k}$ represents the measured 2D image coordinates and $w^{i} \in R^4$ stands for homogenous coordinates of the $j$th landmark in world frame. $T_{cis}^{k}$ is the transformation matrix from IMU sensor frame to the $i$th camera frame. $T_{SW}^{k}$ is the transformation matrix from world frame to the IMU sensor frame.

The IMU measurement error term $e^{S}_s(x^k_{R}, x^{k+1}_{R}, z^k_s)$ is a function of platform pose states at image time $k$ and $k+1$, which are denoted as $x^k_{R}, x^{k+1}_{R}$ respectively, and $z^k_s$, all the IMU measurements between successive camera measurements.

The estimated platform states $x_R$ is written as:

$$x_R := [w^T r_s, q^T_{WS}, v^T_{WS}, b^T_g, b^T_a]^T$$

where $w^T r_s$ is the platform position in the world frame, $q_{WS}$ denotes the body orientation quaternion, $v_{WS}$ represents the velocity expressed in the sensor frame, and $b_g$ and $b_a$ are the biases of the gyroscopes and accelerometers, respectively.

For the keypoint extraction and matching, OKVIS applies an improved Harris corner detector (Harris and Stephens, 1988) to enforce a uniform keypoint distribution in the image followed by BRISK descriptor extraction (Leutenegger et al., 2011). A pose estimation is propagated from the last pose using IMU measurements. Given this state prediction, 3D-2D correspondences are found between 3D positions of landmarks in the current local map and 2D positions of landmarks extracted from the current frame using brute-force descriptor matching. Next, 2D-2D point correspondences are established for 2D keypoint association.
For the keyframe selection, a frame is selected as keyframe if the image area spanned by matched points are less than 70% of the area spanned by the extracted keypoints or the ratio of matched versus detected keypoints is less than 35%.

3.1.3 Operations on Server

The main requirements of the server are using received keyframe messages to refines keyframes poses, generate maps in a collaborative way, and sends updated poses to the clients. As stated in section 3.1.1, there are \( n \) keyframe handlers receiving messages from \( n \) clients, and one loop closing thread accomplishing the loop closure detection within or between each client.

**Keyframe Handler**

The initialization of a map is accomplished using the first two received keyframes from the message. The keyframe pose and velocity in the message is transformed to the newly created map on the server. The estimated pose is updated using the least squares method to minimize the reprojection errors of all visible landmarks from current keyframe. Local bundle adjustment is also applied to optimize the states and landmarks of most recent consecutive keyframes.

**Loop Closing Thread**

The loop closing thread robustly detects and closes loops in maps once it recognizes a revisited place. In the loop closing thread, selected keyframes from multiple clients are examined to see whether a loop exists in or between clients. The visual overlap between two keyframes is calculated as a similarity score. The similarity scores grow as the overlap
area increases. A pair of keyframes is considered as a loop candidate if the similarity score is larger than the threshold. A loop detection is defined as three consecutive loop candidates show.

When a correspondence of keyframes is detected as causing a loop, a rigid transformation is calculated as the relative constraints between these two frames. If the loop is detected between two maps in two clients, these two local maps are merged using this calculated rigid transformation. If the loop detection happens within one local map, related keyframes and landmarks are fused, and the pose graph is optimized using the loop constraint.

3.2 Test Strategy

There are two types of testing adopted on the thesis. The system performance testing is the system qualification testing and is conducted on a complete system which is comprised of many units. This testing provides a method to verify and validate the system requirements. The evaluation of this type of testing is considered on an integrated basis. The outputs of each unit (client and server, in this thesis) are examined and evaluated. Another testing is generally performed after the system performance testing. It is an approach to analysis the impact of sensor parameters on system performance. Tests are conducted to gain further insight into the system.

For the collaborative SLAM system, the integrated system testing involves unit testing on the client, server, and client-server communication.

System Performance Testing
The client unit is required to be able to operate in real time (about 30 Hz) and achieve a relatively good accuracy of the estimated keyframe poses. Meanwhile, the client unit could select keyframes from the input visual-inertial data. The testing on the client part aims to verify its real-time processing ability, poses estimated accuracy and keyframes selection function.

In the offline mode of the system, the one-way communication between the client unit and server unit is accomplished through sending keyframe messages from clients to the server. The message, which is recorded on the client side, contains map ID, camera parameters, keyframe poses, velocities and biases, as well as IMU data. The unit testing on keyframe message is to extract information from keyframe messages. The correctness and completeness of the client-server communication can be verified by examining the message length and keypoint distribution.

The server is designed to refine state estimations for each client and close the detected loop within or between maps. The unit testing can be operated on processing multiple keyframe messages generated by clients. The outputs of the server involving keyframe poses, velocities and biases and are analyzed to measure the quality of the algorithm.

**Testing the Impact of Sensor Parameters on System Performance**

This testing aims to determine the impact of sensor parameters of the crowd-sourced data on the accuracy of the estimated states. Parameters can be image blur, camera frame rate, IMU data sampling rate.
**Image blur**

Crowd-sourced datasets can be collected using a variety of platforms, such as smartphones of different brands or models. The optical resolution of the collected data may vary with the platform. Image size changes or unfocused blur of images produced by a lens can cause the change of optical resolution. To emulate the change of optical resolution, a Gaussian blur is added to camera frames. A Gaussian blur is the result of blurring an image by a Gaussian smoothing operator. Gaussian smoothing operator is a 2D convolution operator for blurring images and removing details and noise. The use of Gaussian blur usually makes the image unfocused. It uses a two dimensions Gaussian function and builds a convolution kernel with a Gaussian distribution from the center point of the kernel. By applying the convolution matrix to the original image, each pixel in the filtered image is set to a weighted average of its neighborhood, which indirectly changes the optical resolution of the image.

To study the impact of optical resolution on the performance of the collaborative SLAM system, different sizes of Gaussian blur are added to the collected visual data to emulate different optical resolution.

**Camera frame rate**

To simulate different camera frame rates, we consider only selected image frames as the input visual data. The image frames are selected every two frames, four frames, and so forth. Figure 5 presents examples of different camera frame rates.
Figure 5. Examples of Simulated Different Camera Rates. The highlighted frames are frames used as the visual data for the client unit, and $t$ refers to time. **Top:** Original camera rate. **Bottom:** $\times 0.5$ camera frame rate.

**IMU Data Sampling Rate**

High-frequency sensor data can provide us a rich information about object’s motion. Given that the IMU data is collected using the onboard IMUs in the smartphone, the data may have various frequencies. We study the impact of different frequency on collaborative SLAM under certain velocities and accelerations. The changes in the data frequency are emulated by data sampling. Figure 4. illustrates the simulation of different IMU data frequency.

Figure 6. Examples of Simulated Different IMU Data Sampling Rate. Each stick stands for one piece of IMU data and $t$ refers to time. The highlighted sticks are used as the inertial data for the client unit. **Top:** Original IMU data frequency (upper one). **Middle:** $\times 0.5$ sampling rate. **Bottom:** $\times 0.25$ sampling rate.
Chapter 4: Experiment and Results

In the previous section, we describe the collaborative SLAM system (Section 3.1) and requirements for each component in the system (Section 3.2), which is the client unit, the server unit, and the client-server communication. In this chapter, we first continue with a discussion of the evaluation criterion and test data used in the experiment. Then, the experimental results of the two types of testing discussed in the previous section are presented and reviewed. The performance is studied on image blur, camera frame rate, and IMU data sampling rate.

4.1 Experiment Set-up

The experiments on collaborative SLAM system involve unit testing on the Client side, where a visual-inertial odometry method is deployed, unit testing on Server side, where a collaborative SLAM method is implemented, as well as the unit testing on the client-server communication. Two types of testing are conducted on crowdsourced data. The dataset contains real-world outdoor scenes. It is collected by vehicles equipped with Samsung Galaxy S6 smartphone and GPS antenna/receiver on a parking lot at The Ohio State University. The visual data are recorded at full resolution (1920×1080) and full frame rate.
(30 Hz) of the Samsung Galaxy S6’s front-facing camera (secondary camera). The focal length and shutter speed maintain fixed during the period of collecting data by inactivating the auto focus and auto exposure functions. The inertial data are measured at full sensor resolution (200Hz) of onboard gyroscopes and accelerometers of the smartphone.

GPS positioning solution is calculated using RTKLIB (http://www.rtklib.com/). One long duration kinematic trajectory (~20min) is acquired after a calibration period of 11 minutes. Table 1 provides information about the collected dataset. Figure 7 shows the trajectory collected by SPIN Lab at the Ohio State University. Figure 8 shows the example scenes from the video data. As we can see in Figure 8, the dataset contains a diversity of scenes. One scene type includes structured scenes, which have objects with distinctive edge boundaries (e.g., parking spots on the ground) and the other contains repeated textures (e.g., trees and plants).

The gyroscope bias and accelerometer bias are noted as \(b_g\) and \(b_a\). The rapid fluctuations in the gyroscope and accelerometer signal are modeled with a zero-mean, independent, continuous-time Gaussian white noise of strength \(n_g\) and \(n_a\), respectively. \(n_{bg}\) and \(n_{ba}\) are the Gaussian white noise driving the random walk processes of the gyroscope bias \(b_g\) and accelerometer bias \(b_a\).

\[
\begin{align*}
n_{bg} & : = \dot{b}_g, \quad n_{ba} : = \dot{b}_a
\end{align*}
\]

Table 2 presents the root power spectral density of the IMU parameters stated above (Huai, 2017). Table 3 shows the standard deviation of initial values of IMU biases.
Figure 7. The trajectory of the data which contains several repeated loops.
Table 1. Dataset information

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Duration (s)</th>
<th>Camera frame rate (Hz)</th>
<th>Traveled distance (m)</th>
<th>Image resolution</th>
<th>IMU data frequency (Hz)</th>
<th>GPS solution frequency (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samsung Galaxy S6</td>
<td>678</td>
<td>30</td>
<td>3229.6</td>
<td>1920×1080</td>
<td>200</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2. IMU Characteristics. Root PSD stands for the root power spectral density.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Root PSD</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gyro noise density</td>
<td>$n_g$</td>
<td>$2.4 \times 10^{-3}$</td>
<td>rad/(s$\sqrt{Hz}$)</td>
</tr>
<tr>
<td>Gyro drift noise density</td>
<td>$n_{bg}$</td>
<td>$2.0 \times 10^{-5}$</td>
<td>rad/(s$^2$$\sqrt{Hz}$)</td>
</tr>
<tr>
<td>Accelerometer noise density</td>
<td>$n_a$</td>
<td>$1.6 \times 10^{-2}$</td>
<td>m/(s$^2$$\sqrt{Hz}$)</td>
</tr>
<tr>
<td>Accelerometer drift noise density</td>
<td>$n_{ba}$</td>
<td>$5.5 \times 10^{-5}$</td>
<td>m/(s$^3$$\sqrt{Hz}$)</td>
</tr>
</tbody>
</table>

Table 3. Standard Deviation of IMU Biases Initial Value

<table>
<thead>
<tr>
<th>IMU bias</th>
<th>Standard deviation</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>of initial value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_g$</td>
<td>0.03</td>
<td>rad/s</td>
</tr>
<tr>
<td>$b_a$</td>
<td>0.1</td>
<td>m/s$^2$</td>
</tr>
</tbody>
</table>
Figure 8. Examples of Scenes from Collected Dataset.

4.1.1 Calibration and Synchronization

For our experiment, camera calibration, IMU sensor calibration, and camera-IMU sensor-to-sensor calibration have been already completed, and therefore, not repeated in this experiment. For the camera calibration, the intrinsic camera parameters are estimated based on several frames from collected dataset using Agisoft PhotoScan (http://www.agisoft.com). The time delay between the onboard camera and IMU sensors of the Samsung Galaxy S6 smartphone was determined by aligning smartphone inertial data and video data with Microstrain IMU data which has good GPS timestamps. The synchronization between GPS and visual-inertial sensor are also required for the evaluation part since the GPS positioning solution is adopted as the ground truth for the trajectory in this experiment. The constant time delay between GPS timestamps and video frames were obtained by aligning the gyroscope data with the estimated angular rates using camera video data.

4.1.2 Evaluation Metrics

SLAM is a problem for a moving platform to estimate its trajectory and to generate a map of its surrounding. The outputs of a SLAM system include an estimated trajectory and
a generated resulting map. To evaluate the system, the analysis of these two outputs are necessary. However, the evaluation of a resulting map is much more challenging in practice than in theory, due to the complication of acquiring a ground truth map. Therefore, the SLAM system is evaluated in this experiment only by the quality of its trajectory outcomes, which simplifies the evaluation progress but reserves the probability that a right trajectory does not come with a good map.

Before comparing the ground truth and estimated data of the trajectory, we need to make sure that they are equally sampled and with same data length. Therefore, additional data association and interpolation are necessary given the possibility of different sampling rates, lengths and missing data between these two sequences. The estimated data contains poses, described as $4\times4$ homogeneous transformation matrices, of the platform. The ground-truth GPS data is three-dimensional position vector in Earth-Center Earth-Fixed (ECEF) coordinate system, while the estimated pose of the body frame is expressed in the global frame (which is identical to the body frame at the starting point). After the preprocessing of these two sequences of trajectory, we get an estimated trajectory poses sequence $P_1, P_2, \ldots, P_n$, and a ground truth position sequence $Q_1, Q_2, \ldots, Q_n$. For the ground truth sequences in our experiment, we need to notice that the DGPS ground truth data is three-dimensional position data, without the orientation information. The Absolute Trajectory Error (ATE) and the Relative Pose Error are two error metrics used to evaluate SLAM solution (Sturm, 2012). In this experiment, only ATE is adopted.

*Absolute Trajectory Error (ATE)*
Absolute trajectory error is an evaluation metric to validate a SLAM solution. It measures the global consistency of the estimated trajectory. The absolute trajectory error compares the ground truth position of the platform and the estimated position of the platform at the same time instants. The orientation component of the platform’s pose is not considered in the ATE. Since the two pose sequences are expressed in different coordinate frames, we need to do the data alignment before comparing them. The estimated trajectory is aligned with the ground truth by a transformation, as shown next. $S^Q_P$ is the $4 \times 4$ best-fitting rigid transformation matrix that aligns estimated homogeneous position sequence $P_1, P_2, \ldots, P_n \in \mathbb{R}^3$, and a ground truth homogeneous position sequence $Q_1, Q_2, \ldots, Q_n \in \mathbb{R}^3$.

$$S^Q_P := \arg \min_{S \in SE(3)} \sum_{i=1}^{n} (SP_i - Q_i)^2$$ \hspace{1cm} (7)

The optimal transformation is calculated using algorithm stated in Table 2. Using transformation computed using Equation (7), the absolute trajectory error at position step $i$ can be defined as

$$F_i := S^Q_P P_i - Q_i$$ \hspace{1cm} (8)

For the sequence of $n$ positions, we calculate the root mean square error over the sequence. The RMSE is given as

$$RMSE(F_{1:n}) := \left( \frac{1}{n} \sum_{i=1}^{n} ||F_i||^2 \right)^{1/2}$$ \hspace{1cm} (9)
Table 4. Alignment between Two Sets of Corresponding Points. \( S_P^Q \) is the best-fitting rigid transformation that aligns two sets of corresponding points, \( P = \{ p_1, p_2, ..., p_n \} \) and \( Q = \{ q_1, q_2, ..., q_n \} \). \( S' \) is the best-fitting similarity transformation that aligns these two point sets. \( \bar{p} \) and \( \bar{q} \) are the centroids of point sets \( P \) and \( Q \), respectively. \( R \) is the optimal rotation and \( t \) stands for the optimal translation. \( \sigma \) represents the computed scale factor.

**Algorithm: Least-Squares Points Alignment Using SVD**

**Problem statement:** Let \( P = \{ p_1, p_2, ..., p_n \} \) and \( Q = \{ q_1, q_2, ..., q_n \} \) be two sets of corresponding points in \( \mathbb{R}^3 \). Find a rigid transformation that optimally aligns the two sets in the least squares sense.

\[
S_P^Q := \arg \min_{S \in \text{SE}(3)} \sum_{i=1}^{n} \| Sp_i - q_i \|^2
\]  

(10)

1. Compute the centroids of both point sets:

\[
\bar{p} = \frac{1}{n} \sum_{i=1}^{n} p_i, \quad \bar{q} = \frac{1}{n} \sum_{i=1}^{n} q_i.
\]  

(11)

2. Compute the centered vectors \( x_i \) and \( y_i \):

\[
x_i := p_i - \bar{p}, \quad y_i := q_i - \bar{q}, \quad i = 1, 2, ..., n
\]  

(12)

3. Compute the \( 3 \times 3 \) covariance matrix \( H \). \( X \) and \( Y \) are the \( 3 \times n \) matrices that have \( x_i \) and \( y_i \) as their columns, respectively.

\[
H = XY^T,
\]  

(13)

4. Compute the singular value decomposition

\[
H = U \Sigma V^T,
\]  

(14)

and the \( 3 \times 3 \) rotation matrix is

\[
R = VU^T
\]  

(15)

5. Compute the scale factor \( \sigma \)

\[
a_i := Rx_i
\]  

(16)

Continued
Table 4. continued

\[ \sigma = \frac{\prod_{i=1}^{n} a_i q_i}{\prod_{i=1}^{n} a_i \| a_i \|^2} \]  \hfill (17)

1. Compute the 3×1 translation vector
\[ t = \bar{q} - R \bar{p} \]  \hfill (18)

2. The 4×4 rigid transformation matrix is
\[ S_p^q = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \]  \hfill (19)

The 4×4 similarity transformation matrix is
\[ S' = \begin{pmatrix} R & t \\ 0 & \sigma \end{pmatrix} \]  \hfill (20)

4.2 System Performance Testing

This section shows the experiment and results of the system performance testing (described in Section 3.2) on the collaborative SLAM system. In this experiment, camera frames are down-sampled by a factor of 2 (see Table 5).

Table 5. Characteristics of Data Segments.

<table>
<thead>
<tr>
<th>Image resolution of segments</th>
<th>Start video index</th>
<th>End video index</th>
<th>Traveled distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start-segment 1</td>
<td>(1920/2) × (1080/2)</td>
<td>20000</td>
<td>28900</td>
</tr>
<tr>
<td>Start-segment 2</td>
<td>(1920/2) × (1080/2)</td>
<td>20000</td>
<td>32000</td>
</tr>
<tr>
<td>End-segment</td>
<td>(1920/2) × (1080/2)</td>
<td>28700</td>
<td>40339</td>
</tr>
</tbody>
</table>

The visual-inertial data are divided into two segments, start-segment one and end-segment for the simulation of multiple clients. Note that these two segments have an
overlap of 2000 frames. The trajectories of the start-segment one and end-segment are shown in Figure 9. Another data segment, start-segment two is used in the performance testing of the client unit.

Figure 9. GPS trajectories of the dataset collected at The Ohio State University. (a) Start-segment one trajectory (blue) and End-segment trajectory (red). (b) Start-segment one Trajectory. (c) End-segment trajectory
Testing on Client Unit

The start-segment one data are used as the input of the testing on the client unit. Point based comparison results are shown in Table 7 and Figure 11. The estimated position and GPS reference position are expressed in the local North-East-Down frame with the origin at the starting point. It was observable that the estimated trajectory has significant drift at the beginning, see Figure 10 (c). In Figure 10 (b), the error reached around 30 meters in about 10 seconds. This is due to the fact that the initial velocity of this data segment is not zero and it needed time for the visual-inertial odometry algorithm to get the velocity estimation to converge. Also, the visual-inertial algorithm in the client has no loop closure constraint in this segment, and thus, the drifts due to the velocity initialization were preserved and maintain unfixed in the client. See Figure 10 (a), compared with a beginning portion, which had a significant error, the rest of the trajectory displays better shape and scale.

Table 6. Error statistics of point-based comparison between the estimated positions and GPS reference solution.

<table>
<thead>
<tr>
<th></th>
<th>3D point distance</th>
<th>e_x</th>
<th>e_y</th>
<th>e_z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (m)</td>
<td>31.3</td>
<td>40.8</td>
<td>18.6</td>
<td>41.4</td>
</tr>
<tr>
<td>Mean(m)</td>
<td>37.1</td>
<td>65.9</td>
<td>65.7</td>
<td>40.1</td>
</tr>
<tr>
<td>RMSE (m)</td>
<td>41.4</td>
<td>19.2</td>
<td>36.3</td>
<td>46.3</td>
</tr>
</tbody>
</table>
Figure 10. (a) Top view of the estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. (b) 3D position error at each epoch. (c) Zoom in of the starting point surrounding.

To get a better view how the trajectory drifts over time, we fixed the velocity initialization problem with the help of the initial velocity from the GPS reference solution. And in this comparison, we used the start-segment two, which is longer. The resulting trajectory and GPS reference trajectory are converted to the local North-East-Down frame.
of the starting point, see Figure 11. The error statistics of this point-based comparison is summarized in Table 7. As can be seen in Figure 11 (b), the overall trend of the error is increasing as a function of time. The cyclic pattern in this error is studied later. It is clear that the last loop was not closed, resulting in significant drifts from the GPS reference solution.

Figure 11. (a): Top view of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. (b): 3D position distance error is calculated at each epoch. (c): 3D view of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point.
Table 7. Error statistics of point-based comparison between the estimated positions and GPS reference solution.

<table>
<thead>
<tr>
<th></th>
<th>3D point distance</th>
<th>$e_x$ (m)</th>
<th>$e_y$ (m)</th>
<th>$e_z$ (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median</td>
<td>59.2</td>
<td>40.4</td>
<td>28.5</td>
<td>-0.8</td>
</tr>
<tr>
<td>Mean</td>
<td>62.9</td>
<td>45.3</td>
<td>33.4</td>
<td>-1.9'</td>
</tr>
<tr>
<td>RMSE</td>
<td>73.7</td>
<td>54.7</td>
<td>49.3</td>
<td>4.1</td>
</tr>
</tbody>
</table>

Figure 12. (a): Zoom in between 160 seconds and 245 seconds of Figure 11 (b). The peak values and valley values are marked. (b): The corresponding portion of estimated trajectory and GPS reference trajectory in the local North-East-Down frame of the starting point. Peak points and valley points from (a) are noted in the two trajectories. Pink lines between the trajectories represent the error vectors at every 10 points.

As seen in Figure 11(b), aside from the increasing trend over time, the error also displays cyclic patterns. We selected a portion of the trajectory to study the reason for this pattern, see Figure 12. The valley values and peak values were marked in (a), and a 3D view of the portion was presented in (b) together with the valley points and peak points from (a). As can be seen in (b), the length of the error vectors (pink lines) enlarged from the 1st point to the 2nd point, dropped a little from the 2nd point to the 3rd point, grew again...
to the 4\textsuperscript{th} point, and had a sudden decline between the 4\textsuperscript{th} point and the 5\textsuperscript{th} point. The length of the vector was affected by the misalignment between the two trajectories and thus caused the cyclic pattern.

In the point-based comparison, the estimated trajectory and GPS reference trajectory are compared in a local NED frame, which represents an absolute localization error and shows how the trajectory drifts over time. In the following context, we use the overall fitted rigid transformation between the two trajectories to gain an understanding of the relative position error. Figure 13 shows the top view comparison of estimated trajectory of the GPS reference solution with the estimated trajectory after a rigid transformation. Table 8 presents quantitative results of absolute trajectory error and scale error. Table 9 provides the error statistics for the estimated trajectory on the client. The visual-inertial odometry on client side reaches a stable frame processing rate about 25 Hz, which meets the real-time requirement. Given a trajectory scale of 0.94, the fusion of IMU measurements solves the scale problem of monocular visual odometry. Given the ATE of 26.4m and ratio of 1.6\%,

we consider that the visual-inertial odometry algorithm in the client unit can provide a relatively fair position estimation of the trajectory. However, the accuracy of the VIO algorithm is not sufficient for personal navigation or vehicle navigation.

The average number of keyframes selected per second is about 8. This number is related to the object space. In this experiment, the video scene contains sky (sparse keypoint distribution), trees and cars (relatively dense keypoint distribution), and ground (keypoint centered along the parking lines). In the VIO algorithm, a frame is selected as
keyframe if the image area spanned by matched points is less than 70% of the area covered by the extracted keypoints.

Figure 13. **Left:** Top view of trajectory estimated by the VIO on Client aligned to the GPS reference solution with the same starting point. **Right:** Top view of trajectory estimated by the VIO on Client aligned to the GPS reference solution with a rigid transformation using algorithm stated in Table 4.

Figure 14. Error between client-side position estimates (after a rigid transformation) and GPS reference positions in x, y, and z directions
Table 8. Error Metrics of the VIO Algorithm on Client. Note for the ATE, the ratio is the error divided by the trajectory length.

<table>
<thead>
<tr>
<th>Start-segment data</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Frames</td>
<td>12000</td>
</tr>
<tr>
<td># of Keyframes</td>
<td>3247</td>
</tr>
<tr>
<td># of Keyframes per Second</td>
<td>8.1</td>
</tr>
<tr>
<td>Trajectory Length (m)</td>
<td>2332.0</td>
</tr>
<tr>
<td>Runtime (s)</td>
<td>481.928</td>
</tr>
<tr>
<td>Frame rate (Hz)</td>
<td>24.9</td>
</tr>
<tr>
<td>ATE (m) / Ratio</td>
<td>26.42 / 1.13%</td>
</tr>
<tr>
<td>Scale error</td>
<td>0.94</td>
</tr>
</tbody>
</table>

Table 9. Error Statistics for Estimated Trajectory on Client. 3D point distance refers to the distance between the estimated position (after a rigid transformation using algorithm stated in Table 4) and the ground truth position. \( e_x, e_y, e_z \) refers to errors between the x, y, z components of the estimated trajectory with the x, y, z components of the ground truth data. RMSE stands for root mean square error.

<table>
<thead>
<tr>
<th>3D point distance</th>
<th>( e_x )</th>
<th>( e_y )</th>
<th>( e_z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (m)</td>
<td>17.9</td>
<td>-2.3</td>
<td>-2.6</td>
</tr>
<tr>
<td>RMSE (m)</td>
<td>26.4</td>
<td>17.9</td>
<td>17.6</td>
</tr>
</tbody>
</table>

*Testing on Communication Unit*

To verify the completeness and correctness of the client-server communication context, the keypoint information in keyframe messages generated at client side is extracted and examined. The length of messages was according to the number of keyframes selected by
the client. As shown in Figure 15, corners on the tree and along the parking plot were correctly detected and stored in the keyframe message. We consider that the communication unit between clients and server meets the designed requirement stated in Section 3.2.

![Figure 15. Examples of corner information contained in keyframe messages. The green boxes are the detected corner in the keyframe.](image)

*Testing on Server Unit*

The server unit was tested on two keyframe messages generated by two clients using start-segment one and end-segment data respectively. The server processed the keyframe messages from two clients simultaneously and detected loops and merged the local maps successively. The top view of the comparison between estimated trajectory and GPS ground truth trajectory are shown in Figure 17. The shape of the estimated trajectory was very close to the reference GPS trajectory. The scale of the estimated trajectory was not well calculated for the entire time. The left figure in Figure 16 provides the distance error between estimated positions with GPS reference positions in the local North-East-Down frame of the starting point. Errors in the loop of departure have lower peak values than
errors in the last loop. It is evident in Figure 17 (b) that the second top long loop has a better estimation than the first top long loop. The scale difference of certain loop was observable from the figure.

Table 11 provides evaluation results of the server unit testing, revealing the clear improvement when using the Collaborative SLAM approach. The ATE of the estimated trajectory is 8.7, and the ratio is 0.26% which is much better than the results on the client. As expected, the loop closure method detected and closed all the loops, thus constrained the absolute trajectory error within 10 meters. The number of keyframe on the server was significantly reduced with the contribution of the loop closing thread on the server. The number of keyframes on the server is 1180, which is much less than 4211, the total amount of keyframes provided by two clients. The average keyframe selected per second is 1.7, which is much less than the average keyframe selected per second of 8 in the client. This is because that the repeated keyframes from the revisiting of loops are discarded in the server.

As shown in Figure 9, there are some repeat loops between the start-segment and the end-segments of trajectory data.

Figure 18 shows the error between estimated position after a rigid transformation and GPS ground truth in x, y, and z directions. As shown in the figure, there are points with large error in either x, y, or z direction exists in the trajectory. Table 12 summarizes the statistic results of the error in Figure 18. Given of median error in each direction is around 1 meter, the estimated position is of good accuracy. However, the root mean square errors in each direction indicate that points with significant position error affect the overall accuracy of the trajectory.
Figure 16. **Left:** Point based comparison between the estimated positions and GPS reference solution in a local NED frame. Errors are calculated as 3D point distance at each epoch. **Right:** Top view of estimated trajectory and GPS reference trajectory in the local NED frame of the starting point.

Table 10. Error statistics of point-based comparison between the estimated positions and GPS reference solution.

<table>
<thead>
<tr>
<th></th>
<th>3D point distance</th>
<th>$e_x$</th>
<th>$e_y$</th>
<th>$e_z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (m)</td>
<td>21.2</td>
<td>-5.7</td>
<td>-18.3</td>
<td>0.8</td>
</tr>
<tr>
<td>Mean (m)</td>
<td>22.9</td>
<td>-3.8</td>
<td>-19.6</td>
<td>0.94</td>
</tr>
<tr>
<td>RMSE (m)</td>
<td>25.4</td>
<td>11.1</td>
<td>22.8</td>
<td>1.1</td>
</tr>
</tbody>
</table>
Figure 17. **Left:** Top view of trajectory estimated by Collaborative SLAM aligned to the linear-interpolated GPS reference trajectory with the same starting point. **Right:** Top view of trajectory estimated by Collaborative SLAM aligned to the linear-interpolated GPS reference trajectory with a rigid transformation using algorithm stated in Table 4. The blue point density (represents the density of keyframe) in the figure is much less than the blue point density in Figure 13.

<table>
<thead>
<tr>
<th>Table 11. Error metrics of the Collaborative SLAM. Note for the ATE (absolute trajectory error), the ratio is the error divided by the trajectory length.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value</strong></td>
</tr>
<tr>
<td># of Keyframes from Client 1</td>
</tr>
<tr>
<td># of Keyframes from Client 2</td>
</tr>
<tr>
<td># of Keyframes on Server</td>
</tr>
<tr>
<td># of Keyframes per Second</td>
</tr>
<tr>
<td>Trajectory Length (m)</td>
</tr>
<tr>
<td>ATE (m) / Ratio</td>
</tr>
<tr>
<td>Scale</td>
</tr>
</tbody>
</table>
Figure 18. Error between Collaborative SLAM position estimates (after a rigid transformation) and GPS ground truth in x, y, and z directions

Table 12. Error Statistics for Estimated Trajectory on Sever. 3D point distance refers to the distance between the estimated position (after a rigid transformation using algorithm stated in Table 4) and the ground truth position. $e_x$, $e_y$, $e_z$ refers to the errors between the x, y, z components of the estimated trajectory with the x, y, z components of the ground truth data. RMSE stands for root mean square error.

<table>
<thead>
<tr>
<th></th>
<th>3D point distance</th>
<th>$e_x$</th>
<th>$e_y$</th>
<th>$e_z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median (m)</td>
<td>6.65</td>
<td>-1.23</td>
<td>0.84</td>
<td>1.41</td>
</tr>
<tr>
<td>RMSE (m)</td>
<td>8.71</td>
<td>6.54</td>
<td>3.49</td>
<td>4.58</td>
</tr>
</tbody>
</table>
4.3 Effect of Sensor Parameters on Performance Testing

The experimental results of operational testing described in Section 3.2 are provided in the following context to investigate the robustness on the system performance on a selected set of parameters.

*Influence of Image Blur*

Three varied sizes of Gaussian filtering were applied to the video frames before the operation on clients began. The image data used in this experiment has a resolution of 960×540. Figure 19 shows the accumulated position errors when blurring images with distinct sizes of the kernel while keeping the same frame rate (30Hz or 15Hz) and the same IMU frequency (200Hz). Clearly, using this dataset, a small size of image blur kernel, such as 3×3 or 5×5, helped improved the accuracy of the approach, which can also conclude from Table 12. The absolute trajectory error under a 3×3 kernel size of image blur, which is 5.4 (camera frame rate of 30Hz) and 14.9 (camera frame rate of 15Hz), shows improvement from the original result (with image blur), which is 8.7 (camera frame rate of 30Hz) and 22.4 (camera frame rate of 15Hz). This is due to the property of Gaussian filtering which removes the high-frequency noise in the image. The keypoint detector and descriptor worked better with the elimination of noise. However, a larger size of blur kernel affected the system in a negative way by removing more corner information. The tracking by the visual-inertial odometry on the client even lost when 13×13 or larger size of Gaussian filtering were applied. Note that the definition of ‘small’ and ‘large’ is related to the scale of the object in the video. In the case that camera is far away from objects, it is still possible that a 3×3 Gaussian blur declines the system accuracy by removing too much
keypoint information. For the dataset collected with a camera located near to objects, a 9×9 Gaussian blur would not have such considerable influence as in this experiment. The selection of Gaussian blur size is relevant to the object size in the video.

Figure 19. Different image blur size and camera frame rate. Accumulated position error between GPS reference trajectory with estimated trajectory after a rigid transformation when changing the image blur size and camera frame rate. G1 stands for no image blur and G3, G5, G9 represent a 3×3, 5×5 or 9×9 Gaussian blur applied to the image frame, respectively.

The influence of a large size, such as 9×9, image blur was also shown in the trajectory scale. Table 12 provide scale errors of the estimated trajectory under camera rate of 30 fps. The scale factor of estimated trajectory almost reached 1.5 in the case of applying a 9×9 Gaussian blur to the experiment data under a 30-fps camera frame rate. The scale estimation of the Collaborative SLAM causes a severe problem when the input visual data is blurred.
Table 13. Error metrics of the Collaborative SLAM Trajectory under the impact of image blur, camera frame rate and IMU data sampling frequency. The first two rows have IMU sampling rate of 200Hz. The last row has camera frame rate of 30fps. ATE is the absolute trajectory error. The scale is calculated using Equation (17).

<table>
<thead>
<tr>
<th></th>
<th>G1 FPS=30</th>
<th>G3 FPS=30</th>
<th>G5 FPS=30</th>
<th>G9 FPS = 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATE (m)</td>
<td>8.71</td>
<td>5.43</td>
<td>7.20</td>
<td>30.48</td>
</tr>
<tr>
<td>Scale</td>
<td>0.96</td>
<td>0.95</td>
<td>0.93</td>
<td>1.48</td>
</tr>
<tr>
<td></td>
<td>G1 FPS=15</td>
<td>G3 FPS=15</td>
<td>G5 FPS=15</td>
<td>G9 FPS = 15</td>
</tr>
<tr>
<td>ATE (m)</td>
<td>22.40</td>
<td>14.86</td>
<td>16.10</td>
<td>46.30</td>
</tr>
<tr>
<td>Scale</td>
<td>0.80</td>
<td>1.19</td>
<td>1.19</td>
<td>2.04</td>
</tr>
<tr>
<td></td>
<td>G1 100Hz</td>
<td>G3 100Hz</td>
<td>G5 100Hz</td>
<td></td>
</tr>
<tr>
<td>ATE (m)</td>
<td>14.34</td>
<td>4.02</td>
<td>32.57</td>
<td></td>
</tr>
<tr>
<td>Scale</td>
<td>1.18</td>
<td>0.97</td>
<td>1.47</td>
<td></td>
</tr>
<tr>
<td></td>
<td>G3 50Hz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATE (m)</td>
<td>9.65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scale</td>
<td>0.95</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 20 and Figure 21 report the statistics on the absolute error (norm of position error) as a function of time traveled. The $3\times3$ Gaussian blur shows a notably better performance than the original one or a larger size of Gaussian blur on improving the accuracy of collaborative SLAM.
Figure 20. Position error statistics of the image blur: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. G1 stands for no image blur and G3, G5, G9 represent a 3×3, 5×5 or 9×9 Gaussian blur applied to the image frame, respectively. The camera frame rate is constant and equals to 30 fps in the figure.

Figure 21. Position error statistics of the image blur: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. The camera frame rate is constant and equals to 15 fps in the figure.
Influence of Camera frame rate

A standard 30Hz camera frame rate and a 15Hz camera frame rate were adopted in this experiment, with several sizes of image blur. A smaller camera frame rate such as 10Hz or 7.5Hz of the experiment data caused the server breakdown due to the algorithm configuration or some unfixed bugs. Thus, only two different camera frame rates are discussed in this thesis. Accumulated position errors are shown in Figure 19. Lower frame rate, which means less overlapping between consecutive frames, had adverse effects on the system accuracy. In this experiment, given an average velocity of 5 m/s, the effect of camera frame rate is much less than the effect of a large size of image blur. The impact of camera frame rate would be significant with the increase of platform velocities.

Influence of IMU Sampling Rate

Three different sampling rate were applied to the IMU data in the experiment. Figure 22 displays a comparison regarding accumulated position errors of applying 200Hz, 100Hz and 50Hz IMU data sampling rate to the result as shown above with the original settings. Figure 23 provides the accumulated position errors when blurring images with distinct sizes of the kernel while keeping the same IMU data sampling rate (200Hz or 100Hz). Interestingly, the test of 100Hz IMU data with video data blurred with 3×3 Gaussian kernel outperformed all the other tests, while a high sampling rate (200Hz) seems to perform slightly worse in this respect. This may be due to erratic behaviors such as selection of point correspondences in the algorithm. We can also conclude that under an average velocity of 5 m/s, the effect of IMU data frequency on the system performance is slight from the result that 100Hz outperformed the 200Hz.
Figure 22 shows that the result of 50Hz IMU sampling rate combined with 30Hz camera frame rate is better than the result of 200Hz IMU sampling rate coupled with 15Hz camera frame rate, which indicating that the influence of IMU data frequency is much less than the camera frame rate. This may due to the relatively high frequency of the IMU data compared to the visual data. The contribution on scale estimation of IMU measurements is also shown in Table 13 that the 100Hz IMU data sampling rate (the second row) relates to larger scale drift.

Figure 22. Different camera frame rate and IMU frequency. Accumulated position error when changing the camera frame rate and IMU frequency. G3 stands for a 3×3 Gaussian filter which applied to image frames.
Figure 23. Different image blur size and IMU frequency. Accumulated position error when changing the image blur size and IMU frequency. G1 and G3, G5 represent no blurring, 3×3 or 5×5 Gaussian blurring applied to the image frame, respectively.

Figure 24. Position error statistics of the IMU data sampling rate: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. G1 stands for no image blur and G3, G5 represent a 3×3, 5×5 Gaussian blurring applied to the image frame, respectively. The camera frame rate is constant and equals to 30 fps in the figure.
We tested the collaborative SLAM system with two clients using the start-segment and end-segment of the dataset with different sensor parameters. To cover different situations, more tests are conducted to evaluate the performance of the system when two clients use visual-inertial data with various camera frame rate, image blur kernel size and IMU data sampling rate. The error metrics are summarized in Table 14.

Table 14. Error metrics of the Collaborative SLAM Trajectory under the impact of image blur, camera frame rate and IMU data sampling rate. The default camera frame rate is 30fps and default IMU sampling rate is 200Hz, if not mentioned. ATE is the absolute trajectory error described in Equation (9). The scale is calculated using Equation (17). The first element accepted by operator “+” is the data parameter of the first client, and the second element is the data parameter of the second client. For example, G3+G9 (30FPS) stands for the case that the first client uses 30fps start-segments data filtered by a 3×3 Gaussian smoother, and the second client uses 30fps end-segment data filtered by a 9×9 Gaussian smoother, both clients adopted a 200Hz IMU data sampling rate.

<table>
<thead>
<tr>
<th></th>
<th>G3 + G9 (30FPS)</th>
<th>G3 + G9 (15FPS)</th>
<th>30FPS+15FPS (G3)</th>
<th>200Hz+100Hz (G3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATE (m)</td>
<td>13.0383</td>
<td>8.7811</td>
<td>8.6173</td>
<td>17.9612</td>
</tr>
<tr>
<td>Scale</td>
<td>0.9456</td>
<td>1.0903</td>
<td>1.0934</td>
<td>0.8119</td>
</tr>
</tbody>
</table>

The scale drift which showed in the cases that both clients have a 9×9 kernel size of image blur was improved by the alternation of one 3×3 kernel size of image blur. The client with less image information (after filtering the image with a 9×9 Gaussian smoother) benefited from the client with sufficient image information by removing the scale drift. The scale estimation using 15fps camera data was also improved by the participant of another client with 30fps camera data, which also contributed to smaller absolute trajectory error. The results proved that the cooperative localization contributed to improving individual
localization accuracy. However, the combination of 200Hz IMU sampling rate and 100Hz IMU sampling rate perform worse in both the scale and the position estimation.

Figure 25. Position error statistics of two clients using different camera frame rate, image blur or IMU data sampling rate: median, 5th and 95th percentiles. Position error refers to the distance between estimated position and its corresponding GPS reference position. G3 represent a 3×3 Gaussian blurring applied to the image frame. 200Hz, 100Hz are the IMU data sampling rate.
Chapter 5: Conclusions and Recommendations

5.1 Conclusions

As indicated in Chapter 1, the main object of this thesis is to test and evaluate a Collaborative SLAM approach proposed in (Huai, 2017). Two types of testing are conducted on the visual-inertial dataset collected by the SPIN lab at The Ohio State University.

System performance testing aims to verify and validate that the proposed system meets the designed requirements of the system described in Section 3.2. As the experiment in Chapter 4.2 demonstrated, the visual-inertial odometry method on the client unit achieved a close to real-time frame processing rate. The data used in the experiments was collected and sensor calibrated by the SPIN Lab at The Ohio State University. By using this data, the estimation accuracy is around 27 meter, which is not suitable for personal navigation or vehicle navigation. The client-server communication unit provided effective, complete and correct keyframe messages to the server which is proved by extracting and examining the keypoint information contained in the keyframe messages. Moreover, the server unit obtained a higher accuracy position estimates of 8 meter with the help of loop closure constraints.
Testing the impact of sensor parameters on system performance provides further studies of the robustness and sensitivity of the system. We showed how optical resolution, camera frame rate, and IMU data sampling rate do affect the navigation accuracy. Under a slow motion such as 5 meters per second velocity in this experiment, when employing lower IMU frequency, such as 100Hz or 50Hz, we do not observe a dramatic performance decrease. The impact of camera frame rate is also somewhat similar to that of the moving velocity of the platform. In this experiment, the reduction in camera frame rate from 30Hz to 15Hz showed a not significant effect on the accuracy. However, under a higher velocity, the impact of camera frame rates would show a larger decline in the accuracy.

Moreover, a small size of image blur kernel such as 3×3 Gaussian filter helps to improve the accuracy by eliminating the high-frequency noise in the image. However, a larger kernel size of image blur shows a significant effect on the accuracy, especially on the scale estimation. The effect of Gaussian blur is related to the object size in the video frame. For another dataset which has smaller object size compared to the object size in the dataset we used, a 3×3 Gaussian blur might not improve the estimation accuracy due to the loss of enough keypoint information.

5.2 Recommendations

In this thesis, the Collaborative SLAM approach is tested on a single dataset. To thoroughly evaluate the system, more datasets are needed to verify the system performance under different motion state and object space.
The Collaborative SLAM approach is tested on a deployment of two clients. More tests may be needed to evaluate the system performance on running more clients and maintaining more maps on the server. The situations of multiple clients with different quality visual-inertial data may be worth to be investigated.

The Collaborative SLAM system works in an offline mode, where client and server do not run simultaneously. Tests need to be conducted in the online mode to get complete validation of the system.

The outcomes of the Collaborative SLAM system involve the pose estimates and the map. Experiments in this thesis only provide test and evaluation results of the position estimates, due to the unavailability of a full 6DOF ground truth trajectory and a map of the surrounding environment. Future works can be done on calculating a reliable 6DOF pose trajectory and generating a detailed reference map for this dataset. The pose estimates and landmark estimates still need testing and evaluation.
References


