FEDERAL UNIVERSITY OF SANTA CATARINA

DEPARTMENT OF AUTOMATION AND SYSTEMS ENGINEERING

GRADUATE PROGRAM

Seyed Jamal Haddadi

Improvement of Visual-Inertial ORB_SLAM using Correction in Initial States Estimation

Florianópolis

2021

Seyed Jamal Haddadi

Improvement of Visual-Inertial ORB_SLAM using Correction in Initial States Estimation

Thesis submitted to the Postgraduate Program at the Federal University of Santa Catarina to obtain the title of Doctorate degree in Control, Automation Engineering and Systems.
Coordinator: Dr. Eugênio de Bona Castelan Neto

Florianópolis
2021

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Haddadi, Seyed Jamalaldin
Improvement of Visual-Inertial ORB-SLAM using Correction in State Estimation / Seyed Jamalaldin Haddadi ; orientador, Eugenio de Bona Castelan Neto, coorientador, Werner Kraus Junior, 2021.
104 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia de Automação e Sistemas, Florianópolis, 2021.

Inclui referências.

1. Engenharia de Automação e Sistemas. 2. Visual Inertial ORB-SLAM. 3. Initialization. 4. Benchmark. 5. Initialization. I. Castelan Neto, Eugenio de Bona. II. Junior, Werner Kraus. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Seyed Jamalaldin Haddadi

Improvement of Visual-Inertial ORB-SLAM using Correction in State Estimation

Florianópolis, 2021.

This work is dedicated to my dear parents.

Thanks

Throughout the writing of this thesis, I have received a great deal of support and assistance.

I would first like to thank my supervisor, Professor Eugênio B. Castelan, whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. Also, you provided me with the tools that I needed to choose the right direction and successfully complete my dissertation.

I would particularly like to single out my teacher at linear dynamic course, Professor Daniel Coutinho who, kindly guided me in this course.

In addition, I would like to thank my parents for their wise counsel and sympathetic. You are always there for me.

RESUMO

O objetivo desta tese de doutorado é melhorar a precisão de posicionamento de sistemas robóticos que usam uma câmera e IMU (sistema visual-inercial) para se localizar em um ambiente interno. Para este fim, ORB-SLAM2 é escolhido como a solução mais confiável e completa para Visual SLAM monocular, e como o mais representativo SLAM visual de última geração que é usado para se fundir as informações IMU. Desta forma, decidimos aumentar a precisão na estimativa dos estados iniciais, tendo um impacto muito importante no resultado final. Uma vez que a estratégia é usar a otimização fortemente acoplada no sistema SLAM visual-inercial, dois parâmetros ajustáveis L e P são incorporados na equação de estimativa de estados. Esses dois parâmetros são utilizados como coeficientes do primeiro termo e do termo de gravidade da fórmula proposta, respectivamente. A razão para empregar esses parâmetros é de obter graus de liberdades adicionais para regular o sistema com base no nível de dificuldade do ambiente para obter o melhor desempenho na busca dos melhores valores de inicialização, levando à melhor precisão na saída final. Para avaliar o desempenho do algoritmo proposto, um teste de *benchmark* usando o conjunto de dados EuRoC é executado e seus resultados comprovam o papel eficiente de parâmetros ajustáveis adicionais para melhorar a precisão. Além disso, a análise de Hardware - uso da CPU durante a implementação - por dois hardwares diferentes é outra prática realizada nesta tese que mostra como o tipo de ambiente afeta o uso da CPU. Finalmente, os parâmetros ajustáveis obtidos no *benchmark* são aplicados em um experimento do mundo real. Este experimento foi realizado com realizada com sucesso, sem falhar e perder os recursos durante o experimento. Além disso, a redução do tempo de inicialização é outra conquista deste estudo que, em comparação com o V-SLAM / VI-SLAM de última geração.

Palavras-chave: Visual-inercial. SLAM. Inicialização de estado. Dados de EuRoC. Parâmetros ajustáveis.

RESUMO EXPANDIDO

Introdução

À medida que dispositivos artificiais - carros autônomos, quadricópteros, robôs em tamanho real ou mesmo sistemas de realidade virtual e aumentada - começam a interagir ou se adaptar ao mundo 3D ao nosso redor, eles precisam da capacidade de perceber, reconstruir e, finalmente, entendê-lo de forma semelhante. Tome o Veículo Aéreo Não Tripulado (UAV) como exemplo, cujo interesse em controlá-lo e convertê-lo em um robô inteligente está aumentando significativamente entre os pesquisadores. As aplicações desses robôs foram amplamente definidas em tarefas como monitoramento de tráfego, operações de resgate, vigilância, transporte e hobistas. Os robôs que podem operar autonomamente em ambientes internos / externos são concebidos para serem úteis para várias aplicações que foram mencionadas acima. Além de todas essas aplicações, um imperativo para um robô autônomo é a capacidade de auto-localização no ambiente. Devido à ausência do Sensor de Posicionamento Global (GPS) no ambiente interno e à falta de auto-localização, a manobra autônoma do Quadrotor enfrenta mais desafios do que ao ar livre. Assim, é necessário empregar estratégias alternativas de localização para cumprir a função do GPS em ambientes internos. A forma mais simples é utilizar técnicas de Odometria adequadas para estimativa de movimento de curto prazo, mas se o objetivo for a localização no mesmo ambiente para longo prazo, seria desejável ter um mapa, permitindo uma localização sem deriva. Mapeamento é o processo de criação do mapa usando sensores integrados, resultando em localização conhecida. É importante que este mapa seja obtido simultaneamente ao mesmo tempo que o robô navega. Esse problema é conhecido como Localização e Mapeamento Simultâneo (SLAM) ou Estrutura e Movimento (SaM). Por exemplo, suponha que a informação anterior de um ambiente interno seja considerada como um mapa que foi usado na localização visual e navegação. Considerando essa suposição em termos de mapeamento, às vezes, por causa de restrições ambientais (por exemplo, caso de relevo), é difícil navegar por um mapa preciso do ambiente pré-existente que já foi adquirido. Assim, em tal situação, uma solução mais eficiente poderia ser construir o mapa ao mesmo tempo que o robô se move, liderando o SLAM. O processo SLAM é realizado por medições de dados de sensores disponíveis que constroem o mapa do ambiente, e essas medições são utilizadas simultaneamente para re-estimar a posição do robô. Comparado ao GPS, Unidade de Medição Inercial (IMU), raio laser, sensores ultrassônicos e outros sensores tradicionais, os sensores visuais como câmeras podem adquirir informações valiosas dos arredores, envolvendo cor, textura e outras informações visuais. SLAM usando câmeras é referido como SLAM visual (VSLAM), porque é baseado apenas em informações visuais. Este tipo de SLAM é uma tecnologia fundamental para uma ampla diversidade de aplicações e tem sido amplamente discutido em visão computacional, Realidade Aumentada (AR) e robótica na literatura.

Objetivos

Nesta tese, nosso foco é resolver alguns problemas na área de Visual SLAM monocular incorporando informações de câmeras com medidas de IMU. Especificamente, esses problemas são falha de recurso contra movimento rápido e rotação pura em cenas, onde há pouca textura ou desfoque na imagem. Uma vez que nossa aplicação pretendida é superar esses desafios em robótica, nosso foco também será principalmente restrito a algoritmos e abordagens que são em tempo real e, portanto, podem ser estendidos para sincronização de câmera IMU. Além disso, uma nova formulação são empregada na estimativa de inicialização do estado do sistema SLAM visual-inercial, auxiliando na melhoria da precisão de localização. Os principais objetivos desta tese são:

- Inicializar com precisão os estados na etapa de inicialização no sistema SLAM visual-inercial usando a otimização não linear de IMU visual fortemente acoplado e pré-integrado. Neste processo, para obter mais precisão, as informações antigas e os resultados das estimativas são marginalizados juntos. Esta prática será executada empregando uma nova formulação na qual dois parâmetros ajustáveis são usados para alcançar a melhor precisão de estimativa de estado inicial.
- Mostrar a superioridade do algoritmo proposto em comparação com os sistemas de Odometria Visual-Inercial de última geração e SLAM Visual-Inercial no conhecido conjunto de dados de benchmark EuRoC.
- Mostrar a menor tendência do algoritmo proposto ao acúmulo de erros na trajetória de longo prazo em comparação com os algoritmos Visual-Inercial de última geração.
- Investigação do desempenho do algoritmo proposto do ponto de vista do hardware, como o uso da CPU.
- Experimentar o algoritmo proposto em um ambiente interno real e apresentar o bom desempenho do algoritmo proposto no mundo real.
- Redução no tempo de inicialização antes de capturar os recursos ORB.

Metodologia

Nesta tese, a metodologia proposta é utilizar o ORB-SLAM2 como sistema Visual SLAM base devido à sua robustez e capacidade de trabalhar em tempo real indoor ou outdoor, bem como sua capacidade de fechar loops. Este método é robusto contra cenários difíceis, inserindo quadros-chave o mais rápido possível e removendo depois os redundantes, para evitar o custo extra. No entanto, acreditamos que ele sofre de vários problemas, como a inconsistência na inicialização e, às vezes, devido ao movimento rápido e rotação pura, os recursos tomados pelo ORB-SLAM são perdidos no loop de rastreamento e, posteriormente, resulta em falha na estimativa da pose da câmera, particularmente em um ambiente sem textura ou ambiente com menos textura. Para resolver esses problemas, propomos um SLAM visual-inercial no âmbito da fusão de sensores, combinando as medições IMU e as informações da câmera. de fato, em nossa metodologia, fornecemos um novo método de inicialização, no qual o usuário pode regular os dois parâmetros ajustáveis com base na dificuldade do ambiente.

De fato, essa melhoria é aplicada na etapa de inicialização que para não subestimá-la, leva a uma baixa precisão na estimativa dos valores iniciais após o mapa 3D e sua posição de saída não serem confiáveis. Para tanto, apresentamos uma nova técnica na qual, primeiramente, na etapa de inicialização, é utilizado o método de otimização gradiente descendente em vez de SVD e, em segundo lugar, para estimar a velocidade, o vetor de gravidade e a escala métrica, uma nova formulação é apresentada. em que o usuário pode ajustar os pesos dos parâmetros para obter o melhor desempenho para encontrar os melhores valores de inicialização. No próximo capítulo, a abordagem proposta é implementada por dois métodos: benchmark e usando uma câmera IMU real.

Resultados e Discussão

Neste seção, detalhamos a implementação completa da inicialização visual-inercial para avaliar qualitativa e quantitativamente o desempenho do algoritmo proposto. No geral, a implementação pode ser dividida em duas seções: Benchmark e Experiment. Com base na seção de metodologia apresentada do algoritmo projetado, nosso algoritmo precisa ser regulado por meio de dois parâmetros ajustáveis L e P na faixa de 0 e 1. Esse processo é realizado por tentativa e erro até que o melhor desempenho e o erro mínimo de perda sejam alcançados. Embora essa estratégia possa consumir um pouco de tempo devido à execução e avaliação do desempenho muitas vezes, seus resultados são valiosos, pois esses parâmetros podem ser utilizados no ambiente categorizado, como textura e sem textura com base nas relações entre L

e P. No benchmark, observa-se que o uso de parâmetros ajustáveis adicionais L e P pode desempenhar um papel eficiente em alguns cenários com base no nível de dificuldade do ambiente. Além disso, o algoritmo proposto mostrou que tem uma tendência menor de acúmulo de erros na trajetória de longo prazo em comparação com algoritmos de inércia visual de última geração. Embora esse menor acúmulo de erros não tenha sido observado em todas as sequências, pode ser um caminho inicial para posterior estudo e aprimoramento do mesmo. Além disso, o algoritmo proposto no processo de implementação do benchmark é investigado a partir de uma perspectiva de hardware. A quantidade de CPU usada durante o benchmark é medida em dois tipos de computadores: Laptop e um hardware chamado Raspberry-Pi 3 single-board. Os resultados mostraram que o Raspberry-Pi 3 às vezes tende a ficar sobrecarregado, parando na captura de recursos e falha devido à menor capacidade da CPU. Neste experimento interno real, um dos principais objetivos era empregar os parâmetros ajustáveis obtidos no benchmark e aplicá-los em um experimento do mundo real. Esta prática é realizada com sucesso, sem falhar e perder os recursos durante o experimento. Outra conquista neste experimento foi mostrar o tempo de inicialização reduzido para 3 segundos antes da captura do recurso, o que comparado ao método de inicialização proposto no VO / VIO / SLAM de última geração, como um tempo de inicialização aceitável. De fato, ao contrário do ORB-SLAM original, cujo tempo de inicialização é em torno de 10 segundos, nosso algoritmo proposto leva um tempo menor para inicializar os estados.

Considerações Finais
A tese trata do desenvolvimento de métodos de inicialização para sistemas baseados em otimização Visual-Inertial Simultaneous Localization and Mapping (SLAM), visando melhorar a acurácia e precisão de tais sistemas. Uma revisão da literatura sobre SLAM e métodos relacionados, tais como Visual SLAM, Visual Odometry, Visual-Inertial SLAM usando métodos baseados em filtragem e otimização, é apresentada a fim de contextualizar os desafios a serem resolvidos neste projeto de pesquisa. Portanto, uma abordagem baseada em otimização para inicializar com precisão os estados no método ORB-SLAM2 é proposta, na qual parâmetros de peso são incluídos para atingir o desempenho desejado. Para corroborar o método proposto, uma análise de comparação com alguns métodos VSLAM é realizada utilizando o conjunto de dados EuRoC. Além disso, uma análise da carga computacional foi realizada com duas plataformas: o sistema computacional embarcado Raspberry Pi 3 e um laptop. Além disso, resultados experimentais foram obtidos em ambiente interno, validando o bom desempenho do método proposto. Foi demonstrado empiricamente que o método proposto reduz em três vezes o tempo de inicialização em comparação ao algoritmo ORB-SLAM padrão. Nossa sugestão para trabalhos futuros é encontrar uma gama apropriada de parâmetros ajustáveis para usar no algoritmo proposto em qualquer ambiente. Essa estratégia facilitará o uso desse algoritmo no posicionamento de qualquer ambiente interno, incluindo escuro / claro, bagunçado e escadas (usando uma câmera para baixo). Além disso, é possível combinar o algoritmo proposto com abordagens de Inteligência Artificial (IA), como aprendizado de máquina e aprendizado profundo, de forma que as posições obtidas por SLAM visual-inercial pudessem ser treinadas por meio de redes neurais artificiais, levando a predizer a posição em um ambiente interno invisível.

Palavras-chave: Visual-inercial. SLAM. Inicialização de estado. Dados de EuRoC. Parâmetros ajustáveis.

ABSTRACT

The fusion of monocular visual and IMU has gained a lot of attention from robotic systems. Recent results have shown that optimization-based fusion approaches outperform filtering approaches. However, poor initialization can lead to inaccurate state estimation in optimization-based visual-inertial Simultaneous Localization and Mapping (SLAM) systems. Therefore, due to the nonlinearity of visual-inertial systems, initial values (visual scale, gravity, velocity, and Inertial Measurement Unit (IMU) biases) play a crucial role. For this reason, this thesis aims to improve the initial states estimation using two adjustable parameters *L* and *P*. In fact, these additional parameters are employed as coefficients of first term and gravity term of the proposed formula, respectively. Based on the difficulty level of the environment (texture, mid-texture, and texture-less), the indoor room is categorized into easy, medium, and difficult, and then two adjustable parameters are regulated based on this difficulty level. This strategy has been tested in two types of implementation; Benchmark with the public EuRoC dataset and real-world experiment. In benchmark, by employing the right adjustable parameters, in some scenarios, we could attain satisfactory results compared to state-of-the-arts visual-inertial Odometery and SLAM in terms of positioning accuracy and reduction of accumulative error. In this part, also, from point of hardware's view, some measurements are performed. While the proposed algorithm is being executed, the maximum CPU usage in each sequence is measured on a Raspberry-Pi single-board and a Laptop. The results proved that the Raspberry-Pi 3 - because of poor hardware configuration - is under more pressure in terms of CPU usage. The second part is concerned with a real-world experiment in which a monocular-inertial RealSense ZR300 sensor is utilized. The outcomes were satisfactory so that the initialization time was very short and the proposed algorithm could quickly obtain the ORB features.

Keywords: Visual-Inertial. SLAM. State Initialization. EuRoC dataset. Adjustable Parameters.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AR	Augmented Reality
BA	Bundle Adjustment
BoW	Bag of Words
CF	Closed-Form
CKF	Cubature Kalman Filter
CPU	Central Processing Unit
DoF	Degree of Freedom
DPPTAM	Dense Piecewise Parallel Tracking and Mapping
DSO	Direct Sparse Odometry
DTAM	Dense Tracking and Mapping
EKF	Extended Kalman Filter
GPS	Global Positioning Sensor
GPU	Graphics Processing Unit
IMU	Inertial Measurement Unit
LIDAR	Light Detection and Ranging
LSD-SLAM	Larg-Scale Direct SLAM
MAP	Maximum a Posterior
MAV	Micro Aerial Vehicle
MSCKF	Multi-State Constrained Kalman Filter
ORB	Oriented FAST and rotated BRIEF
PCA	Principal Component Analysis
PL-SLAM	Points and Line SLAM
PTAM	Parallel Tracking and Mapping
RAM	Random-Access Memory

RGB-D	Red-Green-Blue-Depth
ROS	Robot Operating System
MPC	Model Predictive Control
ROVIO	Robust Visual Inertial Odometry
RSME	Root-Mean-Square Error
SaM	Structure and Motion
SfM	Structure from Motion
SLAM	Simultaneous Localization and Mapping
SVD	Singular Value Decomposition
SVO	Semi-direct monocular Visual Odometry
UAV	Unmanned Aerial Vehicles
VINS	Visual-Inertial Navigation System
VIO	Visual Inertial Odometry
VI-SLAM	Visual-Inertial SLAM
VO	Visual Odometry
VR	Virtual Reality
VSLAM	Visual SLAM

Contents

1 INTRODUCTION

As artificial devices – autonomous cars, quadcopters, full-sized robots or even virtual and augmented reality systems – start to interact with, or adapt to, the 3D world around us, they need the ability to perceive, reconstruct and ultimately understand it similarly. A car that drives itself needs to know where it is, and it needs to recognize and avoid obstacles, both dynamic and static. The best example is to convincingly display a virtual object standing on a table in which both the pose of the observer, as well as the shape of the table need to be known.

An impressive interest in controlling Unmanned Aerial Vehicles (UAVs) is increasing among researchers significantly. Applications of these robots have been widely defined in tasks such as traffic monitoring, rescue operations, surveillance, transportation, and hobbyist. For a hobbyist, they may be high-tech toys for flying around and recording videos while for researchers, they are a low-cost platform for the fulfillment of software design such as positioning, navigation, control, etc.

Robots that can autonomously operate in outdoor/indoor environments are envisioned to be useful for various applications that have been mentioned above. In addition to all these applications, an imperative for an autonomous robot is the ability of self-localization in the environment. Indeed, a robust accurate localization is crucial to achieve high-performance flight and to interact with the environment. Also due to denying the Global Positioning Sensor (GPS) in the indoor environment and lack of self-localization, the autonomous maneuver of Quadrotor faces more challenges (J. Pestana, 2012). Thus, it is necessary to employ alternative localization strategies to perform the duty of GPS in indoor environments. The simplest way is called Odometry which is a method to localize a robot by processing the sensor information to compute incremental motion. This allows us to retrieve the trajectory of the robot, which will inevitably accumulate error making the estimated trajectory to drift from the real trajectory performed by the robot. Odometry techniques are appropriate for short-term motion estimation, but if the aim is the localization in the same environment for the long-term, having a map would be desirable, allowing drift-free localization. Mapping is the creation process of the map using onboard sensors, resulting in known localization. It is important that this map is obtained simultaneously at the same time that the robot navigates. This problem is known as Simultaneous Localization and Mapping (SLAM) or Structure and Motion (SaM). For instance, suppose that the prior information of an indoor environment is considered as a map that has been used in visual localization and navigation. Considering this assumption in terms of mapping, sometimes, because of environmental constraints (for instance, relief case), it is demanding to navigate by a pre-existent accurate map of the environment which has already been acquired. Thereby, in such a situation, a more efficient solution could be to build the map at the same time that the robot moves (Y. Lu et al., 2018), leading SLAM.

The SLAM process is performed by available sensors data measurements that build the map of the environment, and these measurements are simultaneously utilized for re-estimating the position of the robot. Compared to GPS, Inertial Measurement Unit (IMU), laser lightning, ultrasonic sensors, and other traditional sensors, visual sensors like cameras can acquire rich information of surroundings, involving color, texture, and other visual information. Meanwhile, they are cheaper and easier to deploy and also able to obtain environmental perception as a map. SLAM using cameras is referred to as visual SLAM (VSLAM), because it is based on only visual information. This type of SLAM can be exerted as a fundamental technology for a wide diversity of applications and has been comprehensively discussed in computer vision, Augmented Reality (AR), and robotics in the literature.

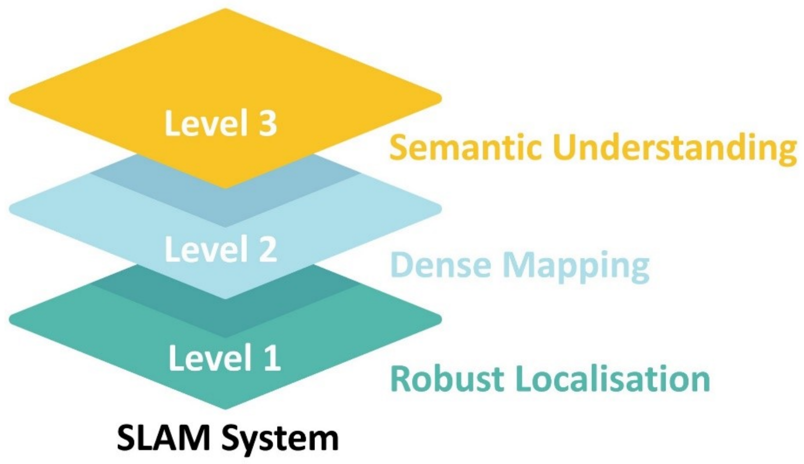
1.1 SLAM AND ITS HISTORY

As it has been explained in the previous section, SLAM is the problem of navigating a vehicle in an unknown environment, by building a map of the area and using this map to deduce its location without the need of a priori knowledge of the location. The solution to this problem is of great importance to the field of autonomous robots operating in GPS denied environments, and therefore, since its introduction in the scientific community, SLAM has been the subject of many research efforts. In other words, SLAM is the problem of building a map of an unknown environment from onboard sensor data while simultaneously using the data to estimate the robot position (C. Cadena et al., 2016).

A typical SLAM system is constituted of two modules; Front-end and Back-end. The front-end is in charge of fulfilling the data association for the back-end module, and the Back-end in SLAM is in charge of state inference after data association. From a probabilistic framework perspective, the purpose of the back-end is to generate the output of MAP (Maximum a Posterior) estimates given the measurements from the front-end (Y. Liu et al., 2018). For this purpose, the back-end solutions have evolved from filter-based approaches to graph optimization methods that we will represent in the future.

Figure 1 shows the schematic diagram of SLAM. In this figure, as it is observable, the SLAM system consists of three levels including robust localization, dense mapping, and semantic understanding in which, level 1 is a prerequisite for level 2, and level 2 is a prerequisite for level 3. Here, 3 and level 2 are placed into the front-end module, and level 1 is included in the back-end module.

Figure 1 - schematic diagram of SLAM



In the following, looking back, and discussing the history of SLAM, - its foundation and origin - the important stages of SLAM evolution are summarized from the last decades until recently. However, a more comprehensive description of the most recent state of the art will be provided in chapter 2.

Approaches for computing 3D structure from 2D images date back more than 100 years, long before the advent of digital photography or even computers. Very early work includes that of (Kruppa, 1913), where he proposed an analytic approach to compute the relative pose between images from five manually labeled point-correspondences. In the following decades, both the number of points and the number of images rose, and practical methods for solving the resulting mathematical systems were developed. The general term "Bundle Adjustment" (D. C. Brown, 1976), is still operating on manually labeled point correspondences in analog images. Bundle adjustment is the problem of refining a visual reconstruction to produce jointly optimal 3D structure and estimating the parameters such as camera pose and/or calibration. Here, optimal refers to the estimated parameters that are found by minimizing some cost function that quantifies the model fitting error and, jointly, that the solution is simultaneously optimal concerning both structure and camera variations (B. Triggs et al., 2000).

Formally, SLAM was introduced by (Smith et al., 1986) and then, with the emergence of the digital revolution in image processing and the appearance of Visual SLAM (in which the input of SLAM is visual information only), feature-less (direct) method formulations for structure and motion have been proposed. In feature-less methods, the process is performed based on the optimization of 3D geometry on the raw intensity images without any abstraction, using handcrafted feature detectors and descriptors. In this way, (K.J. Hanna, 1991) proposes a direct formulation for estimating dense depth as well as the camera motion from a monocular image sequence, minimizing a photometric error. An early example for direct and dense geometry estimation is the work of (Matthies et al, 1988), which proposes a method to estimate dense depth from a (calibrated) sequence of images, using pixel-wise filtering interleaved with spatial smoothing. A sparse and direct monocular SLAM system capable of running in real-time was presented by (H. Jin et al., 2003), which optimizes sparse patch positions, as well as camera, poses for a sequence of images, using a photometric consistency as an error measurement formulation.

Simultaneously to the appearance of feature-less methods, feature-based (indirect) methods – another type of Visual SLAMs - has been introduced. Compared to the feature-less method, the feature-based method separates correspondence estimation from geometric optimization, and the initial step of manually selecting and matching suitable points (key-points) was replaced by automatic feature detection. In general, in feature-based methods, geometric consistency such as positions of feature points in an image is used.

The first approaches to find key-points were presented by (M.A. Fostner et al., 1987) and (C. Harris et al., 1988). Consequently, in the following years, other algorithms such as FAST corners, provided by (E. Rosten et al., 2010), have been performed, which are considerably faster in computation. Investigating in earlier researches, it is explicitly observed that the initial method to track the selected key points was to minimize the photometric error between small patches around them. This is commonly known as the Kanade-Lucas-Tomasi feature tracker (KLT), which was first proposed by (C. Tomasi et al., 1991). After this, a global research in the field of descriptor spaces is carried out instead of the local optimization approach. These descriptors were SIFT (D.G. Lowe et al., 1999), SURF (H. Bay et al., 2006), and ORB (E. Rublee et al., 2011). These effectively allow us to solve the matching problem globally by approximating it with a nearest-neighbor search, replacing gradient-based local optimization (J. Engel, 2016).

After key-points detection and matching them in the first step, 3D geometry estimation – camera motion and key-point positions – is executed from 2D correspondences that have already been found. In this way, real-time is an important issue that has to be considered. The earliest real-time capable, incremental methods were proposed based on the Kalman filter, which accumulates information about the world as joint Gaussian distribution on all involved parameters. (H. Jin et al., 2000 & A.J. Davison et al., 2007) are examples of such filtering-based algorithms. Parallel Tracking and Mapping (PTAM) provided by (G. Klein et al., 2007) is the most well-known SLAM algorithm in which the tracking estimates camera motion in real-time, and the mapping estimates accurate 3D positions of feature points with a computational cost. PTAM is the first method that incorporates non-linear optimization (Bundle Adjustment) into the real-time SLAM algorithms. Then ORB-SLAM (R. Mur-Artal et al., 2015) takes advantage of PTAM, improving it.

Although monocular Visual SLAM has made great achievements in localization and mapping, it is a partially observable problem, in which sensors do not offer the depth of landmarks, resulting in lack of accurate initial estimation of the process, inability to recover the metric scale and drift of scale, all of which take their toll on the camera pose estimation accuracy. The main reason that caused the vision-only systems to be incapable of observing the metric scale, is due to the theoretical constraints in the camera's projective nature. As well as these issues, Visual SLAM algorithms are vulnerable to some circumstances such as motion blur from fast motions and the lack of scene texture. Therefore, these conditions can make Visual SLAM methods to fail or present very poorly performance. To address these problems, one of the most effective strategies to boost localization accuracy is visual-inertial fusion, which has been taken from a combination of IMU and visual measurements (M. Quan et al., 2019). In this way, one of the most important challenges is how to fuse the measurement information from the camera and IMU together so that the best initialization accuracy could be obtained. The importance of this issue shows itself when the IMU measures acceleration and angular velocity to obtain a position and orientation, in which measurements include the noise inherent and will lead to estimates of position and orientation drift away of true values. Also, though the camera can estimate the pose (position and orientation) accurately during a long time under slow-motion, it suffers heavily from motion blur. Another challenge is due to different measurement rates of camera and IMU, respectively, in which the low frequency of camera may cause the problem in synchronization between the camera and IMU).

1.2 PROBLEM STATEMENT

In this thesis, our focus is on solving some problems in the area of monocular Visual SLAM incorporating camera information with IMU measurements. Specifically, these problems are feature failure against fast movement and pure rotation in scenes, where there is little texture or blur on the image. Since our intended application is to overcome these challenges in robotics, our focus will also be mainly narrowed to algorithms and approaches which are real-time, and therefore, can be extended to IMU-camera synchronization. Besides, a new formulation would be employed in the state initialization estimation of the visual-inertial SLAM system, aiding to improve the localization accuracy.

In the next chapter, Visual SLAM (VSLAM) algorithms and challenges ahead have been explained in detail.

1.3 MAIN OBJECTIVES

The main objectives of this thesis are:

- To accurately initialize the states in the initialization step in the visual-inertial SLAM system using non-linear optimization of tightly-coupled visual and pre-integrated IMU. In this process, to get more accuracy, the old information and estimation results are marginalized together. This practice will be executed by employing a new formulation in which two adjustable parameters are used to reach the best initial state estimation accuracy.
- To show the superiority of the proposed algorithm compared to the state-of-the-art Visual-Inertial Odometry and Visual-Inertial SLAM systems on the well-known EuRoC benchmark dataset.
- To show the lower tendency of the proposed algorithm to error accumulation in long-term trajectory compared to the state-of-the-art Visual-Inertial algorithms.
- Investigation of the performance of the proposed algorithm from a point of hardware's view such as CPU usage.

- To experiment with the proposed algorithm in a real indoor environment, and present the good performance of the proposed algorithm in the real world.
- Reduction in the initialization time before capturing the ORB features.

1.4 CONTRIBUTIONS

The contributions of this thesis are given in the following ways:

- Obtaining the best adjustable parameters based on the difficulty level of the environment.
- Accuracy improvement during implementation of the proposed method in some scenarios of the EuRoC benchmark dataset.
- To reduce the accumulative error during benchmark, leading to improve accuracy.
- Finding the hardware limitations in using a single-board processor during benchmark
- Performing the successful experiment in an indoor real-world environment without feature failure in low-texture sense during the trajectory.
- Reduction in the initialization time before capturing the ORB features.

1.5 THESIS ORGANIZATION

This thesis is organized into several chapters:

- Chapter 2 discusses the structure of SLAM in detail, the introduction of Visual SLAM, visual sensors, and several forms of Visual SLAM algorithm: RGB-D SLAM, stereo SLAM, and Monocular SLAM. This chapter also deals with the classification of Visual SLAM methods and their limitations. The advantage and disadvantages of each system are described by an explanation of the most representative Visual SLAM of each category.
- Chapter 3 introduces visual-inertial SLAM algorithms incorporating IMU measurements with visual information. The main subject in this chapter is to increase the accuracy in state initialization, highly affecting the pose estimation obtained by visual-inertial SLAM.
- Chapter 4 explains how to implement the proposed algorithm in the real world and its results.

- Chapter 5 includes the conclusion in which the obtained results are analyzed and some recommendations to improve the visual-inertial SLAM are provided.

1.6 PUBLICATIONS

a) **Seyed Jamal Haddadi**, Eugênio de Bona Castelan Neto, "Sensor Fusion and Autonomous Indoor Navigation of a Quadrotor Using ORB-SLAM", IEEE, Latin American Robotic Symposium, Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE), Brazil, November 2018.

b) **Seyed Jamal Haddadi**, Eugênio de Bona Castelan Neto, "Evaluation of Monocular Visual-Inertial SLAM: Benchmark and Experiment", IEEE 7th RSI International Conference on Robotics and Mechatronics (ICROM), Iran, November 2019.

c) **Seyed Jamal Haddadi**, Eugênio de Bona Castelan Neto, "Improvement of Visual-Inertial ORB_SLAM using Correction in Initial States Estimation", Robotics and Autonomous Systems, ELSEVIER Journal, March 2021. (Submitted).

2 VISUAL SLAM (VSLAM)

2.1 INTRODUCTION

As mentioned in the previous chapter, visual SLAM is the branch where a SLAM system uses a camera as the only extrinsic sensor. Employing the camera is highly practical due to its low power consumption and lightweight. VSLAM also provides a rich representation of the environment. Visual SLAM incorporates the SLAM research in robotics with the investigation of computer vision, and it is being used widely. Besides, these algorithms use the visual-based perception devices to execute the SLAM research, and it aims to construct a map and a full trajectory for the camera (or the robot). Also, they are suitable for camera pose estimation in AR systems due to their simple configuration which allows it to be relatively easy such as camera-mounted tablets or smart-phones. Indeed, the field of VSLAM has become fast developed because the computing power of CPUs has been improving and many novel solutions have been proposed.

In this chapter, VSLAM algorithms, classification of VSLAMs, and their limitations are discussed in detail. This chapter aims to figure out the challenges of the state-of-the-art of VSLAM algorithms.

2.2 ELEMENTS OF VISUAL SLAM

Based on (T. Taketomi, et al., 2017), the framework of VSLAM algorithms is composed of five modules including three basic modules and two additional complimentary ones. Note that these basic modules are largely related to figure 1 where there have been robust localization, dense mapping, and semantic understanding.

2.2.1 Basic Modules

These modules are main cores of VSLAM:

2.2.1.1 Initialization

One of the most essential issues to initialize the VSLAM is to define a certain coordinate system for camera pose estimation and 3D reconstruction in an unknown environment. An unknown environment is an environment where a robot that is moving in that, doesn't have any prior knowledge of it. In such an environment, the robot has to make a 3D reconstruction from the objects such as their color, shape, etc. Therefore, the determination of the global coordinate system is the first stage in initialization that should be defined to acquire an initial map with respect to the global coordinate system and reconstruct part of the environment.

2.2.1.2 Tracking

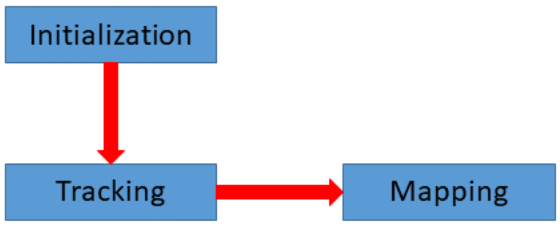
After the initialization, tracking and mapping are performed to continuously estimate the camera pose. In the tracking module, the reconstructed map is tracked to estimate the camera pose of the image according to the global map. To perform this, firstly, 2D-3D correspondences between the image and map are obtained through feature tracking or feature matching in the image. Next, by solving the perspective-n-point (PnP) problem (R. Klette et al., 1998 & D. Nister, 2004), the camera pose is computed from the correspondences. There is an assumption among most VSLAM methods implying that intrinsic camera parameters have already been calibrated. As a result, a camera pose is normally equivalent to extrinsic camera parameters with translation and rotation of the camera in the global coordinate system.

2.2.1.3 Mapping

The mapping is the third module in which, by computing the 3D structure of an environment, the map is expanded while the camera observes unknown regions where the mapping was not performed before.

Figure 2 shows a simple schematic of elements of SLAM.

Figure 2 – a simple schematic diagram of SLAM



2.2.2 Complementary Modules

These modules are exerted for stable and accurate VSLAM:

2.2.2.1 Relocalization

Sometimes, due to the existence of some disturbances or fast camera motion, the tracking of camera pose is subjected to failure. In such cases, an alternative method to continue computing the camera pose is to use relocalization concerning the map again. Incorporating relocalization into VSLAM would guarantee that the system works even after the tracking is lost, and such a strategy makes VSLAM systems practically advantageous.

2.2.2.2 Global Map Optimization

When the VSLAM system starts moving in the environment while estimating the camera pose, the accumulative estimation error during localization is inevitable according to the camera movements, leading to gross error in pose estimation which often takes place in long-term camera movement. In such a circumstance, performing the global map optimization could be a reasonable technique to suppress the error. In this process, the map is rectified by optimizing the pose estimation with the consistency of whole map information to reduce accumulative error.

In continuation of the suspension of accumulative error in camera pose estimation in VSLAM, Pose graph optimization is one of the most commonly-used approaches (G. Grisetti, et al., 2010 & R. Kümmerle, et al., 2011), and it is a way solve the loop closure in Visual SLAM. In this method, a pose graph is considered as a graph with nodes representing robot poses and edges linking the nodes, among which odometry measurements are available, and then the consistent graph is build to suppress the error in the optimization (K. Li, et al., 2019). One demerit of global-pose optimization is a strong dependence on the accuracy of the initialization point for the graph. Besides, there are several instances when poor initialization led to arising the pose estimation error (J. Jackson, et al., 2019).

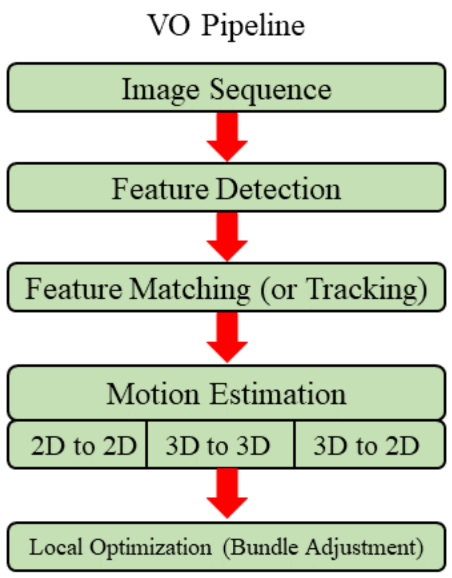
2.3 OTHER RELATED TECHNIQUES

2.3.1 Visual Odometry (VO)

Besides Visual SLAM, there exists another approach similar to SLAM, which is Visual Odometry (VO). A prerequisite for understanding VO is to perceive the Odometery (see chapter 1).

To be more accurate, VSLAM has been constituted of global map optimization and Visual Odometry (T. Taketomi, et al., 2017). The idea of VO was first introduced in the 1980s for the Mars Rover project, and the term was not popularized in the engineering context until around 2004 (D. Nister, et al., 2004). The goal of VO is to estimate the motion of the camera in real-time using sequential images. Figure 3 shows the VO process in a pipeline.

Figure 3 - VO process in a pipeline



Generally, VO can be performed by passive cameras (monocular, stereo, and omnidirectional) and active sensors (Light Detection and Ranging(LIDAR), time-of-flight –

Active optical Time-of-Flight, is a remote-sensing method to estimate range between a sensor and a targeted object by measuring the travel time from the emitter to the object and back to the receiver.) and Red-Green-Blue-Depth (RGB-D). Since our focus is on a monocular camera, hence, it is essential to mention that the most critical problem in such cameras is that the motion scale is unobservable, which must be synthesized. In other words, the scale of trajectory cannot be directly estimated as the depth is unobservable from monocular cameras (although for in hybrid methods such as visual Inertial Odometry, integrating raw IMUs measurements can provide noisy, short-term estimates of scale) (E. Jared, et al., 2018).

2.3.2 Structure from Motion

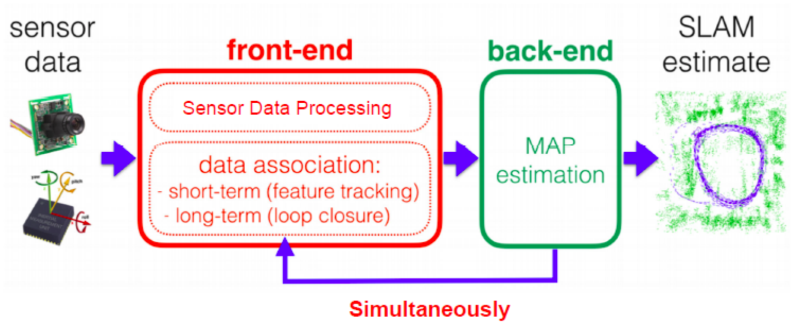
Structure from Motion (SfM) employs a set of 2D images acquired by a moving camera to estimate the 3D geometry of a scene and the camera motion is a technique to estimate the pose of camera motion and 3D structure of the environment in a batch manner (S. Agarwal, et al., 2011). SfM approaches often have to work on an unordered set of images without time constraints and might employ different cameras, whereas VSLAM is supposed to work in real-time on an ordered sequence of images acquired from a fixed set-up. In SfM, the focus is on the accuracy of the 3D reconstruction; however, it offers an off-line solution with high reconstruction quality and uses an accurate feature detector and descriptor to obtain higher quality features. Besides, SfM can exhaustively use all input images to find feature correspondences and perform global reconstruction optimization applying bundle adjustment. From a technical point of view, VSLAM, Visual-Inertial Odometry (VIO), and SfM are similar and share many common components.

2.4 VSLAM VERSUS VO

VO and VSLAM are fundamentally the same in the sense that both methods attempt to estimate the camera pose and reconstruct the scene structure. VO aims at recovering the camera pose incrementally, and potentially, it may build a local map and trajectory using local optimization. SLAM, on the other hand, tries to produce a globally consistent map and trajectory, such that the system can detect when the camera returns to a previously explored position and correct the drift accordingly. Global consistency is achieved by realizing that a previously mapped area has been re-visited (loop closure) and this information is exerted to decline the drift in the estimates.

So far, the main components of Visual SLAM are introduced and it turned out that the SLAM can be comprised of five modules (mentioned in 2.2). Now, knowing the VSLAM and VO, the structure of VSLAM can be generally divided into two main components: front-end and back-end; the front-end is comprised of VO (D. Nistér, et al., 2004) module and the mapping module, and the back-end is responsible for performing the optimization module. There may also be an additional loop-closure detection module (A. Angeli, et al., 2008).

Figure 4 - The structure of visual SLAM



According to Figure 4, in the front-end module, the VO module estimates the approximate 3D camera pose and structure from adjacent images to provide better optimization of the initial value for the back-end. Then the optimization module in the back-end estimates the trajectory and map state from noisy data. This can be viewed as a maximum a posteriori problem (D.M Greig, et al., 1989). Finally, the mapping module creates a map that will be used mainly in SLAM and can be used for navigation, visualization, and interaction. Also, as it was explained in section 2.2, the loop-closure detection module determines whether the camera arrives at a scene that has been previously captured, so that loop-closure solves the problem of drifting of the estimated positions over time.

The downside of VSLAM is that the computing cost is high and advanced algorithms are needed. Also, the technical difficulty of VSLAM is higher than that of other sensor-based SLAMs because the view range of the camera is smaller than the laser (360) for example, and as a result, the camera can acquire less visual input. From such input, camera poses need to be continuously estimated, and the 3D structure of an unknown environment is simultaneously reconstructed. In this way, “feature-based (indirect) method” has been introduced in 2000, which is the early work (G. Klein et al., 2007) of VSLAM using a monocular camera, and is based on tracking and mapping feature points. On the flip side, there is another approach which is called the “direct method ”. This method can be exerted to tackle the texture or feature-less environment where VSLAM has been proposed to tracking and mapping without the need to feature detection of points, and it deals with a whole image for its process.

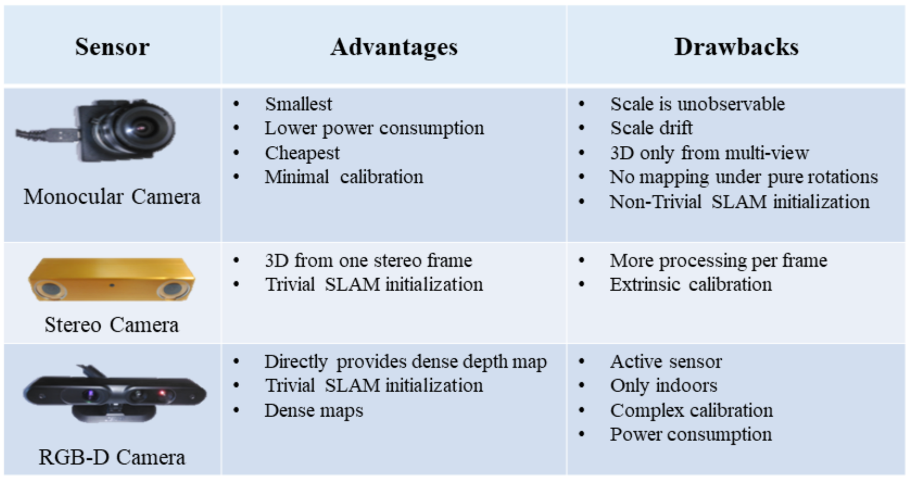
2.5 SENSORS FOR VISUAL SLAM

As we described at the Introduction Section, SLAM is a way for a robot to localize itself in an unknown environment, while incrementally constructs a map of its surroundings. SLAM has been extensively studied in the past couple of decades resulting in many different solutions using different sensors, including non-visual types such as an ultrasonic sensor or a LIDAR, or visual ones such as a monocular camera, a stereo camera, or a RGB-D camera, which can provide not only the color but also the depth of every pixel. On the other hand, if sensors are categorized based on data output, then the mathematical problem behind SLAM is not so different in using various types of sensors. For instance, the data output of the ultrasonic sensor which could be considered as non-visual is based on the meter. Also, a monocular camera, which is a visual sensor, can provide data output according to the meter. Therefore, the emplyment of a different kind of measurement unit of sensors does not greatly modify the mathematical formulation of SLAM. The visual SLAM sensors are explained further in the next section.

2.5.1 Visual SLAM Sensors

Nowadays, with technological development, robots are getting much smaller before, which restricts their payload capacity to certain scopes. Hence, in the visual SLAM area, using simple (single and multiple) cameras has been shown more interest rather than the traditional complex laser radar and sonar, etc among researchers. RGB-D cameras (T. Schops et al., 2019), stereo cameras (S. Se et al., 2002), time-of-flight camera (S. May et al., 2009), and Monocular cameras (S. Weiss et al., 2013), are four main approaches in the literature differing in the number or type of cameras, that among them, Mono, Stereo, and RGB-D are the most likely applied in SLAM techniques. A list of advantages and drawbacks of the most common types of cameras including Monocular, Stereo, and RGB-D, for visual SLAM, is shown in Table 1.

Table 1 - The most common types of cameras for Visual SLAM



Sensor	Advantages	Drawbacks
Monocular Camera	• Smallest • Lower power consumption • Cheapest • Minimal calibration	• Scale is unobservable • Scale drift • 3D only from multi-view • No mapping under pure rotations • Non-Trivial SLAM initialization
Stereo Camera	• 3D from one stereo frame • Trivial SLAM initialization	• More processing per frame • Extrinsic calibration
RGB-D Camera	• Directly provides dense depth map • Trivial SLAM initialization • Dense maps	• Active sensor • Only indoors • Complex calibration • Power consumption

Recently, (L. Han., et al., 2019) presented Real-time globally consistent camera localization for visual SLAM using a RGB-D camera in which each observation (three-dimensional point feature) has to be linearized based on its local coordinate (camera poses), which is nonlinear and dynamically changing, resulting in extensive computation during optimization. However such nonlinearity is decoupled into a linear (feature position) and nonlinear components (camera poses).

Also, recently, structured light-based RGB-D cameras (J. Geng, 2013) such as Microsoft Kinect (Z. Zhang, 2012) have become cheap and small. Since such cameras provide 3D information in real-time, these cameras are also used in VSLAM algorithms.

By using RGB-D cameras, the 3D structure of the environment with its texture information can be obtained directly. Also, in contrast to monocular VSLAM algorithms, the scale of the coordinate system is known because the 3D structure can be acquired in the metric space. The basic framework of depth (D)-based VSLAM is as follows. An Iterative Closest Point (ICP) algorithm (P. J. Besl., et al., 1992) has widely been used to estimate camera motion. Then, the 3D structure of the environment is reconstructed by combining multiple depth maps. To incorporate RGB into depth-based VSLAM, many approaches have been proposed as explained below. However since these approaches project IR (Infrared) patterns into an environment to measure the depth information, they are reliable and developed for indoor applications only, and it is difficult to detect emitted IR patterns in outdoor environments. Note that the Note that IR emitters are small wired transmitters for repeating an infrared signal from your remote to an isolated piece of A/V equipment. Furthermore, there is a limitation in the range of the depth measurement such that the RGB-D sensors can capture the environment.

Stereo cameras are other types of cameras that provide a passive 3D depth estimation that typically involves the use of stereo calibration procedure to compute projection matrix that will transform a 2D point(generally observed in left camera) into a 3D point in the left camera coordinate system. (R. Gomez-Ojeda., et al., 2019) propose PL-SLAM, a stereo visual SLAM system that combines both points and line segments to work robustly in low-textured environments. Although stereo camera setups perform better than RGB-D in the case of outdoor applications, the amount of data that they provide is high and requires large amounts of computational capacity.

As a result, the advantage of measuring the depth by using stereo or RGB-D cameras is to eliminate the several challenges encountered when relying only on a monocular camera. One particular drawback of all depth-measuring devices is the very limited range and the light problem at which they can operate accurately - to navigate in large, open spaces (e.g. a factory building, or outside of GPS-denied environments such as narrow streets). On the other hand, monocular cameras are especially suited for applications where compactness and minimum weight are critical. In addition to that, lower prices and flexible deployment make them a suitable option for robots and unmanned vehicles. For the reasons above, the remaining of this section focuses on the monocular Visual SLAM algorithms, where the only information is coming from a monocular camera.

2.6 MONOCULAR VISUAL SLAM

There are two approaches to solve the monocular visual SLAM: Feature-based methods and Direct methods.

2.6.1 Direct (feature-less) methods

Direct methods use directly all the pixel intensity information in the image. These methods can exploit visual information without relying on key-point detectors, and therefore, they are expected to be more accurate and robust when there is little texture in the scene or blur on the image (R. Mur-Artal 2017). Direct SLAM construction is based on the computation of depth associated with each pixel on selected cameras by minimizing an error measure such as brightness or brightness-based cross-correlation (M. Irani et al., 1999). DTAM (R. A. Newcombe et al., 2011) is an example of a direct visual SLAM algorithm, which constructs a dense map of the scene. Defining the photometric error function, which combines the errors for each pixel along with the camera motion and the projection function, leads to:

$$E_{photo} = \sum_{p \epsilon \Omega} \| I_i(p_i) - I_{i+1}(\pi \, (\xi, P)) \| \, , \qquad (1)$$

where $I_i(p_i) = I_{i+1}(p_{i+1})$, and I_i, I_{i+1} are consecutive frames. Meanwhile, p_i and p_{i+1} denote the projections of a world point $P_W = (X \; Y \; Z)^T$ in those frames. Also, camera motion ξ represents elements of SE(3) - quaternation and position - and the projection function is introduced as π (ξ, P).

Since the DTAM restores the dense map for each pixel and applies global optimization, the computational burden is very high. Engel et al. proposed Large-Scale Direct SLAM (LSD-SLAM) (J. Engle, et al., 2014) and Direct Sparse Odometry (DSO) (J. Engle, et al., 2016) based on the direct method.

LSD-SLAM is the most popular algorithm in visual SLAM based on direct methods. This algorithm describes a direct monocular visual SLAM algorithm that allows us to build large-scale, consistent maps of the environment. Along with highly accurate pose estimation based on direct image alignment, the 3D environment is reconstructed in real-time as a pose-graph of keyframes with associated semi-dense depth maps. Compared to approaches that parameter optimization is performed without scale, LSD-SLAM uses pose graph optimization which explicitly considers the scale factor, allowing for correction of scale drift and loop closure detection in real-time. In contrast to DTAM, LSD-SLAM and DSO use fewer pixels, and, since each pixel depth is calculated independently, they are more efficient than DTAM. LSD-SLAM employs three parallel threads after initialization takes place: i) tracking; ii) depth map; iii) map optimization.

The direct-method-based SLAM has the following advantages: it does not extract feature points, it can be used even in the case of a small number of feature points or a blurred image, and it can generate a depth map. However, it is ineffective for fast motion and changes in grayscale values, depends on higher hardware requirements for the camera, and also it is slower than the feature-based method.

2.6.2 Feature-based Method

Unlike the direct (feature-less) methods using images directly, feature-based methods firstly detect and extract distinctive interest points (keypoints) from images. A descriptor, typically a vector of binary or real values of a certain length, is computed for each keypoint by operating on a patch of pixels around the key point. This allows us to match keypoints across images just by comparing their descriptors. The incorporation of a keypoint and its descriptor is called a feature. Indeed, once features are extracted, the image can be discarded so that only the feature-based method can operate on these features (R. Mur-Artal et al., 2015), since these features are considered as inputs for motion estimation and localization procedures. Normally, it is expected that features be invariant to rotation and viewpoint changes, as well as robust to motion blur and noise. The feature correspondences are stored in a map, ready to be used for localizing newly tracked frames. For monocular systems, the features can be stored either in 2D, as detected, or in estimated 3D, after back projection; this results are obtained from two approaches, for expressing the motion between the previous feature vector f_{i-1} and the current one f_i:

- 2D-to-2D: when both f_{i-1} and f_i are specified as 2D image frames; the transformation, accomplished by Bundle Adjustment (Section 1.1) is an iterative refinement to obtain a more accurate map of the local trajectory, and also, it is a windowed algorithm as it tries to minimize an error function over the last frames. This is calculated as the image re-projection error as:

$$\underset{X^k, C_i}{\operatorname{argmin}} \sum_{k,i} \left\| p_i^k - g(X^k, C_i) \right\|^2, \qquad (2)$$

where p_i^k is the point corresponding to the landmark X^k in image i, and $g(X^k. C_i)$ is the re projection of the same landmark according to the current camera pose C_i. The choice of window size depends on the computational capabilities of the system, since the re-projection error is a nonlinear function requiring an expensive algorithm such as Levenberg Marquardt (H. Gavin 2013). A small window limits the number of parameters, making bundle-adjustment tractable in real-time.

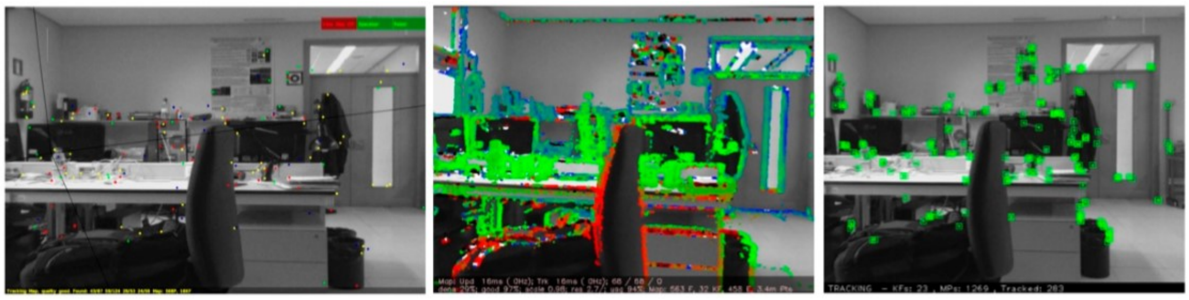
- 3D-to-2D: when f_{i-1} is specified as 3D frames and f_i is in 2D. Similarly, to the previous case, the transformation corresponds to the minimal reprojection error, but it also needs a preparation step to triangulate p_{i-1} from two adjacent camera views, I_{i-2} and I_{i-1}. The error function for the transformation T_i then becomes:

$$T_i = \underset{T_i}{\operatorname{argmin}} \sum_k \left\| p_i^k - p_{i-1}^k \right\|^2. \tag{3}$$

Eventually, the algorithms should accomplish optimizations and corrections. PTAM (G. Klein et al., 2007) is a feature-based algorithm that most monocular visual SLAM algorithms for MAVs rely on it. This algorithm has been the standard of modern feature-based visual SLAM that divides the SLAM system into two parallel independent threads: i) tracking; ii) mapping. For tracking, it uses fast, every-frame visual odometry with a robust estimator and coarse-to-fine optimizations. For mapping, it uses the batch technique of bundle adjustment, which is more accurate than incremental approaches but also more expensive computationally. However, it is tractable in real-time since it is decoupled from tracking and can be run only at key-frames when there is sufficient motion to update the map.

Three of the most popular monocular VSLAM algorithms, two feature-based including PTAM (G. Klein et al., 2007), ORB-SLAM (R. Mur-Artal et al., 2015) and one Direct method (LSD-SLAM (J. Engel et al., 2014)) with features taken indoor can be seen in Figure 5.

Figure 5 - The most popular monocular VSLAM algorithms.

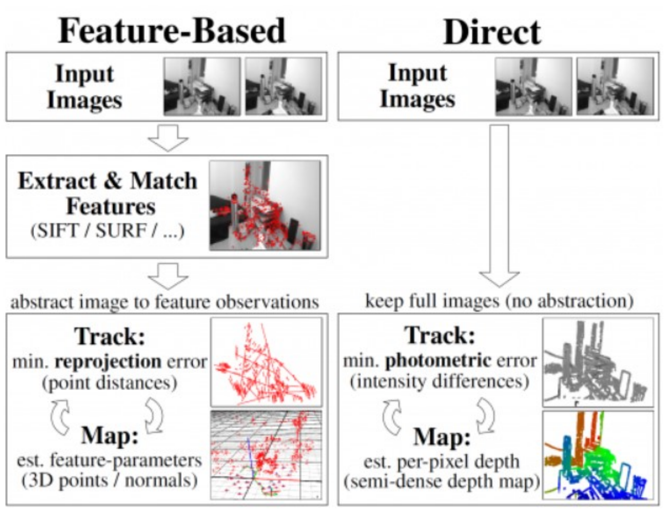


a) PTAM b) LSD-SLAM c) ORB-SLAM

The newest feature-based VSLAM algorithm is the ORB_SLAM in which Oriented FAST and rotated BRIEF (ORB) features are used. This algorithm, that is shown in Figure 5.c, can estimate the 6-DoF of the robot and reconstruct a sparse environment model. The main achievements of ORB_SLAM is to use ORB features in real-time, re-localization with invariability to the viewpoint, and a place recognition module that uses a Bag of Words (BoW) for loop detection. The Bag-of-Words (BoW) approach (N. Kejrival et al., 2016) is one of the most popular methods in this category. In this method, an image is represented as a histogram of words present in a dictionary

Since, most of the feature-based methods extract only distinct feature points from images, which can at most reconstruct a specific set of points (traditionally, corners), and then it can only reconstruct a sparse scene map. Thus, this kind of method can be called sparse indirect methods. One of the differences between direct and indirect methods is that the direct methods normally find dense correspondences, so it can reconstruct a dense map at an extra cost of computation. It is one of the reasons that, currently, researchers have been looking forward to the dense indirect methods that can reconstruct dense maps (Y. Lu et al., 2018). The difference between feature-based and direct methods is shown as a block diagram in Figure 6.

Figure 6 - Difference to key-point-based methods


Feature-Based
Direct
Input Images
Input Images
Extract & Match Features (SIFT / SURF / ...)
abstract image to feature observations
keep full images (no abstraction)
Track: min. reprojection error (point distances)
Map: est. feature-parameters (3D points / normals)
Track: min. photometric error (intensity differences)
Map: est. per-pixel depth (semi-dense depth map)

Source: https://vision.in.tum.de/research/vslam/lsdslam

According to the discussions above and disadvantages of direct-based methods, which the most noticeable ones are the slow execution and high computational cost, we continue our study with feature-based methods. Note that two of the most important advantages that motivated use to use feature-based methods are lower computation cost and faster run speed than direct-based methods.

There exist two types of feature-based methods in the literature: The Filter-based and Keyframe-based approaches.

2.6.2.1 Filter-based Methods

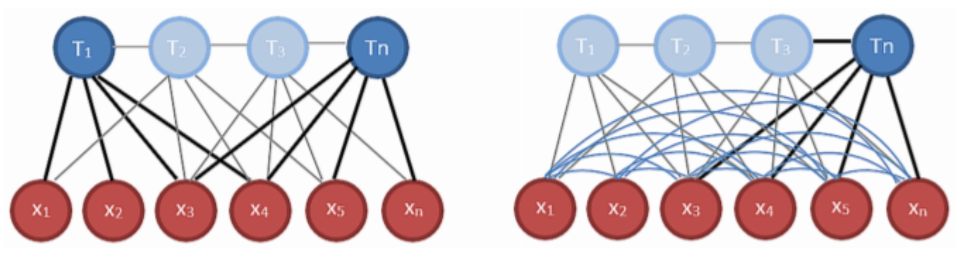
In this kind of method, the information is summarized at each step in a probability distribution, and the camera pose is estimated using the information of all features of the map. This method requires fewer computational resources due to the marginalization of past state and can achieve high accuracy in a short time. However, due to the missing loop closure mechanism, linearization error, and lack of place recognition, it will slowly drift during long term SLAM. It is noteacible that to address problems caused by linearization, the use of an Unscented Kalman Filter (UKF), appearsto be an appealing option (Julier., et al., 2000). Also, it includes uncertainties owing to computational complexity and the linearization process. Based on the measurement process of information, filter-based methods can be categorized into two classes: *i*) Kalman Filter (EKF, UKF and CKF) and Particle filter methods (M. Kleinert., et al., 2010 & J. A Hesch., et al., 2013); *ii*) sliding window filtering approaches (A. Joel., et al., 2017 & M. Li., et al., 2013). Since in EKF-based SLAM, the state vector contains both poses of robot/camera and a set of positions, and as long as these features are continuously observed and contained in the state vector, there is no drift in the estimated pose relative to these features. However, due to the quadratic calculation in the number of features in the state vector, it has high computational complexity. On the contrary, sliding window filtering methods keep a sliding window of the previous camera poses in the state vector, and use the feature measurements to impose probabilistic constraints on these poses (M. Quan et al., 2019).

2.6.2.2 Keyframe-Based Methods

This method estimates states by using the entire data, and with a loop closure mechanism, it can achieve high accuracy drift by leveraging the estimated 3D map; however, the demand for much more computational resources limits the efficiency in computation. Also, the keyframe method retains the optimization approach of global Bundle Adjustment (BA) (R. Mur-Artal, 2015), but unlike the filter-based method (relying on all the features in the map marginalize out past poses and summarize the information gained over time with a probability distribution), keyframe method computationally must select only a small number of past frames (Key-frames) to calculate the current pose (H. Strasdat, et al., 2015). Although this method is computationally expensive, it exploits the sparse structure, and thus, it enables fast computation by using sparse linear solvers.

Figure 7 shows the difference between two types of feature-based methods including keyframe-based and filter-based approaches, where we can notice the interactions among the pose in time and features.

Figure 7 - Keyframe BA (left) vs filter-based (right): T is a pose in time, x is the feature/landmark



There also exist various approaches to combine the front-end and back-end in visual-SLAM such as a parallel real-time system that has been presented by G. Keilin, et al., in 2019, in which the system consists of the tracking and the mapping threads. The following, (R. Mure, 2015) inspired by the parallel process provided an improved system with the concept of a co-visibility graph for local mapping to keep the consistency for the large-scale environment. Furthermore, (C. Forster., et al., 2014) provided a parallel solution by fusing direct tracking for pose estimation and depth filter for feature estimation. Besides these strategies, (E. Hong., et al., 2018) have shown that the sliding window method keeps the computational time bounded by marginalizing out old states.

As a consequence of these differences, Strasdat et al. in 2010 showed that keyframe-based methods outperform filter-based ones, and it is therefore not surprising to note that most new releases of monocular SLAM systems are keyframe-based.

2.6.3 Related Works

Monocular Visual SLAM, a system that uses a camera as its data input sensor is widely used in platforms moving in indoor environments. Compared to radar and other range-finding instruments, a visual sensor has the advantages of low power consumption and small volume, and it can provide more abundant environmental texture information for a moving platform. In general, all monocular SLAM systems aim to construct a globally consistent representation of the environment. It has drawn the attention of researchers due to its low cost and small size. The solution way of Monocular visual SLAM had been presented by some researchers. (A. Chiuso, et al., 2003) proved the stability of an algorithm for reconstructing three-dimensional structure and motion causally in real-time from monocular sequences of images. Then, (E. Eade, et al., 2006) addressed the cost of maintaining estimates, which rises rapidly with the number of landmarks mapped. Their strategy was to apply a monocular Fast-SLAM algorithm to a single camera that employs a particle filter and top-down search to allow real-time performance while mapping large numbers of landmarks.

In 2007, Davison et al introduced the most considerable improvement to tackle the Visual-SLAM problems. They were the pioneer in this area, who proposed the Mono-SLAM - a method of capturing the path of a freely moving camera while generating a sparse map - for the first time to implement a monocular real-time SLAM system. In this kind of visual SLAM, which is on Extended Kalman Filter (EKF)-based method, the produced map is sparse and consists of image patches as features.

One of the most significant achievements to solve monocular Visual SLAM is the Parallel Tracking and Mapping (PTAM) that is presented by Klein et al., in 2007. This robust SLAM solution mainly focused on accurate and fast mapping in a similar environment to Mono-SLAM. PTAM is the first optimization-based solution to split tracking and mapping into separate tasks processed in two parallel threads. So that, the front-end thread only performs pose estimation and feature tracking, while the back-end thread performs mapping and everything else, such as feature initialization and removing unnecessary keyframes. These features enable the algorithm to perform the expensive batch optimization routine of Bundle Adjustment (BA) in real-time. However, similarly to many earlier works, it works only in small scenes and easily suffers from tracking loss. PTAM is the first method incorporating BA into the real-time VSLAM algorithms.

(R. A. Newcombe, et al., 2010) proposed a hybrid monocular SLAM system that relied on PTAM to fit a dense surface estimate of the environment that is refined using direct methods. In this system, to estimate a dense refinement over the base mesh, a parallel process selects a batch of frames that have a potentially overlapping surface visibility using a GPU accelerated implementation of variational optical flow. Then, this algorithm was updated to DTAM that removed the need for PTAM as a front-end to the system and generalized the dense reconstruction to fully solve the monocular SLAM pipeline.

(K. Pirker, et al., 2011) proposed the Continuous localization and mapping in a dynamic world (CD SLAM) in 2011, with the objectives to handle short- and long-term environmental changes and mixed indoor/outdoor environments. To limit the map size and gain robustness against significant rotational changes, CD SLAM suggests the use of a modified Histogram of Oriented Cameras descriptor (HOC) (K. Pirker, et al., 2011), with a GPU, accelerated descriptor update, and a probabilistic weighting scheme to handle outliers. Furthermore, to update the feature descriptors after loop closure, it provided a geometric adaptation and also the use of large-scale nested loop closures with scale drift correction.

In 2013, Robust monocular SLAM in Dynamic environments (RD SLAM) was released by (C. Pirchheim, et al., 2013), and it aimed to cope with dynamic objects and slowly varying scenes. It employs a heavily parallelized GPU accelerated SIFT and stores them in a KD-Tree (C. Silpa-Anan, er al., 2008), that further accelerates feature matching based on the nearest neighbor of the queried feature in the tree. While the KD-tree is meant to accelerate SIFT feature matching, updating it with new features is computationally costly. Also, it performs landmark and keyframe culling using histograms of colors to detect and update changed image locations, while sparing temporarily occluded landmarks.

After one year, in 2014, (C. Forster et al., 2014) proposed a fast Semi-direct monocular Visual Odometry (SVO), which combines the feature point and direct tracking optical flow method. SVO generates a five-level pyramid representation of the incoming frame; data association is first established through iterative direct image alignment, starting from the highest pyramid level up until the third level. Preliminary data association from this step is used as a prior FAST feature matching procedure, similar to PTAM's warping technique, with a Zero-Mean SSD score. However, different from PTAM, to achieve real-time performance, SVO needs to run with a high frame rate camera. It was designed mainly for onboard applications that have limited computation resources.

In LSD SLAM proposed by (J. Engle, et al., 2014), and later in DSO (J. Engle, et al., 2016), a randomly initialized scene's depth from the first viewpoint, use an initialization method that does not require two view geometry; it takes place on a single frame: pixels of interest (i.e., image locations that have high-intensity gradients) are given a random depth value with an associated large variance in the first keyframe. This results in an initially erroneous 3D map. The pose estimation methods are then invoked to estimate the pose of newly incoming frames using the erroneous map, which in return results in erroneous pose estimates. However, as the system process more frames of the same scene, the originally erroneous depth map converges to a stable solution. The initialization is considered complete when the depth variance of the initial scene converges to a minimum. According to a report of (R. Mur-Artal et al., 2015), they still need features for loop detection, and their camera localization accuracy is significantly lower than PTAM.

DPPTAM, which stands for Dense Piecewise Parallel Tracking and Mapping, was proposed by (A. Concha, et al., 2015), and it borrows from LSD SLAM's initialization procedure and, therefore, also suffers from the problem of random depth initialization, where several keyframes must be added to the system before a stable configuration is reached.

ORB-SLAM (R. Mure, 2015) and ORB-SLAM2 (R. Mure, 2017), as a successful application of SLAM based on feature tracking and one of the most complete keyframe-based monocular SLAM algorithms, can overcome all the limitations and shortcomings which were observed in above methods. The basic architecture of ORB-SLAM2 is based on PTAM, in which in addition to the tracking and mapping threads, a loop-closure detection thread is added. It is an oriented FAST and rotated BRIEF (ORB) (E. Rublee, et al., 2011) feature-based monocular SLAM, leveraging three main parallel threads: tracking thread is responsible for tracking feature points in real-time, local mapping thread to perform g local Bundle Adjustments (BA) map, and loop closing thread is considered to suspend the accumulated drift in-camera trajectory and performing a pose graph optimization. It is noteworthy that, at the time, ORB-SLAM2 including map reuse, loop closing, and re-localization capabilities, is one of the best Visual SLAM frameworks in which its superiorities are visible to all researchers in the computer vision field. ORB-SLAM is relatively stable and accurate, and it can be adapted to various environments, such as indoor/outdoor and large/small scale, and can be executed in real-time on a PC. Also, it is a good reference for learning and studying SLAM methods, supporting automatic map initialization, and the keyframe and map point management mechanisms are relatively comprehensive. However, like other Visual SLAM algorithms, it still has many shortcomings, and it is vulnerable to some circumstances such as motion blur from fast motions, the lack of scene texture, which can make Visual SLAM methods fail or perform very poorly.

2.7 WELL-KNOWN CHALLENGES

In general, Visual SLAM algorithms confront some challenges, which are listed in the followings subsections:

2.7.1 Pure Rotation

Pure rotational is the most prevalent problem in Visual SLAM methods, as it would occur when the user moves a device in handheld augmented reality applications. This problem explicitly can be observed in monocular cameras owing to a lack of ability in observation of disparities during purely rotational motion. It is necessary to mention that purely rotational motion cannot be considered as a problem in RGB-D Visual SLAM, and this is due to fact that the tracking and mapping processes can be executed by using the depth obtained in maps.

2.7.2 Initialization

Since SLAM aims to solve localization and mapping at the same time, this is a challenge of robot localization during the first moments when no map is available. This problem can be straightforward solved using stereo cameras and lasers or could be a serious challenge when a monocular camera is used.

2.7.3 Real-time

The goal of SLAM is to provide localization and map information to the robot, so that it can be used to accomplish its mission. Therefore, all algorithms have real-time constraints and should be scaled well both in long-term and large-scale operation.

2.7.4 Intrinsic Camera Parameters Estimation

In most of VSLAM algorithms, intrinsic camera parameters have been considered known. This implies that camera calibration should be carried out before the use of VSLAM applications, and intrinsic camera parameters have to be regulated during the estimation process.

2.7.5 Scale Ambiguity

Absolute scale information is a requirement in some VSLAM applications, using a monocular camera. Indeed, in monocular vision systems, lack of knowledge about metric distances caused by the inherent scale ambiguity could be a major limitation in some applications. In other words, when a 3-D scene is captured by the camera and projected into a 2-D frame, depth information is lost. To tackle this problem, one method is to fuse inertial measurements with monocular odometry or tracking thread to estimate metric distance. Having an acceptable estimation of metric distance results in an increase of pose estimation accuracy.

2.8 SUMMARY

In this chapter, a general view of monocular Visual SLAM was presented. We discussed some important Visual SLAM approaches such as PTAM, LSD-SLAM, SVO, DTAM, DDTAM, CD-SLAM, Mono-SLAM, RD-SLAM and Fast-SLAM. The advantages and disadvantages of each of these methods have been identified. For instance, the drawback of the PTAM algorithm was that the re-localization is based on the correlation of low-resolution thumbnails of the key-frames and yields a low invariance to a viewpoint, and it did not detect large loops. Another approach was related to LSD-SLAM that had been reported that this method still needed features for loop detection, and camera localization accuracy is significantly lower than PTAM. Also, ORB-SLAM and ORB-SLAM2 algorithms have comprehensively been explained.

So far, from the literature, we realized that the ORB-SLAM2 is the most reliable and complete solution for monocular Visual SLAM, and it can be considered as the most representative state-of-the-art visual SLAM in feature-based methods. The significant merit of this algorithm is that it supports monocular cameras, stereo cameras, and RGB-D cameras, and can produce high-precision results in real-time.

In this thesis, ORB-SLAM2 (Raul mure, 2017) is chosen as a base Visual SLAM system due to its robustness and ability to work in real-time indoor or outdoor, as well as its ability to close loops. This method is robust against difficult scenarios by inserting key-frames as quickly as possible, and removing later the redundant ones, to avoid the extra cost. However, we believe that it suffers from several problems such as the inconsistency in initialization, and sometimes, owing to fast movement and pure rotation, the features took by ORB-SLAM are lost in the tracking loop and subsequently results in failure camera pose estimation, particularly in a textureless environment or less texture environment. To tackle these problems, in the next chapter, we propose a visual-inertial SLAM in the framework of sensor fusion, combining the IMU measurements and the camera information.

3 VISUAL-INERTIAL SLAM

In the previous chapter, we explained the Visual-SLAM (V-SLAM) algorithm specifications. In this chapter, to tackle the challenges in V-SLAM and improvement of its accuracy in the indoor environment, Visual-Inertial SLAM (VI-SLAM) is introduced. In the following, this chapter aims to focus on how to fuse the monocular Visual-SLAM with IMU, and to tackle the challenges in state initialization in VI-SLAM, heavily affecting the performance of the system.

In this way, the aim is to provide a tightly coupled and optimization-based VI-SLAM system, leveraging an accurate initial states estimation in which a novel formulation using adjustable hyperparameter which considers the gravity norm. Besides, the refinement of scale and gyroscope bias is another approach that would boost the accuracy and robustness of initialization states and the overall system. To this end, in this thesis, ORB-SLAM2 is considered as the base SLAM system due to its robustness and its ability to work in real-time, in a GPS-denied environment, besides its ability to close loops treatment.

3.1 INTRODUCTION

In a dynamic system, such as robotics, the states change over time. As an example; the states of the quadrotor might consist of the current position, orientation, and velocity. To achieve the accurate control of such a system, specific sensors are utilized to gather information and derive the current state as accurately as possible. Generally, the challenge of visual-sensor-based SLAMs is higher than non-visual-sensor-based SLAMs. It is because of less visual input can be acquired by cameras (such as monocular), providing a limited field of views compared to laser scanner measuring up to 360°, for instance. In this kind of inputs, the camera poses need to be accurate and also can be continuously estimated. For this reason, to increase accuracy, one possibility is to combine visual sensors with other sensors. In general, combining the multi-sensors is performed to acquire high accuracy and, finally, to get a robust state estimation.

On one hand, V-SLAM algorithms can provide good tracking and rich map information in visually distinguishable environments. However, these algorithms are vulnerable to some circumstances such as motion blur from fast motions, the lack of scene texture, scale ambiguity in a monocular setup, occlusions, illumination changes, and these conditions can make V-SLAM methods to fail or perform very poorly. On the other hand, to be more accurate, inertial sensors, such as IMU as a complementary sensor, can provide self-motion estimation at high frequency, robust to aggressive motion, allow to recovery of the global roll, pitch, angles provide absolute scale in motion, and estimate gravity direction. Indeed, fusing the inertial sensors with V-SLAM systems can provide accurate and robust state estimation in different situations, which results in an appropriate solution to mitigate all these problems above, originated from the complementary nature of these two types of sensors. Because IMU can temporarily track the motion, of the sensor even when there are no enough features or texture for visual tracking, consistent visual observations assist in rectifying the biases in the inertial measurements (M. Hsiao, at al., 2018).

All the explanations mentioned above lead to a hot topic in the sensor fusion field that is known as visual-inertial SLAM (VI-SLAM), fusing camera and IMU data for localization, which has gained much attention for various reasons. Firstly, in the fast-paced world of technology, the robotic systems are being increasingly used, particularly in a wide variety of intelligent systems and applications involving autonomous exploration and navigation of Micro Aerial Vehicles (MAVs). Secondly, Augmented Reality (AR) and Virtual Reality (VR) systems have become much popular among AR developers.

To carry out the improved comparison and identify the pros and cons of VI-SLAM algorithms, the literature has been divided into filtering-based methods and optimization-based methods.

3.2 ESSENTIALS ON VI-SLAM

VI-SLAM algorithms have been categorized into two approaches: filtering-based and optimization-based. Maplab (T. Schneider, at al., 2018 & S. Lynen, at al., 2015) and VINS-mono (T. Qin, et al., 2017, Y. Lin, et al., 2017 & P. Li, et al., 2017) are typical examples of these two approaches which both of them are open source. Filtering-based methods are frequently used for Visual-Inertial Navigation (VIN), in which marginalization is performed to all the previous IMU states, including poses, velocities, and biases, to achieve fast computation. Compared to the filter-based methods, optimization-based methods with bundle adjustment exceed with high accuracy estimation than the filters, whereas, there is no doubt that the filtering method has the superiority of robustness and high response frequency.

3.2.1 Filtering-Based Methods

According to the type of sensor fusion, VI-SLAM approaches can be grouped into either loosely or tightly coupled based on.

3.2.1.1 Filtering-based and loosely-coupled

The Loosely-coupled approach (S. Weiss et al., 2013) considers the Visual-Odometry (VO) module (explained in Chapter 2) as a 'black box' which yields the 6-DoF pose as output and integrates this result with IMU measurements using filtering-based methods (for instance, EKF) to estimate the states, including IMU biases and absolute scale. In simple terms, the loosely coupled systems take the visual sensor and IMU as two separate modules and calculate the orientation and possible the change in position (but not full pose), using the filtering-based methods for fusion. The main advantages of loosely coupled approaches are their relative simplicity and high scalability which can join a variety of sensor information to combine. However, the most important drawback of them is low fusion accuracy.

In filtering-based and loosely coupled methods, EKF plays a key role so that in 2011, S. Weiss et al., added an IMU to monocular Visual SLAM, and presented a metric state estimation based on an Extended Kalman Filter, in which their proposed strategy decouples the two frameworks, visual pose estimate and metric scaled state estimation.

(M. Achtelik, et al., 2011) presented a solution to address the issue of having a low frequency onboard visual pose update versus the high agility of a MAV. This problem is solved by filtering visual information with inputs from inertial sensors. Indeed, since the proposed system is based on monocular Visual SLAM, they performed a separated monocular visual SLAM framework, which estimates the absolute scale with the assist of an air pressure sensor and IMU from the Flight Control Unit (FCU).

Multi-Sensor-Fusion Extended Kalman Filter (MSF-EKF) is provided by (S. Lynen, et al., 2013) in a general EKF framework for fusing the data from different sensors in a state estimate, being able to process delayed measurements, both relative and absolute, from a theoretically unlimited number of different sensors and sensor types with online self-calibration.

In 2016, R. Munguía, et al proposed a visual-aided inertial navigation and mapping system, in which a filtering-based algorithm is employed to solve the full state of autonomous robots. The system, which relies on Kalman filtering, is designed to fuse the measurements obtained from a monocular camera, an IMU, and a loosely coupled strategy is used for incorporating data provided by the GPS, whenever it is possible.

Since real-time 6-DoF motion tracking is essential for the registration between virtual scenes and the real world, visual-inertial fusion has been employed in AR/VR systems. Hence, (W. Fang, et al., 2017) presented a visual-inertial-based real-time motion tracking for mobile AR/VR in which, by combining a monocular camera and an IMU, 6-DoF motion tracking is estimated by sensor-fusion in real-time. Moreover, to able real-time and smooth 6-DoF motion tracking, an adaptive filter framework is proposed to balance the jitter and latency phenomenon.

The metric distance originated from the scale ambiguity severely affects the accuracy of vision systems and is a strong limitation for some applications. In this way, (A. Spaenlehauer, et al., 2017) addressed this problem by fusing inertial measurements with monocular odometry (ORB-SLAM algorithm for monocular tracking input) to estimate the metric distance in a loosely coupled manner.

3.2.1.2 *Filtering-based and tightly-coupled*

Compared with the loosely coupled method which optimizes the 6-DoF output from the VO methods instead of the raw visual measurements together with the inertial measurements, the tightly coupled approach is a method that jointly optimizes the state of the camera and IMU together into a motion and observation equation in each iteration of the nonlinear optimization and then performs state estimation. Although the tightly-coupled method is more costly than the loosely-coupled, thanks to advances in computer technology, it can achieve more accurate results. Tightly-coupled methods constitute the main focus of the research area in sensor fusion.

Multi-State Constrained Kalman Filter (MSCKF) (A.I. Mourikis, et al., 2007) is an early method and popular EKF-based Visual-Inertial Odometry (VIO) approach which is a tightly coupled algorithm. In this system, the visual information and IMU data are combined into a filter and the body poses are updated by a 3D key-point processing with high accuracy. The MSCKF extracts and matches the SIFT feature, and maintains 30 camera poses in the filter state, and uses visual measurements of the same feature across multiple camera views to form a multi constraint update. In this way, another primarily filter-based method for visual-inertial fusion was proposed by (E. S. Jones, et al., 2011), which focuses on the integration of visual and inertial sensors and their use in the ego-motion estimation, localization, and mapping.

In 2013, M. Li, et al proved that, compared to standard methods (such as EKF-SLAM), the MSCKF algorithm outperforms in terms of accuracy, consistency, and computational efficiency. However, they believe that the MSCKF makes no Gaussianity assumptions on the features' positions, which is required in EKF-SLAM. Then, they concentrated on improving the consistency of the MSCKF which could be the main core for boosting the estimation states' accuracy. To this end, a closed-form expression for the IMU's error-state transition matrix is derived that could be employed in any case in which an IMU is used for estimation.

(J. A. Hesch, et al., 2013) developed an Observability Constrained MonoSLAM (OC-MonoSLAM), which explicitly enforces the unobservable directions of the system, hence preventing spurious information gain and decrease inconsistency. This framework can be exerted to several variants of the VINS problem such as Visual SLAM and VIO systems using the MSCKF.

Apart from robotic systems, AR systems such as mobile phones are other platforms that have been used in visual-inertial algorithms. As an example, (X. Chao, et al., 2014) introduced an interpolation-based camera measurement model, targeting visual-inertial navigation using cell phones containing low-grade rolling-shutter cameras. From the perspective of the type of sensor fusion, an Observability-Constrained Extended Kalman filter (OC-EKF) has been employed to improve the VINS consistency and accuracy of the system. Besides, an interpolation model is proposed to express the camera pose of each visual measurement, as a function of adjacent IMU poses, that are included in the estimator's optimization window.

The square root inverse sliding window filter (SR-ISWF) for Visual-inertial navigation systems (VINS) is another filtering-based and tightly coupled approach that has been implemented on a mobile phone (L. Wu, et al., 2015). This algorithm maintains the upper triangular Cholesky factor of the Hessian matrix of a sliding window of recent states, and achieved more than double the speed, with comparable accuracy, of the MSCKF provided by (A.I. Mourikis, et al., 2007). (P. Tanskanen, et al., 2015) presented a combination of advantages of EKF-based approaches with those of direct photometric error minimization methods, in which inertial data and vision-based surface measurements are used simultaneously during camera pose estimation. In this algorithm, IMU information is tightly integrated with direct surface measurements, allowing to track image regions that are difficult to tackle with approaches that rely on feature trackers. Whereas, their approach starts tracking without a special initialization sequence such as IMU biases estimation.

(M. Bloesch, et al., 2015) proposed the Robust Visual Inertial Odometry (ROVIO) which is an EKF-based monocular system that updates the pose state using multi-level patches around feature points with the propagated IMU motion and minimization of photometric errors. This system, which is open source[1], is used to directly detect luminosity error to obtain accurate, robust tracking from image matching. A significant disadvantage of ROVIO is the lack of loop closure to attenuate the accumulated errors.

(I. Sa, et al., 2017) built a visual-inertial vertical takeoff and landing (VTOL) platform to practically combine the ROVIO (M. Bloesch, et al., 2015) as a robust visual odometry and a state-of-the-art controller (MPC) with traditional dynamic. In their implementation, the IMU biases and camera–IMU extrinsic is also included in the filter state and, final states are co-estimated online for higher accuracy. Even though they considered the scale estimation initialization, the effect of the gravity vector is neglected, which could improve the accuracy of the system.

Robust and accurate state estimation has always been a challenge in robotics. If the system can obtain accurate pose estimation based on a prior map, then system adaptability will improve. To this end, (T. Schneider, at al., 2018) proposed a VI-SLAM system called Maplab, an open-source algorithm[2], in which pre-integrated IMU measurements are inserted into the optimization, and the algorithm includes a map merging, loop closure, and visual-inertial optimization. The system extensibility is suitable for research and provided the evaluation method for the selection of system mining components.
Although Maplab attained acceptable results in benchmark, the gravity and accelerometer bias are not distinguished from each other in initialization.

Recently, a novel filter-based and a tightly coupled monocular VI-SLAM has been introduced by (M. Quan, et al., 2019), combining the advantages of filtering-based and optimization-based approaches, which ensures the fast response of the system to the highly dynamic motion of robots and tracking the motion through the visual-inertial EKF (as an assistant). Since the filter becomes inconsistent due to linearization errors, the globally

[1] https://github.com/ethz-asl/rovio
[2] https://github.com/ethz-asl/maplab

consistent map is constructed and then it feeds back the map to the EKF state vector. Simultaneously, in a parallel thread, a global map is performed and a keyframe-based visual-inertial bundle adjustment are executed to optimize the map. Furthermore, a loop closure detection and correction module are also employed in a parallel thread to eliminate the accumulated drift when revisiting an area. The main drawback of this algorithm is that the gravity vector is neither refined nor constrained.

In tightly integrated filters, the prediction step typically propagates the current camera state estimate using the IMU measurements. The state is recursively corrected based on the camera images. V. Usenko, et al. (2019) did a research in which, the significant drawback of filters is that the linearization point for the non-linear measurement and state transition models cannot be changed, once a measurement is integrated.

3.2.2 Optimization-Based Methods

In the fast-paced world and with the advancement of computers, the use of optimization-based VI-SLAM has steeply increased. In optimization-based methods, the entire SLAM frame is divided into a front-end and back-end according to the image processing; the front-end is responsible for map construction, whereas the back-end is responsible for pose optimization. Compared to filter-based methods, optimization techniques enable fast computation by using sparse linear solvers such as g2o (R. Kümmerle, et al., 2011), Ceres (S. Agarwal, et al., 2017), iSAM (M. Kaess, et al., 2008), and GTSAM (F. Dellaert, 2012), which can solve the optimization problems with tens thousands of variables near-realtime in terms of calculation speed.

OKVIS (Leutenegger, et al., 2014, 2015) is a classic tightly-coupled - optimization-based V-I SLAM - an open-source algorithm that is available in a ROS-compatible package[3], and utilizes non-linear optimization on a sliding window of keyframe poses. In this algorithm, the inertial measurements are tightly integrated into the keyframe-based visual SLAM system, in which the cost function comprising the IMU error term and the reprojection error term was jointly optimized. Furthermore, to maintain a bounded sized optimization window, the old states are marginalized. Therefore, OKVIS could enhance accuracy and robustness in real-time operation. However, since as a first step to initialization

[3] https://github.com/ethz-asl/okvis

and matching, the last pose is propagated using acquired IMU measurements to obtain a preliminary uncertain estimate of the states in this system. IMU integration is needed to be repeatedly computed when the linearization point changes.

One of the main important issues in visual-inertial mapping is to perform the global consistent optimization, which receives less attention in the computer vision community. While in principle, the optimization can be formulated as Bundle Adjustment (BA) along with additional IMU information, this approach would rapidly become computationally infeasible owing to the large number of frames which would lead to a high number of optimization parameters in a naive formulation. To tackle this problem and keep the computational burden in bounds, BA makes the high-frame-rate images of the camera to a smaller set of keyframes. Therefore, the common strategy in VIO is to pre-integrate IMU measurements between two consecutive frames (V. Usenko, et al., 2019).

To reduce the computation and avoid the repeated constraints caused by the parameterization of relative motion integration, pre-integration was the first time proposed by (T. Lupton, et al., 2012). In their system, IMU data were changed between two frames by pre-integrating the constraints, which could significantly decline the computation. This theory was further developed by (C. Forster, et al., 2015) and applied to the VI-SLAM framework to reduce bias. In fact, by using IMU pre-integration, it is not needed to reintegrate all IMU measurements each time that bias changes, and this practice can be performed by updating the bias changes using Jacobians w.r.t. bias. However, using a formulation from (C. Forster, et al., 2015), reintegration can be avoided integrating IMU measures only once, and updating the pre-integrated terms through a linear approximation.

The first direct tightly coupled algorithm for the VI-SLAM system was proposed by (A. Concha, et al., 2016) that could run in real-time under a standard CPU. The processing is split into three threads. The first thread, which runs at frame rate, is responsible for the estimation of camera motion by a joint non-linear optimization from visual and inertial data given a semi-dense map. The second one creates a semi-dense map of high-gradient areas only for camera tracking purposes. Eventually, a fully dense reconstruction of the scene at a lower frame rate is estimated in the last thread. However, the initialization was not introduced in this strategy.

VI-ORB-SLAM (R. Mur-Artal, et al., 2017) can be considered the most reparative of tightly coupled and nonlinear optimization approach, which contains an ORB sparse front-end, graph optimization back-end, loop closure, and relocation. This system can close the loop and reuse the previously constructed 3D map. This system leverages a specific initialization that can estimate the initial states such as scale, gravity direction, velocity, and accelerometer and gyroscope biases. The local map module uses local BA to optimize the latest N keyframes and all points observed on these N keyframes after a new keyframe is inserted. Local maps are then retrieved based on the time series of the keyframe. The fixed window connects the N+1st keyframe and the co-visibility graph. Although the gravity and bias acceleration is estimated together, it is necessary to wait up to 10 seconds to observe all the values, and it uses much CPU time. Besides, sometimes, it suffers from feature loss in extreme environments which leads to failure in pose estimation.

(T. Qin, et al., 2017) proposed a versatile monocular visual-inertial odometry system VINS-Mono, a novel real-time, non-linear optimization-based sliding window estimator, which performs local BA in one thread to estimate the state of the platform, and closes loops in a lightweight manner in a parallel thread. In this system, a loosely-coupled sensor fusion initialization procedure is provided to bootstrap the estimator from arbitrary initial states. A tightly coupled approach is employed to combine the pre-integrated IMU measurements and feature observations. Besides, to improve the accuracy, a tightly-coupled procedure for re-localization is proposed. VINS-Mono is a real-time and open-source algorithm[4] that is available. Although VINS-Mono is an appropriate initialization state estimator, (M. Quan, et al., 2019) have proved that in VI-SLAM, due to having the additional unobservable directions (i.e. the scale of the environment and the direction of local gravity) in the special motion VINS-Mono become unable to provide a good initial value for the dataset, resulting in the drift of the estimated trajectory.

An adaptive monocular visual-inertial SLAM for AR applications in mobile devices is introduced by (J. Piao, et al., 2017) which combines data from a mobile device camera and inertial measurement unit sensor. Besides this VI-SLAM, an optical-flow-based fast visual odometry is proposed that can estimate the pose in real-time. The main goal of this strategy is to provide an adaptive execution module that dynamically selects visual-inertial odometry or optical-flow-based fast visual odometry.

(Y. Liu, et al., 2017) presented a tightly coupled visual-inertial SLAM system that can run with real-time performance in an unknown environment. To this end, authors employed a parallel framework with a novel IMU initialization method, which benefits from the novel IMU factor, the continuous pre-integration method, the vision factor of directional error, the

[4] https://github.com/HKUST-Aerial-Robotics/VINS-Mono

separability trick, and the robust initialization criterion which reliable estimation of states can be performed on Central Processing Unit (CPU). However, in one of their implementations which was in a difficult scenario (texture-less) environment, their strategy could not work well and obtain the result.

A robust initialization and online scale estimation in VIO is an algorithm that recently has been introduced by (E. Hong, et al., 2018). The main contribution of their work is to stabilize the initialization of scale and gravity using relative pose constraints. To attain this, a local scale parameter is adopted in the online initialization process which is responsible for taking into account the ambiguity and uncertainty of VIO initialization. However, at the beginning of their algorithm, gyroscope bias has not been estimated, and also, the gravity vector is not rectified.

To increase the accuracy of visual-inertial SLAM algorithms, (C. Campos, et al., 2019) have concentrated on the estimation of the initial state and proposed a fast joint monocular-inertial initialization method, based on the work of (A. Martinelli, 2014) and (J. Kaiser, et al., 2017).

Their outcomes showed that the original Martinelli-Kaiser technique is not good enough in initialization in most practical scenarios. Hence, they have proposed two visual-inertial BA steps to improve the solution and two novel tests to detect bad initializations. Even though their strategy declined the scale error down and rejected bad initializations, due to the use of two visual-inertial BA steps, it rises the computational cost.

(W. Huang, et al., 2018) considered the camera-IMU extrinsic parameters in the initialization stage of monocular visual-inertial SLAM techniques. In this way, since the repeating, the calibration of extrinsic parameters can be challenging while the sensor changes slightly, the authors propose an online initialization method to automatically estimate the initial values and the extrinsic parameters without knowing the mechanical configuration. Their method automatically estimates the visual scale, velocity, gravity, biases of gyroscope and accelerometer, and calibrates the camera-IMU extrinsic parameters while the system is performing free motion in environments. Besides, it can automatically identify the convergence of the calibration parameters so that the initialization stage can be terminated.

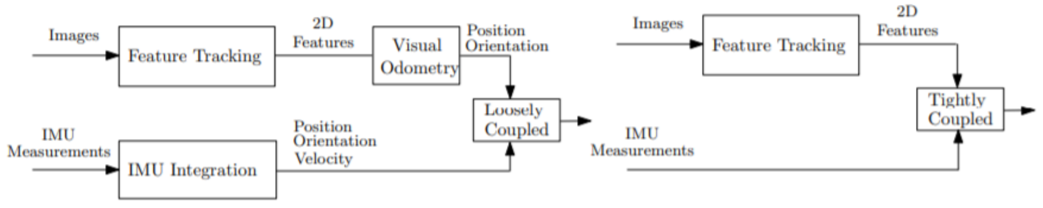
Recently, (B. Nisar, et al., 2019) provided a visual-inertial model-based odometry system in which a relative motion constraint combining the robot's dynamics and the external force in a pre-integrated residual has been taken into account, resulting in a tightly coupled, sliding-window estimator exploiting all correlations among all variables. The comparison of their system with the original pipeline (VINS-Mono) without motion constraints proved that the estimator not only improves odometry accuracy on real-world data, but also estimates time-varying external forces without increasing the computation time. Since datasets, such as EuRoC MAV, does not provide rotor speed measurements or commanded thrust, the proposed algorithm has not been evaluated on them.

The poor performance of initialization in Visual-Inertial SLAM algorithms heavily affects the accuracy of initial values. For this reason, (X. Mu, et al., 2018) addressed this problem so that they refine the estimated gravity vector by optimizing the two-dimensional (2D) error state on its tangent space. Then, the accelerometer bias is separately estimated, which is difficult to be distinguished under small rotation. Although their approach to initialize the states is robust, they used the Singular Value Decomposition (SVD) in a separate step to refine the gravity, which computationally is expensive and slower than optimization methods.

With a more accurate comparison, it can be concluded that in a tightly coupled method, IMU pre-integration is used to predict the 2D feature locations in the next frame, leading to facilitate feature tracking. In contrast, loosely-coupled approaches do not consider the visual and inertial information coupling, making them incapable of eliminating the accumulated drift from the usability of inertial measurements, which leads the resulted estimate to be sub-optimal.

The difference between these two approaches is explicitly illustrated in Figure 8.

Figure 8 - Comparison of loosely (left) and tightly coupled (right) paradigms for VIO



In terms of comparison of filtering-based and optimization-based methods, the filtering approaches suffer from drift and exhibit only a limited representation of the global environment. Although they are able to close loops topologically and reuse its map, the global metric consistency is not enforced in real-time. In addition, the maintaining of the dense covariance matrix in EKF is very expensive so that the size of features has to be very limited. In contrast, optimization-based methods jointly perform the nonlinear optimization overall coupled sensor states, which can also perform loop closures to compensate for drifting behavior and provide globally consistent maps. However, they often demand costly computations for platforms that are small, and they demonstrate the state estimations without feedback control.

In optimization-based methods, full smoothing is an approach for estimating the entire history of the states by solving a large nonlinear optimization problem (Jung and Taylor, 2001; Sterlow and Singh, 2004; Bryson et al., 2009; Indelman et al., 2013; Patron-Perez et al., 2015). Even though full smoothing guarantees the highest accuracy, since it can update the linearization point of the complete state history as the estimate evolves, it suffers from the complexity of the optimization problem which is approximately cubic concerning the dimension of the states. Then, the real-time operation quickly becomes infeasible as the trajectory and the map grow over time (D. Scaramuzza, et al., 2019).

As a result of related works, it can be explicitly observed that by integrating the advantages of various branches of SLAM techniques (such as filtering and optimization-based approaches and loosely and tightly coupled methods), the robustness and accuracy of the system would greatly be improved. Besides, the initial state estimation of VI-SLAM systems plays a key role in boosting the accuracy of the system, which its weakness results in incorrect estimates and subsequently drift in the estimated trajectory.

3.3 VI-SLAM THEORETICAL BACKGROUND

To start off and dealing with the proposed VI-SLAM system, some preliminaries are required as the basis of our contribution. In fact, to a large extent, these preliminary discussions are common beginning among most of the VI-SLAM literature.

To estimate the real-time state in visual-inertial odometry, the monocular camera and IMU which form a sensing device should be correctly aligned with each other. Therefore, the first step to start working with VI-SLAM system is to model the camera integrated with inertial sensors.

Having both camera and IMU causes that features are observed at different poses. For this reason, camera and IMU should be correctly aligned to each other which states of the sensing system to be concurrently estimated.

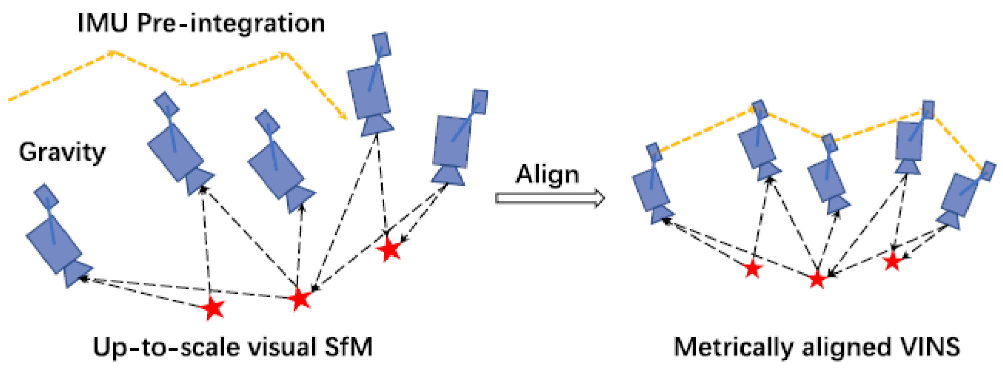
Looking at the Figure 9, it can be observed that the several keyframes and IMU measurements are maintained in a sliding window. In this process, to jointly optimize the camera and IMU states, as well as feature location, a local bundle adjustment (BA) would be implemented when a new keyframe is inserted into the sliding window.

In this figure, the basic ideas are: i) to make the well-aligned transformation between different coordinate frames and multiple states of the camera–IMU, ii) to match the up-to-scale visual structure with IMU pre-integration. In this way, Vision-Only measurement and IMU pre-integration are two key issues that are explained in this section. However, before it, some notations concerning camera-IMU alignment need to be described.

In this study, $(\cdot)_C$ is considered as the camera frame, which is an arbitrarily fixed frame in a visual structure. $(\cdot)_W$ denotes the world reference frame where gravity vector is along with z-axis, in which $\mathbf{g}^W = [0; 0; g]^T$ is the gravity vector in the world frame. and $(\cdot)_B$ represents body frame. Besides, we treat the IMU frame as the body frame, which means the IMU frame is aligned with the body frame. In this step, we perform vision-only structure from motion (SfM), then loosely align IMU measurements with SfM results to get metric initial states.

Figure 9 demonstrates how to align the up-to-scale visual SfM with IMU measurement.

Figure 9 - Camera-IMU alignment strategy in visual-inertial fusion.



Besides, there is a camera calibration which estimates the parameters of a lens and image sensor and can be utilized to correct for lens distortion, size measurement of an object in world frame or determine the position of the camera in the scene. In camera calibration, there are two types of calibrations; i) extrinsic calibration, and ii) intrinsic calibration. Extrinsic parameters specify the translation and rotation vectors of the camera in the world coordinate system, while intrinsic parameter can be considered as a diagonal matrix including the focal length, the number of pixels per x and y unit of image coordinates and principal points (R. J. Radke, 2009). In this study, we assume that the intrinsic calibration of the camera and extrinsic calibration between the camera and IMU is known in the initialization step. We do not need a very precise extrinsic calibration, since we will continuously refine it in the nonlinear optimization.

3.3.1 Visual Measurement

The initialization procedure starts with a vision-only structure, which estimates a graph of up-to-scale camera poses and feature positions. In order to model the visual measurement, a re-projection function $\pi(\cdot) : \Re^3 \rightarrow \Re^2$ has been considered, which maps a 3D landmark $g_l^c = [x_l^c.\ y_l^c.\ f_l^c]^T$ in the camera frame to a 2D points $f_l = [u_l.\ v_l]^T \in \Re^2$ in image coordinate frame. Note that the l and c stand for camera and landmark respectively. By considering the 2×1 measurement noise with covariance $\sum_{\sigma_l}$ in conventional pinhole-camera model (R. Hartley, et al., 2003), the reprojection function is expressed as $\tilde{z}_l = f_l + \sigma_l$. Please notice that the formulation of the pinhole model without considering noise ($\sum_{\sigma_l} = 0$) becomes:

$$\tilde{z}_l = \pi(g_l^c) = \begin{bmatrix} f_u \frac{x_l^c}{z_l^c} + c_u \\ f_v \frac{y_l^c}{z_l^c} + c_v \end{bmatrix}, \tag{4}$$

where $[f_u \quad f_v]^T$ and $[c_u \quad c_v]^T$ denote the focal length and principal points, respectively, which are obtained during the camera calibration. Leveraging the visual measurement model in (4), the re-projection error function $\boldsymbol{r}_C(g_l.\chi_k) \in \Re^2$ is given by:

$$\boldsymbol{r}_C(g_l.\chi_k) = \pi\left(\mathbf{R}_C^B \ \ (\mathbf{R}_\mathrm{B}^\mathrm{W} \ \ (\boldsymbol{g}_l - \mathbf{p}_B^W) - \mathbf{p}_C^B)\right) - \tilde{\mathbf{z}}_l. \tag{5}$$

where $\mathbf{R}_\mathbf{B}^\mathbf{W}$ and $\mathbf{p}_\mathrm{B}^\mathrm{W}$ are the orientation from frame {B} to {W} and 3D position of {B} with respect to {W}, respectively. Moreover, $\mathbf{R}_C^B$ and $\mathbf{p}_C^B$ denotes the rotation and translation between the mounted camera-IMU sensor that is computed from the calibration.

3.3.2 IMU pre-integration

In practice, the IMU frequency is much higher than the camera. For instance, when the IMU frequency is 200 Hz, the camera frequency maybe 20 Hz. Therefore, the IMU Pre-Integration is a common strategy in tightly-coupled approaches to fuse the IMU and camera information.

In general, an IMU measures the angular rate and the acceleration of the sensor in the body frame {B}, combining the force for countering gravity dynamics of the system. These measurements are exposed to acceleration and gyroscope noises, such as white noises $\boldsymbol{\eta}_{\mathrm{a}}$ and $\boldsymbol{\eta}_{\mathrm{g}}$, respectively. As well as noises, measurements are affected by gyroscope bias, $\mathbf{b}_{\mathrm{g}}$, and acceleration bias, $\mathbf{b}_{a}$, which should be considered in the IMU pre-Integration model. The raw gyroscope and accelerometer measurements, $\widehat{\boldsymbol{\omega}}$ and $\hat{\mathbf{a}}$ at time *t,* are given by:

$$\begin{aligned} \hat{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{b}_{\mathrm{a}_t} + \mathbf{R}_{\mathrm{W}}^{t}\mathbf{g}^{\mathrm{W}} + \boldsymbol{\eta}_{\mathrm{a}}. \\ \widehat{\boldsymbol{\omega}}_t &= \boldsymbol{\omega}_t + \mathbf{b}_{a_t} + \boldsymbol{\eta}_{\mathrm{g}}. \end{aligned} \tag{6}$$

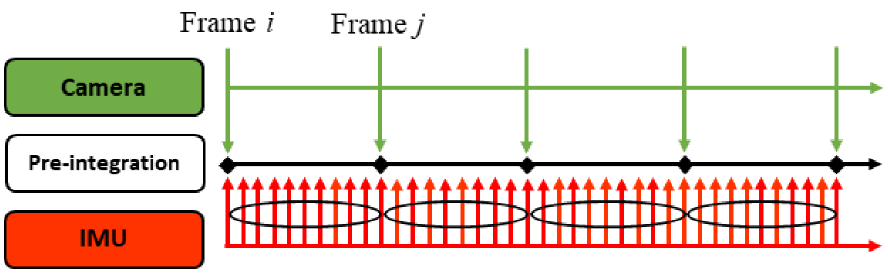
As it is explained earlier, the IMU pre-integration was extended by Forster et al. (2015) on the manifold space. Based on this concept, given two sequence keyframes at the time *i, j,* the IMU orientation $\mathbf{R}_{\mathrm{WB}}$, ${}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}$ and translation ${}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}}$ (between world frame {W} and body frame {B}) can be estimated by using measurement equation as follows

$$\begin{aligned} &\mathbf{R}_{\mathrm{WB}}^{\mathrm{j}} = \mathbf{R}_{\mathrm{WB}}^{\mathrm{i}} \prod_{k=i}^{j-1} \mathrm{Exp}\left(\left(\boldsymbol{\omega}_{\mathrm{B}}^{k} - \mathbf{b}_{\mathrm{g}}^{k} - \boldsymbol{\eta}_{\mathrm{g}}^{k}\right)\Delta\mathrm{t}\right), \\ &{}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{j} = {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i} + {}_{\mathrm{W}}\mathbf{g}\Delta\mathrm{t}_{ij} \sum_{k=i}^{j-1} \mathbf{R}_{\mathrm{WB}}^{k}\left(\mathbf{a}_{\mathrm{B}}^{k} - \mathbf{b}_{\mathrm{a}}^{k} - \boldsymbol{\eta}_{\mathrm{a}}^{k}\right)\Delta\mathrm{t}, \\ &{}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}}^{j} = {}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}}^{i} + {}_{\mathrm{W}}\mathbf{g}\Delta\mathrm{t}_{ij} \sum_{k=i}^{j-1} {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{k}\,\Delta\mathrm{t} + \frac{1}{2}{}_{\mathrm{W}}\Delta\mathrm{t}^{2} + \frac{1}{2}\mathbf{R}_{\mathrm{WB}}^{k}(\mathbf{a}_{\mathrm{B}}^{k} - \mathbf{b}_{\mathrm{a}}^{k} - \\ &\boldsymbol{\eta}_{\mathrm{a}}^{k})\Delta\mathrm{t}^{2}, \end{aligned} \tag{7}$$

where Exp (.) denotes the exponential map operator, and $\Delta\mathrm{t}$ represents the IMU sampling interval, and $\Delta\mathrm{t}_{ij} \doteq (i-j)\Delta\mathrm{t}$.

Figure 10 illustrates the diagram of the IMU pre-integration procedure integrating all the information measured by IMU between two consecutive frames *i* and *j*.

Figure 10 - The diagram IMU pre-integration.



By neglecting the effect of IMU measurements noise, and considering a constant noise during the pre-integration process, a small bias correction $\delta\mathbf{b}^i_{(,)}$ could be taken into account to refine the obtained results from IMU pre-integration. Thus, equations (7) can be rewritten as

$$\begin{aligned}
&\mathbf{R}^j_{\mathrm{WB}} = \mathbf{R}^{\mathrm{i}}_{\mathrm{WB}}\Delta\bar{\mathbf{R}}_{ij}\mathrm{Exp}(\mathbf{J}^g_{\Delta\bar{\mathbf{R}}_{ij}}\delta\mathbf{b}^i_j),\\
&{}_{\mathrm{W}}\mathbf{v}^j_{\mathrm{B}} = {}_{\mathrm{W}}\mathbf{v}^i_{\mathrm{B}} + {}_{\mathrm{W}}\mathbf{g}\Delta\mathrm{t}_{ij} + \mathbf{R}^{\mathrm{i}}_{\mathrm{WB}}(\Delta\bar{\mathbf{v}}_{ij} + \mathbf{J}^g_{\Delta\bar{\mathbf{v}}_{ij}}\delta\mathbf{b}^i_g + \mathbf{J}^a_{\Delta\bar{\mathbf{v}}_{ij}}\delta\mathbf{b}^i_a), \qquad (8)\\
&{}_{\mathrm{W}}\mathbf{p}^j_{\mathrm{B}} = {}_{\mathrm{W}}\mathbf{p}^i_{\mathrm{B}} + {}_{\mathrm{W}}\mathbf{v}^i_{\mathrm{B}}\Delta\mathrm{t}_{ij} + \frac{1}{2}\,{}_{\mathrm{W}}\mathbf{g}\Delta t^2_{ij} + \mathbf{R}^{\mathrm{i}}_{\mathrm{WB}}(\Delta\bar{\mathbf{p}}_{ij} + \mathbf{J}^{\mathrm{g}}_{\Delta\bar{\mathbf{p}}_{ij}}\delta\mathbf{b}^i_g + \mathbf{J}^a_{\Delta\bar{\mathbf{p}}_{ij}}\delta\mathbf{b}^i_a),
\end{aligned}$$

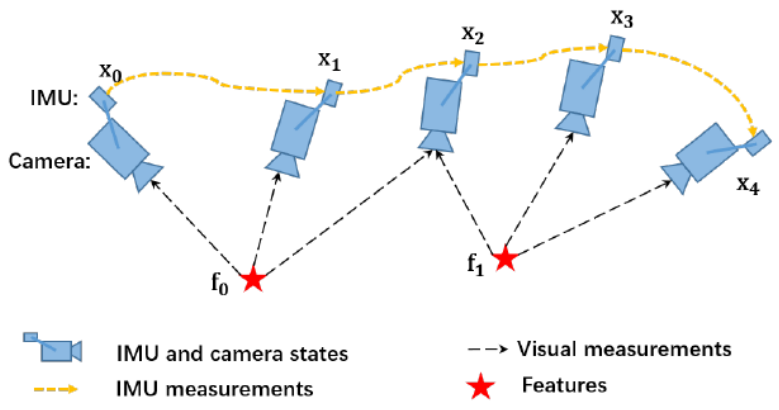
where $\mathbf{J}^g_{(,)}$ and $\mathbf{J}^a_{(,)}$ are exerted to indicate how the $\Delta(,)$ change with a bias change in bias estimation. The terms of $\Delta\bar{\mathbf{R}}_{ij}$, $\Delta\bar{\mathbf{v}}_{ij}$ and $\Delta\bar{\mathbf{p}}_{ij}$ are independent of the states a time i and the gravity, and to be more accurate, they can be computed directly from the IMU sensor between to keyframes, as follows

$$\begin{aligned}
&\Delta\bar{\mathbf{R}}_{ij} = \prod_{k=i}^{j-1}\mathrm{Exp}\left((\boldsymbol{\omega}^k_{\mathrm{B}} - \bar{\mathbf{b}}^i_{\mathrm{g}})\Delta\mathrm{t}\right),\\
&\Delta\bar{\mathbf{v}}_{ij} = \sum_{k=i}^{j-1}\Delta\bar{\mathbf{R}}_{ik}\,(\mathbf{a}^k_{\mathrm{B}} - \bar{\mathbf{b}}^i_{\mathrm{a}})\Delta\mathrm{t}, \qquad (9)\\
&\Delta\bar{\mathbf{p}}_{ij} = \sum_{k=i}^{j-1}(\Delta\bar{\mathbf{v}}_{ik}\,\Delta\mathrm{t} + \Delta\bar{\mathbf{R}}_{ik}((\mathbf{a}^k_{\mathrm{B}} - \bar{\mathbf{b}}^i_{\mathrm{a}})\Delta t^2),
\end{aligned}$$

where $\bar{\mathbf{b}}^i_{\mathrm{g}}$ and $\bar{\mathbf{b}}^i_{\mathrm{a}}$ remain constant during the pre-integration and can be recomputed at the time i.

Note that the sliding window is a strategy that keeps the computational time bounded by marginalizing out past states. As depicted in Figure 11, the sliding-window-based nonlinear optimization framework would process visual and inertial measurements in a tightly-coupled way. The nonlinear optimization parts of this process would be explained in the next sections.

Figure 11 - An illustration of sliding-window based monocular VIO. The local window keeps several keyframes and IMU measurements between consecutive keyframes. A local bundle adjustment (BA) jointly optimizes keyframes poses, velocity, IMU bias as well as feature depths.



3.4 VI INITIALIZATION

In this section, the idea is to align the up-to-scale camera pose with metric information of IMU pre-integration. In the first step of the proposed approach, ORB-SLAM2 provides visual measurements for a few keyframes. At the same time and according to the explanation detailed in section 3.3.2, the IMU pre-integration between these keyframes are computed. Indeed, the procedure is that, when a new keyframe is generated, the proposed visual-inertial initialization algorithm is executed to initialize and iteratively update the states including gyroscope bias, velocity, gravity vector, accelerometer bias and metric scale (including scale refinement). This procedure continues until the termination criterion is achieved. Firstly, the gyroscope bias is approximately estimated between two keyframes. Secondly, the initialization of velocity, gravity vector, and metric scale should be performed. Note that the metric scale estimated here is a rough scale that needs to be refined, and the gravity vector has been estimated without bias consideration. Then, by refining the gravity vector, constraining the magnitude to refine the estimated gravity vector through forming a relatively new non-linear linear least-square problem, the velocities, the gravity vector, and scale parameter are initialized. After that, in order to estimate the acceleration bias and rectify the scale, we estimate the scale acquired in the previous stage with acceleration bias together. Finally, the gyroscope bias is recomputed which would improve the overall initialization accuracy.

It is noticeable that the technical contribution of this thesis, which is to consider the adjustable parameters to estimate the initial states, is explained in this section.

3.4.1 Gyroscope Bias Initialization

Possessing the orientation data $\mathbf{R}_{\mathrm{wB}}^{\mathrm{i}}$ and $\mathbf{R}_{\mathrm{wB}}^{\mathrm{j}}$ from Visual ORB-SLAM2, and rotation changes $\Delta\bar{\mathbf{R}}_{ij}$ from IMU pre-integration in (9), the gyroscope bias estimation between two consecutive keyframes *i, j* can be described by minimization of errors between relative rotation from camera gyroscope integration as (W. Huang, et al., 2018)

$$\delta\mathbf{b}_{\mathrm{g}}^{*} = \underset{\delta\mathbf{b}_{\mathrm{g}}}{\operatorname{argmin}} \textstyle\sum_{k=i} \left\| Log(\Delta\bar{\mathbf{R}}_{ij} Exp(\mathbf{J}_{\Delta\bar{\mathbf{R}}_{ij}}^{g} \delta\mathbf{b}_{\mathrm{g}}))^{T} \mathbf{R}_{\mathrm{WB}}^{\mathrm{i}} \mathbf{R}_{\mathrm{BW}}^{\mathrm{j}} \right\|^{2}, \tag{10}$$

where *k* denotes the number of keyframes, $\|\cdot\|$ is the L2-norm, and $\mathbf{R}_{\mathrm{WB}}^{(\cdot)} = \mathbf{R}_{\mathrm{WC}}^{(\cdot)}\mathbf{R}_{\mathrm{CB}}$ can be computed by transforming the pose of the IMU to the world coordinate system. Then, the optimal value of gyroscope bias $\delta\mathbf{b}_{\mathrm{g}_{3\times1}}^{*}$ can be obtained by solving (10) through the Gauss-Newton algorithm (R. Kummerle, et al., 2011). The numerical solution of the equation (10) is described in Algorithm 1.

Algorithm 1: Solving Equation (10) by Using the Gauss-Newton Algorithm

```
Input: Orientation data $\mathbf{R}_{\mathrm{wB}}^{\mathrm{i}}$ and $\mathbf{R}_{\mathrm{wB}}^{\mathrm{j}}$ from Visual ORB-SLAM2, and rotation changes
$\Delta\bar{\mathbf{R}}_{ij}$ from IMU pre-integration in (5) ;
Output: gyroscope bias estimation between two consecutive keyframes i, j;
Process:
    solving the normal equation:
        $\mathbf{H}\delta\mathbf{b}_{\mathbf{g}}^{*} = -\mathbf{F}$
    where $\mathbf{F} := \frac{h(\delta\mathbf{b}_{\mathbf{g}}^{*})}{\delta\mathbf{x}}$, and $\mathbf{H} = \mathbf{F}^{\mathbf{T}}\mathbf{F}$. Hessian of the cost function ;
    Update: $\delta\mathbf{b}_{\mathbf{g}}^{*} \leftarrow \delta\mathbf{b}_{\mathbf{g}}$ ;
```

3.4.2 Velocity, Gravity Vector, and Metric Scale Initialization

Once the gyroscope bias is estimated, then, we should initialize velocity, gravity vector, and metric scale. To this end, the state vector for optimization can be defined

$$\boldsymbol{X}_{\mathbf{v},\ s,\ \mathbf{g}_0} = \left[{}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{0}\ {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{1}\ \ldots.\ {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{n}\ {}_{\mathrm{W}}\mathbf{g}\ s\right]^{T}, \tag{11}$$

where ${}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i}$, $i = 1,..,n,$ is the velocity in body frame while taking the *k*th image, with $i, j \in k$ image, ${}_{\mathrm{W}}\mathbf{g}$ indicates gravity vector in the camera frame, and s denotes global scale. Note that, if the acceleration bias is incorporated in (11), the chance of having ill-conditioned system will increase since the gravity and accelerometer bias are hard to be distinguished. For this reason, in this stage, the gravity is estimated without consideration of acceleration bias.

Due to the small rotation in initialization, separation of accelerometer bias from gravity is very challenging. Therefore, we estimate the gravity and global scale without bias consideration. Then, by using ${}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}}^{j}$ from (8), applying zero bias accelerometer and substituting ${}_{\mathrm{W}}\mathbf{p}_{\mathrm{B}} = s\,{}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}} + \mathbf{R}_{\mathrm{WC}}\,{}_{\mathrm{C}}\mathbf{p}_{\mathrm{B}}$ as the inclusion of scale s when transforming the camera frame C to body frame B, we can obtain the equation of two consecutive keyframes *i, j* in the window as

$$s\,{}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{i+1} = s\,{}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{i} + {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i}\Delta \mathrm{t}_{i,j} + \frac{1}{2}\,{}_{\mathrm{W}}\mathbf{g}\Delta t_{i,j}^{2} + \mathbf{R}_{\mathrm{WB}}^{\mathrm{i}}\Delta\bar{\mathbf{p}}_{ij} + \left(\mathbf{R}_{\mathrm{WC}}^{\mathrm{i}} - \mathbf{R}_{\mathrm{WC}}^{\mathrm{j}}\right)\,{}_{\mathrm{C}}\mathbf{p}_{\mathrm{B}}\ , \tag{12}$$

where it can be denoted as

$$\hat{\mathbf{z}}_{i.j} = \mathbf{H}_{i.j}\boldsymbol{X}_{\mathbf{v}_i.s.\mathbf{g}_0}\ , \tag{13}$$

and

$$\hat{\mathbf{z}}_{i,j} = \left[-\mathbf{R}_{\mathrm{WB}}^{\mathrm{i}\ T}\Delta \mathrm{t}_{i,j}\quad \mathbf{R}_{\mathrm{WB}}^{\mathrm{i}\ T}\left({}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{j} - {}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{i}\right)\quad \frac{1}{2}\mathbf{R}_{\mathrm{WB}}^{\mathrm{i}}\Delta t_{i,j}^{2}\right]\begin{bmatrix} {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i} \\ s \\ {}_{\mathrm{W}}\mathbf{g} \end{bmatrix}, \tag{14}$$

The angular rate ${}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{(\cdot)}$ and $\mathbf{R}_{\mathrm{WB}}^{(\cdot)}$ are being supplied by ORB-SLAM2, and $\Delta \mathrm{t}_{i.j}$ is the time interval between two consecutive keyframes. Note that in (Xufu Mu, et al., 2018), firstly, the velocity estimation has not been considered in this step, and then, the gravity and scale are estimated between three consecutive keyframes, and finally, (14) is solved by Singular Value Decomposition (SVD). But, in this thesis, we take into account the velocity in this step and estimate it between two keyframes to improve the accuracy of $\boldsymbol{X}_{\mathbf{v}_i.s.\mathbf{g}_0}$, and then, solve it by the Gradient descent optimization method. The reason to use optimization is due to the fact that the SVD is slow and computationally expensive and requires care dealing with missing data. Conversely, since our problem is a convex optimization, the Gradient approach - an iterative optimization algorithm for finding the minimum of a function - is used which is faster and deals well with missing data. Also, SVD is used to perform PCA that aims to decompose a matrix (usually a set of observations) in order to find the directions in which the observations have the largest variance.

Therefore, we form (14) to a least-square problem, and the optimization problem can be described as

$$\boldsymbol{X}_{\mathbf{v}.s.\hat{\mathbf{g}}_0} = \underset{\boldsymbol{X}_{\mathbf{v}.s.\mathbf{g}_0}}{\operatorname{argmin}} \sum_{i.j \in K} \left\| \hat{\boldsymbol{z}}_{i.j} - \mathbf{H}_{i.j} \boldsymbol{X}_{\mathbf{v}_i.s.\mathbf{g}_0} \right\|^2, \tag{15}$$

By solving the least square problem (15), we can estimate the rough gravity, scale and velocity. However, it is noticeable that from output of (15), we only deal with the scale. Also, note that in the above formula, the accelerometer bias is not considered during the computation of scale, gravity, and velocity (as it cannot be observed in (14)). This is done in order to increase the chance of having an ill-conditioned system since the gravity and accelerometer bias are hard to be distinguished (C. Campos, et al., 2019). Therefore, at the next step, it would be explained how the gravity magnitude is considered in our system.

Algorithm 2 shows the solution process of equation (15).

Algorithm 2: Solving Equation (15) by Using the Gradient Decent Algorithm

```
**Input:** ${}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{(\cdot)}$ and $\mathbf{R}_{\mathrm{WB}}^{(\cdot)}$ that are supplied by visual ORB-SLAM2, $\Delta \mathrm{t}_{i.j}$ is time interval between two consecutive keyframes;
**Output:** velocity ${}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i}$, gravity vector ${}_{\mathrm{W}}\mathbf{g}$ and scale $s$;
**Process:**
    solving the linear least square problem:
        $\boldsymbol{X}_{\mathbf{v}.s.\hat{\mathbf{g}}_0} = \boldsymbol{X}_{\mathbf{v}.s.\mathbf{g}_0} - \alpha \nabla f(\boldsymbol{X}_{\mathbf{v}.s.\mathbf{g}_0})$
    where $\alpha$ and $\nabla$ denote learning rate and gradient at current position, respectively;
    Update: $\boldsymbol{X}_{\mathbf{v}.s.\hat{\mathbf{g}}_0} \leftarrow \boldsymbol{X}_{\mathbf{v}.s.\mathbf{g}_0}$ ;
```

After obtaining the rough scale, we need to acquire the final gravity $\hat{\mathbf{g}}$ without considering the accelerometer bias, which is rectified by considering gravity magnitude information $\mathbf{g}^T\mathbf{g} = 9{,}82\ ^2$ (Y. Liu, et al., 2017), where $\mathbf{g} = [0 \quad 0 \quad G]^T$, $G = 9{,}82$.

In general, there are two common ways to consider gravity magnitude in (15): i) to model the gravity, substituting in ${}_{\mathrm{W}}\mathbf{g}$ in (9) and it is re-optimized the (15) as done in (Y. Liu, et al., 2017, W. Huang, et al., 2018), ii) to directly add the gravity magnitude as a constraint (J. Kaiser, et al., 2017, A. Martinelli, 2014).

To rectify the states in the initialization section of this thesis, the second way which is to directly add the gravity magnitude to optimization problem has been selected, in which two adjustable parameters L and P are employed into the equation. As a part of this process, a Closed-Form (CF) method is employed that is provided by (J. Kaiser, et al., 2017, A. Martinelli, 2014). In fact, we used the basic equations that characterized the closed-form solution to obtain local gravity and velocity. To perform this strategy, a short interval of time should be determined that in our case by using trial and error, it is chosen 1 seconds. The reason to use of this value of interval time is that we realized that in our case, and by trial and error, for any short time less than 1 second, local convexity of the states leads failing.

At any short interval, the camera would observe *N* point-features in which the camera displays an image of these points. In this way, the times of this interval are denoted by $\boldsymbol{t_1}, \boldsymbol{t_2}, \dots, \boldsymbol{t_n}$,and the following equation is derived from (A. Martinelli, 2014):

$$\boldsymbol{S_j} = \boldsymbol{\lambda_1^i \mu_1^i} - \boldsymbol{V t_j} - \boldsymbol{G}\frac{\boldsymbol{t_j^2}}{\boldsymbol{2}} - \boldsymbol{\lambda_j^i \mu_j^i}\,, \tag{16}$$

where vectors $\boldsymbol{S_j}$ denotes the camera-IMU integration in the interval $[\boldsymbol{t_1}, \boldsymbol{t_j}]$ that is determined by accelerometer and gyroscope. Also, vector $\boldsymbol{\lambda_1^i}$ represents the distance to the point features *i* at time $\boldsymbol{t_j}$. Besides, vectors $\boldsymbol{\mu_j^i}$ denotes fully determined by visual and gyroscope measurements in the interval $[\boldsymbol{t_1}, \boldsymbol{t_j}]$ that are required to express the bearing at time $\boldsymbol{t_j}$ in the frame at time $\boldsymbol{t_1}$.

Given that the scale is not considered in (16), we kept rough scale obtained from (15), and continue the process from (16) onwards. Also, equation (16) provides three scalar equations for each point feature $i = 1, \dots, N$, and it can be written in the following compact form:

$$\Xi \boldsymbol{X} = \boldsymbol{S}\,, \tag{17}$$

where for simplicity, $\boldsymbol{X}$ is the same $\boldsymbol{X}_{\mathbf{v},\hat{\mathbf{g}},\lambda} = \left[\hat{\mathbf{g}}^{\boldsymbol{T}}, \mathbf{v}^{\boldsymbol{T}}, \boldsymbol{\lambda_1^1}, \dots, \boldsymbol{\lambda_1^N}, \dots, \boldsymbol{\lambda_n^1}, \dots, \boldsymbol{\lambda_n^N}\right]^{\boldsymbol{T}}$, vector $\boldsymbol{S}$ is fully determined by the measurements, and Ξ is provided by (A. Martinelli, 2014). Then by making an optimization problem, equation (17) is described as the follows

$$\boldsymbol{X} = \underset{X_{\mathrm{v},\mathbf{g_0},\lambda}}{\mathbf{argmin}} \textstyle\sum_{i,j\in K} \left\|\Xi \boldsymbol{X}_{\mathrm{v},\mathbf{g_0},\lambda} - \boldsymbol{S}\right\|^2 \tag{18}$$

Furthermore, as an important part, the gravitational magnitude $\mathbf{g}$ is considered as an extra constraint $\mathbf{g}^{\boldsymbol{T}}\mathbf{g}$ that it can be expressed in matrix form

$$\boldsymbol{L}|\boldsymbol{\Gamma X}|^2 = \mathbf{g}^{\boldsymbol{T}}\mathbf{g}\ , \tag{19}$$

where $\boldsymbol{\Gamma} \equiv [\boldsymbol{I_3}, \boldsymbol{0_3}, \dots, \boldsymbol{0_3}]$.

Finally, we have an optimization problem as follows

$$\begin{cases} \boldsymbol{X} = \underset{X_{\mathrm{v},\mathbf{g_0},\lambda}}{\mathbf{argmin}} \textstyle\sum_{i,j\in K} \boldsymbol{P}\left\|\Xi \boldsymbol{X}_{\mathrm{v},\mathbf{g_0},\lambda} - \boldsymbol{S}\right\|^2 \\ \qquad \mathbf{s.t}\text{: } \boldsymbol{L}|\boldsymbol{\Gamma X}|^2 = \mathbf{g}^{\boldsymbol{T}}\mathbf{g} \end{cases}, \tag{20}$$

where P and L are the adjustable parameters considered as the weights for objective and constraint terms, respectively. Also, to solve this equation, it has been converted into problem form that is solvable for Fmincon (*https://uk.mathworks.com/help/optim/ug/fmincon.html*) being a nonlinear package programming solver to find a minimum of a constrained nonlinear multivariable function, and it turns out that the Interior Point method would find the nearest results to minimum values.

It is noticeable that to the best of our knowledge, it is the first time the formulation above is represented in visual-inertial fusion, where P and L have been considered as adjustable parameters for each term. Indeed, by this strategy, we can to do a trial-and-error method to attain the best value of $\boldsymbol{X}_{\mathbf{v},\hat{\mathbf{g}},\boldsymbol{\lambda}}$.

Algorithm 3 describes the approach used to solve (20) by Interior Point method.

Algorithm 3: Solving Equation (20) by Using the Fmincon Algorithm

```
Input: vector $\boldsymbol{S}$ is fully determined by the measurements of accelerometer and
Gyroscope in the interval $[t_1 . t_j]$, and $\Xi$ is provided by (A. Martinelli, 2014).
Output: the rectified gravity vector $\hat{\mathbf{g}}$, velocity $\boldsymbol{v}$ at time $j$, and $\lambda_n^N$ distance to the
keyframe $i$ at time $t_j$;
Process:
    solving the constrained least square problem using fmincon solver
        $min\, f(\boldsymbol{X}_{\mathbf{v}..\mathbf{g}.\boldsymbol{\lambda}})$
        s.t: $|\boldsymbol{\Gamma}\boldsymbol{X}_{\mathbf{v}..\mathbf{g}.\boldsymbol{\lambda}}|^2 = \mathbf{g}^T\mathbf{g}$
    where $\boldsymbol{\Gamma} \equiv [\boldsymbol{I}_3 . \mathbf{0}_3 . \dots . \mathbf{0}_3]$;
Update: $\boldsymbol{X}_{\mathbf{v},,\hat{\mathbf{g}},\lambda} \leftarrow \boldsymbol{X}_{\mathbf{v},,\mathbf{g},\lambda}$ ;
```

3.4.3 Accelerometer Bias Estimation and Scale Refinement

To compute the accelerometer bias and scale refinement, we have to consider $\mathbf{g}_0$ that has already been obtained from (20) as a fixed vector. Assuming that the estimated scale has an approximation value, then, to rectify the scale, the accelerometer bias and scale can be estimated together. The considered variables to be estimated are

$$\boldsymbol{X}_{s,\mathbf{b}_a} = [s, \mathbf{b}_a]^T \ , \tag{21}$$

By adding $\mathbf{b}_a$ as the accelerometer bias to (14), and making a new optimization problem similar to (15), we have

$$\hat{\mathbf{z}}_{i,j} = \begin{bmatrix} -{\mathbf{R}_{\mathrm{WB}}^{\mathrm{i}}}^T \Delta \mathrm{t}_{i,j} & {\mathbf{R}_{\mathrm{WB}}^{\mathrm{i}}}^T ({}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{j} - {}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{i}) & -\mathbf{J}_{(.)}^{a} \end{bmatrix} \begin{bmatrix} {}_{\mathrm{W}}\mathbf{v}_{\mathrm{B}}^{i} \\ s \\ \mathbf{b}_a \end{bmatrix} \tag{22}$$

Since the velocities are estimated in Section 3.4.2, the updated $\hat{\mathbf{z}}_{i,j}$ would be as follows

$$\hat{\mathbf{z}}_{i,j} = \begin{bmatrix} \mathbf{I}_{3\times3}({\mathbf{R}_{\mathrm{WB}}^{\mathrm{i}}}^T ({}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{j} - {}_{\mathrm{W}}\mathbf{p}_{\mathrm{C}}^{i}) \Delta \mathrm{t}_{i,j}) & -\mathbf{J}_{(.)}^{a} \end{bmatrix} \begin{bmatrix} S \\ \mathbf{b}_a \end{bmatrix} \tag{23}$$

where optimization solution is used to find the $\boldsymbol{X}_{s.\mathbf{b}_a}$, the refined scale and the estimated bias are obtained.

3.4.4 Gyroscope Bias Refinement

In the final step, after initializing all the states, to increase the accuracy of initialization, we reinitialize gyroscope biases by using the updated values obtained in the initialization process and recompute (15) to improve the accuracy of initialization states as much as possible.

3.5 TERMINATION CRITERION

Before the end of the visual-inertial initialization process, this question may come up that that when the initialization should be terminated. Unfortunately, in many works, this problem and its solution have been ignored. However, in our strategy, the visual-inertial initialization process is automatically terminated when all the estimated states are convergent. For instance, one of the convergence indicators is to leverage the norm of the nominal gravity which is a constant 9.806 m/s^2. Once the termination criterion is performed, the initialization process will be automatically finished. Then, to feed the nonlinear tightly coupled visual-inertial SLAM system, the estimated initial state values can be used.

3.6 SUMMARY

In this chapter, to overcome the challenges in VSLAM and improve its accuracy in the indoor environment, a new states initialization estimation method is introduced to improve the position accuracy. Indeed, this improvement is applied in the initialization step which not to underestimate it, leads to having a low accuracy in the estimate of initial values following that the 3D map and its output position be unreliable. To this end, we presented a new technique in which, firstly, in the initialization step, the Gradient descent optimization method is used instead of SVD, and secondly, to estimate the velocity, gravity vector, and metric scale, a new formulation is presented in which the user can tune the parameter weights to attain the best performance in finding the best initialization values. In the next chapter, the proposed approach is implemented by two methods: benchmark and using a real IMU-camera.

4 PERFORMANCE EVALUATION

In this chapter, we detail the complete implementation of visual-inertial initialization to qualitatively and quantitatively evaluate the performance of the proposed algorithm. Overall, implementation can be divided into two sections: Benchmark and Experiment.

Based on the designed algorithm presented in the previous chapter, our algorithm requires to be regulated via two adjustable parameters L and P in the range of 0 and 1. This process is performed using trial and error until the best performance and minimum loss error are reached. Although this strategy might be a little time-consuming due to run and evaluate the performance many times, its results would be valuable since these parameters can be utilized in the categorized environment such as texture and texture-less based on the relations between L and P.

In benchmark, the EuRoC dataset (M. Burri, at al., 2016) is utilized. This dataset contains 11 sequences of 2 scenes including six "Vicon Room" and five industrial "Machine Hall" sequences, which was synchronously recorded from the firefly micro-aerial vehicle equipped with a global shutter WVGA camera at 2fps and an IMU at 200Hz at high flying speed in two different environments. The first environment is a 30 m^2 indoor room which contains objects including the chair, table, chess-board, and any others. Second environment is 300 m^2 in which industrial machines can be observed. Both datasets contain ground-truth positions measured by the Leica MS50 laser tracker and Vicon motion capture system, which are well-calibrated systems to be employed as the benchmark dataset to evaluate the efficiency of various state-of-the-art VO/VIO/SLAM approaches. Also, depending on illumination, texture, and motion dynamic, the dataset is categorized into three classes including easy, medium, and difficult.

It is necessary to mention that the EuRoC dataset does not provide an explicit ground-truth scale, so we require to compute the true scale according to the trajectory generated from visual ORB-SLAM2 and the ground-truth data. In fact, in this process, firstly, the initialization of the ORB-SLAM2 system is completed, and then an initial translation between two keyframes is generated. At this stage, the true translation on basis of corresponding ground truth states can be calculated. After this, the benchmark scale, which is the same true scale, can be considered as the ratio of the true translation to the initial translation.

Then, to corroborate the superiority of our algorithm, the proposed system has been compared with the state-of-the-art methods: Monocular Visual-Inertial System (VINS-MONO) (Y. Lin, at al., 2016), Monocular visual-inertial System with loop closure (VINS-Loop), VI ORB-SLAM (R. Mur-Artal, at al., 2017), and Multi-State Constrained Kalman Filter (MSCKF) (A.I. Mourikis, et al., 2007). In the following, some explanation is presented for each of these state-of-the-art.

a) VINS-MONO: a non-linear optimization-based sliding window estimator, tracking robust corner features.
b) VINS-Loop: the VINS-MONO with closure loop.
c) VI ORB-SLAM: a tightly coupled and nonlinear optimization approach that contains an ORB sparse front-end, graph optimization back-end, loop closure, and relocation. This system is able to close the loop and reuse the previously constructed 3D map.

We also evaluated the proposed algorithm from a point of hardware's view in which a different hardware configuration - embedded single-board hardware called Raspberry-Pi - is employed. This experiment is carried out to measure the performance of the proposed algorithm such as pose estimation accuracy and CPU usage while processing the EuRoC dataset.

Finally, we validate the performance of our algorithm by performing an indoor real-world experiment using the sensor of Intel RealSense ZR300 which contains an IMU and a global shutter camera. Both tests are performed on a laptop with Intel Core i7 3.0 GHz CPU and a 16GB RAM configuration, which has been equipped with Robot Operation System (ROS) – Kinetic version - and implemented in C++ and Python.

In this context, the objectives in benchmark are:

- To validate the efficient role of additional adjustable parameters to improve accuracy.
- To show the lower tendency of the proposed algorithm to accumulate error in long-term trajectory compared to state-of-the-art visual-inertial algorithms.
- Investigation of the proposed algorithm performance from a point of hardware's view such as CPU usage.
- Investigation of limitations of employing the Raspberry-Pi 3 in the benchmark.

Besides, the objectives of indoor real-world experiment are:

- Employing the adjustable parameters obtained in benchmark and applying them in real-world experiment.
- Reduction in the initialization time before capturing the ORB features.

4.1 BENCHMARK

4.1.1 Implementation

As presented in the previous chapter, our strategy is to adjust the parameter *L* and gravity coefficient *P,* so that the best performance and minimum error could be achieved. For this reason, by using trial and error, we can attain the appropriate parameters of the formula provided for three types of environments in the EuRoC dataset. Note that we opted for three different sequences as the representative of the EuRoC dataset based-off on their difficulty level (easy, medium, and difficult), and then, the parameters obtained in this step will apply to the entire sequences of the dataset. These parameters are constrained to the range between 0 and 1. Thus, by increasing and declining each of them, the best performance and accuracy can be achieved. To do this, we use

- Operating system of Ubuntu 16.0 LTS
- ROS – kinetic version[5]

In terms of the benchmark process and measuring the approach, it is noticeable that, first of all, the data is immediately recorded using ROSBAG command in ROS framework before running algorithm, and then after finishing the path traveled, this data is converted to a CSV file to exploit in error computation and visualization.

Table 2 shows the parameters used in our algorithm in the EuRoC dataset.

Table 2 - Adjustable parameters obtained in this study

Difficulty Level	*P*	*L*
Easy	0.852	0.987
Medium	0.813	0.935
Difficult	0.962	0.897

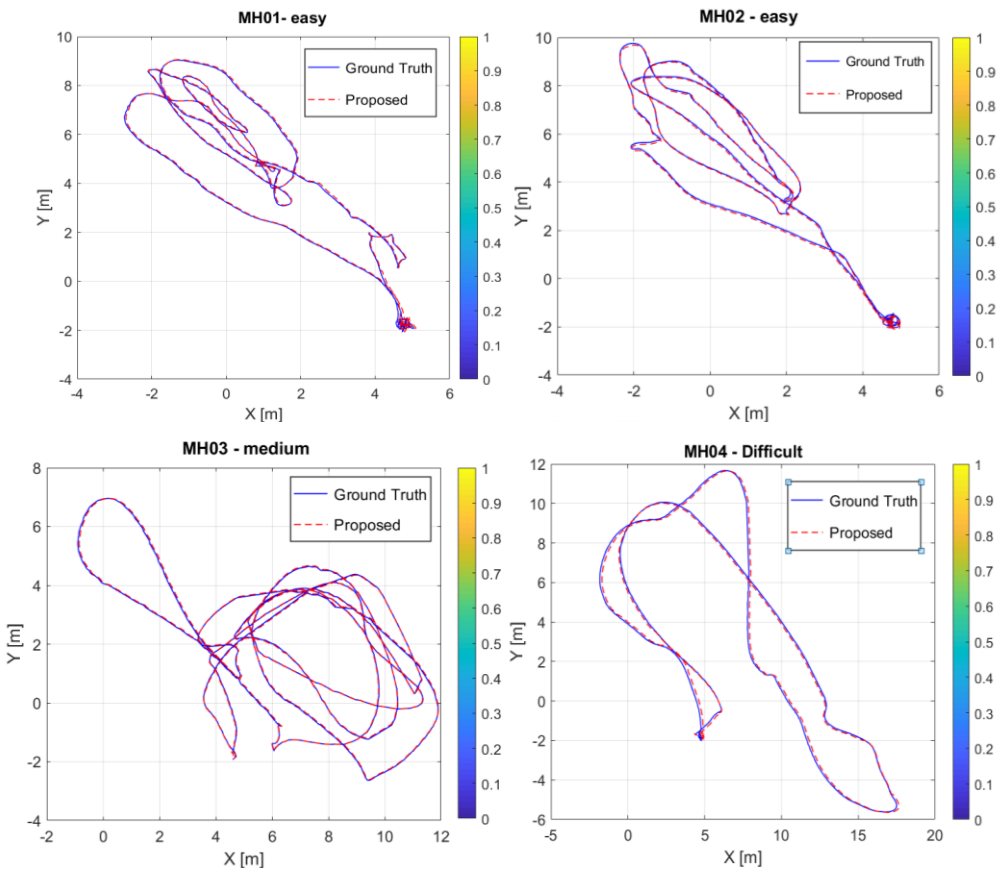
It is clear that, by increasing the difficulty of the environment from medium to difficult, the amount of *P*-value should increases. Simultaneously, by decreasing the difficulty level of the environment, the amount of *L* value should increase. By knowing the relation

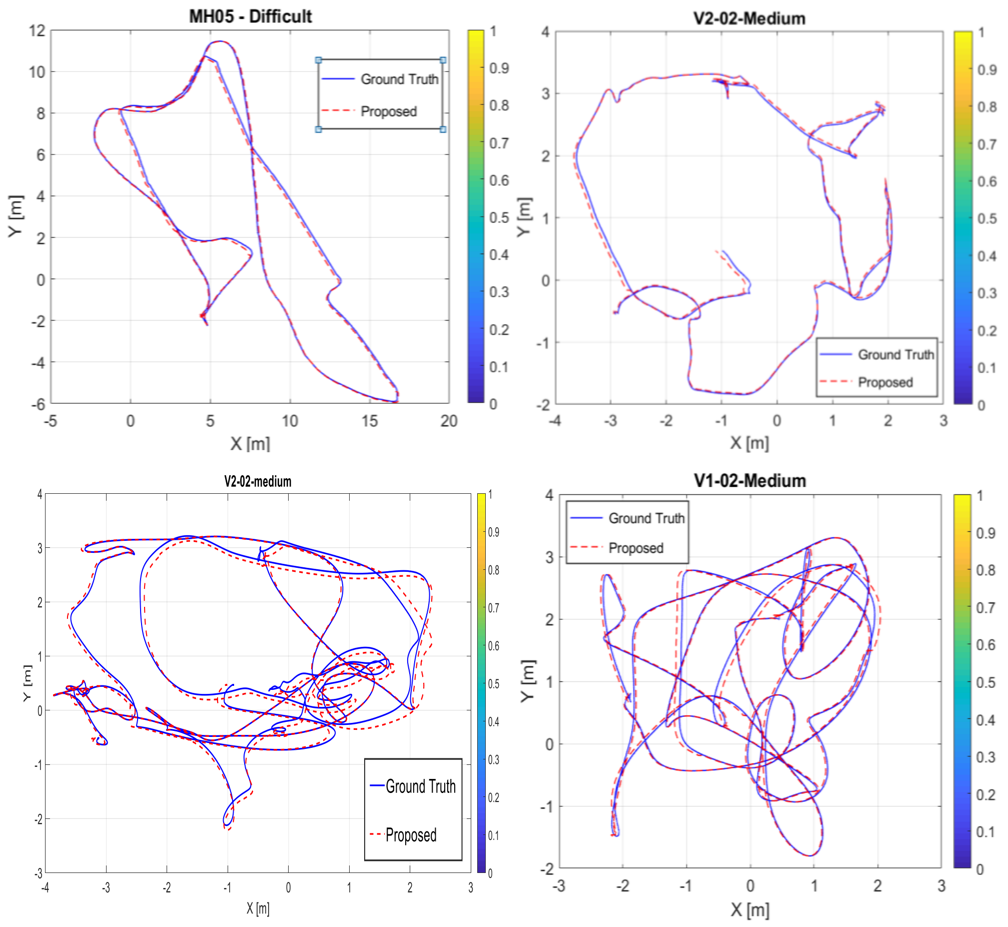
[5] http://wiki.ros.org/kinetic

between *L* and *P* in different environments in terms of texture, lightness, or any others, leads to reduce the drifting away from the ground-truth.

Now, we can apply our new formulation with additional parameters *L* and *P* to the EuRoC dataset and evaluate its accuracy. To this end, the accuracy of our algorithm, which has been evaluated in 11 sequences of the EuRoC dataset, shows that it has a successful performance compared to some sequences in other algorithms. The results of the use of selected parameters can be observed in Figure 12, in which the trajectory traveled by MAV, programmed via the proposed algorithm, has been compared to ground-truth. Note that in this figure, *MH* and *V* are the names of scenarios based on the difficulty of the environment.

Figure 12 - Trajectory of the proposed system in EuRoC dataset





Fortunately, in all the sequences except *V1-01-Difficult*, the path travelled by our algorithm followed the ground truth, implying the proposed system had an appropriate performance.

Also, to have a deeper insight in the obtained results, a numerical comparison table (Table 3) has been created, in which by using absolute translation root-mean-square error (RSME) criteria, the performance of the proposed method has been compared to VINS-MONO, VINS-Loop, and VI ORB-SLAM algorithms for all the 11 sequences.

We choose the *RSME* criterion, since it is a commonly-used index to assess the error between the predicted and observed values. To calculate the *RMSE*, the following equation is used:

$$RSME = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f_i - o_i)^2} \quad , \qquad (20)$$

where n is the number of samples, and f and o denote the predicted and observed values, respectively.

Note that the *RSME* is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are. In other words, *RMSE* is a measure of how to spread out these residuals are.

To a certain extent, one of the most critical problems in this study is to attempt to reduce *RSME* intensity by adjusting the parameters considered as weights in the proposed approach presented in Chapter 3.

In this table, scale error is calculated as $|s^* - \hat{s}|/|\hat{s}| \times 100\%$, where s^* denotes the optimum scale and $\hat{s}$ corresponds to the ground-truth scale.

Table 3 - Comparison of translation *RMSE* (m)

	Proposed		**VI ORBSLAM2**		**VINS-MONO**	**VINS-LOOP**
Sequences	Scale Error (%)	RSME (m)	Scale Error (%)	RSME (m)	RSME (m)	RSME (m)
V1-01 (easy)	1.32	0.065	0.9	0.027	0.088	0.081
V1-02 (medium)	1.16	0.088	0.8	0.028	0.068	0.042
V1-01 (difficult)	x	x	x	x	0.160	0.156
V2-01 (easy)	2.05	0.059	0.2	0.032	0.068	0.063
V2-02 (medium)	2.04	0.067	1.4	0.041	0.084	0.066
V2-03 (difficult)	2.1	**0.072**	0.7	0.074	0.159	0.157
MH-01 (easy)	0.64	**0.064**	0.5	0.075	0.301	0.098
MH-02 (easy)	0.81	**0.069**	0.8	0.084	0.249	0.152
MH-03 (medium)	1.25	0.090	1.5	0.087	0.173	0.080
MH-04 (difficult)	1.14	**0.091**	3.4	0.217	0.323	0.129
MH-05 (difficult)	0.8	0.078	0.5	0.082	0.257	0.077

It can explicitly be observed that the worse result is related to sequences *V1-02-Medium* because its *RSME* is higher than VINS-MONO, VINS-LOOP and VI ORB-SLAM over the alignment trajectory to the ground-truth pose via SE (3). Whereas in contrast, the proposed algorithm experienced the lower RSME in sequences *MH-01-Easy*, *MH-02-Easy*, *V2-03-Difficult* and *MH-04-Difficult* achieved acceptable results and outperforms the VINS-

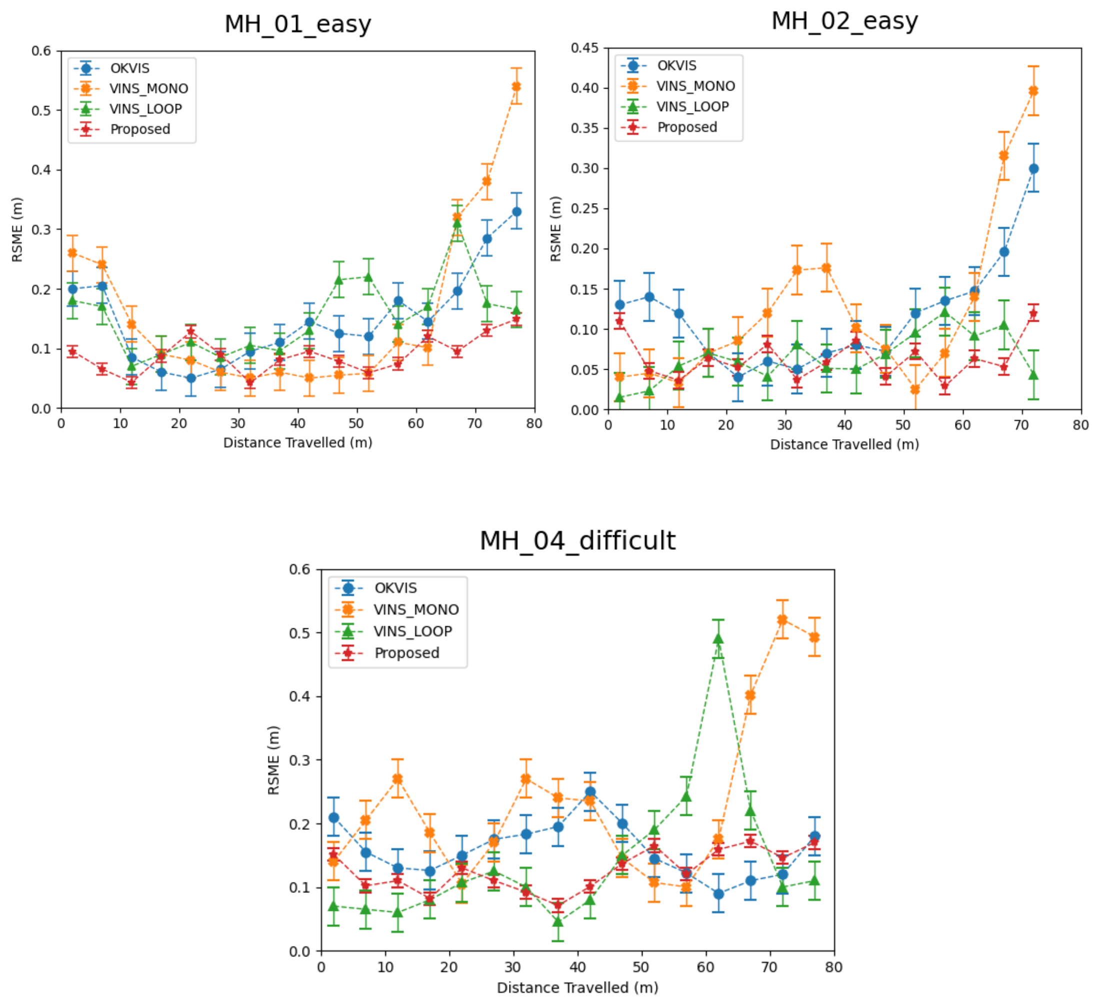
MONO, VINS-Loop, and VI ORB-SLAM. Note that in our comparison, no full BA results of VI ORB-SLAM are considered.

Although our algorithm obtained high accurate results in benchmark, it fails to perform the *V1-01-Difficult* sequence. This is because, in this sequence, the MAV has extreme movements at the beginning, resulting in initialization failure in the algorithm. About other data sequences, our visual-inertial SLAM algorithm can be executed in real-time without any lost-tracking. However, in comparison, OKVIS, which employed a stereo camera, can run without performing initialization leading the algorithm to handle this data sequence easily. However, in contrast, OKVIS obtained the lowest accuracy during to process *V2-03-Medium*.

On the other hand, VINS-MONO utilizes the sparse optical flow tracking as an independent front-end module to retrieve data association. In videos, optical flow can be considered as a robust approach for tracking features, leading to successful initialization and let the algorithm run all the data sequence in the EuRoC dataset.

In addition, in terms of consistency, the proposed algorithm is evaluated. In the obtained results, it is noticeable that our algorithm in some sequences has a lower tendency to error accumulation in long-term trajectory compared to other algorithms as it does not leverage the full Bundle Adjustment, because it increases the computational cost of the system. For instance, in the sequence MH-01-*easy*, and VIMS_MONO system, by increasing the trajectory, the accumulated error increases from about 3 m to 5.5 m in terms of Root Mean Square Error (*RSME*). Figure 13 shows the trajectory traveled by the proposed method compared with OKVIS, and VINS-MONO, VINS with loop closure (VINS-LOOP) in terms of consistency.

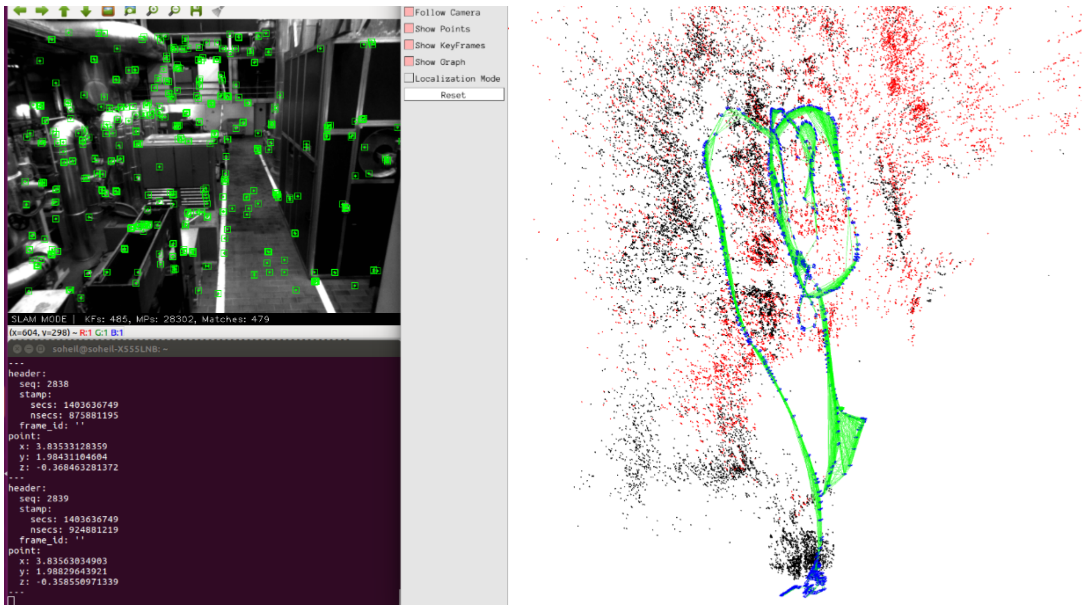
Figure 13 - Comparison of the proposed method versus OKVIS, VINS-MONO, and VINS with loop closure (VINS-LOOP)



Furthermore, compared with the method proposed as VI ORB-SLAM (R. Mur-Artal, et al., 2017), our method requires initialization time of 1 or 2 seconds instead of 15 seconds.Also, our algorithm uses less CPU time, and can successfully initialize in sequence *MH-05-Difficult*, where the previous method (S. J. Haddadi, et al., 2019) had high *RSME*.

Figure 14 illustrates the features and global map obtained by the proposed algorithm.

Figure 14 - Features and the map obtained by proposed algorithm.



```
SLAM MODE |  KFs: 485, MPs: 28302, Matches: 479
(x=604, y=298) ~ R:1 G:1 B:1
```

```
---
header:
  seq: 2838
  stamp:
    secs: 1403636749
    nsecs: 875881195
  frame_id: ''
point:
  x: 3.83533128359
  y: 1.98431104604
  z: -0.368463281372
---
header:
  seq: 2839
  stamp:
    secs: 1403636749
    nsecs: 924881219
  frame_id: ''
point:
  x: 3.83563034903
  y: 1.98829643921
  z: -0.358550971339
---
```

4.1.2 CPU Usage Measurement In Benchmark

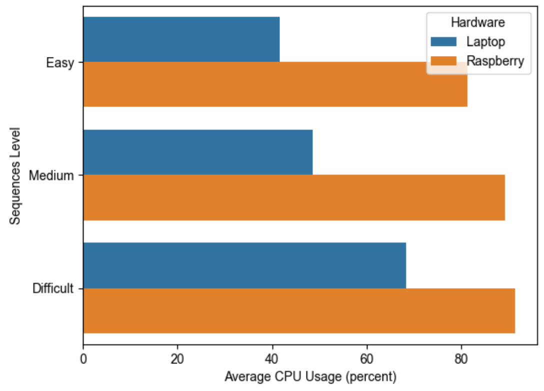
From a hardware perspective, it is crucial how the *Central Processing Unit* (CPU) is used during the visual-inertial SLAM process. Suppose that this algorithm tends to be employed for indoor positioning of a Quadrotor. In this case, single-board computers come up.

In this section, in point of hardware's view, the CPU used in this benchmark is evaluated via two types of computer: Raspberry-Pi single-board and Laptop. The Raspberry-Pi is the right choice since it is much cheaper than the existing commercial single-board processors. Also, relative to its price, in terms of power, Raspberry-Pi is appropriate to execute processor-heavy tasks such as image processing and sensor fusion. The Raspberry-Pi 3 Model B single-board used in this study, has the following configuration:

- 1.4GHz 64-bit quad-core CPU
- Dual-band wireless LAN
- Bluetooth 4.2/BLE
- Power-over-Ethernet support (with separate PoE HAT)
- 1 GB SDRAM
- Video Core 250-400MHZ GPU

In this way, the scenarios are categorized into three levels: easy, medium, and difficult. The reason for this practice is to assess the CPU usage while the difficulty of the environment is increasing and the rooms are darker and more unrecognizable. Since in the dataset, there are four easy environments, three medium and four difficult scenarios, we decided to compute an average percentage of CPU usage for each category, which is illustrated in Figure 15.

Figure 15 - CPU usage during executing the proposed algorithm for each category in EuRoC dataset



To monitor the CPU usage, ***htop*** tool is used which is an interactive process viewer, and allows user to scroll vertically and horizontally so that all the processes running on the system, along with their full command lines could be observable. As well as this, users can view them as a process tree, selecting multiple processes, and acting on them all at once. Also, ***htop*** can be used to determine percentage CPU usage along with memory and swap usage statistics.

Table 4 - CPU usage Comparison between Laptop and Raspberry-pi

Sequences Level	Hardware	Average CPU Usage (percent)
Easy	Laptop	41.57
Medium	Laptop	52.69
Difficult	Laptop	68.45
Easy	Raspberry-Pi	81.39
Medium	Raspberry-Pi	89.24
Difficult	Raspberry-Pi	94.47

Also, Table 4 shows the percent of average CPU usage in each class and computer. From this table, it can be noticed that as much as the difficulty of environments increases, the

CPU used to process the algorithm raises. Also, it shows the Raspberry-pi is attempting to use a major amount of its capacity to handle the computing process.

From Table 4, it can be concluded that since the CPU capacity of Raspberry-Pi is lower than Laptop, hence, its capacity is quickly filled which means this single-board is sometimes prone to be overloaded and fail. To overcome this problem, the suggestion is to employ the Raspberry-Pi 4 which includes a faster CPU (1.5 GHz), GPU (500 MHZ), and RAM (4GB). We think that in a newer version of Raspberry-Pi, the CPU, GPU, and memory speeds are higher which means it could be a robust system to perform projects that need more processing power and are challenging for Raspberry-Pi 3. However, due to the inaccessibility of this computer in this study, the experiment has been executed via Raspberry-Pi 3.

4.1.3 Benchmark Conclusion

In this section, it is observed that the use of additional adjustable parameters *L* and *P* can play a relatively efficient role in some scenarios based on the difficulty level of the environment. Also, the proposed algorithm showed that has a lower tendency to error accumulation in long-term trajectory compared to state-of-the-art visual-inertial algorithms. Although this lower error accumulation has not been observed in all the sequences, it could be an initial way for further study and improvement of it. Besides, the proposed algorithm in the benchmark implementation process is investigated from a hardware perspective. The CPU amount used during the benchmark is measured by two types of computers: Laptop and a hardware called Raspberry-Pi 3 single-board. The results showed that the Raspberry-Pi 3 is sometimes prone to be overloaded, stopping in feature capturing and failure due to its lower CPU capacity.

4.2 INDOOR REAL-WORLD EXPERIMENT

The second experiment is to perform the proposed visual-inertial SLAM algorithm in the real indoor environment (with random texture) using a hand-held monocular-inertial RealSense ZR300, which is a ready-to-market sensor that provides 20-HZ image and 200-HZ IMU measurements.

Based on hardware components, the most important points about this sensor is that it consists of one fisheye camera with a lens's angular Field of View (FoV) of 133° and 100° in the horizontal and vertical directions, respectively, and streams a 640×480 image at 60 frames per second. Also, there is an onboard IMU which provides 3-axis accelerations and angular velocities in a body frame with timestamps at 20 kHz. It is noticeable that in this experiment, the depth measurement has not been used, and the two IR cameras, the projector, and the RGB camera are disabled for reduced-power operation.

4.2.1 Camera-IMU time-synchronization

Since there are different clock sources in ZR300 for the IMU and image timestamps, direct use of the timestamps from the RealSense library leads to poor estimator performance. To tackle this challenge, the used strategy generates a synchronization message every time the sensor captures an image, including the timestamp and a sequence number of the corresponding image. Then, two ring buffers (for images and synchronization messages) are used to look up the correct timestamp of the image concerning the IMU before publishing it over ROS. Different sampling rates of IMU is another important issue that should be considered in synchronization. Suppose that the IMU works on different sampling rates of gyroscopes (~200 Hz) and accelerometers (~250 Hz). Therefore, it is obvious that poor state estimation would be accrued. To mitigate this problem, an IMU message is published containing both sensors at the rate of the gyroscopes, and by buffering the messages, linearly incorporates the accelerometer messages. This process is performed using a technique provided by (J. Rehder, at al., 2016).

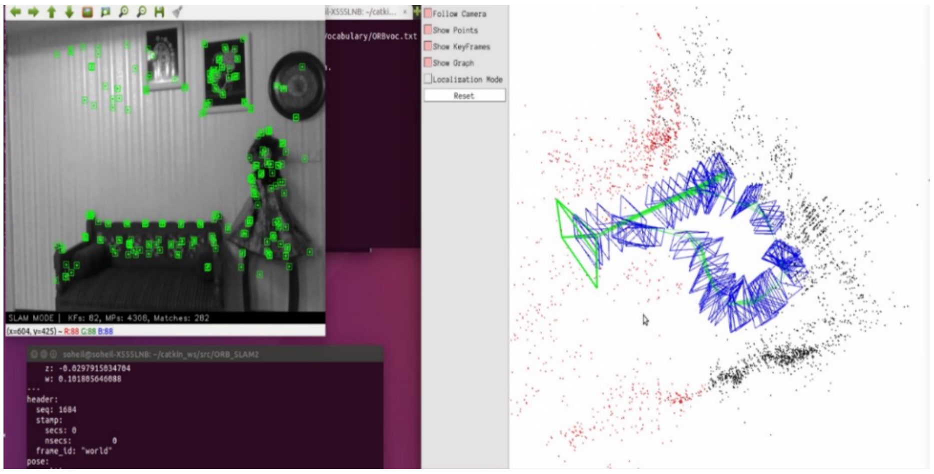
In summary, when two sensors are running at different sampling rates with their time sources, the faster update rate is used as the master time (I. Sa, at al., 2018).

Also, about estimating the camera exposure time, it is noticeable to mention that a constant offset between the IMU and camera is employed using Kalibr (P. Furgale, at al., 2013).

Looking back to the experiment, since the knowledge about the difficulty level of the environment is extensively decisive, the selected indoor environment has a size of 4 by 4 m and it is considered as a medium in terms of having texture. For this reason, we chose *L* and *P* based on Table 4, such that *P* and *L* are 0.813 and 0.913, respectively.

To achieve the best performance, the intrinsic parameters of the camera, including focal length and principal points are calibrated as: $f_u = 535{,}4\,.\,f_v = 539{,}2$ and $c_u = 320{,}1$, $c_v = 247{,}6$ respectively.

Figure 16 - A screen shot of the proposed visual-inertial ORB-SLAM algorithm during experiment.



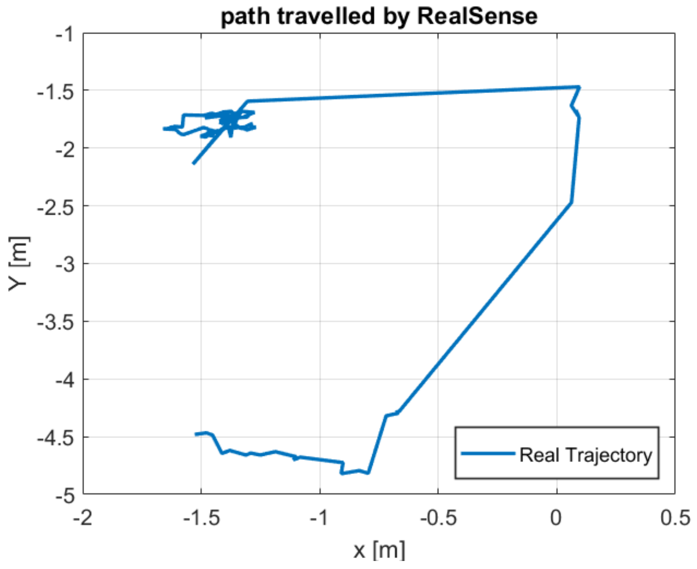
```
SLAM MODE |  KFs: 82, MPs: 4308, Matches: 282
(x=604, y=425) ~ R:88 G:88 B:88
```

```
    z: -0.0297915034704
    w: 0.101805646088
---
header:
  seq: 1684
  stamp:
    secs: 0
    nsecs:         0
  frame_id: "world"
pose:
```

Figure 16 depicts a screenshot of features taken by visual-inertial ORB-SLAM and the trajectory traveled by the monocular-inertial sensor during the experiment. Even though the environment lacks the chessboard or any special visual tag, the features are well-distributed in the current image.

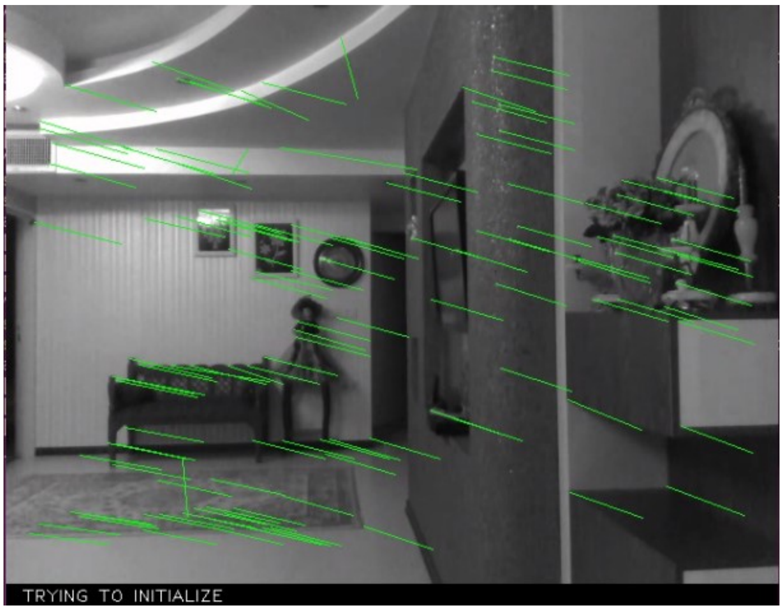
In our implementation procedure, we kept the sensor in our hands, and after initialization and finding the features by the sensor, we started to make a square trajectory ($16m^2$) by walking at a normal pace. In Figure 21, which depicts the trajectory traveled by the monocular-inertial sensor, it is explicitly observed that there are no significant drifts occurred when we move around the room. The total amount of path traveled is 16 m, which is an appropriate distance to perform this test.

Figure 17 - Path travelled using IMU-Camera in real indoor environment.



Another positive achievement in this experiment is to decrease the initialization time. Before this, we had to wait at least 10 seconds to initialize the system and obtaining ORB features. However, with this new algorithm, we need just around 3 seconds to obtain ORB features after the beginning of the video capturing. Figure 18 shows the initialization step during this experiment in the real-world some seconds before obtaining ORB features. As it can been observed in this Figure, while green lines are appeared, it means that the algorithm is trying to initialize the states.

Figure 18 - Initialization step during this experiment in real world.



4.2.2 Real-World Experiment Conclusion

In this section, the process of the indoor real-world experiment test is explained. One of the main objectives of this section was to employ the adjustable parameters obtained in the benchmark and apply them in a real-world experiment. Fortunately, this practice is successfully performed without failure and losing the features during the experiment. Another achievement in this experiment is to show the reduced initialization time to 3 seconds before feature capturing, which compared to the initialization method proposed in state-of-the-art VO/VIO/SLAM, such as (R. Mur, et al., 2017) is an acceptable initialization time. Indeed, in contrast to the original ORB-SLAM, which its initialization time is around 10 seconds, our proposed algorithm takes a lower time to initialize the states.

5 CONCLUSION

In this study, the evaluation of the proposed visual-inertial SLAM with a new formulation is presented. As it is mentioned at the beginning of this study, the main objectives of this thesis are: i) To accurately initialize the states in the initialization step of the visual-inertial SLAM, ii) To reduce the tendency of the proposed algorithm to error accumulation in long-term trajectory, iii) Investigation of the performance of the proposed algorithm from a point of hardware's view, such as CPU usage, iv) To evalute the proposed algorithm using a benchmark and in a real indoor environment.

To attain these objectives, a new formulation is proposed in framework of a visual-inertial SLAM system, in which a camera and IMU are employed to localize the robot in a GPS-denied environment. To this end, this study explains how to attain this achievement step by step.

Chapter 1 describes how robots can autonomously operate in outdoor/indoor environments. The major section of this chapter is dedicated to Simultaneous Localization and Mapping (SLAM) and its structure as one of the most popular positioning methods in the robotic system. In fact, this chapter is an introduction to identify the SLAM elements and entering the world of positioning using localization and mapping.

Then, in chapter 2, the visual SLAM, which contains how to take the advantages of the camera is described. The goal of this chapter is to show the usage and advantage of a single camera as the main sensor to generate the visual information input of the visual SLAM system. To better understanding, visual SLAM, some of the most important visual SLAM (VSLAM) approaches, and their pros and cons are reviewed. In the following of this chapter, it is concluded that the ORB-SLAM2 could be the most reliable and complete solution for monocular visual SLAM, and is the most representative state-of-the-art visual SLAM.

Every visual SLAM system has some challenges, leading to inaccurate positioning in an indoor environment. Chapter 3 explains how to leverage the fusion of camera and IMU in a SLAM system under the title visual- inertial SLAM (VI-SLAM) to reduce the inaccuracy or tackle the challenges in a V-SLAM system. In this chapter, the importance of state initialization accuracy is introduced as one of the most challenging steps, heavily affecting the performance of the system. To the best of our knowledge, this chapter presents a new formulation, in which the user can regulate the adjustable weights to attain the best performance in finding the best initialization values.

To assess the performance of our presented algorithm, the best values for adjustable parameters are obtained based on the difficulty level of the indoor environment. In other words, by using this strategy, the user can use the obtained adjustable parameters in every environment based on being texture or textureless. To this end, two types of test, including a benchmark and experimental, are executed. In the benchmark test, the proposed algorithm is applied to the EuRoC dataset. Fortunately, the difficulty level of this dataset has already been determined based on the existence of objects, light, being texture or textureless. By employing the right parameters L and P, in some scenarios, we could attain satisfactory results compared to state-of-the-arts visual-inertial Odometery and SLAM. Since the P parameter is the coefficient of gravity term, it can be concluded that the value of gravity plays a key role in improving the state estimation accuracy based on the difficulty of the environment. Also, from point of hardware's view, while the proposed algorithm is being executed, the maximum CPU usage in each sequence is measured on a Raspberry-Pi single-board and a Laptop. Results demonstrated that the Raspberry-Pi 3 - because of a worse hardware configuration - is under more pressure in terms of CPU usage. Indeed, the CPU capacity of Raspberry-Pi is lower than Laptop, hence, its capacity is quickly filled, which means this single-board is sometimes prone to be overloaded and fail. Therefore, Raspberry-Pi 4 is suggested to overcome this problem, which includes faster CPU, GPU, and RAM. Besides, to illustrate the goodness of our algorithm in the real world, an experimental test is performed. In this way, a monocular-inertial RealSense ZR300 sensor is employed. The outcomes were satisfactory, so that the initialization time was very short and the proposed algorithm could quickly obtain the ORB features. Also, for the betterment of accuracy in the real-world experiment, a good suggestion is to find a limited range for adjustable parameters to be adapted in every environment in terms of having or not the texture.

Unfortunately, due to the existence of the Covid-19 Virus, it was not possible to fulfill the experimental test at the laboratory. To this end, we carried out this test in a residential indoor room with a dimension of 5×5 meter. As well as this, since the Vicon cameras to measure the ground truth was not available, the comparison with ground truth measurement was impossible, and we were content with just our RealSense ZR300 sensor.

Our suggestion for future work is to find an appropriate range of adjustable parameters to use in the proposed algorithm in any environment. This strategy will facilitate the usage of this algorithm in the positioning of any indoor environment including dark/light, messy, and stairs (using a down camera). Also, it is possible to combine the proposed algorithm with Artificial Intelligence (AI) approaches, such as machine learning and deep learning, so that the positions obtained by visual-inertial SLAM could be trained using artificial neural networks, leading to predict the position in an unseen indoor environment.

Acknowledgment

This experiment is performed under the financial coverage of Coordination for the Improvement of Higher Education Personal (CAPES) including providing the registration fee and participating in the Brazilian Symposium on Robotics (SBR) symposium and purchasing some hardware such as battery and camera.

REFERENCES

P. Puerta, “On-board control algorithms for autonomous indoors navigation of Multirotor Micro Air Vehicles”. A Final Master Project submitted for the degree of Master in Robotics and Automation, 2012.

Y. Lu, Z. Xue, G. S. Xia, L. Zhang, “A survey on vision-based UAV navigation.” Taylor & Francis Group, ISSN: 1009-5020 (Print) 1993-5153 (Online) Journal, Published online: 12 Jan 2018.

Cadena, C. Carlone, L. Carrillo, H. Latif, Y. Scaramuzza, D. Neira, J. Reid, I. Leonard J.J. “Past, Present, and Future of Simultaneous Localization and Mapping: Towards the Robust-Perception Age”. IEEE Trans. Robot. 2016, 32, 1309–1332.

Yi Liu , Zhong Chen, Wenjuan Zheng, Hao Wang and Jianguo Liu, “Monocular Visual-Inertial SLAM: Continuous Preintegration and Reliable Initialization”, *Sensors* 2017, *17*, 2613; doi:10.3390/s17112613.

E. Kruppa. “Zur Ermittlung eines Objekts aus zwei Perspektiven mit innerer Orientierung”. In: *Sitzungsberichte der math.-naturw. Kl. der kaiserlichen Akademie der Wissenschaften, Abt. IIa* 122 (1913), pp. 1939–1948.

D.C. Brown. “The bundle adjustment – progress and prospects”. In: *International Archives Photogrammetry* 21.3 (1976), pp. 1–1.

M.A. Fostner ¨ and E. Gulch ¨ , “A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centers of Circular Features”. In: *Proceedings of the ISPRS Intercommission Workshop*. 1987

C. Harris and M. Stephens. “A combined corner and edge detector”. In: *Proceedings of the Fourth Alvey Vision Conference*. 1988, pp. 147–151.

E. Rosten, R. Porter, and T. Drummond. “FASTER and better: A machine learning approach to corner detection”. In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32.1 (2010), pp. 105–119,

C. Tomasi and T. Kanade. “Detection and Tracking of Point Features”. Tech. rep. Carnegie Mellon University, 1991.

H. Bay, T. Tuytelaars, and L.V. Gool. “SURF: Speeded up robust features”. In: *European Conference on Computer Vision (ECCV)*. 2006.

D.G. Lowe. “Object Recognition from Local Scale-Invariant Features”. In: *International Conference on Computer Vision (ICCV)*. 1999.

E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. “ORB: an efficient alternative to SIFT or SURF”. In: *International Conference on Computer Vision (ICCV)*. 2011.

H. Jin, P. Favaro, and S. Soatto. “Real-time 3-d motion and structure of point features: Front-end system for vision-based control and interaction”. In: *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2000.

A.J. Davison, I. Reid, N. Molton, and O. Stasse. “MonoSLAM: Realtime single camera SLAM”. In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 29.6 (2007), pp. 1052–1067.

L. Matthies, R. Szeliski, and T. Kanade, “Incremental estimation of dense depth maps from image Image Sequences”. In: *International Conference on Computer Vision and Pattern Recognition (CVPR)*. 1988

H. Jin, P. Favaro, and S. Soatto, “A semi-direct approach to structure from motion”. In: *The Visual Computer* 19.6 (2003), pp. 377–394

K.J. Hanna. “Direct multi-resolution estimation of ego-motion and structure from motion”. In: *Proceedings of the IEEE Workshop on Visual Motion*. 1991.

Jakob-Julian Engel, “Large-Scale Direct SLAM and 3D Reconstruction in Real-Time”, PhD thesis, 2016.

H. Jin, P. Favaro, and S. Soatto, “Real-time 3-d motion and structure of point features: Front-end system for vision-based control and interaction”. In: *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2000.

A.J. Davison, I. Reid, N. Molton, and O. Stasse, “MonoSLAM: Realtime single camera SLAM”. In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 29.6 (2007), pp. 1052–1067.

K.J. Hanna. “Direct multi-resolution estimation of ego-motion and structure from motion”. In: *Proceedings of the IEEE Workshop on Visual Motion*. 1991

H. Jin, P. Favaro, and S. Soatto, “A semi-direct approach to structure from motion”. In: *The Visual Computer* 19.6 (2003), pp. 377–394.

R. C. Smith and P. Cheeseman, “On the representation and estimation of spatial uncertainty”, International Journal of Robotics Research, 5(4):56–68, 1986.

R. Mur-Artal, J. M. M. Montiel, “ORB-SLAM: a Versatile and Accurate Monocular SLAM System” IEEE TransactionsonRobotics, vol. 31, pp. 1147 – 1163, October 2015.

D. Nist´er, O. Naroditsky, and J. Bergen, “Visual odometry,” in CVPR, vol. 1. Ieee, 2004, pp. I–I.

Takafumi Taketomi, Hideaki Uchiyama and Sei Ikeda, “Visual SLAM algorithms: a survey from 2010 to 2016”, ISPJ Transaction on Computer Vision and Applications, DOI 10.1186/s41074-0027-2, 2017.

Mur-Artal, R.; Tardós, J.D, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras". *IEEE Trans. Robot.*, *33*, 1255–1262, 2017.

E. Jared Shamwell, Sarah Leung, William D. Nothwang, "Vision-Aided Absolute Trajectory Estimation Using an Unsupervised Deep Network with Online Error Correction", International Conference on Intelligent Robots (IROS), Spain, 2018.

James Jackson , Kevin Brink, Brendon Forsgren, David Wheeler, and Timothy McLain, "Direct Relative Edge Optimization, A Robust Alternative for Pose Graph Optimization", 2019.

H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I", Robotics & Automation Magazine, IEEE, 13(2):99–110, 2006.

S. Yang, S. A. Scherer, S.A, A. Zell, "An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle", Journal of Intelligent & Robotic Systems 69 (1–4) (January 2013) 499–515.

Klette R, Koschan A, Schluns K, (1998) Computer vision "three-dimensional data from images". 1st edn 11.

Nister. D, "A minimal solution to the generalised 3-point pose problem", In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Vol. 1. pp 560–5671.

Brian Williams, MarkCummins, JoséNeira, PaulNewman, IanReid, JuanTardós, "A comparison of loop closing techniques in monocular SLAM", Volume 57, Issue 12, 31 December 2009, Pages 1188-1197.

Grisetti G, Kümmerle R, Stachniss C, Burgard W, "Atutorialon graph-based SLAM". Intell Transp Syst Mag IEEE 2(4):31–43 13, 2010.

Kümmerle R, Grisetti G, Strasdat H, Konolige K, Burgard W, "g2o:A general framework for graph optimization", In: Proceedings of International Conference on Robotics and Automation. Pp 3607–3613, 2011.

Kailai Li, Johannes Cox, Benjamin Noack, and Uwe D. Hanebeck "Improved Pose Graph Optimization for Planar Motions Using Riemannian Geometry on the Manifold of Dual Quaternions", 2019.

D. Nistér, O. Naroditsky, J. Bergen, "Visual odometry", In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 27 June–2 July 2004.

A, Angeli, D. Filliat, S. Doncieux, J-A. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words". IEEE Trans. Robot. 2008, 24, 1027–1037.

Thomas Schops, Torsten Sattler, Marc Pollefeys, "BAD SLAM: Bundle Adjusted Direct RGB-D SLAM" The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 134-144.

Lei Han ; Lan Xu ; Dmytro Bobkov ; Eckehard Steinbach ; , "Real-Time Global Registration for Globally Consistent RGB-D SLAM", IEEE Transactions on Robotics, Volume: 35 , Issue: 2 , April 2019.

Z. Zhang, "Microsoft kinect sensor and its effect", Multi Media IEEE 19(2):4–10, 2012.

J. Geng, "Structured-light 3d surface imaging: a tutorial", Adv Opt Photon3(2):128–160.

P. J. Besl, ND. McKay, "A method for registration of 3-D shapes". IEEE Trans Pattern Anal Mach Intell14(2):239–256, 1992.

Ruben Gomez-Ojeda, Francisco-Angel Moreno, David Zuñiga-Noël , "PL-SLAM: A Stereo SLAM System Through the Combination of Points and Line Segments", IEEE Transactions on Robotics, Volume: 35 , Issue: 3 , June 2019.

Markus Kleinert and Sebastian Schleith. "Inertial aided monocular SLAM for GPS-denied navigation", In 2010 IEEE International Conference on Multi sensor Fusion and Integration for Intelligent Systems, pages 20–25, Sep 2010.

Joel A. Hesch, Dimitrios G. Kottas, Sean L. Bowman, and Stergios I. Roumeliotis, "Consistency analysis and improvement of vision-aided inertial navigation", IEEE Transactions on Robotics, 30(1):158–176, 2017.

J. A Hesch and S. I Roumeliotis, "Consistency analysis and improvement for single-camera localization", In Computer Vision and Pattern Recognition Workshops, pages 15–22, 2013.

Mingyang Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry", 32(6):690–711, 2013.

Meixiang Quan, Songhao Piao, Minglang Tan, Shi-Sheng Huang, "Accurate Monocular Visual-inertial SLAM using a Map-assisted EKF Approach", IEEE, 2019.

R. Mur-Artal, J. D. Tardós, "Visual-Inertial Monocular SLAM With Map Reuse", IEEE Robot. Autom. Lett., 2, 796–803, 2017.

M. Irani and P. Anandan, "About direct methods. In Vision Algorithms: Theory and Practice" pages 267–277. Springer, 1999.

Newcombe R A, Lovegrove S J, Davison A J, "DTAM: dense tracking and mapping in real time", In: Proceedings of International Conference on Computer Vision .pp 2320–2327, 2011.

J. Engel, V. Koltun, D. Cremers, "Direct Sparse Odometry", CoRR. abs/1607.02565, 2016.

J. Engel, T. Schöps, D. Cremers, "LSD-SLAM: large-scale direct monocular SLAM", In: Proceedings of European Conference on Computer Vision.pp834–849, 2014.

G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces, Mixed and Augmented Reality", 6th IEEE and ACM International Symposium on, pp. 225-234, 2007.

R. Mur-Artal, “Real-Time Accurate Visual SLAM with Place Recognition” PhD thesis, university of Zaragoza, 2017.

Hauke Strasdat, J. M. M. Montiel and Andrew J. Davison, “Real-time Monocular SLAM: Why Filter”, 2015.

S. Agarwal, K. Mierle, “Others. Ceres Solver”, Available online: http://ceres-solver.org accessed on 14 November 2017.

C. Forster, M. Pizzoli, D. Scaramuzza, “SVO: Fast semi-direct monocular visual odometry”, In Proceedings of the IEEE International Conference on Robotics and Automation, pp. 15–22 Hong Kong, China, 2014.

G. Sibley, L. Matthies, G. Sukhatme, “A sliding window filter for incremental SLAM”, In Unifying Perspectives in Computational and Robot Vision; Springer: Berlin, Germany, pp. 103–112, 2008.

G. Sibley, L. Matthies, G. Sukhatme, “Sliding window filter with application to planetary landing”. J.Field Robot, 27, 587–608, 2010.

T. Qin, S. Shen, “Robust initialization of monocula rvisual-inertial estimation on aerial robots”. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017.

H. Strasdat, J. Montiel, A. Davison, “Scale drift-aware large scale monocular SLAM”, The MIT Press, URL: http://www.roboticsproceedings.org/rss06/., 2010.

A. Davison, I. Reid, N. Molton, O. Stasse, “MonoSLAM: Real-time 47 single camera SLAM”, Pattern Analysis and Machine Intelligence, IEEE Transactions, 29(6):1052–1067. doi:10.1109/TPAMI.2007. 49 1049, 2007.

R. A. Newcombe, A. J. Davison, “Live dense reconstruction with a single moving camera”, in: Computer Vision and Pattern Recognition (CVPR), IEEE Conference on, IEEE, 2010, pp. 1498–1505, 2010.

K. Pirker, M. R ¨uther, H. Bischof, “CD SLAM - Continuous localization and mapping in a dynamic world”, in: IEEE International Conference on Intelligent Robots Systems (IROS), IEEE, pp. 3990–3997, 2011.

K. Pirker, “Histogram of oriented cameras - a new descriptor for visual slam in dynamic environments”, in: Proceedings of the British Machine Vision Conference, BMVA Press, pp. 76.1–76.12. Doi:10.5244/C.24.76, 2010.

C. Pirchheim, D. Schmalstieg, G. Reitmayr, “Handling pure camera rotation in keyframe-based SLAM”, in: Mixed and Augmented Reality (ISMAR), IEEE International Symposium on, 2013, pp. 229–238. doi:10.1109/ISMAR.2013.6671783, 2013.

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

C. Forster, Z. Zhang, M. Gassner, M. Werlberger, D. Scaramuzza, “SVO: Semidirect visual odometry for monocular and multicamera systems”, IEEE Trans. Robot. 33 (2) 249–265, 2017.

A. Concha, J. Civera, “DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence”, in: Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on, 2015, pp. 5686–5693. doi:10.1109/IROS.2015.7354184, 2015.

E. Rublee, V. Rabaud, K. Konolige, G. Bradski, “ORB: An efficient alternative to SIFT or SURF”, in: 2011 International Conference on Computer Vision, pp. 2564–2571, 2011.

S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, R. Szeliski, “Building Rome in a day”. Common ACM 54 (10): 105–112. 2011.

G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, W. Burgard, “Efficient estimation of accurate maximum likelihood maps in 3D”. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, pp. 3472–3478, 2007.

M. Kaess, A. Ranganathan, F. Dellaert, “ISAM: incremental smoothing and mapping”, IEEE Trans. Robot. 24(6), 1365–1378, 2008.

E. Olson, J. Leonard, S. Teller, “Fast iterative alignment of pose graphs with poor initial estimates”. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 2262–2269, 2006.

Hong. E., Lim. J.: “Visual-Inertial Odometry with Robust Initialization and Online Scale Estimation, Sensors”, 18 (12), 4287, 2018.

Ming Hsiao, Eric Westman, and Michael Kaess, “Dense Planar-Inertial SLAM with Structural Constraints”, 2018 IEEE International Conference on Robotics and Automation (ICRA) May 21-25, Brisbane, Australia, 2018.

Schneider, T.; Dymczyk, M.; Fehr, M.; Egger, K.; Lynen, S.; Gilitschenski, I.; Siegwart, R. “maplab: An Open Framework for Research in Visual-inertial Mapping and Localization”, IEEE Robot. Autom. Lett, 3, 1418–1425, 2018.

Lynen, S.; Sattler, T.; Bosse, M.; Hesch, J.; Pollefeys, M.; Siegwart, R. Get Out of My Lab: “Large-scale, Real-Time Visual-Inertial Localization”, In Proceedings of the Robotics: Science and Systems, Rome, Italy, 13–17 July 2015.

Qin, T.; Li, P.; Shen, S. “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator”, arXiv, arXiv:1708.03852, 2017.

Lin, Y.; Gao, F.; Qin, T.; Gao, W.; Liu, T.; Wu, W.; Yang, Z.; Shen, S, “Autonomous aerial navigation using monocular visual-inertial fusion”, J. Field Robot, 35, 23–51, 2017.

Li, P.; Qin, T.; Hu, B.; Zhu, F.; Shen, S, “Monocular Visual-Inertial State Estimation for Mobile Augmented Reality”, In Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, Natnes, France, 9–13, pp. 11–21, October 2017.

S. Weiss, M. W. Achtelik, S, Lynen, M, C. Achtelik, L. Kneip, M. Chli, and R. Siegwart. “Monocular vision for long-term micro aerial vehicle state estimation: A compendium”, Journal of Field Robotics, 30(5):803–831, 2013.

Weiss, S.; Siegwart, R. “Real-time metric state estimation for modular vision-inertial systems”, In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13, pp. 4531–4537, May 2011.

Achtelik, M.; Achtelik, M.; Weiss, S.; Siegwart, R. “Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments”, In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, pp. 3056–3063, China, 9–13 May 2011.

Lynen, S.; Achtelik, M.W.; Weiss, S.; Chli, M. “A robust and modular multi-sensor fusion approach applied to MAV navigation”, In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3923–3929, Tokyo, Japan, 3–7 November 2013.

Munguía, R.; Nuño, E.; Aldana, C.I.; Urzua, S, “A Visual-aided Inertial Navigation and Mapping System”, Int. J. Adv. Robot. Syst, 13, 94, 2016.

Fang, W.; Zheng, L.; Deng, H.; Zhang, H. “Real-Time Motion Tracking for Mobile Augmented/Virtual Reality Using Adaptive Visual-Inertial Fusion”, Sensors, 17, 1037, 2017.

Spaenlehauer, A.; Fremont, V, “Sekercioglu, Y.A.; Fantoni, I. A Loosely-Coupled Approach for Metric Scale Estimation in Monocular Vision-Inertial Systems”, Cornell University arXiv Institution: Ithaca, NY, USA, 2017.

Forster, C.; Carlone, L.; Dellaert, F.; “Scaramuzza, D. IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation”, In Proceedings of the Robotics: Science and Systems, Rome, Italy, 13–17 July 2015.

Mourikis, A.I.; Roumeliotis, S.I. “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation”, In Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14, pp. 3565–3572, April 2007.

Jones, E.S.; Soatto, S. “Visual-inertial navigation, mapping and localization: A scalable real-time causal approach”, Int. J. Robot. Res, 30, 407–430. 2011.

Li, M.; Mourikis, A.I. “High-precision, consistent EKF-based visual-inertial odometry”, Int. J. Robot. Res, 32, 690–711, 2013.

Guo, C.; Kottas, D.; Dutoit, R.; Ahmed, A.; Li, R.; Roumeliotis, S. “Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps”, In Proceedings of the Robotics: Science and Systems, Berkeley, CA, USA, 12–16 July 2014.

Wu, K.; Ahmed, A.; Georgiou, G.; Roumeliotis, S. “A Square Root Inverse Filter for Efficient Vision-aided Inertial Navigation on Mobile Devices”, In Proceedings of the Robotics: Science and Systems, Rome, Italy, 13–17 July 2015.

Tanskanen, P.; Naegeli, T.; Pollefeys, M.; Hilliges, O. “Semi-direct EKF-based monocular visual-inertial odometry”. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 6073–6078, Hamburg, Germany, 28 September–2 October 2015.

Bloesch, M.; Omari, S.; Hutter, M.; Siegwart, R. Robust visual inertial odometry using a direct EKF-based approach. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 298–304, Hamburg, Germany, 28 September–2 October 2015.

Sa, I.; Kamel, M.; Burri, M.; Bloesch, M.; Khanna, R.; Popovic, M.; Nieto, J.; Siegwart, R. “Build Your Own Visual-Inertial Drone: A Cost-Effective and Open-Source Autonomous Drone”, IEEE Robot. Autom. Mag, 25, 89–103, 2017.

Hesch, J.A.; Kottas, D.G.; Bowman, S.L.; Roumeliotis, S.I, “Consistency Analysis and Improvement of Vision-aided Inertial Navigation”, IEEE Trans. Robot, 30, 158–176, 2013.

Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W, “g2o: A general framework for graph optimization”, In Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3607–3613, Shanghai, China, 9–13 May 2011.

Kaess, M.; Ranganathan, A.; Dellaert, F, “iSAM: Incremental Smoothing and Mapping”, IEEE Trans. Robot, 24, 1365–1378, 2008.

Dellaert, F, “Factor graphs and GTSAM: A Hands-on Introduction”, Technical Report; Georgia Institute of Technology: Atlanta, GA, USA, 2012.

Leutenegger, S.; Furgale, P.; Rabaud, V.; Chli, M.; Konolige, K.; Siegwart, R, “Keyframe-Based Visual-Inertial SLAM using Nonlinear Optimization”, In Proceedings of the Robotics: Science and Systems, pp. 789–795, Berkeley, CA, USA, 12–16 July 2014.

Lupton, T.; Sukkarieh, S. “Visual-Inertial-Aided Navigation for High-Dynamic Motion in Built Environments without Initial Conditions”, IEEE Trans. Robot, 28, 61–76, 2012.

Concha, A.; Loianno, G.; Kumar, V.; Civera, J. Visual-inertial direct SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21; pp. 1331–1338, May 2016.

Vladyslav Usenko, Nikolaus Demmel, David Schubert, Jorg St, uckler, Daniel Cremers, “Visual-Inertial Mapping with Non-Linear Factor Recovery”, 2019.

Mur-Artal, R.; Tardós, J.D, “Visual-Inertial Monocular SLAM with Map Reuse”, IEEE Robot. Autom. Lett, 2, 796–803, 2017.

Piao, J.; Kim, S, “Adaptive Monocular Visual-Inertial SLAM for Real-Time Augmented Reality Applications in Mobile Devices”, Sensors, 17, 2567, 2017.

Y. Liu, Z. Chen, W. Zheng, H. Wang, J. Liu, “Monocular Visual-Inertial SLAM: Continuous Preintegration and Reliable Initialization”, Sensors, 17, 2613, 2017.

Xufu Mu, Jing Chen, Zixiang Zhou, Zhen Leng and Lei Fan, “Accurate Initial State Estimation in a Monocular Visual–Inertial SLAM System”, Sensor 2018.

Euntae Hong and Jongwoo Lim, “Visual-Inertial Odometry with Robust Initialization and Online Scale Estimation”, Sensors 2018.

Carlos Campos, Jose M.M. Montiel and Juan D. Tardos, “Fast and Robust Initialization for Visual-Inertial SLAM”, 2019 International Conference on Robotics and Automation (ICRA) Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019.

Barza Nisar, Philipp Foehn, Davide Falanga, and Davide Scaramuzza, “VIMO: Simultaneous Visual Inertial Model-Based Odometry and Force Estimation”, IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 4, NO. 3, JULY 2019.

A. Martinelli, “Closed-form solution of visual-inertial structure from motion,” International Journal of Computer Vision, vol. 106, no. 2, pp. 138–152, 2014.

J. Kaiser, A. Martinelli, F. Fontana, and D. Scaramuzza, “Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation,” IEEE Robotics and Automation Letters, vol. 2, no. 1, pp. 18–25, 2017.

SH. Jung, C. Taylor, “Camera trajectory estimation using inertial sensor measurements and structure fom motion results”, In: IEEE Int. Conf. Comput. Vis. Pattern Recog. (CVPR), 2001.

D. Sterlow, S. Singh, “Motion estimation from image and inertial measurements”. Int J Robot Research, 2004.

M. Bryson, M. Johnson-Roberson, S. Sukkarieh, “Airborne smoothing and mapping using vision and inertial sensors”. In: IEEE Int. Conf. Robot. Autom. (ICRA), pp 3143–3148, 2009.

V. Indelman, S. Wiliams, M. Kaess, F. Dellaert, “Information fusion in navigation systems via factor graph based incremental smoothing”. J Robot and Auton Syst 61(8):721–738, 2013.

Patron-Perez A, Lovegrove S, Sibley G, “A spline-based trajectory representation for sensor fusion and rolling shutter cameras”. Int J Comput Vis 113(3):208–219, DOI 10.1007/ s11263-015-0811-3, 2015.

Davide Scaramuzza and Zichao Zhang, “Visual-Inertial Odometry of Aerial Robots”, Springer Encyclopedia of Robotics, 2019.

Weibo Huang, Hong Liu, “Online Initialization and Automatic Camera-IMU Extrinsic Calibration for Monocular Visual-Inertial SLAM”, in Proc. IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, May 21-25, 2018.

R. K¨ummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g2o: A general framework for graph optimization,” in Proc. IEEE Int. Conf. Robot, pp. 3607–3613, Autom., 2011.

Hartley, R.; Zisserman, A. “Multiple View Geometry in Computer Vision”, Cambridge University Press: Cambridge, pp. 1865–1872, UK, 2003.

R. J. Radke. “CHAPTER 1 – Multi-View Geometry for Camera Networks”, Multi-Camera Networks, Principles and Applications. Pages: 3-27, 2009.

J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, “Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes,” in IEEE International Conference on Robotics and Automation (ICRA), pp. 4304–4311, IEEE, 2016.

P. Furgale, J. Rehder, and R. Siegwart, “Unified temporal and spatial calibration for multi-sensor systems,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1280– 1286, IEEE, 2013.

S. J. Haddadi and E. B. Castelan, “Accuracy Improvement of Monocular Visual-Inertial SLAM: Benchmark and Experiment”. IEEE International Conference of Robotic, Tehran, 2019.

M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, R. Siegwart. “The EuRoC micro aerial vehicle datasets”. Int. J. Robot. Res, 35, 1157–1163, 2016.

Y. Lin, F. Gao, T. Qin, W. Gao, T. Liu, W. Wu, Z. Yang, and S. Shen, “Autonomous aerial navigation using monocular visual-inertial fusion,” J. Field Robot., 2017.

R. Mur-Artal, J. D. Tardós. “Visual-Inertial Monocular SLAM with Map Reuse”, IEEE Robotics and Automation Letters, Volume: 2, Issue: 2,. 2017, 796–803.

Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. “Keyframe-based visual-inertial odometry using nonlinear optimization”, International Journal of Robotics Research, 34(3):314– 334, 2015.

S. Umeyama, “Least-Square estimation of transformation parameters between two-point patterns”, *IEEE Trans*. Pattern Anal. Mach. Intell. 1991, 4, 376-380.

Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon, “Bundle Adjustment —A Modern Synthesis”, Springer, LNCS 1883, pp. 298–372, Verlag Berlin Heidelberg, 2000.

Henry Gavin, “The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems”, Computer Scienc, Corpus ID: 5708656, 2013.

Nishant Kejriwal, Swagat Kumar and Tomohiro Shibata, “High Performance Loop Clouser Detection using Bag of Words Pairs”, Robotics and Autonomous Systems, volume 77, pages 55- 65, March 2016.

S. Julier, and J. Uhlmann, “A new method of the nonlinear transformation of means and covariances in filters and estimators”, IEEE Transactions on Automatic Control, 2000.

I. Arasaratnam and S. Haykin, “Cubature kalman filters”, IEEE Transactions on Automatic Control, 54(6), 1254–1269, 2009.

C. Silpa-Anan, and R. Hartley, “Optimised KD-trees for fast image descriptor matching”. In CVPR, 2008.