OPTIMIZATION BASED VISUAL-INERTIAL SLAM AND ODOMETRY SYSTEMS IN GPS-DENIED ENVIRONMENTS

by
FATİH MEHMET DEMİREL

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfilment of
the requirements for the degree of Master of Science

Sabancı University
December 2021

OPTIMIZATION BASED VISUAL-INERTIAL SLAM AND ODOMETRY SYSTEMS IN GPS-DENIED ENVIRONMENTS

Approved by:

Date of Approval: 15/12/2021

OPTIMIZATION BASED VISUAL-INERTIAL SLAM AND ODOMETRY SYSTEMS IN GPS-DENIED ENVIRONMENTS

FATİH MEHMET DEMİREL

MECHATRONICS M.A. THESIS, DECEMBER 2021

Thesis Supervisor: Prof. Dr. Mustafa Ünel

Keywords: Visual-Inertial Simultaneous Localization and Mapping, Visual-Inertial Odometry, Navigation in GPS Denied Environments, Bundle Adjustment

ABSTRACT

Visual Inertial Simultaneous Localization and Mapping (VI-SLAM) and Visual Inertial Odometry (VIO) systems are widely used in various areas such as augmented reality, autonomous cars and aerial vehicles' navigation systems where there is need for navigating the platform in the absence of the GPS information. There are many different configurations of the VI-SLAM and VIO systems in the literature in terms of the sensor types, methods that are used in estimating the states of the platform, sensor fusion methods and the front and back end structures of the visual-inertial systems. In this thesis, the focus will be on monocular graph optimization based VI-SLAM and VIO systems. For this purpose, end-to-end VI-SLAM and VIO structures have been built and the trajectory results have been evaluated using the Euroc-Mav dataset. Moreover, as a contribution to the current studies, a solution to the problem of neglecting dynamic objects in the environment has been proposed to increase the robustness of the visual-inertial navigation systems.

ÖZET

GPS BAGIMSIZ ORTAMLARDA OPTIMIZASYON TABANLI GORSEL ATALETSEL ES ZAMANLI HARITALAMA VE KONUMLANDIRMA SISTEMLERI

FATIH MEHMET DEMIREL

MEKATRONIK MÜHENDİSLİĞİ YÜKSEK LİSANS TEZİ, ARALIK 2021

Tez Danışmanı: Prof. Dr. Mustafa Ünel

Anahtar Kelimeler: Gorsel-Ataletsel Es Zamanli Haritalama ve Konumlandirma Sistemleri, GPS Bagimsiz Ortamlarda Navigasyon, Görsel-Ataletsel Odometri

Görsel-ataletsel eş zamanlı haritalama ve konumlandırma sistemleri ve görsel odometri artırılmış gerçeklik, otonom arabalar, GPS bağımsız ortamlarda hava araçlarının navigasyon sistemleri gibi çeşitli alanlarda yaygın olarak kullanılmaktadır. Literatürde görsel-ataletsel eş zamanlı haritalama ve konumlandırma sistemleri ve görsel odometri sistemlerinin sensör tipleri, platformun durumlarını hesaplamak için kullanılan yöntemler, sensör füzyon yöntemleri, sistemin önyüz ve arkayüz yapıları konularında farklı konfigürasyonlar bulunuyor. Bu tezde odak noktası monoküler poz grafiği optimizasyonu üzerine görsel-ataletsel eş zamanlı haritalama ve konumlandırma sistemleri olacaktır. Bu amaçla uçtan uca görsel ataletsel eş zamanlı haritalama ve konumlandırma sistemi oluşturuldu ve poz sonuçları Euroc-Mav veri seti kullanılarak değerlendirildi. Ayrıca, mevcut çalışmalara katkı olarak, görsel-ataletsel navigasyonun sağlamlığını artırmak için çevredeki dinamik nesnelerin elimine edilmesi üzerine bir çözüm önerilmiştir.

ACKNOWLEDGEMENTS

I would like to thank to my thesis supervisor Prof. Dr. Mustafa Ünel for his guidance and assistance during this thesis. Furthermore, I would like to thank to my family for their support and encouragement.

to my family

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Today, autonomous systems are getting involved in many different areas, from manufacturing to the daily used devices. Autonomous cars, drones, vacuum cleaners, transportation robots, aerial vehicles could be given as autonomous systems that are getting more popular day to day from different areas. These autonomous systems could be categorized into surveillance systems, transportation, precision agriculture, home electronics etc. The application area of autonomous systems is getting wider day after day thanks to the smaller and more energy-efficient, cheap devices that capable of supporting those systems' autonomy. In common, the autonomous systems should perceive its surrounding and navigate in their environment autonomously. In that aspect, several different structural and algorithmic designs are studied in the literature, making the autonomous system perceive its environment and locate itself in that environment.

The focus of this study is building end to end visual inertial SLAM system and investigating the state-of-the-art approaches in visual-inertial slam and odometry systems while contributing to the algorithm in eliminating dynamic objects from the scene in order to increase the robustness of the algorithm in dynamic environments. In addition to that, some modules of the SLAM systems are replaced and tested with recent studies in the literature. Moreover, the studied state-of-the-art algorithms is compared with the experimented results on both public datasets and real-world experiments. Based on that comparison, results are discussed in terms of the different and common approaches in their algorithm structure and the used novel methods. In order to test the algorithms and the contributed system module, a visual-inertial monocular platform is built with Nvidia Jetson Nano, raspberry pi monocular camera, and 9250 IMU with ROS in the backend.

The described navigation system is based on the visual and inertial sensor data fusion. Moreover, the system output will be the map of the environment and the location of the aerial vehicle in that environment. The location consists of the position and the orientation of the aerial vehicle in that map. In literature, this problem is called visual-inertial navigation.

Even though several state-of-the-art works propose a solution for visual-inertial nav-

igation systems for aerial vehicles, in this thesis the most well-known ones (Kimera, ORB-SLAM3 and VINS-MONO) that give the most accurate results are considered.

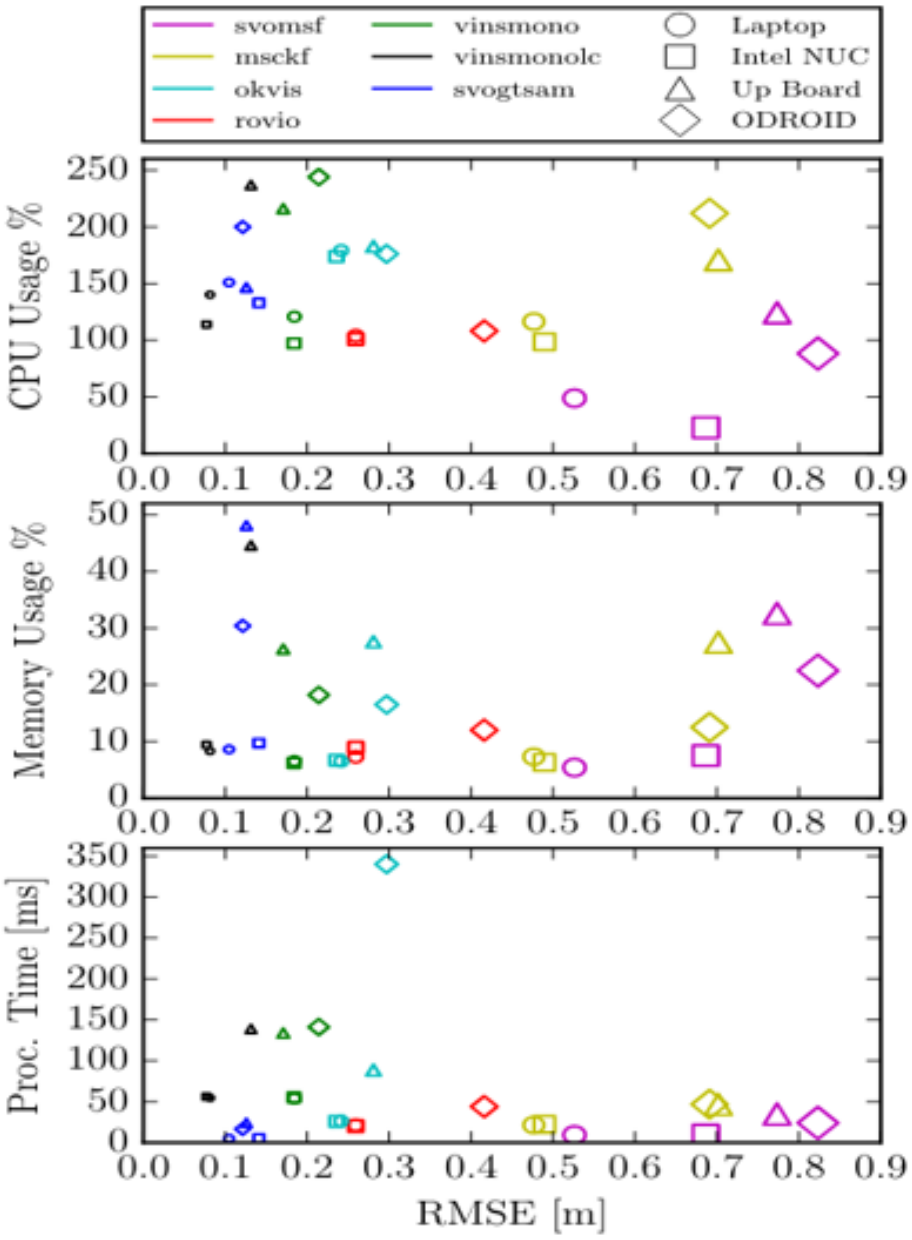


Figure 1.1 Performance results of common VIO algorithms based on root mean square error. Adapted from "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots" by J. Delmerico and D. Scaramuzza, IEEE International Conference on Robotics and Automation(ICRA), 2018

According to Figure 1.1 Delmerico & Scaramuzza (2018) there isn't any comparison of the latest visual-inertial SLAM algorithms such as Kimera Rosinol et al. (2020) and ORB-SLAM3 Campos et al. (2020). In this thesis, the state-of-the-art algorithms is compared in terms of the APE(Absolute Pose Error) and RPE(Relative Pose Error) with the corresponding computational power needs on several public

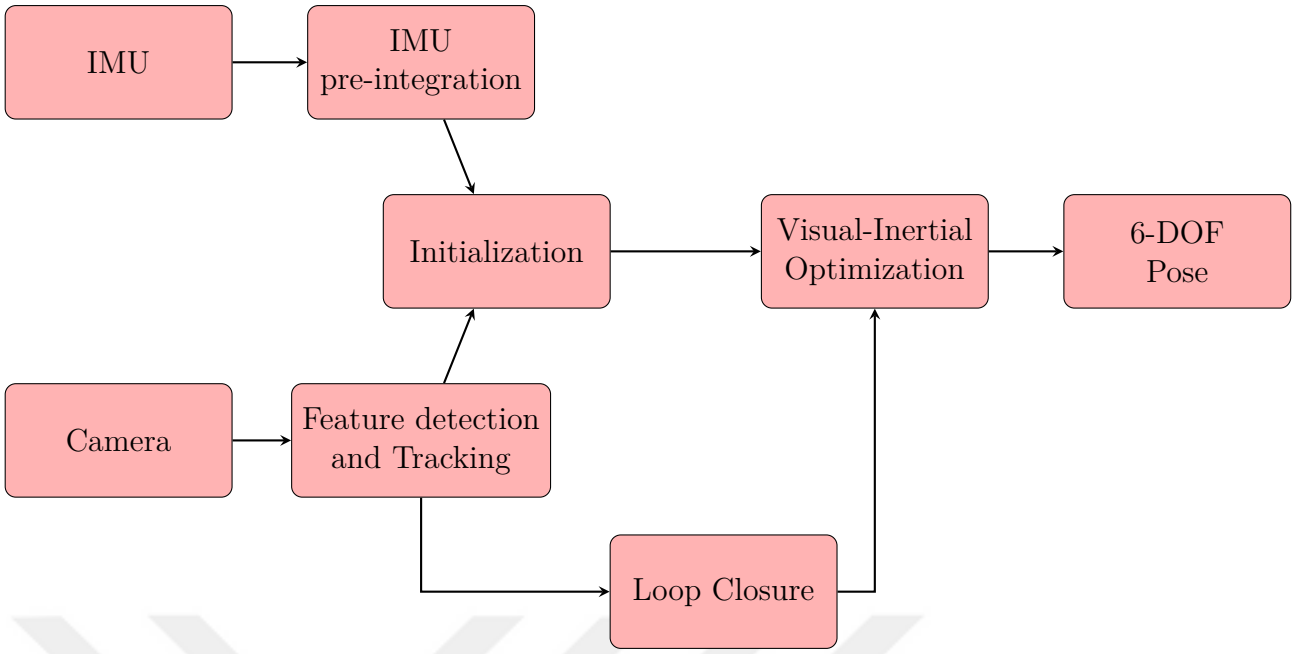


Figure 1.2 Tightly-Coupled Fusion

datasets such as kitti, tum-rgbd and euroc. Moreover, stated algorithms compared with the custom dataset obtained with the built platform mentioned above.

According to the survey study results throughout the thesis, I have decided to focus on a tightly-coupled monocular visual-inertial odometry system to solve the GNSS problem. In Figure 1.2, the overall schema of tightly coupled fusion in visual inertial SLAM pipeline is given by modules. In the steps of building the VIO system structure, several variables need to be decided—those selecting according to the needs of the system that it will be applied. As an outcome of our literature survey, the most suitable configuration for our system is figured out.

As an outcome of the literature survey, I have figured out the most suitable configuration for the system. There are reasons behind the selection of the system properties. In section 2 the rationales behind those selections will be explained in detail.

Furthermore, in the proposed visual-inertial system design, I plan to increase the system's accuracy in real-world experiences by considering the dynamic object in the environment such as humans, animals, moving vehicles, etc. Because in the application areas of the visual-inertial systems, such as mobile robots in industries, autonomous vehicles in public areas, etc., there are moving, dynamic objects around most of the time. The features detected on those objects will not be stable in the point cloud, which will affect the relative pose calculations in the pose graph. The aim is to detect those objects with semantic segmentation in the frames. After

detecting those objects, features detected on those objects will not be saved in the point cloud. In the end, only features from the static objects will be taken into consideration in the visual measurements. By following that feature elimination strategy, I aim to increase the robustness of the visual-inertial system in real-world experiences.

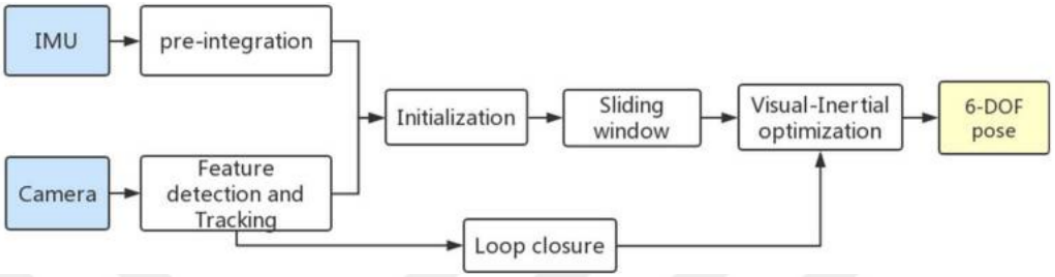


Figure 1.3 General base structure of the VIO algorithm. Adapted from "Robust Stereo Visual-Inertial Odometry Using Nonlinear Optimization" by S. Ma, X. Bai, Y. Wang, R. Fang, Sensors(Basel), 2019, 19(17): 3747

Figure 1.3 represents the general framework of the visual-inertial pose estimation structure. Most of the VIO systems use a camera for visual data and the IMU sensor. In terms of the type of camera, there are two approaches. One of them is building the system using stereo cameras, and the other one is using monocular cameras. In our case, I will go over the monocular camera approaches. There is a need for a considerable baseline length for accurate results in the stereo camera approaches. The baseline is the distance between the two lenses of the stereo camera, and it influences the depth range that can be observed and depth resolution. In that sense, in high altitudes, there is a need for massive baselines. That is why in our case, the system should be independent of this feature. Thus, I decided to go over monocular camera approaches.

In this general structure in Figure 1.3, inputs are the image from the monocular or stereo cameras and the IMU data, which contains the acceleration and angular velocity measurements. Based on those inputs, the output is the 6-DOF pose estimation of the platform. The system starts with feature extraction and IMU pre-integration. The inertial and visual poses are combined to estimate the platform with its pose, velocity, gyroscope bias, gravity vector. The visual-inertial odometry algorithm iteratively updates these values. In the end, the 6-DOF pose of the platform is calculated.

1.1 Outline of the Thesis

The rest of the thesis is organized as follows. Chapter 2 introduces the notation that used throughout the thesis and gives background information about the methods used in the visual-inertial navigation SLAM systems.

In chapter 3 state of the art algorithms are discussed and analyzed in terms of the standard and different approaches in the methodologies they have used for different modules of the visual-inertial SLAM systems that they proposed. Those modules could be summarized as; visual-inertial system initialization, visual-inertial process loop, loop closure, and graph optimization.

In chapter 4, proposed visual-inertial SLAM pipeline is introduced and details given about the sub modules that implemented in the system. Moreover logic behind the connections of those submodules are also discussed.

Experimental results are presented in chapter 5. In chapter 6, the thesis is concluded with several remarks and possible future directions are provided.

2. LITERATURE SURVEY AND BACKGROUND

In this chapter, different system configurations in the modules of the visual inertial structure in the literature are discussed. Furthermore, visual structure, the methods used in the literature and inertial structure are discussed in separate sections. Finally, fusion methodology of inertial and the visual data are discussed in the last section.

2.1 VISUAL-INERTIAL SYSTEM CONFIGURATIONS

In this section, general system configurations of the visual inertial structure are discussed in detail. In addition, the advantages and disadvantages of the selections are also discussed.

2.1.1 Sensor Types

First of all, there are several options to collect the measurements from the environment in SLAM systems. For that purpose, the most used sensors are LIDAR and the cameras. For the cameras, the most common ones in the SLAM systems are the monocular and stereo cameras. There are pros and cons of each sensor type regarding the measurement properties for different use cases. LIDARs and stereo cameras have the advantage of collecting scale data by measuring the depth in the environment. This property would be beneficial in the data association process of any SLAM system. However, for instance, for LIDARs, the type of sensor is an active sensor, and its sending rays to the environment in taking the measurements. In that aspect, LIDARs can not be used in defensive industries because it prevents the system from being stealth. Therefore, to preserve the stealth property of the aircraft, we will use the camera as a passive sensor.

In terms of the camera, there are stereo and monocular cameras. Stereo cameras are advantageous and robust in small-scale, mostly indoor applications of the SLAM. They are not applicable in large environments. To get accurate depth information in stereo cameras, the camera baseline and distance to the landmark ratio should not be too small. In aerial case, when thinking about the baseline - altitude ratio, it is too small to get the accurate depth information from the stereo camera directly. There

is a disadvantage of monocular cameras that they are incapable of recovering the metric scale. The IMU comes into play to solve this lack of capability of monocular cameras to observe the metric scale. In order to observe the metric scale, we need acceleration in at least two axes, and this information will provided by the IMU.

In the visual-inertial SLAM system, a map of the environment and the 6-DOF pose state of the agent are estimated by fusing the visual and inertial measurements. Those sensors are complementary. Cameras have a limited output rate of around 100Hz, and scale could not be determined from monocular cameras. IMUs are scene independent and have high output rate around 1000Hz. But they suffer from noise in signals and bias. Those disadvantages results in accumulated drift in the calculations. Therefore, in order to overcome those disadvantages' of both sensors, they are used as complementary sensors in the visual-inertial slam systems.

Another type of camera that recently started to be used in SLAM systems is event cameras, which have advantages and disadvantages compared with standard cameras. The first commercial event cameras is proposed Lichtsteiner et al. (2008) and in study Vidal et al. (2018) event cameras are used in visual-inertial slam system. The most important advantage of event cameras upon standard cameras is that they can work in fast movements, blurred inputs, and low light conditions thanks to their capability to capture a high dynamic range of inputs. On the other hand, as the name describes, they capture the event, and those inputs are asynchronous. Therefore fusing asynchronous data from the event camera with another sensor could be harder to implement to build a robust system. In addition to their advantages, the latency is very low compared to the standard cameras, where, in event cameras, the latency is in the order of microseconds.

2.1.2 Front-End Structure

We can state that there are two main parts of the generic SLAM algorithm. Those are stated as front-end and back-end. In literature, they are also called data association and optimization parts, respectively. In the front-end part, the pose graph is built by using the measurements from the sensors. In that pose graph, nodes are representing the platform's position and the landmarks in the environment. Moreover, there are edges between those nodes, which represents the measurement constraints between those nodes.

The front-end structure is mostly based on visual data processing. As described in Cadena et al. (2016), in the front-end structure, the features are extracted from the frames and the landmarks in the environment are related with those extracted fea-

tures. In order to relate 3D landmarks with extracted 2D features from the image, the triangulation method is used. The assumption in triangulation is the knows relative pose between the related frames. In order to calculate that relative pose between the related frames, methods that are using epipolar geometry are used by calculating homography, fundamental or essential matrix—making the decision between one of those matrixes in the structure based on the knowns such as camera intrinsics and the scene conditions. The feature correspondences need to be known in order to calculate the relative pose of the related frames. For feature correspondence in literature, there are different methods used in the studies. For instance, in Campos et al. (2020) feature descriptors are matched in order to find feature correspondences between the frames. On the other hand, in Qin et al. (2017), KLT optical flow algorithm Lucas & Kanade (1981) is used in order to track the features. The advantages and disadvantages of those approaches will be discussed in section 2.2.

Data association could be divided into three as short-term, mid-term, and long-term data associations. Typically short-term data association stands for the data association in the feature tracking level, which describes the association in the consecutive frames or in a single window of sliding frames. On the other hand, long-term data association provided with the loop closure methods in the literature—mid term data associations not much common as compared to the short and long term. Mid-term data association approach proposed in the Campos et al. (2020) and Zubizarreta et al. (2020) where data is associated not in consecutive frames but the frames located in the local map of the visual-inertial SLAM system.

2.1.3 Back-End Structure

In the back-end part, there are two approaches stated as filtering-based and optimization-based. In the filtering approach, there is no re-linearization process for old measurements. It uses older state information to estimate the latest state and drop the older states permanently. That results in making the linearization errors and erroneous measurements permanent. In the literature, optimization methods are also called smoothing methods.

On the other hand, in the smoothing methods, there is re-linearization for older states. When the estimate is updated, older measurements re-linearized, and multiple states are estimated instead of the only latest state; that needs more computational power than the filtering approach but more accurate. The recent studies show that the optimization method becomes more accurate and becomes applicable in near real-time applications of the visual-inertial navigation system. Therefore

optimization-based methods became more popular in the last decade. Optimization-based methods are divided into fixed lagged smoothing and full smoothing. Both approaches are explained in detail in the following section.

2.1.3.1 Fixed Lag Smoothing and Full Smoothing

Optimization-based back-end structure in VINS can be divided into two methods which are called fixed lag smoothing and full smoothing Huang (2019). The difference between those two methods relies on the older states' relinearization procedure, which was stated as the prior in optimizing the pose graph in VINS. In fixed lagged smoothing, some of the older states are marginalized, and the rest is relinearized and set as the prior. On the other hand, in the full smoothing approach, all past states are relinearized and set prior to the optimization problem. The increase in the number of states that linearizerd in the optimization means decreasing the reprojection errors in the feature-based in-direct VINS. Therefore full smoothing approaches are more accurate but need more computational power to handle the relinearization process of all older states. In the same perspective, the nonlinear optimization problem to solve in the full smoothing approach getting greater than it is in the fixed lag smoothing approach.

In both optimization methods, fixed lagged smoothing and full smoothing, selecting key-frames to process is common. In three state of art papers that investigated in this study Qin et al. (2017), Campos et al. (2020), Rosinol et al. (2020), all of them used their key-frame selection strategy in their visual-inertial navigation algorithms. In the key-frame selection model, not all of the frames included in the sliding window set in the fixed lag smoothing and the full smoothing. The key-frame selection model is based on deciding whether the frames are essential to consider in processing—this decision strategy is based on the design of the algorithm. There are stated constraints to decide whether the current frame is key-frame or not. Those constraints could be the parallax between consecutive frames, new feature quantity in the current frame different from the older frame, etc. Based on those constraints, for instance, in Qin et al. (2017) if the current frame is stated as key-frame, visual and inertial measurements of that frame are included in the formulated cost function. On the other hand, if the current frame is not selected as a key-frame, visual measurements of that frame are neglected, and only IMU measurements are taken into consideration when solving the related cost function Qin et al. (2017). That helps increase the VINS algorithm's computational efficiency without losing valuable information from the frame with key-frame selecting protocol applied.

2.1.4 Direct and In-Direct SLAM

VINS (visual-inertial navigation systems) are separated into direct and in-direct in terms of the visual residual models. Those methods are distinguished in terms of error types that they are trying to minimize Huang (2019). In in-direct methods, initially, feature points are extracted from the frame, and based on the correspondences of those feature points, geometric re-projection error is calculated. In contrast, in the direct methods, the photometric error is calculated upon all pixel intensities in the image frame. In terms of computational efficiency, in-direct methods are more computationally heavy because of the feature extraction and finding correspondences of those extracted features.

On the other hand, direct methods give better results in the low texture environments, because as mentioned above, the photometric error is calculated from pixel intensities in the image. However, in in-direct methods, feature extraction could be problematic in low detailed texture environments, resulting in not enough features in the scene. Therefore it will not be robust working in low texture areas with the in-direct methods. On the other hand, besides the low texture and blurry environments, in-direct methods are more robust and accurate than direct methods.

In Campos et al. (2020), it is mentioned that direct methods have their limitations in different aspects. In order to increase the accuracy of the tracking, the baseline should be wide enough to overcome the depth constraint. However, in direct methods, due to the photometric consistency, the baseline of the correspondences is restricted to some extent. That limits the tracking accuracy, and therefore, the model of the direct method should be designed perfectly in order to sustain the tracking robustness. On the other hand, in in-direct methods, extracted features could be matched in wider baselines and from different perspectives. That is the one advantage of the in-direct methods over the direct methods.

Moreover, in terms of the error calculation part of the process, in-direct methods are more computationally efficient thanks to it only operates on the extracted features. On the other hand, in terms of the error minimization and the accuracy, with bundle adjustment applied on the indirect methods, results are close as it is in the direct methods.

2.1.5 Fusion Methods

Another aspect of deciding in VIO algorithm is whether the built the structure upon tightly or loosely coupled fashion. Typically there are two main parts that the fusion of the sensor output could be used in the calculation of the state vector.

The first part is the initialization of the visual-inertial navigation algorithm, which bootstraps the system with estimating the initial parameters. The second part is the visual-inertial loop of the system.

In loosely coupled fusion, the pose of the platform is estimated with both camera and IMU measurements separately, and at the end, fusion is applied to their estimation. In a loosely coupled approach, fusing visual and inertial information is not considered in the raw data level; this makes the system incapable of correcting the drifts in vision only pose estimation. On the other hand, in the tightly coupled approach, raw measurements of the camera and IMU are used together to estimate the platform's position. The tightly coupled approach needs more computational power than loosely coupled fusion, but it is more accurate than the loosely coupled approach.

In literature, another naming convention for those two fusion methods are joint and disjoint approaches. Joint approaches stand for the tightly coupled fashion, and the disjoint approach stands for the loosely coupled fashion. As described in the Zubizarreta et al. (2020), in joint methods, the inertial and visual residual are used together in the MAP estimation. On the other hand, in disjoint methods, inertial and visual residuals are not using in the same maximum-a-posteriori(MAP) estimation.

2.1.6 Additional Properties

Additionally, visual-inertial navigation systems could have some properties which enhance the visual-inertial odometry and slam system's accuracy and robustness. Firstly, there is loop closure in the structure. Loop closure means that when the platform becomes the same position as one of the older states, it detects the state and optimizes the whole path.

Moreover, in the data association part, different algorithms using different key-frame selection constraints. Running the algorithm with the key-frames makes it efficient without crucial information from missing frames. Therefore the key-frames should be selected carefully in order not to miss information from elected frames as non-key-frame. Typically, algorithms calculate how much the current frame is essential with calculating the parallax and the number of different features in that frame. Moreover, there are threshold values for both feature count and the parallax, which vary between applications and environments. For instance, if the number of different features exceeds a stated threshold, then that means there is no need to consider this frame as a new frame. That helps us to prevent storing unnecessary data.

Another issue is about the visual-inertial odometry and slam algorithm is being online or offline. In the offline approach, the optimization process runs after the whole data association is completed. On the other hand, in the online approach, data association and optimization are processed in the same frame in nearly real-time.

2.2 VISUAL STRUCTURE

In this chapter, modules of the visual structure in visual inertial SLAM system are discussed. Overall system is named as structure from motion in the literature. Name itself describes the system where structure defines the map of the environment and "from the motion" tells that the environment is built by the motion. There are also vision modules discussed in structure from motion as PnP, tirangulation, epipolar geometry, feature extraction and matching, feature tracking methods as optical flow and descriptor matching.

2.2.1 Structure from Motion

Structure from motion describes the whole visual structure used in the visual-inertial systems. The input in the structure from motion is the image key-frames, and the output of that processed key-frames are rotation, and up-to-scale translation of the camera poses with respect to the reference frame. In all three studies that are the main focus of this thesis, Qin et al. (2017), Campos et al. (2020), and Rosinol et al. (2020) based on sliding windows based processing in their visual structures. There are differences in terms of the methods applied in the sections of the structure of motion, which will be discussed in the related sections below.

2.2.1.1 PnP

The perspective-n-point algorithm is used to determine the pose of the camera using the 3D locations of the landmarks in the environment. The input in the PnP algorithm is the 3D location of the landmarks and the related 2D feature locations in the respective image frame. By using these two pieces of information, the pose of the camera with respect to the reference frame could be calculated. The minimum required point quantity is 3 in the PnP algorithm, but the PnP algorithm gave four solutions with three points. In order to find a unique correct solution among those four solutions, one or more additional feature points are needed. In Qin et al. (2017), P3P is used in the visual front end part.

2.2.1.2 Triangulation

The triangulation method is used to determine the 3D locations of the landmarks related to the corresponded features in two frames. In triangulation, the relative transformation between those two frames should be known to calculate the 3D locations of the landmarks in the environment with respect to the frame taken as the reference. Moreover, the camera matrices need to be known to calculate the rays from camera projection center to the related correspondence point. Intersection of those rays from two different views resulted in the 3D location of the related correspondence point.

In triangulation, the calculated locations of the landmarks are calculated in the respected reference image frame. In order to get the position of the landmarks in the world frame, the transformation between the world frame and the reference image frame should be known.

2.2.1.3 Fundametal Matrix - Epipolar Geometry

In structure from motion, there is a need to know the relative orientation of the frames with respect to each other in order to calculate the pose of the camera with respect to the reference image frame or at the end world frame. Several approaches are used to get the relative transformation between the consecutive frames in the sliding window in visual-inertial systems. Those approaches are based on the epipolar geometry in computer vision which contains the fundamental matrix, essential matrix, and homography matrix calculations. The selection between those matrix calculations is based on the environment observed and the algorithm's structure. For instance, if the observed scene is mostly planar, the homography matrix could be selected to be calculated to get the relative transformation between the consecutive frames. On the other hand, if the observed scene is mostly non-planar, the fundamental or essential matrix is calculated to get the relative transformation between consecutive frames.

There are different methods to calculate the fundamental and homography matrix. Essential and fundamental matrices are convertible to each other with knowing the calibration matrix of the camera. In order to find the unique fundamental matrix, there are at least 8 point correspondences needed to be known in consecutive frames. The fundamental matrix is a 3x3 matrix that contains nine elements, but the last element came as an up-to-scale parameter and is stated as one. Then for the rest of the eight elements, there are eight unknowns in the matrix. Therefore there are at least eight equations needed to solve that eight unknowns. Those equations are built

upon the epipolar geometry constraints with known point correspondences between the consecutive frames. In epipolar geometry, coplanarity constraint is considered to build those equations in calculating the fundamental matrix. In coplanarity constraint, it is assumed that the three vectors, which are the first camera to the correspondent point, the second camera to the correspondent point, and the vector between the two cameras called baseline vector, shape a planar surface. There is a theorem saying that if three vector builds a planar surface, their triple product should be equal to zero. Hartley & Zisserman (2003)

For the homography matrix case, where the corresponding points are assumed to lie on the same planar surface, at least 4 points are known to solve the homography matrix unknown elements. The homography matrix is the same size as the fundamental matrix, which is 3x3 and contains nine elements. Same with the fundamental matrix, the last element of the homography matrix is set as the up-to-scale parameter as 1. Therefore again, there are eight unknowns to be solved in order to get to the homography matrix. The constraint of being on the same plane for the corresponding points, each correspondent point has two constraints in the x and y-direction. Therefore, 4 points have eight constraints to build eight different equations. By using the homography matrix, feature correspondences between related frames could be found. The 2D location of a point in the first frame multiplied with the homography matrix calculated between those two frames; the result of this equation is expected to equal its corresponding 2D location in the second frame.

2.2.1.4 Feature Extraction and Tracking

Feature extraction and feature tracking are two of the main essential parts of the visual and visual-inertial SLAM. The success of the structure from the motion algorithm is based on the accurate feature correspondences between the respected key-frames. Therefore, in terms of feature extraction, features should be extracted consistently and accurately. Another important point about the features is the uniqueness. In order to match the extracted features, their descriptors should be unique among themselves. Moreover, for a single point, its descriptors ideally expected to be same in images that taken from different point of views.

In terms of the accuracy of the feature tracking between the related frames, it is very important to match the features correctly. Moreover, in order not to lose the track, tracking should be stable in different conditions such as fast movements and different distances between the key-frames. In recent studies about visual-inertial SLAM systems, optical flow and descriptor matching methods are the most popular methods that are used in feature tracking. Both methods will be discussed in the

following sections respectively.

In terms of the features that extracted from the frames, the recent studies on visual-inertial SLAM systems mostly used hand-crafted features such as, shi-Tomasi, orb Rublee et al. (2011) which is faster and computationally more efficient than SIFT Lowe (2004), and harris corners. On the other hand, recent studies on feature extraction methods mostly focused on the deep learning based methods. In terms of the computational efficiency most of them are more computationally heavy than the handcrafted methods. But in terms of the accuracy and robustness to the viewpoint changes, learning based methods are more accurate Jin et al. (2020). One of the most popular feature extractor from the Magic Leap company is the Superpoint. Revaud et al. (2019)

In image matching challenge that CVPR presented in 2020 Trulls et al. (2020), for the restricted keypoint section where they have tested with 2k features, as a handcrafted feature descriptor SIFT resulted in with NI(number of inliers) 76.84 and mAA(mean average accuracy) as 0.366 . On the other hand, in the same section, the mentioned learning based feature descriptor SuperPoint resulted with NI-441 and mAA-0.681 . There could be some restrictions in learning based feature extractors. For instance without manipulation, SuperPoint could handle at most 2k features in the image. On the other hand, for instance for Revaud et al. (2019), it can handle up to 8k features in the image. That shows the application properties will be determinant to select the proper extraction method.

2.2.1.5 Optical Flow

Optical flow is using for tracking the extracted features or directly the pixels through the consecutive frames. Optical flow methods could be separated as sparse and dense. In sparse optical flow, tracking is upon the windows on the image, system tracks the given features with specific pixel locations in the image. On the other hand, in dense optical flow, the flow of the all pixels are calculated at the end of the process. Dense methods are more accurate in tracking, but the processing time is not suitable for real-time in high complexity applications such as SLAM systems. In most of the visual SLAM systems in the literature, sparse optical flow is used as a feature tracking method. But in near feature, with the development of the processors and GPUs, sparse optical flow algorithms would be replaced by learning based dense optical flow algorithms.

Other than directly using optical flow for feature tracking, dense optical flow methods can be used to extract the relative depth map of the objects from the image.

Especially, recent learning-based dense optical flow studies such as RAFT Teed & Deng (2020), the flow calculated very accurately, and the object's masks in the image can be extracted in detail. That flow maps can be used to solve the occlusion by using dense flow algorithm to extract object masks in the specific applications.

In terms of the SLAM applications, KLT sparse optical flow algorithm Lucas & Kanade (1981) is very popular among recent studies. VINS-MONO and Kimera are using KLT Optical flow algorithm to track the features between the frames.

2.2.1.6 Descriptor Matching

One of the methods that used to find the correspondent features between the corresponding frames is descriptor matching. In Campos et al. (2020), instead of optical flow as in Qin et al. (2017) and Rosinol et al. (2020), features tracked with descriptor matching method where the features are ORB features.

In the descriptors matching method, each extracted feature has its description array calculated with various methods. Most of the handcrafted methods used the neighbour pixels to calculate the descriptor for an exact point in the image. One of the recent works, called Superpoint DeTone et al. (2018), is one of the learning-based descriptors with 256 length float array as descriptors of one feature point. It is very robust in different environmental conditions, illumination changes and the viewpoint changes. It resulted better than SIFT in terms of the viewpoint changes, and better than FAST in terms of illumination changes as indicated in the SuperPoint paper DeTone et al. (2018). In Jin et al. (2020), it is stated that in terms of the number of inliers in handcrafted feature extraction methods are mostly higher than the average, but in terms of the mAA(mean average accuracy), mostly, the results are not high as the learning based methods.

The step of matching those calculated descriptors is kindly based on finding the similarities between the descriptors. Other than traditional methods to find the similarities between the descriptor array, recently, there are learning-based works which are not only considering the visual descriptor of the point. One of the robust learning-based feature matching methods is SuperGlue Sarlin et al. (2019) where points are evaluated as both in visual descriptors and their spatial appearance in the image frame. Moreover, another way that trying to be improved in feature matching in recent studies is the efficiency. The data getting enormously huge and the matching systems should handle this data in near real time for most of the applications. One of the most popular work in terms of the accuracy and the efficiency to find the similarities between the sequences in work from the Facebook Research

Group, which is called Faiss Johnson, Douze & Jégou (2017) . It is described as finding similarities between million sized arrays for the algorithm and it can work both on CPU and GPU.

2.3 INERTIAL STRUCTURE

In this chapter, inertial structure of the visual inertial SLAM is discussed. The main focus is IMU-preintegration which briefly synchronize IMU data with the camera data.

2.3.1 IMU-PREINTEGRATION

In both VINS-MONO and ORB-SLAM3, the same approach is applied in the inertial integration as a pre-processing part in the VINS systems. In both algorithms, IMU pre-integration method explained in the Forster et al. (2015) used to integrate the inertial measurements between the time period of taking two consecutive image frames.

The rational behind the IMU pre-integration is that the inertial sensor's data collection frequency is much greater than the camera sensor, there is a need to preintegrate those sensor readings between two selected key-frames from the camera sensor. Pre-integrating those inertial sensor readings between two consecutive selected camera frames becomes a single relative transformation constraint used in visual-inertial alignment in the VINS.

2.4 VISUAL INERTIAL FUSION

In this chapter, two methods, tightly and loosely coupled fusion, that used in fusing inertial and visual data in the visual inertial SLAM structure are discussed.

2.4.1 Tightly - Loosely Coupled Visual-Inertial Fusion

Tightly and loosely coupled fusion methods are most common visual inertial data fusion approaches in the literature. In tightly coupled fusion, as it is described in Figure 2.1 inertial and the visual data are fused in feature level.

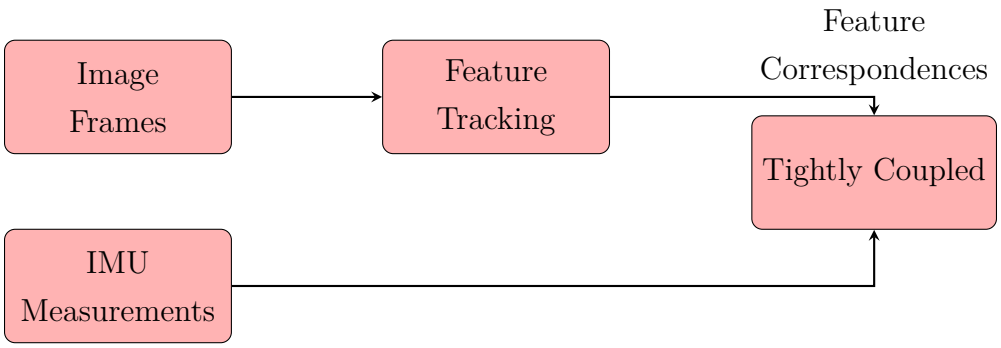


Figure 2.1 Tightly-Coupled Fusion

On the other hand, as modeled in Figure 2.2, in loosely coupled fusion of the visual and the inertial data; position and orientation are estimated separately with visual and inertial data. At the end respected position and the orientation estimations are fused.

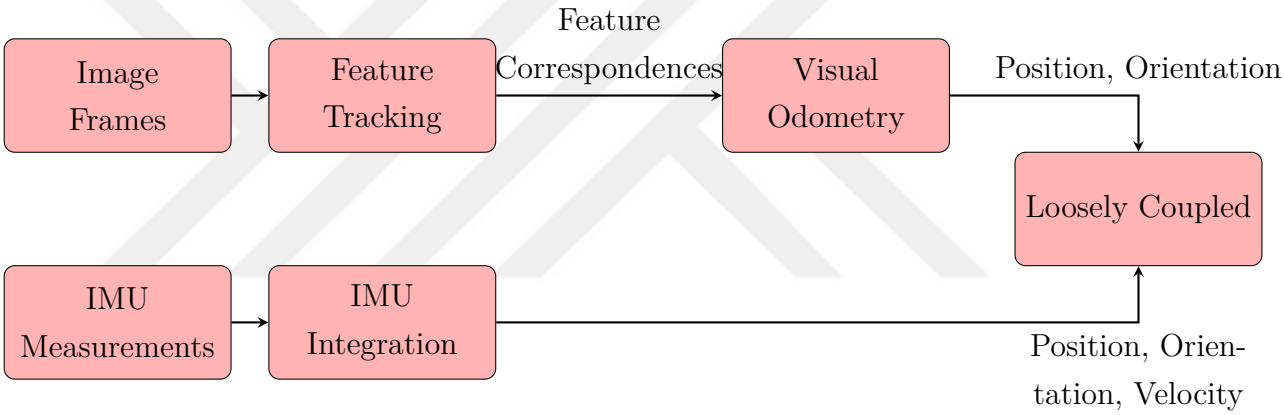


Figure 2.2 Loosely-Coupled Fusion

3. VISUAL-INERTIAL SLAM SYSTEM

In chapter 2 visual-inertial system configurations and the rationale behind those configurations are discussed. In this section, those configurations will be discussed in detail upon recent state of the art studies Qin et al. (2017), Campos et al. (2020) and Rosinol et al. (2020). Throughout the section, mathematical approaches to both visual and inertial structures and the fusion methods used in those studies will be discussed. The discussion will be made on the different and common approaches that those studies have in different sections of the visual-inertial slam structures.

As standard, visual-inertial slam structure starts with an initialization procedure, which bootstraps the whole visual-inertial system to estimate the system's initial parameters. This thesis is based on investigating the monocular camera-based visual-inertial systems, which means there is a need to estimate the scale parameter in the visual side of the system. In the initialization procedure, together with the scale parameter, different parameters are estimated.

After successful initialization of the system parameters, the system continues in the loop called tightly or loosely coupled visual-inertial system based on the MAP estimation. Based on the used fusion structure, which is tightly or loosely, visual and the inertial residuals are combined in the cost function, which will be solved as a minimization problem for each iteration.

In Figure 3.1, visual and inertial pipelines of the visual inertial SLAM systems are represented separately by given the details about the inputs and the outputs of each module. Moreover, in each module examples about the methods of each module from literature are also given.

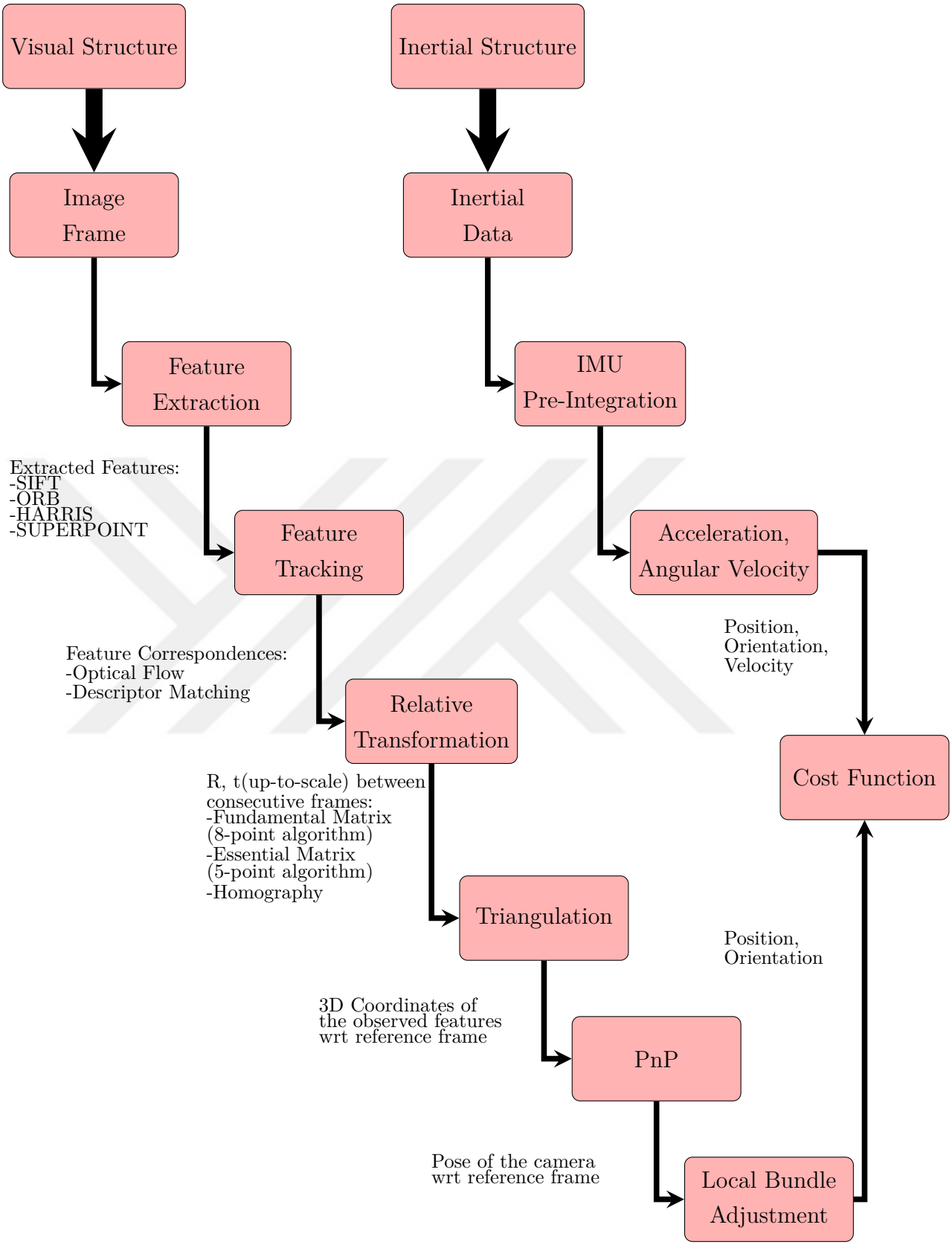


Figure 3.1 Tightly-Coupled Fusion

3.1 Inertial Measurements

In VINS, the IMU sensor readings are integrated for time period between two visual frames. This method is called IMU-preintegration. In IMU pre-integration, velocity, position, rotation changes of the IMU frame are calculated between the time interval of taking two consecutive visual frames. The quaternion-based IMU pre-integration is explained in detail and published in Shen et al. (2015). The acceleration and angular velocity calculation using raw gyroscope and accelerometer measurements are defined as:

$$\hat{a} = a_t + b_{a_t} + R_w^a g^w + n_a \tag{3.1a}$$

$$\hat{w} = w_t + b_{w_t} + n_w \tag{3.1b}$$

Where $\hat{a}$ is accelerometer measurement and the $\hat{w}$ is the gyroscope measurement. b_a, b_w terms represents the bias and the n_a, n_b represents the noise for both accelerometer and the gyroscope. Noises are stated as Gaussian white noise $n_a \sim N(0, \sigma^2)$, $n_w \sim N(0, \sigma^2)$. And derivative of respective noises are stated as the biases.

In the pre-integration part Forster et al. (2015), briefly, the position, velocity, and rotation differences between two consecutive frames are calculated as follows:

$$\alpha_{b_{k+1}}^{b_k} = \int\int_{t_k}^{t_{k+1}} R_t^{b_k}(\hat{a}_t - b_{a_t})\, dt^2 \tag{3.2a}$$

$$\beta_{b_{k+1}}^{b_k} = \int_{t_k}^{t_{k+1}} R_t^{b_k}(\hat{a}_t - b_{a_t})\, dt \tag{3.2b}$$

$$\gamma_{b_{k+1}}^{b_k} = \int_{t_k}^{t_{k+1}} 1/2\Omega(\hat{w}_t - b_{w_t})\gamma_t^{b_k}\, dt \tag{3.2c}$$

In order to find the position difference between two consecutive frames, in equation 3.2a double integration applied to calculated acceleration from raw accelerometer measurement and the bias and noises. α represents the position change in the time interval between two consecutive poses. In the same way, in equation 3.2b and 3.2c Qin et al. (2017), single integration applied to both acceleration and the angular velocity to calculate the velocity and rotation difference represented with β and γ respectively. The accelerometer and gyroscope measurements are integrated with the time interval of the two consecutive image frames taken from the monocular

camera.

3.2 Visual Measurements

The system uses both the visual and inertial measurements in order to get the relative orientation and location of the consecutive frames in the pose graph. Initially, there is no information about the world frame; therefore, c0, the camera frame while taking the first image, is stated as the reference frame for the estimator initialization. The system is highly non-linear; therefore, the initialization of the parameters plays a significant role in the system's robustness. Those parameters are the velocities of the body frame while taking the respective image frames, gravity vector and scale coefficient. In order to estimate those parameters, after obtaining the inertial and visual measurements, visual-inertial alignment is applied.

In obtaining the visual measurements, SfM(structure from motion) is proposed. As an outcome of the SfM, the relative orientation of the poses while taking the corresponding image frames is obtained. However, the scale coefficient could not be obtained through SfM. It is extracted from the visual-inertial alignment. In the SfM structure, the correspondent features of two consecutive frames are initially found with the KLT sparse optical flow algorithm. The relative orientation and translation between two consecutive keyframes are obtained with a five-point algorithm upon those correspondent points. After that, triangulation is applied to the feature detected on those two frames. The outcome of triangulation 3D locations of the features points obtained with respect to the camera coordinate frame. With known 3D locations of the features, with the P3P algorithm (perspective 3 points), the orientations and the locations of the individual poses of the frame could be obtained for the rest of the frames that observe the same set of features. In the end, bundle adjustment applied to minimize the reprojection error in all poses in the pose graph. This structure gives the initial estimation of the orientations and locations of the nodes in the pose graph. Based on this initial estimation of the poses, the system continues with the visual-inertial fusion.

$$q_{b_k}^{c_0} = q_{c_k}^{c_0} \otimes (q_c^b)^{-1} \tag{3.3a}$$

$$s\bar{p}_{b_k}^{c_0} = \bar{p}_{c_k}^{c_0} \otimes R_{b_k}^{c_0} p_c^b \tag{3.3b}$$

In equations 3.3a and 3.3b Qin et al. (2017), where q_c^b and p_c^b are rotation and translation from camera frame to body frame, those matrices are fixed due to the rigid design of the platform. The connection between the IMU(body frame) and the camera is fixed. s represents the scale coefficient that will be obtained from the visual-inertial alignment. $q_{c_k}^{c_0}$ and $\bar{p}_{c_k}^{c_0}$ are obtained from the SfM. They are used to obtain translation and rotation from the current body frame to the initial camera frame.

3.3 Visual-Inertial Alignment

In visual-inertial alignment, the gyroscope bias is calibrated, and the parameter vector is estimated. The parameter vector contains the velocities in the body frame while taking the corresponding images, gravity vector, and scale coefficient. Both gyroscope bias calibration and parameter vector estimation are solved with fusing visual and inertial measurements.

For the gyroscope bias calibration, relative rotation obtained from the SfM and the rotation difference γ obtained from the IMU preintegration are used to estimate the gyroscope bias. They are used in the cost function described below in equation 3.4a Qin et al. (2017). The expected norm of this quaternion tensor product is 1. Therefore tensor product solved as minimization problem to find the change in the bias. After the gyroscope bias is found, the rotation difference γ linearized with found gyroscope bias by multiplying it with its Jacobian 3.4b Qin et al. (2017).

$$\min_{\delta b_w} \sum_{k \in B} \| (q_{b_{k+1}}^{c_0})^{-1} \otimes q_{b_k}^{c_0} \otimes \gamma_{b_{k+1}}^{b_k} \|^2 \tag{3.4a}$$

$$\gamma_{b_{k+1}}^{b_k} \approx (\hat{\gamma})_{b_{k+1}}^{b_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}(\mathbf{J})_{b_w}^{\gamma} \delta b_w \end{bmatrix} \tag{3.4b}$$

Using the new estimated gyroscope bias, the α, β and γ values are calculated again as in the preintegration equations 3.2a, 3.2b and 3.2c.

For the state vector, in other words, parameter vector, the position, velocity, and rotation difference are calculated with the obtained relative orientation vectors from the SfM. Then minimization problem structured by subtracting the position, velocity, orientation values obtained from IMU preintegration and calculated values with SfM outcomes as in equation 3.5b and 3.5c Qin et al. (2017). This linear minimization problem is solved for the parameter vector. Initial parameter vector is stated

in 3.5a Qin et al. (2017).

$$X_I = \left[\; v_{b_0}^{b_0}, ..., g^{c_0}, s \; \right] \tag{3.5a}$$

$$\alpha_{b_{k+1}}^{b_k} = R_{c_0}^{b_k}(s(\bar{p}_{b_{k+1}}^{c_0} - \bar{p}_{b_k}^{c_0}) + \frac{1}{2}g^{c_0}{\Delta t_k}^2 - R_{b_k}^{c_0}v_{b_k}^{b_k}\Delta t_k) \tag{3.5b}$$

$$\beta_{b_{k+1}}^{b_k} = R_{c_0}^{b_k}(R_{b_{k+1}}^{c_0}v_{b_{k+1}}^{b_{k+1}} + g^{c_0}\Delta t_k - R_{b_k}^{c_0}v_{b_k}^{b_k}) \tag{3.5c}$$

$$\hat{Z}_{b_{k+1}}^{b_k} = \begin{bmatrix} \alpha_{b_{k+1}}^{b_k} \\ \beta_{b_{k+1}}^{b_k} \end{bmatrix} = \mathbf{H}_{b_{k+1}}^{b_k}X_I + n_{b_{k+1}}^{b_k} \tag{3.5d}$$

$$\min_{X_I} \sum_{k \in B} ||\hat{Z}_{b_{k+1}}^{b_k} - \mathbf{H}_{b_{k+1}}^{b_k}X_I||^2 \tag{3.5e}$$

Position and velocity change in between two consecutive frames are calculated in equations 3.5b and 3.5c Qin et al. (2017) respectively with the rotations and translations from SfM. $\hat{Z}$ vector contains the position and velocity change calculated from the IMU pre-integration equations. The expectation is both calculations to give the same results and at the and their subtraction is equal to 0 in zero noise, drift case. Therefore minimizing equation 3.5e Qin et al. (2017) gives us the estimation of elements of initial parameter vector X_I .

3.3.1 Initialization In VINS and ORB-SLAM3

In order to determine the state vector before the estimator process of the VINS, the initialization process plays a crucial role in bootstrapping the VINS. In the initialization process, the velocity, direction of the gravity vector, sensor biases are aimed to be stated. Based on this state vector, world frame also stated as an outcome of the initialization process.

In Vins-Mono Qin et al. (2017), the state vector contains the direction of the gravity, the velocity of the body frames in consecutive time stamps, and IMU biases. The initialization procedure is based on loosely coupled sensor fusion of the monocular camera and the IMU. Linear least square function formulated based on Newton kinematics and minimization problem solved for the state vector with the Gauss-Newton method. The fundamental matrix approach is used to get the relative transformation between two consecutive frames in the vision part. In order to determine individual poses in the current sliding windows, triangulation followed by the PnP method is applied. The vision process and linear cost function for VINS-MONO are mentioned in previous reports; therefore, it is not described in that report in detail.

In ORB-SLAM3 Campos et al. (2020), similar to the VINS-MONO, the initializa-

tion procedure is based on formulating a linear cost function to determine the state vector. Unlike the VINS-MONO, in ORB-SLAM3, they are not using the fundamental matrix approach to get the relative transformation between the consecutive images in the fixed sliding window. Their novel approach has the mechanism to determine whether the environment for initialization is proper for homography or fundamental matrix. In a different thread, with using the 8-point algorithm and DLT for fundamental matrix and homography, respectively. As a result of those algorithms, the score is calculated to decide which one to use in the initialization. It is called automatic initialization procedure in their past work Burri et al. (2016).

$$x_c = H_{cr} x_r \tag{3.6a}$$

$$x_c^T F_{cr} x_r = 0 \tag{3.6b}$$

In equation 3.6a and 3.6b Campos et al. (2020) F and H represents the fundamental and homography matrix that calculated in the automatic initialization procedure in ORB-SLAM3 Campos et al. (2020).

$$S_{H,F} = \sum_i (p_M(d_{cr}^2(x_c^i, x_r^i, M)) + p_M(d_{rc}^2(x_c^i, x_r^i, M))) \tag{3.7}$$

In equation 3.7 Campos et al. (2020) d_{cr}^2 and d_{rc}^2 are symmetric transfer errors between corresponding frames. S_H and S_F represent the score values for the homography and fundamental matrix. Indices c and r represents the corresponded frames in the window.

$$R_H = \frac{S_H}{S_H + S_F} \tag{3.8}$$

In the automatic map initialization model, selection of whether homography or fundamental matrix made by checking the R H value that calculated in equation 3.8 Campos et al. (2020). R_H value is calculated in equation 3.8 with the score values of the homography and fundamental matrices which calculated in the equation 3.7. In the decision model, if the computed R_H value is bigger than 0.45, then the homography approach is used because it means that the scene is planar, nearly planar, or low texture or low parallax. In contrast, if the R_H value is calculated as smaller than 0.45, then it means that there is enough parallax. In that case, the fundamental approach gives better results.

In real-world tests of the VINS-MONO, I observed that the estimator could not be initialized due to a low parallax issue in the observed environment. The lack of compatibility to the planar or low parallax scenes of the VINS-MONO algorithm in the initialization, based only on the fundamental matrix, 5-point algorithm, prevents the algorithm estimator from accurately initialized. On the other hand, thanks to the ORB-SLAM3 algorithm's automatic map initialization model, which decides whether to use homography or fundamental matrix according to the environment, it resolves low parallax by selecting using homography in such planar environments. In the next step ORB-SLAM3 aimed to test with the same bag file which used to be tested the VINS-MONO.

3.4 Tightly Coupled Monocular Visual Inertial Odometry

Tightly coupled monocular VIO is based on the non-linear minimization problem where it is modeled in Figure 3.2. The structure is similar to the estimator initialization part. The state vector contains the IMU states, rotation and translation between camera and IMU frame, the inverse distance of the last observation of a specific feature, and its first observation. The state vector is estimated by minimizing the sum of the prior, visual measurement residual and the inertial measurement residual.

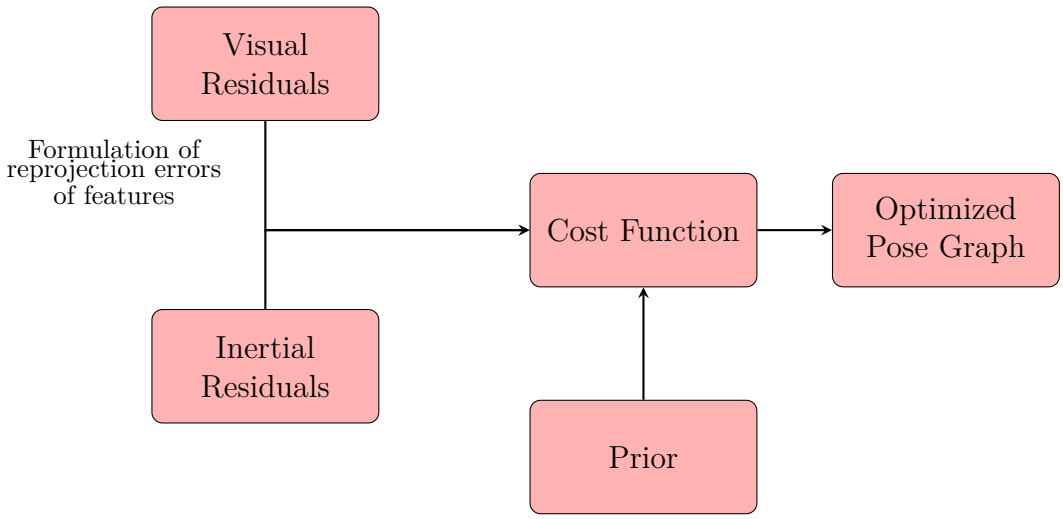


Figure 3.2 Tightly-Coupled Fusion

The visual measurement residual is obtained by calculating the feature location from its first observation to current observation—this calculation is made with the rotation and translation matrices obtained from SfM. For instance, the i^{th} feature initially observed in frame k, and the current frame is j. The rotation and translation from k^{th} frame to j^{th} frame applied to that feature location in k^{th} frame to find

its location in j^{th} frame. After the location is estimated via this described method, residual is found by subtracting this estimated location from the observed i^{th} feature location in j^{th} frame.

$$r_C(\hat{Z}_l^{c_j}, X) = \begin{bmatrix} b_1 & b_2 \end{bmatrix}^T . \left(\hat{\bar{P}}_l^{c_j} - \frac{P_l^{c_j}}{||P_l^{c_j}||} \right) \tag{3.9a}$$

$$\hat{\bar{P}}_l^{c_j} = \pi_c^{-1} \left(\begin{bmatrix} \hat{u}_l^{c_j} \\ \hat{v}_l^{c_j} \end{bmatrix} \right) \tag{3.9b}$$

$$P_l^{c_j} = R_b^c \left(R_w^{b_j} \left(R_{b_i}^w \left(R_c^b \frac{1}{\lambda_l} \pi^{-1} \left(\begin{bmatrix} \hat{u}_l^{c_i} \\ \hat{v}_l^{c_i} \end{bmatrix} \right) + p_c^b \right) + p_{b_i}^w - r_{b_j}^w \right) - p_c^b \right) \tag{3.9c}$$

λ_l represents the inverse distance of the feature l from its first observation to its current observation. π is the back projection from pixel coordinates. Equation 3.9c Qin et al. (2017) calculates the translation of feature l from its initial observation to the current frame j with translations and rotations obtained from SfM. The inertial measurements are obtained by subtracting the estimated position, velocity, and rotation differences from relative orientations obtained by the SfM from the observed values taken from the IMU preintegration process. The equations are similar as they are in the estimator initialization part. Equation 3.9a Qin et al. (2017) states the visual measurement residual which calculated with subtracting the projected feature that calculated in 3.9c from observed feature location obtained from equation 3.9b Qin et al. (2017).

$$r_B(\hat{Z}_{b_{k+1}}^{b_k}, X) = \begin{bmatrix} \delta\alpha_{b_{k+1}}^{b_k} \\ \delta\beta_{b_{k+1}}^{b_k} \\ \delta\theta_{b_{k+1}}^{b_k} \\ \delta b_a \\ \delta b_g \end{bmatrix} = \begin{bmatrix} R_{b_{k+1}}^{b_k} \left(p_{b_{k+1}}^w - p_{b_k}^w + \frac{1}{2} g^w \Delta t_k^2 - v_{b_k}^w \Delta t_k \right) - \hat{\alpha}_{b_{k+1}}^{b_k} \\ R_w^{b_k} \left(v_{b_{k+1}}^w + g^w \Delta t_k - v_{b_k}^w \right) - \hat{\beta}_{b_{k+1}}^{b_k} \\ 2 \left[q_{b_k}^{w^{-1}} \otimes q_{b_{k+1}}^w \otimes (\hat{\gamma}_{b_{k+1}}^{b_k})^{-1} \right]_{xyz} \\ b_{ab_{k+1}} - b_{ab_k} \\ b_{wb_{k+1}} - b_{wb_k} \end{bmatrix} \tag{3.10a}$$

Where r_B stated the inertial measurement residual. After both inertial and visual measurement residuals are calculated, they are solved together in a non-linear minimization problem to find the state vector X.

$$X = [x_0, x_1, \ldots, x_n, x_c^b, \lambda_0, \ldots, \lambda_m] \tag{3.11}$$

Where x is the IMU states and the λ is the inverse distance from the l^{th} feature first observation to the location in current observation. m is the number of the features, and the n is the number of the pose states.

3.5 Marginalization

The marginalization procedure is applied to increase the efficiency of the algorithm. Increasing the efficiency is directly related to how the algorithm becomes closer to run in real-time. In such applications, real-time feedback to the platform controller is very crucial in autonomous systems. In this work, there are two cases of marginalization. If the previous keyframe's previous frame is keyframe, then the initial keyframe of the measurements of the sliding windows removed and becomes prior. On the other hand, if the previous frame of the latest keyframe is not a keyframe, its visual measurements are removed, but IMU measurements are still kept. Because IMU measurements are continuous in time intervals, they should be considered in the algorithm if the frame is not counted as a keyframe.

3.6 Dynamic Object Elimination in SfM and Fundamental Matrix

In this chapter, dynamic object elimination approach in visual SLAM is discussed. Initially points that extracted from the dynamic object is eliminated. In the scope of this thesis, candidate dynamic object stated as human. In further studies these candidates would be increased by training more objects in Yolo Bochkovskiy et al. (2020). Results are also validated with the fundamental matrix calculation upon the correspondence points.

3.6.1 Dynamic Object Elimination

In order to increase the robustness of the visual-inertial slam system, dynamic objects are aimed to eliminate from the scene. Elimination is described as eliminating the detected features on the dynamic objects in the current image frame. Those eliminated features are not considered in the feature correspondences while calculating the fundamental matrix.

In the initial design of this project, humans and animals in the environment are counted as dynamic objects. For further versions, the variety of the objects could be extended. In order to keep the efficiency of the algorithm to process in near real-time, the dynamic objects that have the low possibility of being in the daily

environments are not considered in this project.

YoLo Bochkovskiy et al. (2020) framework is used to detect the dynamic objects in the environment. As an output of the YoLo framework, the bounding boxes of the detected objects are given. Based on those bounding boxes' coordinates, the features detected by the harris corner detector labeled as to whether they are locating in one of those bounding boxes or not. The features that stay inside of those bounding boxes are not considered. The fundamental matrix is calculated with the rest of the features that have correspondences with consecutive frames.

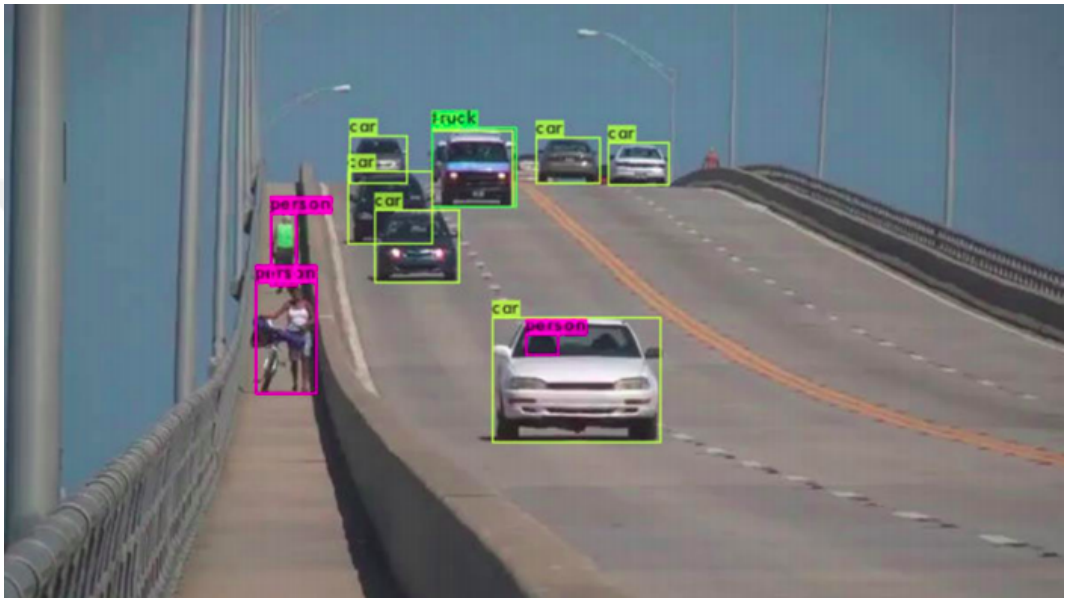


Figure 3.3 YoLo framework output for a single frame

Illustration 3.3 shows the output of the YoLo framework with a pre-trained network and weighs as parameters to the object detection. For further steps, in order to detect the custom desired objects, a custom network could be designed to get the weights to detect those objects. As mentioned, the animals and the humans are counted as the dynamic objects at first hand in this project. Therefore the bounding box coordinates of the humans in this specific single frame scene are taken as an output from the YoLo framework. For this specific frame, the YoLo output for bounding boxes are shown below:

Table 3.1 Detection Output

Type	Output
person	81% (left$_x$: 155top_y : 169$width$: 35$height$: 66)
person	96% (left$_x$: 163top_y : 127$width$: 15$height$: 43)
car	90% (left$_x$: 209top_y : 103$width$: 50$height$: 43)
car	95% (left$_x$: 211top_y : 82$width$: 34$height$: 27)
car	99% (left$_x$: 225top_y : 126$width$: 50$height$: 42)
truck	82% (left$_x$: 259top_y : 76$width$: 50$height$: 48)
car	34% (left$_x$: 260top_y : 79$width$: 48$height$: 45)
car	99% (left$_x$: 296top_y : 189$width$: 99$height$: 74)
person	60% (left$_x$: 315top_y : 200$width$: 19$height$: 12)
car	85% (left$_x$: 322top_y : 83$width$: 38$height$: 27)
car	89% (left$_x$: 365top_y : 86$width$: 36$height$: 25)

Figure 3.4 Detected Harris Corners Before Elimination

Figure 3.4 shows the extracted Harris features without eliminating the features detected on dynamic objects (human or animal).

Figure 3.5 Detected Harris Corners After Elimination

Illustration 3.5 shows the Harris features after eliminating the features detected on dynamic objects.

After obtaining the final set of harris features, the KLT feature tracker was applied to track those features on the consecutive image frames. To find the relative orientation and up to scale translation between those consecutive frames, one approach is using a fundamental matrix. In this report, I have focused on the fundamental matrix where the cameras are not calibrated. The fundamental matrix is calculated based on the correspondences from two consecutive frames. After calculating the fundamental matrix, the calculated fundamental matrix is also verified with those correspondent features. This verification is made by multiplying the vectors from image coordinates to specific feature points on the world frame with the baseline vector. This multiplication should give '0′ because of the coplanarity constraint. There are 3 vectors extracted that make a planar surface, and their triple scalar product should be 0 in theory. Two of those vectors are vectors from the camera projection centers to the corresponding point in the world frame, and the third is between the camera projection centers. Direction vectors are represented as;

$$^{n}x' = (R')^{-1}(K')^{-1}x' \tag{3.12}$$

Where $^{n}x'$ is the normalized direction vector, R is the rotation, and K is the calibration matrix of the first camera. x' is the image coordinates of the correspondence point in the first frame. In the same way, the normalized direction vector of the second camera is extracted. Those two vectors are the two elements of the mentioned

triple scalar product which will be equal to 0 because of the coplanarity constraint.

$$b = X' - X'' \tag{3.13}$$

Where b is the baseline between the camera projection centers. X', X'' represents the camera projection centers of two cameras that took the corresponding frames. b vector is the final element of the triple scalar product that makes a planar surface with the normalized direction vectors. Then, according to the coplanarity constraint;

$$[{}^n x' b {}^n x''] = {}^n x' . (b \times {}^n x'') = {}^n x'^T S_b {}^n x'' = 0 \tag{3.14}$$

When we put the equation for normalized direction in the equation 3.12 into the equation 3.14, the fundamental matrix could be found as,

$$F = (K')^{-T} R' S_b R''^T (K'')^{-1} \tag{3.15}$$

Where K, K are calibration matrices of the cameras. In the fundamental matrix approach, the cameras are not calibrated, and in my case, both calibration matrices are equal because we are using the same camera to capture all frames. R, R are the rotation vectors from the corresponding camera projection center to the $3D$ feature point in the world frame. Finally, S_b represents the baseline vector from the first camera projection center to the second one.

To verify the results with the calculated fundamental matrix, I have initially converted the corresponding point sets into a homogeneous coordinate system. After that, I have multiplied the transpose of the respected correspondence point from the first frame with the fundamental matrix that I have found. After that, I have multiplied this resulted matrix with the respected correspondence point in the second frame. I have checked the results whether they are close to '0′ or not. I observed that the equation gave me numbers very close to the '0′ for all correspondence points in the set. The numbers below stated the results of the multiplication for correspondence points in the set.

$$(x_1)^T F x_2 = 0 \tag{3.16}$$

Where x_1 and x_2 represent corresponding image points respectively and F represents the fundamental matrix.

Table 3.2 Fundamental Matrix Check on Correspondence Points

$x'Fx$
-0.04851716756820679
-0.37185138463974
-0.2456498295068741
-0.1752408146858215
-0.2976421415805817
-0.3518350124359131
-0.4493158459663391
-0.1280926913022995
-0.1153489872813225
-0.5757604241371155
-0.3942645490169525
-0.1491507142782211
-0.2330176681280136
-0.2046927213668823
-0.231176570057869
-0.4219972193241119
0.008003597147762775

Figure 3.6 Tracked Correspondent Points with KLT Optical Flow

Illustration 3.6 shows the tracked eliminated correspondent points in two consecutive frames from the video stream input.

4. PROPOSED VISUAL-INERTIAL SYSTEM STRUCTURE

In this chapter, proposed end-to-end visual inertial SLAM structure is presented and discussed in detail by each module and the relations between these modules. General structure and the relations between the modules are also represented in Figure 4.1

4.1 System Overview

Proposed visual inertial system divided into two main modules which are visual and the inertial. In visual module there structure from motion system is implemented with the keyframe selection. Detailed explanations about each module in Figure 4.1 are given in the following sections.

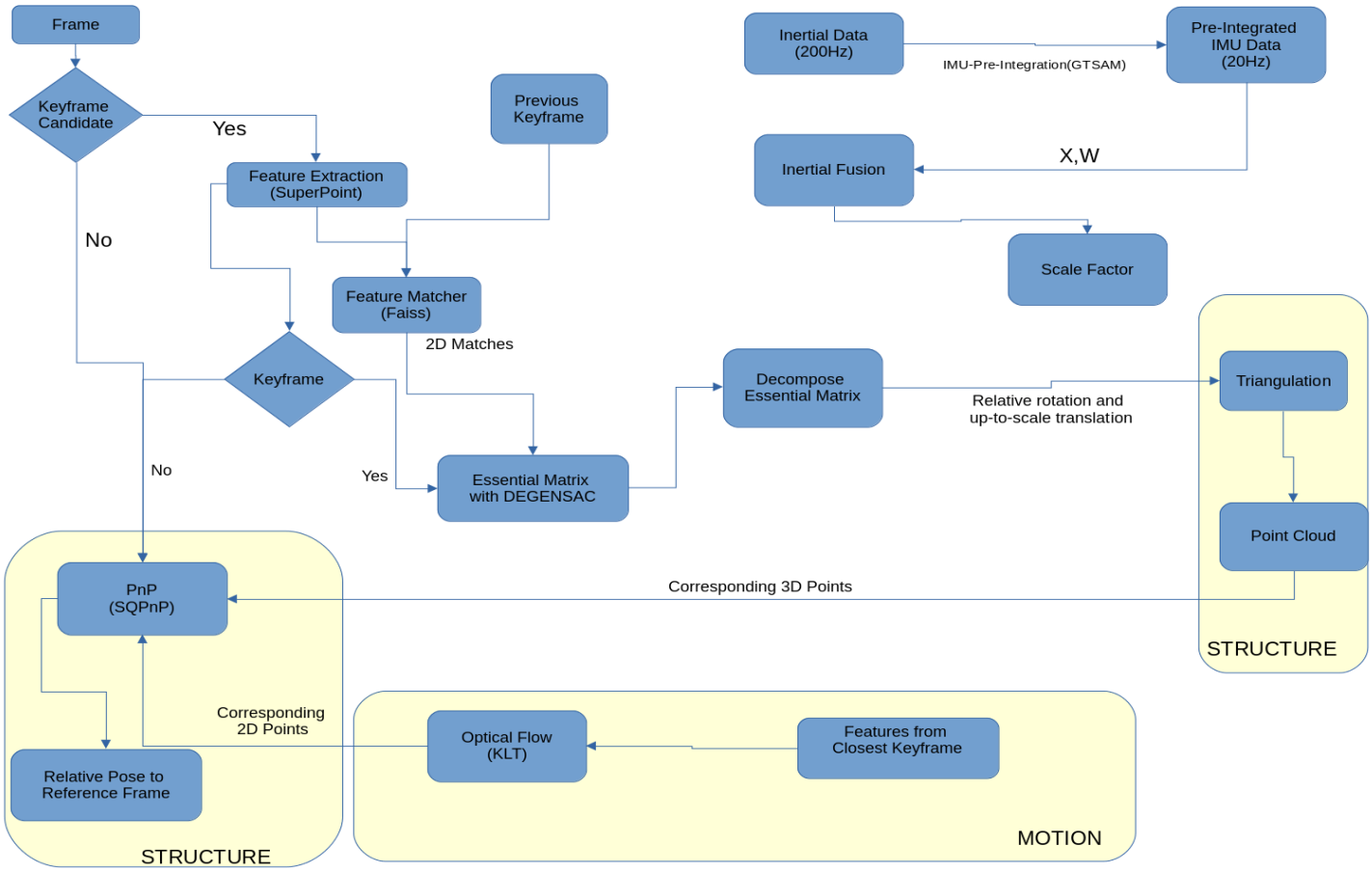


Figure 4.1 System Overview

4.1.1 Visual Module

In the vision section of the structure, pose trajectory is built with respect to the zeroth frame. Zeroth frame is set as reference frame of the system and also it directly set as the first keyframe. Pipeline structured based on the keyframe system. By selecting keyframes to process, it will end up with more efficient and close to real time system.

In proposed system keyframes are selected in two steps. Initially keyframe candidates are selected in a fixed time period between the frame sequence. After the selection of the keyframe candidates, Superpoint features DeTone et al. (2018) are extracted from each of the keyframe candidates. Extracted Superpoint features from a candidate keyframe is represented in Figure 4.2.

Figure 4.2 Extracted Superpoint Features

Superpoint feature extraction is configurable in terms of the uniquness of the feature, feature quantity and the nms. In the current structure Superpoint extraction configuration is as follows; 2000 point per keyframe candidate, keypoint threshold which describes the keypoint quality in Superpoint side set as 0.005 (bigger value is giving more qualified points), and the nms radius is set to 2 which determined the how much that the extracted keypoints could be close to each other in the current frame.

The extracted keypoints are stored by their respected keyframes ids and point ids in order to use them in further processes as triangulation and optimization. After feature extraction, by using Faiss matcher Johnson et al. (2017), consecutive keyframe candidates' features are matched and besed on that match, the ratio of number of matched keypoints to the total extracted keypoints is calculated. Feature matching with Faiss is represented in Figure 4.3.

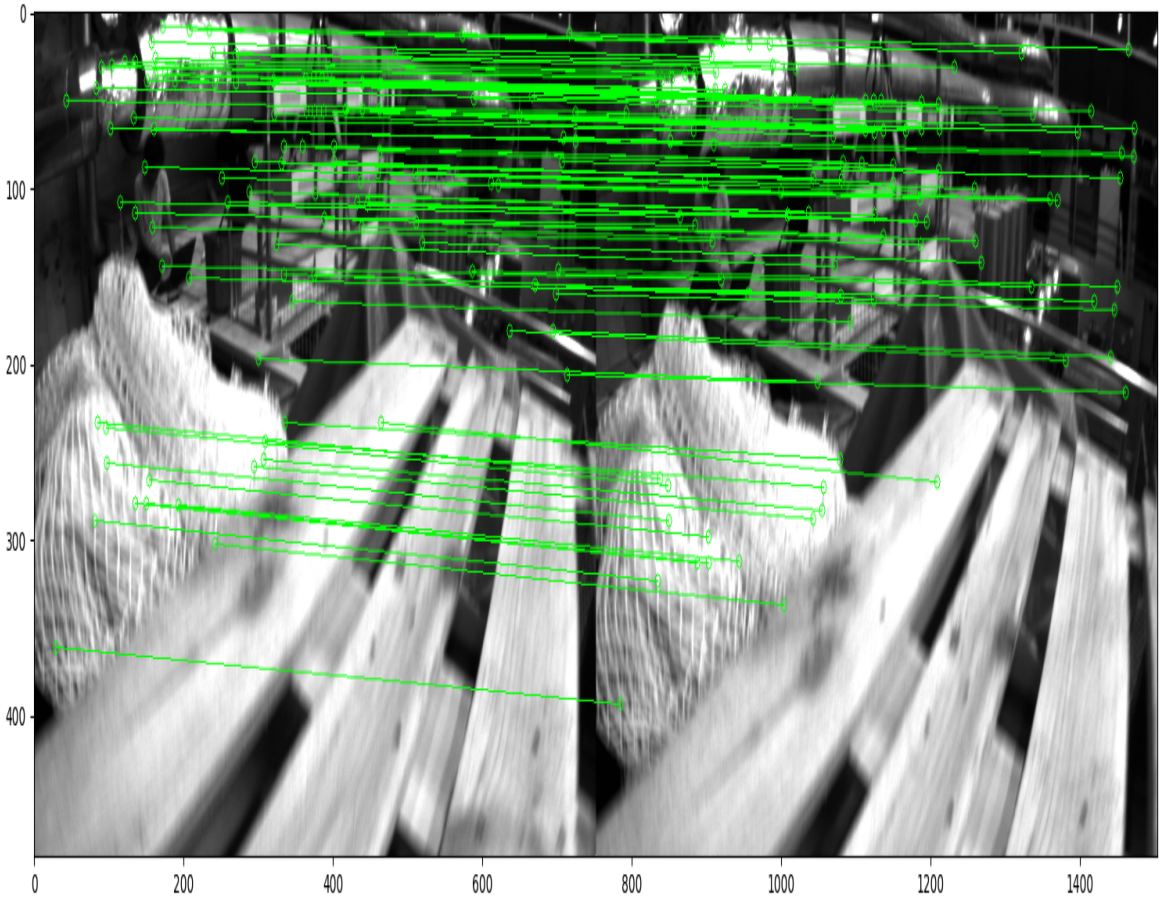


Figure 4.3 Feature Matches with Faiss

Considering the stated threshold which is %75 in the current pipeline, if the calculated ratio is above that threshold then that means the current candidate keyframe are very close to the previous one and there is no need to select and process it as keyframe. This approach is also helps to eliminate rotation only motion issues while calculating the essential matrix in that circumstances. Poses in rotation only motions will be evaluated as ordinary frame and the pose will calculated with the SQPNP Terzakis & Lourakis (2020).

After the keyframe selection process, relative motion between the keyframes are calculated. Initially, with given camera intrinsics essential matrix between the consecutive keyframes are calculated. In order to eliminate the outliers in the process of calculating essential matrix, MAGSAC Barath & Matas (2018) used as a scoring method which scores points in terms of likelihood to become an inlier. Calculated essential matrix and the inlier points are used to get relative rotation and up-to-scale translation by decomposing the essential matrix.

In order to keep the trajectory other than relative rotation and translation matrices, rotation and translations with respect to the reference frame are also need to be calculated. For the current rotation matrix, dot product of previous rotation matrix and the current relative rotation matrix is calculated as in equation 4.1.

$$\vec{curr\_rot} = \vec{rot\_pre} \cdot \vec{relative\_rot} \tag{4.1}$$

In the same manner, for the current translation, previous translation vector is added to the multiplication of scale factor with the dot product of the previous rotation vector with the current relative translation vector as shown in the equation 4.2.

$$\vec{curr\_t} = \vec{t\_pre} + scale\_factor * (\vec{rot\_pre} \cdot \vec{relative\_t}) \tag{4.2}$$

In order to calculate relative scale directly from the vision pipeline, by setting the zeroth frame as reference frame for calculating the relative scale, matched points between consecutive keyframes are trinangulated. Based on two seperate point clouds, the common points are extracted and ramdomly selected point pairs 3D location distance ratios are used for calculating the relative scale between the keyframes. That section is really noise dependent and in order to prevent huge miscalculations there is another step as thresholding the calculated relative scale.

In order to calculate the poses of the non-keyframe frames, 3D point cloud structure is used. Extracting Superpoint feature for each frame is a very high cost operation,

therefore in order to obtain features from the non-keyframe frame, points are tracked with KLT optical flow from the closest keyframe to the current non-keyframe. As the extracted Superpoint features are already recorded by their ids and the keyframes, when they tracked in the optical flow which provide 2D location information of the respected feature points. At the same time by checking the ids of those points, 3D locations of respected feature can also be obtained from the triangulated point cloud. At the end, we have 2D-3D correcpondences of the feature in an image, the pose of that frame is calculated with the SQPNP Terzakis & Lourakis (2020).

4.1.2 Inertial Module

The main structure of the inertial section is the IMU-Preintegration process. As a different type of sensors, Camera and IMU differentiates in terms of the data collection frequency. For a standard camera, the data collection frequency is around 20Hz while IMU sensor could collect data around 200Hz. In order to fuse the sensor outputs of IMU and the camera, the collected data need to be synchronized. The process of IMU-preintegration provides outputs from the IMU sensor in the same frequency with the camera.

In that section GTSAM(Georgia Tech Smoothing and Mapping) library Kaess (2015) is used. The process starts with integrating IMU acceleration and angular velocity measurements in a loop that iterates with time diff equal to the time interval between two consecutive IMU measurements. When it's reached to the camera frequency, relative transformation between initial state and the current state is predicted by also considering the IMU biases.

For the initial calibration of the IMU, based on the dataset documentation and the IMU specs, random white noise parameters are set as described; gyroscope noise density $1.6968e-04 rad/s/sqrt(Hz)$, gyroscope random walk $1.9393e-05 rad/s^2/sqrt(Hz)$, accelerometer noise density $2.0000e-3 m/s^2/sqrt(Hz)$, accelerometer random walk: $3.0000e-3 m/s^3/sqrt(Hz)$.

5. EXPERIMENTAL RESULTS

During the thesis, several different experiments are studied and reported. Proposed visual-inertial SLAM system are tested with the Euroc MAV dataset - MH01. Moreover physical hardware are set up for real world data collection and processing those datas with the state of the algorithms. Three main state of the art algorithms on the visual-inertial SLAM literature are tested and compared in terms of their capabilities and novelties. In addition, selected state of the art algorithms are studied by configuring their SLAM modules by implementations and the results are also reported. All those experiments are reported and discussed in the respective sections below.

5.1 Proposed Visual-Inertial SLAM System on Euroc-MAV Dataset

Machine Hall 01 Dataset from Euroc MAV dataset is processed in the proposed Visual-Inertial SLAM system. Trajectory output, 3DoF translation data output by comparing it with the ground truth measurements are reported in Figures 5.1, 5.2 and 5.3.

Based on the Superpoint and Faiss matcher configurations algorithm can handle around 30fps where the camera frequency in the Euroc dataset is collecting data at 20 fps. Efficient keyframe selection, and the GPU processes in feature extraction and matching makes the algorithm run at that level. Moreover using only inliers in the calculations such as SQPNP with RANSAC and calculating five point algorithm with RANSAC make the process considerably faster, because they are iteration based algorithms and using inliers as input to those algorithm makes the convergence time shorter.

Overall average inlier ratio in matched features is around 0.75 with MAGSAC. On the other hand in RANSAC implementation, the inlier ratio is more than 0.75 but calculated essential matrix not performing well as it is in the MAGSAC implementation. This result indicates that even inlier ratio is resulted higher in the RANSAC implementation, not all of them are real inliers.

Ground truth data of the Machine Hall 01 dataset is obtained from two devices Leica

and Vicon. Leica is working in 20Hz and Vicon is running in 100Hz. They aligned the measurements from Leica and Vicon to the 200Hz to make the measurement fit with the IMU measurements. IMU running at 200Hz and stereo camera that is used in Euroc Dataset is running at 20Hz. Therefore in order to get best fit with the groundtruth measurements which is in 200Hz and the algorithm outputs which is 20Hz, measurements are shifted but it is still not very accurately aligned with the groundtruth measurements.

Algorithm tested on the Machine Hall 01 dataset where is the camera already calibrated with the given intrinsics.

Table 5.1 Camera Intrinsics in Euroc Mac Machine Hall Dataset

Camera Model	Pinhole
Camera Intrinsics	
f_x	458.654
f_y	457.296
c_x	367.215
c_y	248.375
Camera Distortion Parameters	
k1	-0.28340811
k2	0.07395907
p1	0.00019359
p2	1.76187114e-05

There are 3682 images and 36383 IMU readings. Groundtruth measurements are post processed and aligned with the frequency of IMU measurements. Therefore in benchmarking and setting the comparison plots, groundtruth measurements aligned with camera frequency.

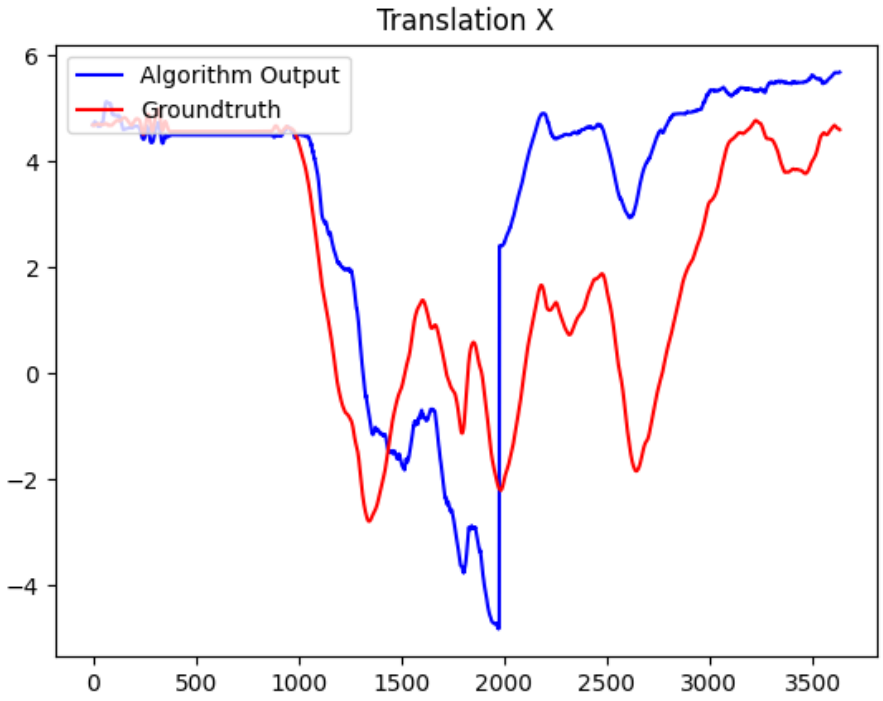


Figure 5.1 Translation X

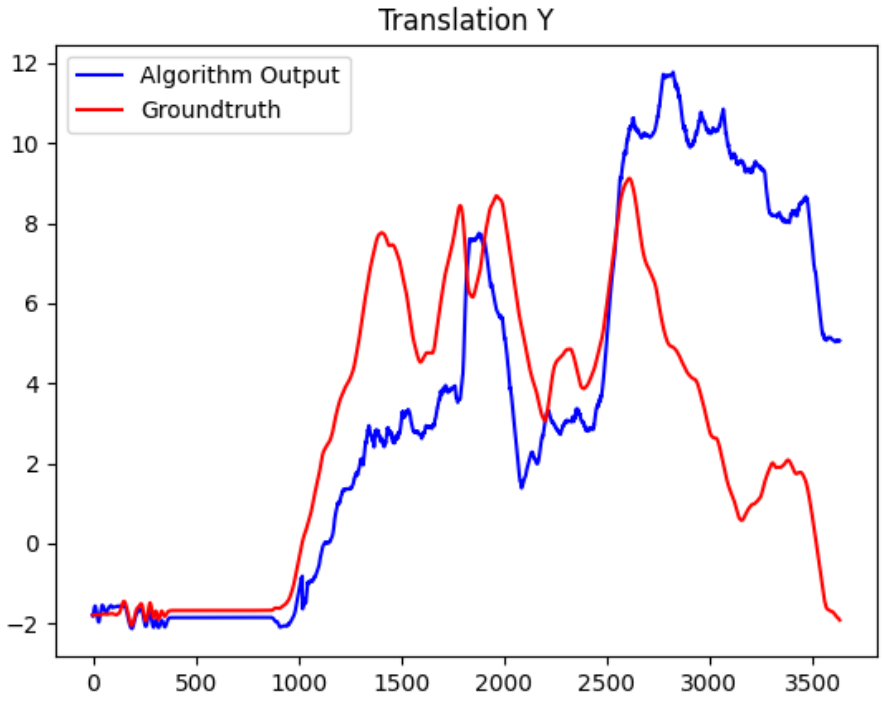


Figure 5.2 Translation Y

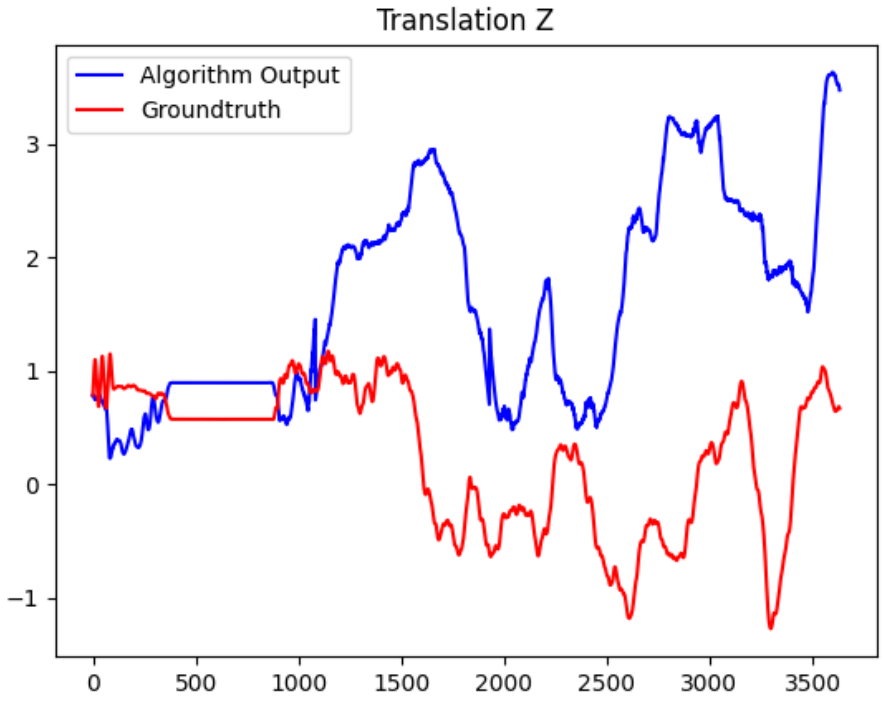


Figure 5.3 Translation Z

Figures 5.1, 5.2, 5.3 are the plots of comparison between the algorithm output and the groundtruth trajectory. The first thing that observed from the plots is that there is an increasing shift in the plots while it proceeds further. That may occur because of the cumulative error in the rotation in while calculating relative motion. Other than that there are some spikes in the plot which may occur because of the noise in matches or not well estimated essential matrix. Optimization on the algorithm in progress with testing more datasets. The additional results will be added to the document.

5.2 Hardware Test Platform Setup

In order to test state-of-the-art VINS algorithms Qin et al. (2017), Rosinol et al. (2020) and Campos et al. (2020) with manipulations in real world cases, the test platform has built with the NVIDIA Jetson Nano, mpu 9250 IMU, and the Raspberry Pi v2 camera. NVIDIA Jetson supports MIPI-CSI(Mobile Industry Processor Interface – Camera Serial Interface) cameras as Raspberry Pi 2. In order to get the raw data from the camera and the IMU, ROS(robot operation system) framework is used. In ROS, two different topics for each sensor are created, which are named as /cam and /imu, respectively. In the same way, in order to send and receive

sensor data, two different publisher and subscribers are written which publish and subscribe to topics /imu and /cam respectively.

In initial tests, it is observed that NVIDIA Jetson's processing power is not sufficient to process the VINS and visualize the output at the same time. Therefore, the structure is adjusted with additional processing power. In the new design, publishers run on the NVIDIA Jetson, to which the sensors are connected. On the other hand, subscribers are run on the laptop where the NVIDIA Jetson and the laptop connected through an ethernet network. The data reading frequency is set to 20Hz for the camera(Raspberry Pi v2) and 100Hz for the inertial sensor(bmu9250). In order to test the platform in an outside environment, bag files were created with publishing IMU and camera data into it. After data collection finished, those bag files processed offline with VINS in the laptop.

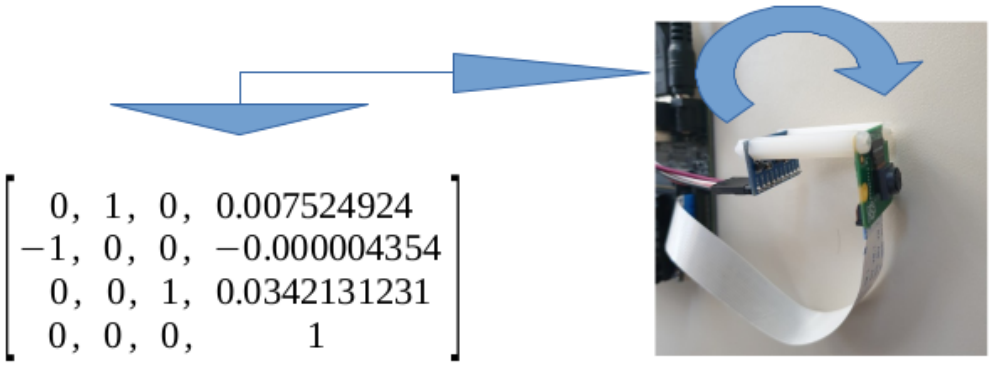
Initially camera calibrated with the algorithm Heng et al. (2013) which is built on the papers Heng et al. (2013), Heng et al. (2014) and Heng et al. (2014). Camera calibration output for Raspberry Pi v2 camera is given in Table 5.2

Table 5.2 Camera Intrinsics in Hardware Experiment

Camera Model	Pinhole
Camera Intrinsics	
f_x	1.1310328085080641e+03
f_y	1.2767137335706250e+03
c_x	3.2866580320967523e+02
c_y	2.2709250804519911e+02
Camera Distortion Parameters	
k1	2.4806261512816249e-01
k2	2.1941456723798769e-01
p1	-5.2838710043973868e-03
p2	-2.0383170940044690e-02

In order to read the IMU measurements, the rtimulib library [10] is used to get the IMU data through the I2C bus. In the VINS structure, there is a fixed transformation between the IMU and the camera connected to the platform. In the designed structure, the transformation matrix between the camera and IMU is represented in 5.4. Moreover topview of the complete hardware setup is shown in the Figure 5.5.

IMU to Camera Transformation Matrix:



$$\begin{bmatrix} 0, & 1, & 0, & 0.007524924 \\ -1, & 0, & 0, & -0.000004354 \\ 0, & 0, & 1, & 0.0342131231 \\ 0, & 0, & 0, & 1 \end{bmatrix}$$

Figure 5.4 Fixed IMU-Camera Structure

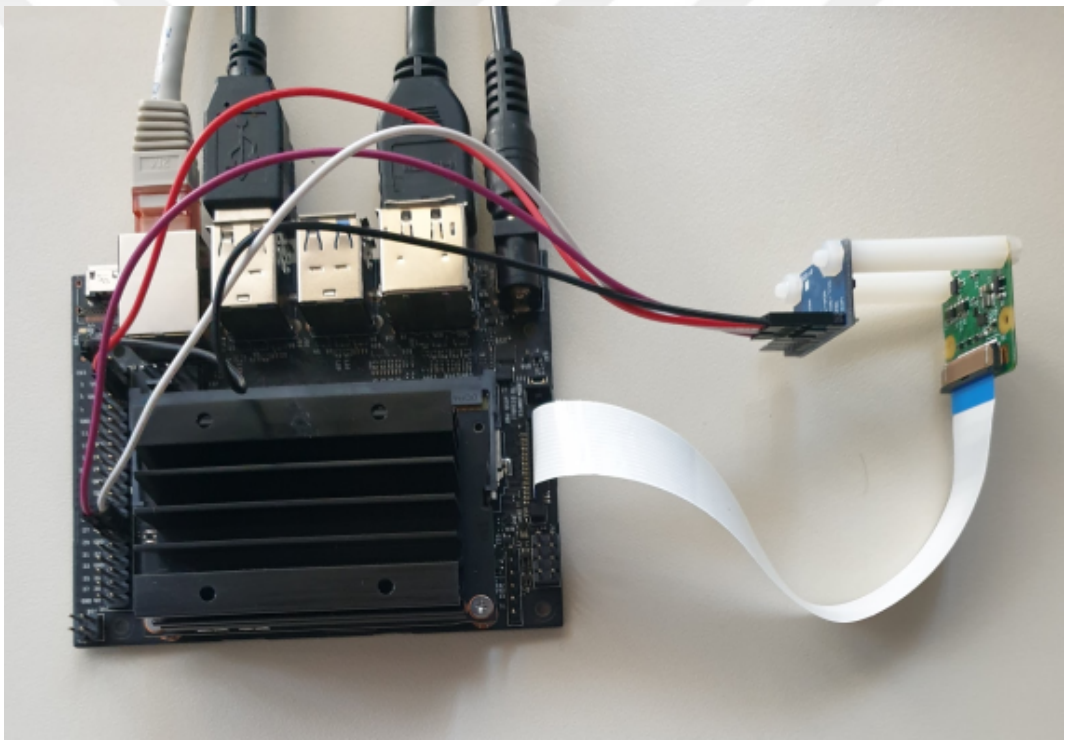


Figure 5.5 Hardware Test Platform Setup

5.2.1 Outputs

In this section, outputs of the experiments are given in detail with the evaluation metrics. Euroc Mav dataset Machine Hall sequences are used for the evaluation.

5.2.1.1 Euroc Mav Dataset

VINS-MONO algorithm works in high accuracy in the Euroc Mav dataset. These results are the initial observations on the public datasets and the real-world experiments with built hardware.

Table 5.3 EuRoC-Mav Dataset Specifications

Dataset	Year	Environment	Carrier	Cameras	IMUs	Time Sync	Ground Truth	stat/props
EuRoC-MAV	2016	indoors	MAV	1xstereogray 2x752x480@20Hz	ADIS1648 3-axis acc/ gyro@200Hz	hw	laser tracker pos@20Hz motioncapture pose@100 Hz acc 1mm	11 seqs,0.9km

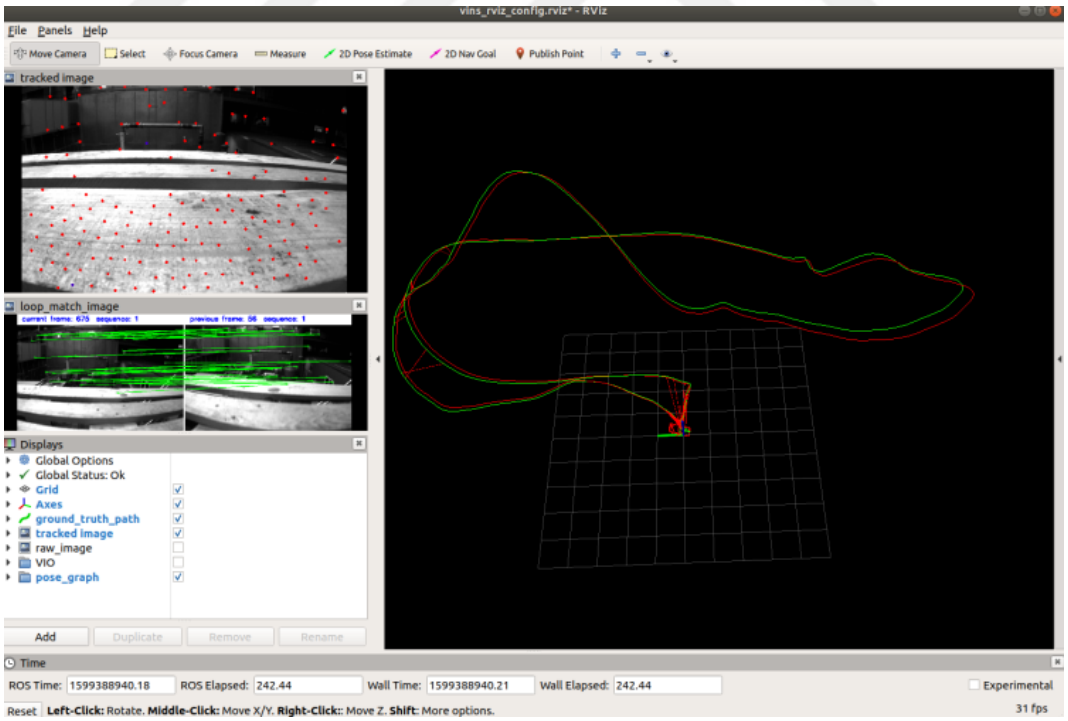


Figure 5.6 Euroc Mav - Machine Hall Dataset RVIZ Output of VINS-MONO

In Figure 5.6, green lines indicated the ground truth and the red line indicated the estimated pose graph with the VINS-MONO algorithm.

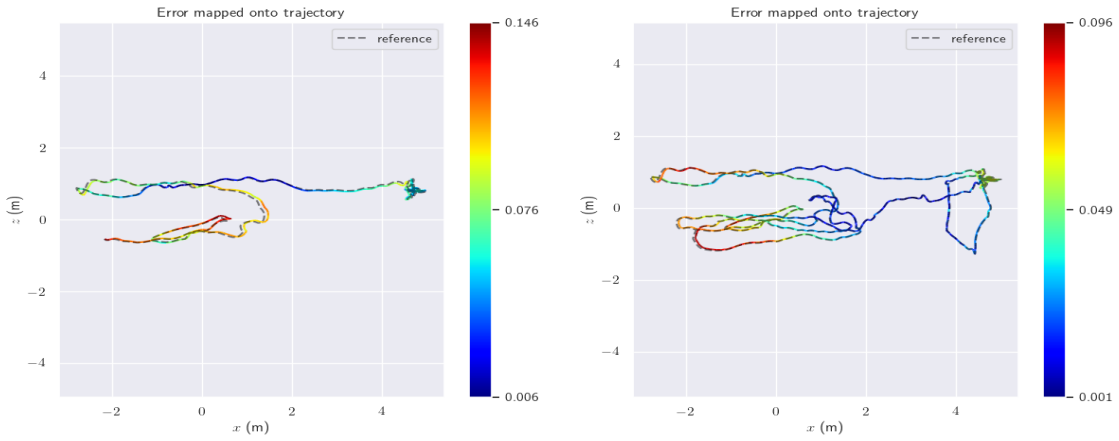
Evaluation Metric:

In order to evaluate the visual-inertial SLAM algorithms, absolute pose error (APE) is calculated, and those algorithms are compared upon the calculated absolute pose error in the trajectory of different datasets. Absolute pose error is calculated as getting the difference between the calculated pose outcome from the visual-inertial algorithm and the ground truth pose stored in the respected dataset. Moreover, statistical values as mean, standard deviation, median, etc., upon calculated absolute pose error in the respected trajectory of the dataset are also presented in the results section below.

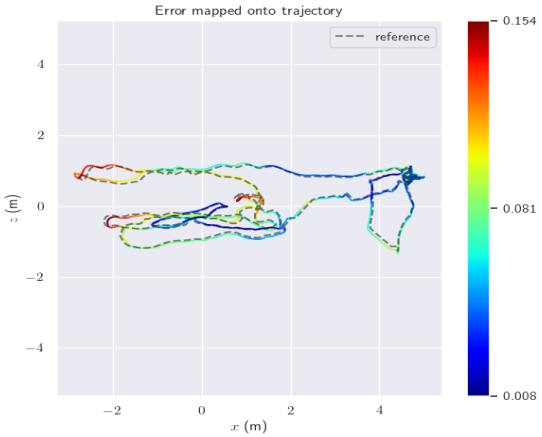
There are six different trajectories are evaluated with the visual-inertial algorithms Qin et al. (2017), Campos et al. (2020) and Rosinol et al. (2020). Datasets are separated as the environments where there are two different environments the datasets are collected, and the within those environments datasets are separated as easy, medium, and difficult. The increase in the difficulty level of the trajectories means more fast motions, motion blur, and quite low light conditions.

The results are presented as follows, for each environment, the trajectories are presented in figures with colored absolute pose errors in the trajectory with a respected algorithm. Moreover, the statistical values of those trajectories and absolute pose errors of all compared algorithms are shown in the same figure. Six different result sets are shown as described above in sequence.

Moreover, 6 trajectory of the Machine Hall sequences in the Euroc Mav Dataset Burri et al. (2016) are also tested with implemented configurations of the state of the art algorithms Vins-Mono and ORB-SLAM3. In Vins-Mono in the localization section, instead of the iterative PnP solution, more accurate solution to the perspective-n-point problem SQPNP is implemented and evaluated. In addition, in ORB-SLAM3 instead of the orb features, SuperPoint features are tested with the Machine Hall sequences.



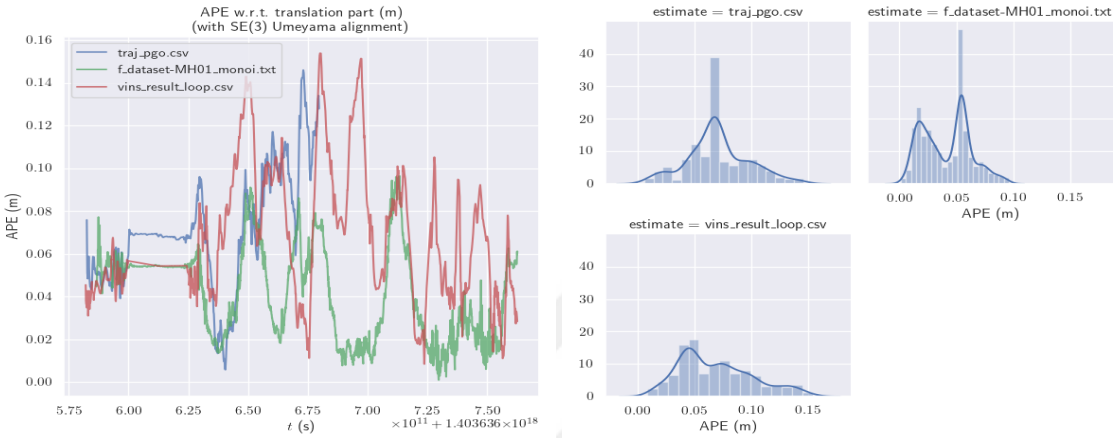
(a) Kimera (b) ORB-SLAM3



(c) VINS-MONO

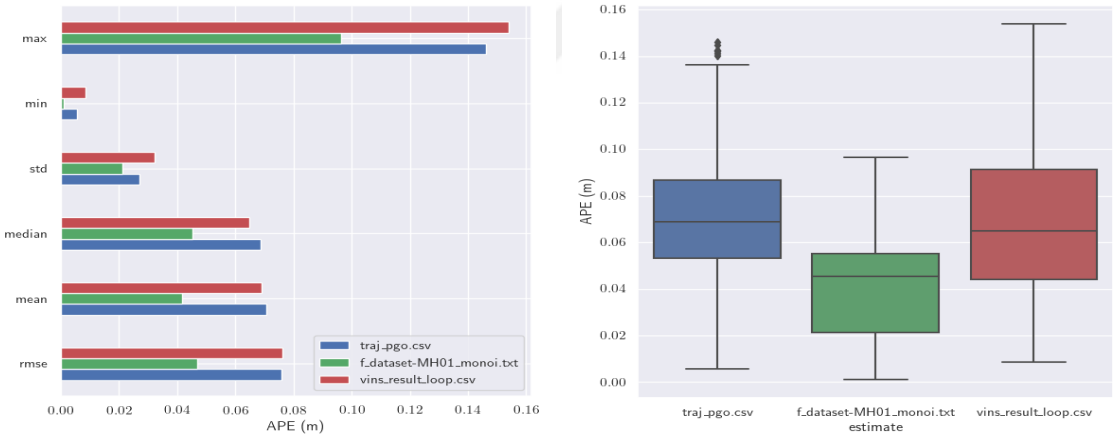
Figure 5.7 Trajectory error maps on MH01-Easy dataset in EuRoC MAV

In Figure 5.7, it can be observed that in Kimera, the trajectory in Machine Hall 01 sequence is partially estimated. Kimera's relocalization and mapping module is not well enough to complete the full trajectory in that sequence. ORB-SLAM3 and Vins-Mono have similar results in MH-01 sequence. Their respected APE's are 0.075 and 0.076 for that sequence.



(a) APE vs Time in MH01-Easy Dataset in EuRoC MAV

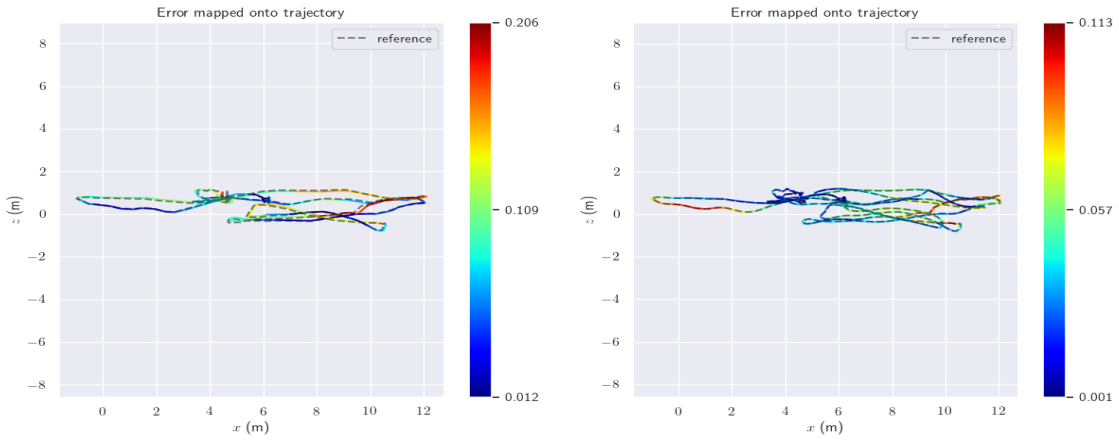
(b) Histogram of APE in MH01-Easy Dataset in EuRoC MAV



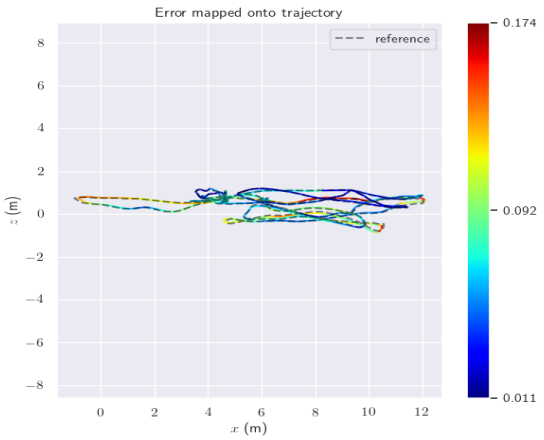
(c) Statistical Values on APE in MH01-Easy Dataset in EuRoC MAV

(d) Box Plot on APE in MH01-Easy Dataset in EuRoC MAV

Figure 5.8 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in MH01-Easy Dataset in EuRoC MAV



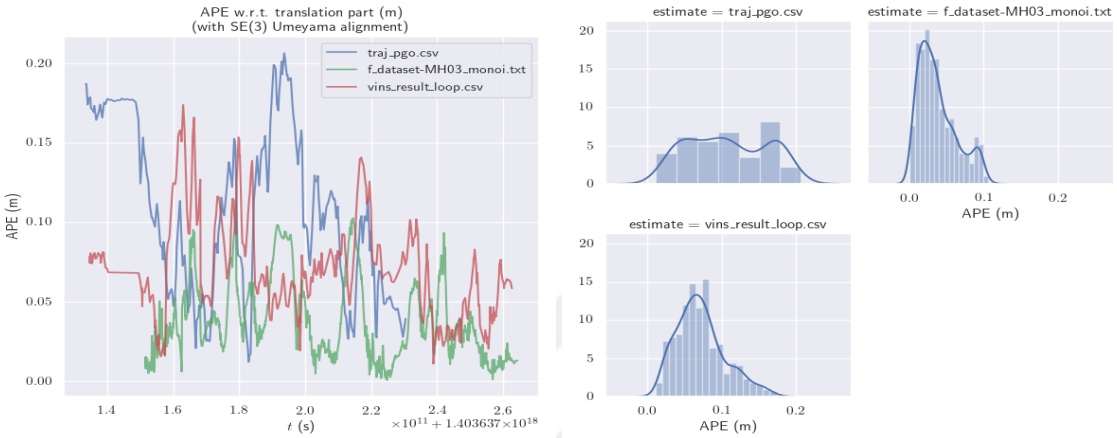
(a) Kimera (b) ORB-SLAM3



(c) VINS-MONO

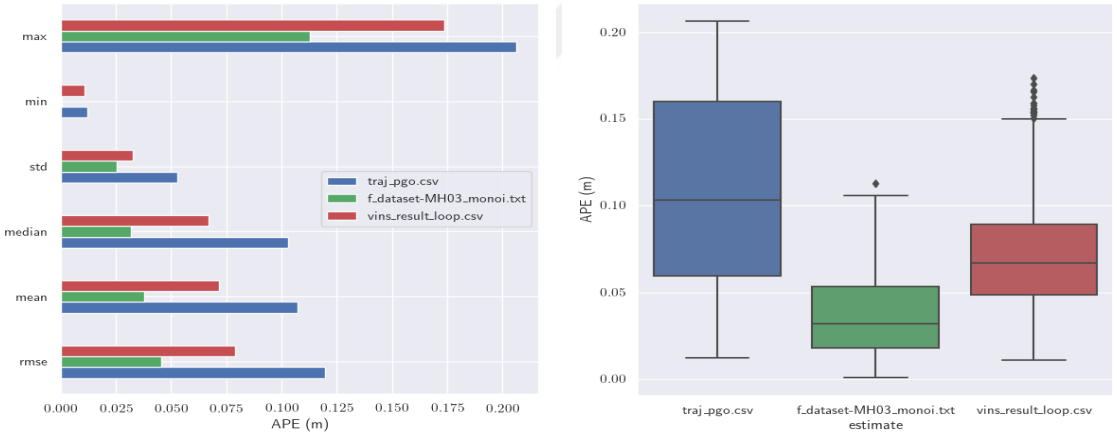
Figure 5.9 Trajectory error maps on MH03-Medium dataset in EuRoC MAV

In between the sequences of the Euroc Dataset, environment gets darker and feature candidates get lower in the surfaces. Machine Hall 03 sequence stated as in medium difficulty in terms of the environment. As it can be seen from Figure 5.10, comparing with the results in Figure 5.8, overall APE results are increased in all of the sequences. Moreover it can be observed from the trajectory results in Figure 5.9 that same with the MH01 results, in Kimera there are poses that couldn't be estimated. In terms of the overall APE by checking Figure 5.10 (b) histogram of the APE indicates that ORB-SLAM3 overperforms the rest in that sequence.



(a) APE vs Time in MH03-MEDIUM Dataset in EuRoC MAV

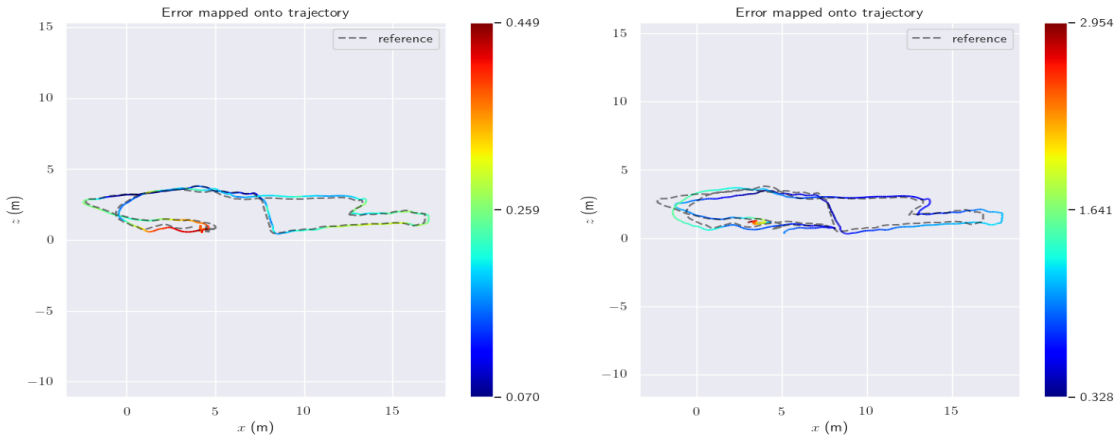
(b) Histogram of APE in MH03-Medium Dataset in EuRoC MAV



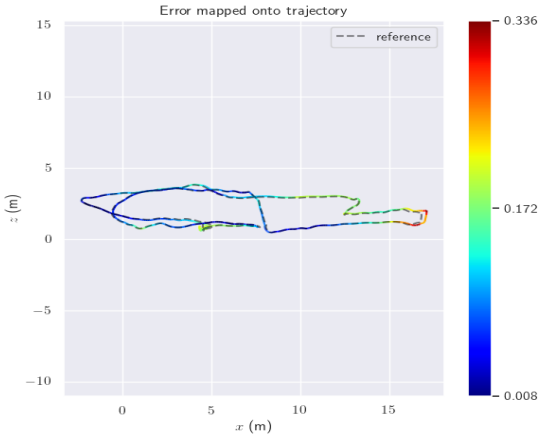
(c) Statistical Values on APE in MH03-Medium Dataset in EuRoC MAV

(d) Box Plot on APE in MH03-Medium Dataset in EuRoC MAV

Figure 5.10 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in MH03-Medium Dataset in EuRoC MAV



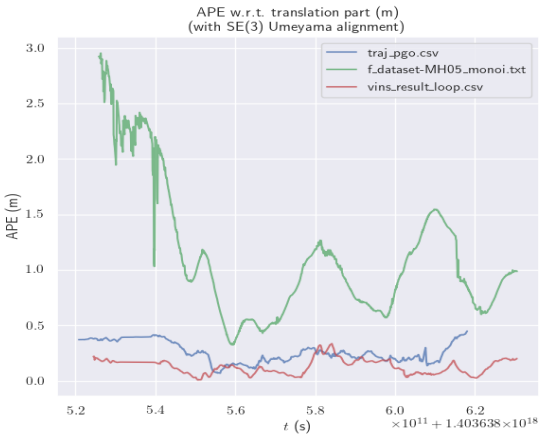
(a) Kimera (b) ORB-SLAM3



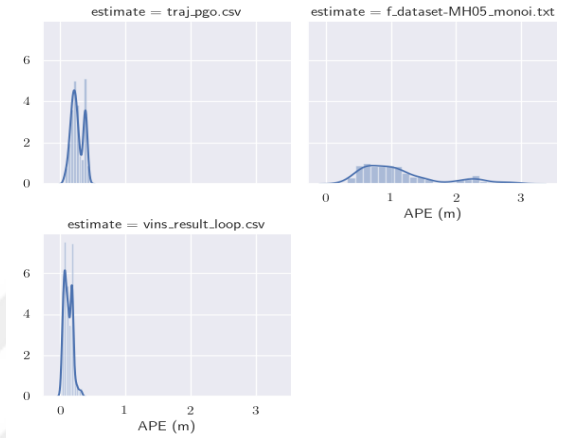
(c) VINS-MONO

Figure 5.11 Trajectory error maps on MH05-Difficult dataset in EuRoC MAV

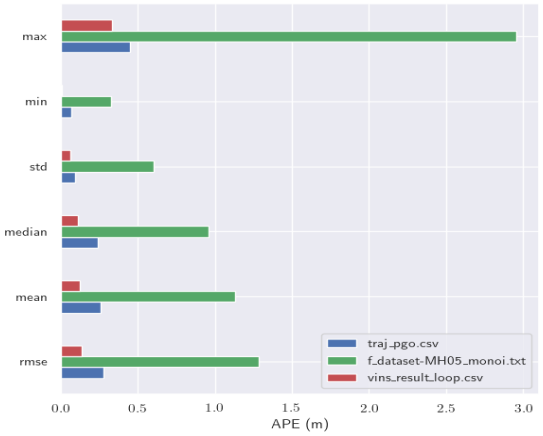
Machine Hall 05 sequence is the most environmetally challenging dataset among the Machine Hall sequences. It is also shorter than the rest. In that sequence, different from the previous ones, Kimera could handle to estimate most of the trajectory. ORB-SLAM3 has the the least accuracy in the trajectory. Based on the trajectory plot in Figure 5.11 Vins-Mono seem overperforms the rest. In addition, in Figure 5.12 (a) which shows the APE error vs time, ORB-SLAM3 seems suffer from wrong calculation about the scale and the system is not accurately initialized.



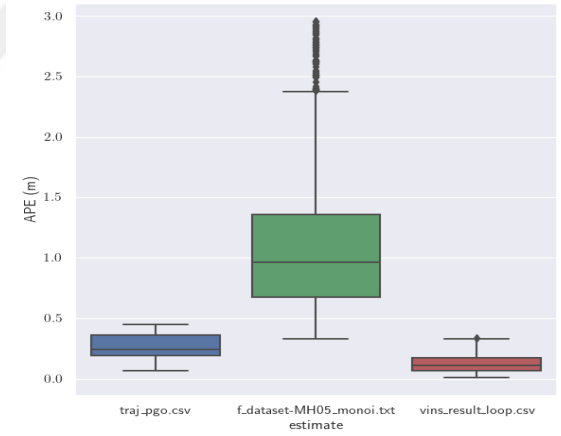
(a) APE vs Time in MH05-Difficult Dataset in EuRoC MAV



(b) Histogram of APE in MH05-Difficult Dataset in EuRoC MAV

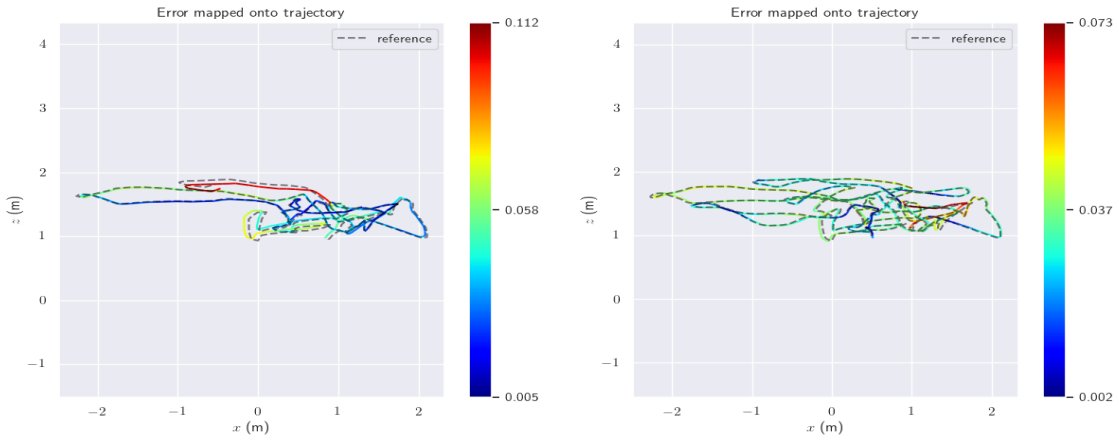


(c) Statistical Values on APE in MH05-Difficult Dataset in EuRoC MAV

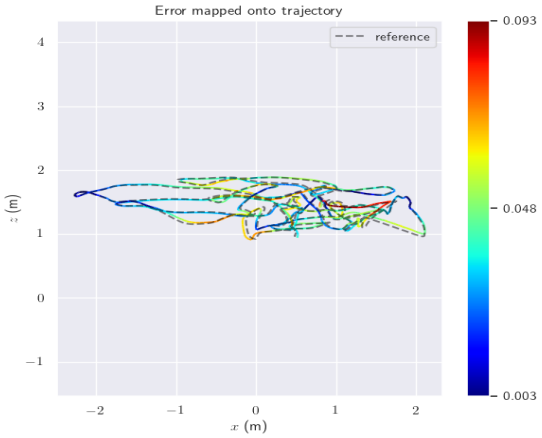


(d) Box Plot on APE in MH05-Difficult Dataset in EuRoC MAV

Figure 5.12 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in MH05-Difficult Dataset in EuRoC MAV



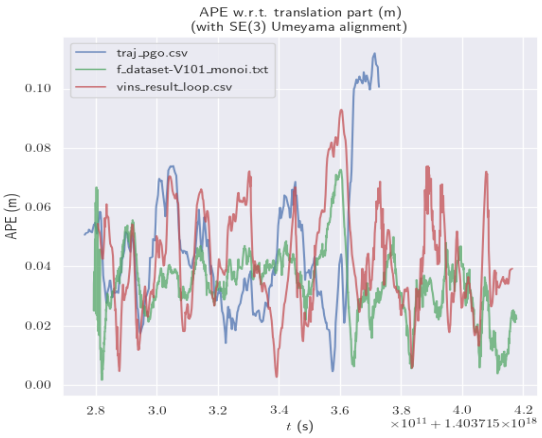
(a) Kimera (b) ORB-SLAM3



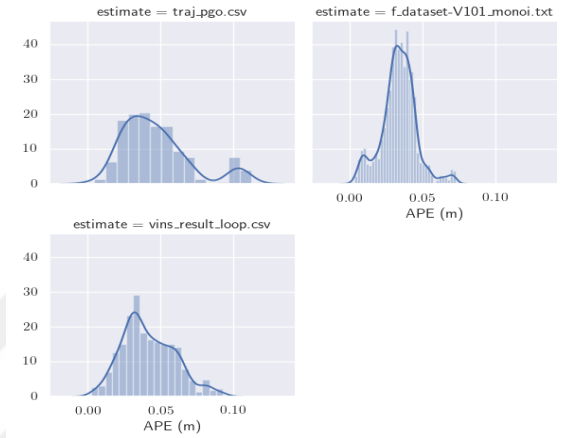
(c) VINS-MONO

Figure 5.13 Trajectory error maps on V101-Easy dataset in EuRoC MAV

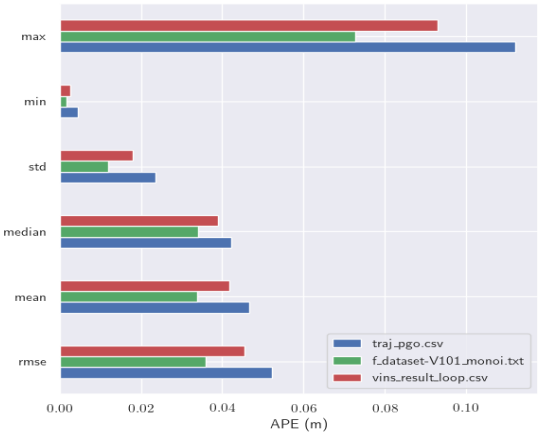
Vicon Room dataset recorded in different environment than Machine Hall sequence. In same way with the Machine Hall sequence, it gets gradually weak by visual clues and increase in visual challenges in the environments. In Vicon Room 101 dataset, same with the Machine Hall, Kimera is not able to estimate the whole trajectory which can be observed directly from the trajectory plot in Figure 5.13 and APE vs time plot in Figure 5.14 (a). Overall performance in Vicon Room 101 sequence is slightly better than the performance than the Machine Hall 01 sequence. As it can be observed in Figure 5.14 (c) ORB-SLAM3 has the lowest mean APE value close to the 0.04 . On the other hand rest is above 0.04 for the Vicon Room 101 sequence.



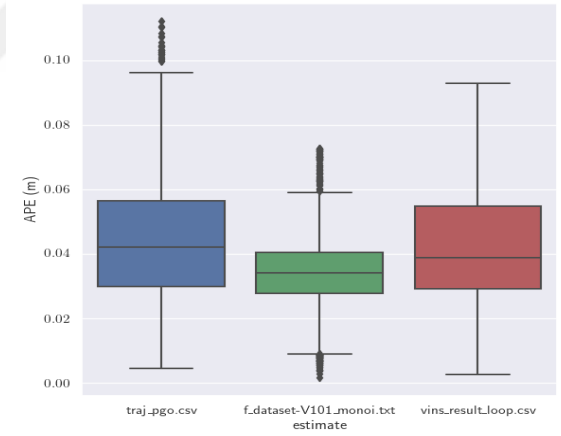
(a) APE vs Time in V101-Easy Dataset in EuRoC MAV



(b) Histogram of APE in V101-Easy Dataset in EuRoC MAV

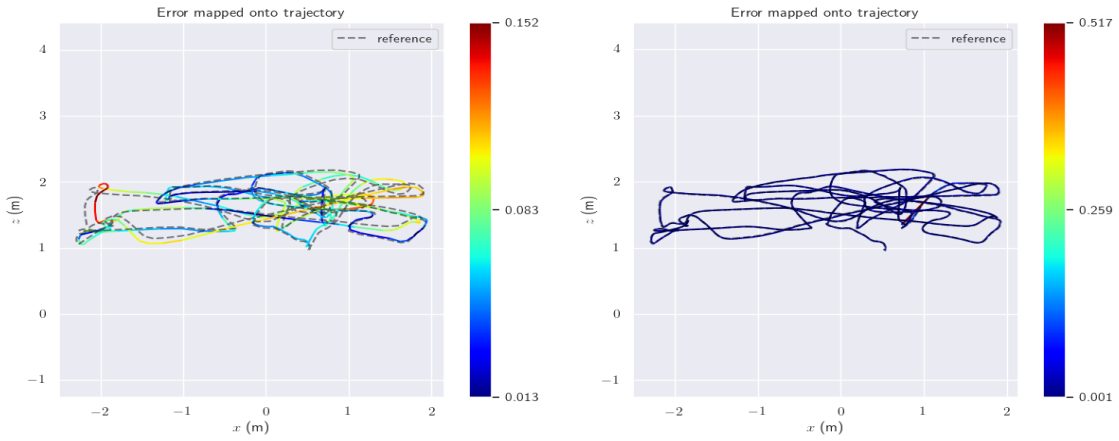


(c) Statistical Values on APE in V101-Easy Dataset in EuRoC MAV



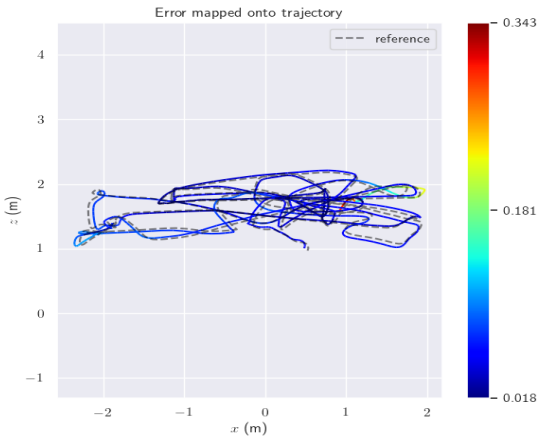
(d) Box Plot on APE in V101-Easy Dataset in EuRoC MAV

Figure 5.14 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in V101-Easy Dataset in EuRoC MAV



(a) Kimera

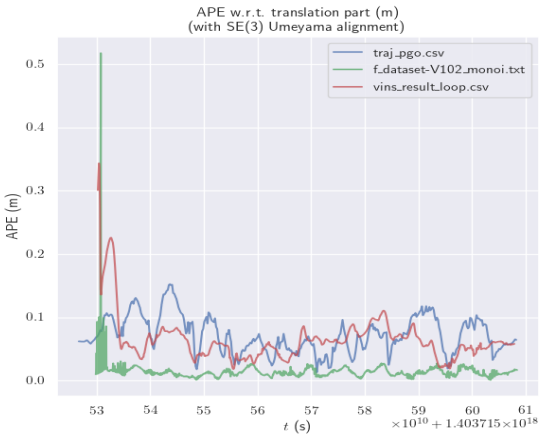
(b) ORB-SLAM3



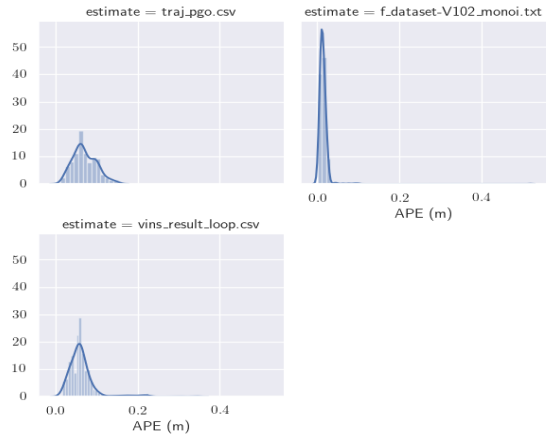
(c) VINS-MONO

Figure 5.15 Trajectory error maps on V102-Medium dataset in EuRoC MAV

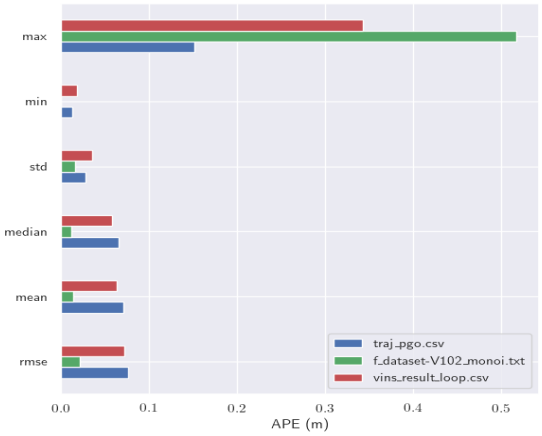
In vicon room 102 sequence which is more difficult than the v101 sequence in terms of the visual perception, Vins-Mono and ORB-SLAM3 overperforms Kimera in the trajectory APE observed in Figure 5.15. In terms of the APE statistics in the trajectory, as it's observed in Figure 5.16 (a), Vins-Mono and ORB-SLAM3 suffers from initialization. Even Kimera handles initialization better, for the rest of the sequence its overall mean APE lower than rest as in 5.16 (c).



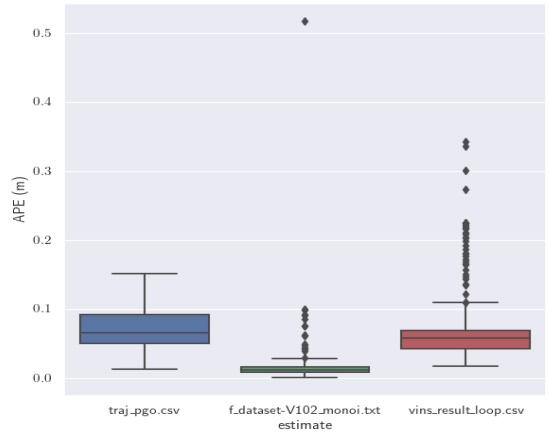
(a) APE vs Time in V102-Medium Dataset in EuRoC MAV



(b) Histogram of APE in V102-Medium Dataset in EuRoC MAV

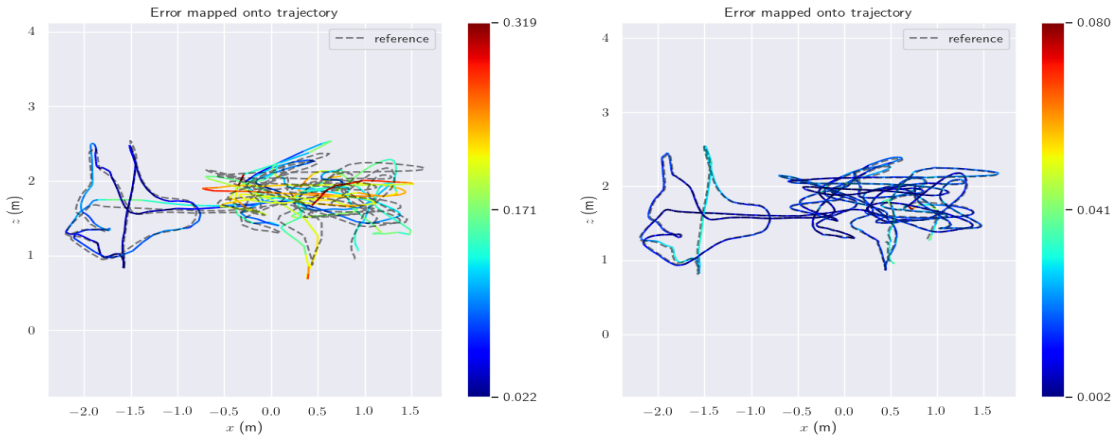


(c) Statistical Values on APE in V102-Medium Dataset in EuRoC MAV

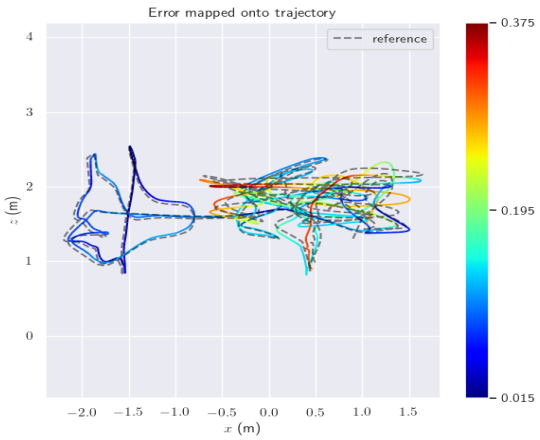


(d) Box Plot on APE in V102-Medium Dataset in EuRoC MAV

Figure 5.16 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in V102-Medium Dataset in EuRoC MAV



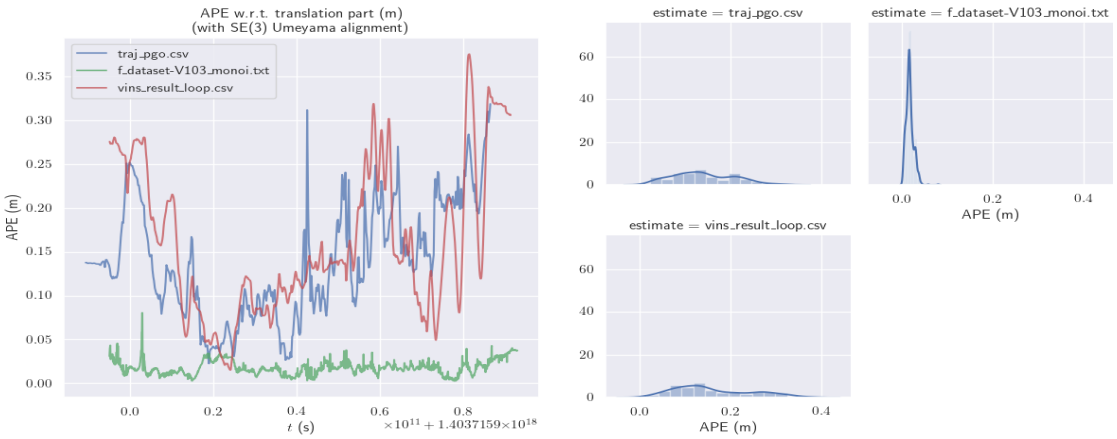
(a) Kimera (b) ORB-SLAM3



(c) VINS-MONO

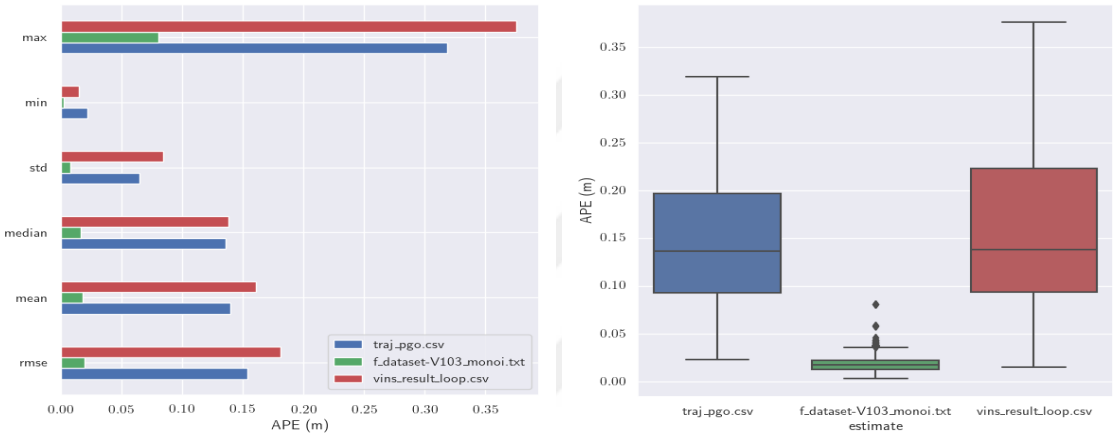
Figure 5.17 Trajectory error maps on V103-Difficult dataset in EuRoC MAV

Vicon room 103 is most difficult and longest sequence among the Vicon Room Dataset. It can clearly observed from the trajectories with errors in Figure 5.17 that ORB-SLAM3 overperforms the rest. This is the sequence which there is the most difference in terms of APE between the best APE with the second closest APE in the results. ORB-SLAM3 has APE mean around 0.01 while Vins-Mono and Kimera's have around 0.16 in Figure 5.18 (c).



(a) APE vs Time in V103-Difficult Dataset in EuRoC MAV

(b) Histogram of APE in V103-Difficult Dataset in EuRoC MAV



(c) Statistical Values on APE in V103-Difficult Dataset in EuRoC MAV

(d) Box Plot on APE in V103-Difficult Dataset in EuRoC MAV

Figure 5.18 Results: Kimera:traj_pgo, ORB_SLAM3:f_dataset, VINS-MONO:vins_result in V103-Difficult Dataset in EuRoC MAV

Table 5.4 APE - RMSE

Algorithms	MH01 EASY	MH03 MEDIUM	MH05 DIFFICULT	V101 EASY	V102 MEDIUM	V103 DIFFICULT
VINS-MONO	0.07617	0.07890	0.13757	0.04549	0.07265	0.18138
ORB-SLAM3	0.04678	0.04564	1.28198	0.03588	0.02171	0.01966
Kimera	0.07584	0.11961	0.27756	0.05232	0.07695	0.15424

As it can be seen from Table 5.4, in most of the sequences, ORB-SLAM3 outperforms both Kimera and VINS-MONO. Only for the MH-05 sequence, Kimera and VINS-MONO end up the sequence with significantly lower rmse of the absolute pose errors by the time. The reason behind this could be the relocalization module of the ORB-SLAM3. In ORB-SLAM3, when both camera and IMU sensors are available, short-term relocalization is implemented as depending only on the motion model based on the inertial data. The absence of visual data in detecting short-term relocalization could result in possible edge cases that will not be covered.

5.2.1.2 Test Platform Experiment

The Nvidia Jetson platform tested with VINS-MONO algorithm with custom camera intrinsics and IMU to camera rotation matrix in the Istanbul Technopark campus. Feature tracking works properly on harris corners in the environment. However, due to the misalignment between the IMU and the camera frequency algorithm, the system frequently lost pose tracking. In order to solve this problem, IMU-camera calibration should be handled for the specific combination of the hardware structures. Below can be seen the RVIZ output screenshot from the experiment.

Table 5.5 Collected Data Information

Path	technopark.bag
Experiment Location	Technopark Istanbul Campus
Duration	5:09s (309s)
Size	2.1 GB
Messages	18545
Types	Sensor_msgs/Image Sensor_msgs/Imu
Topics	/cam 6182 msgs /imu 12363 msgs

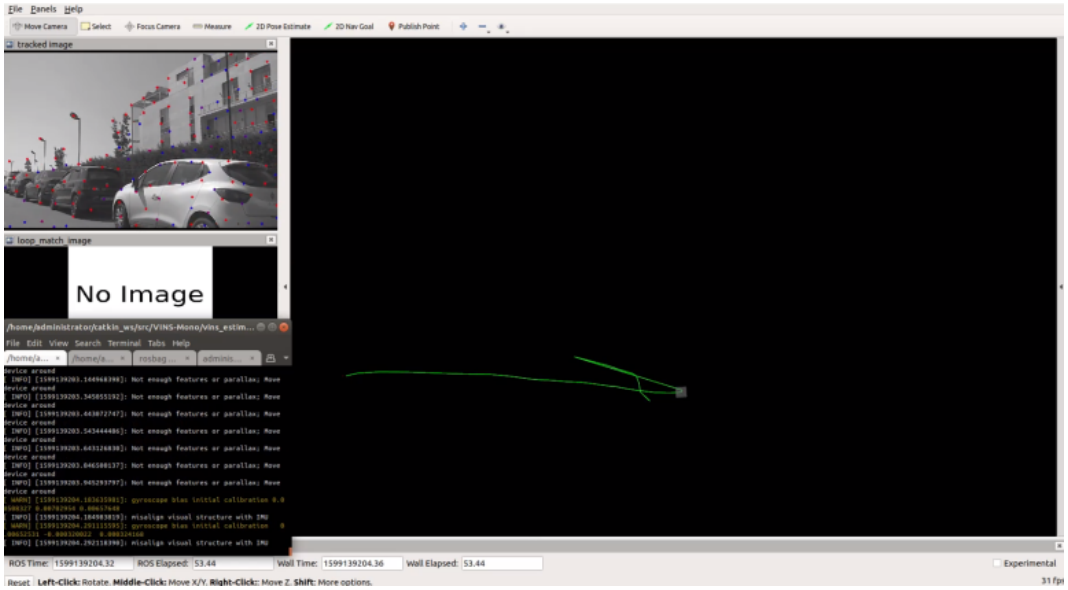


Figure 5.19 RVIZ screen output during real-world experiment in Istanbul Technopark

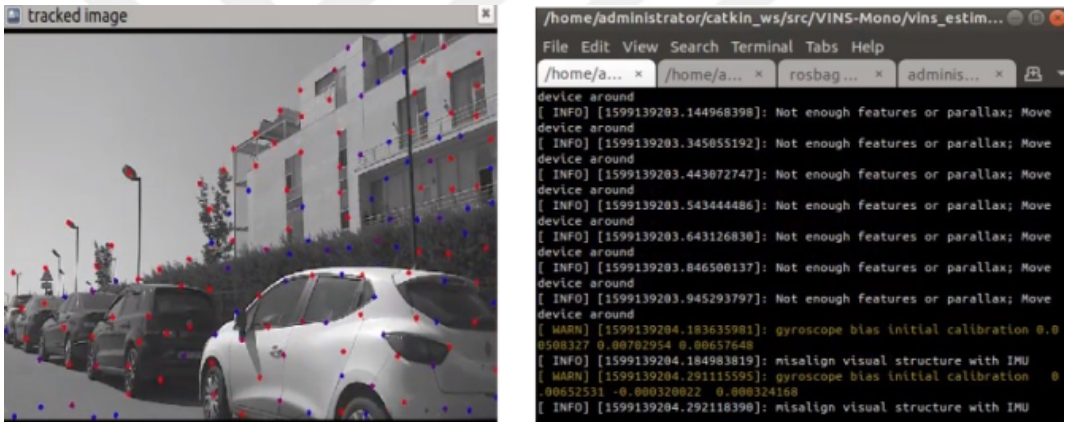


Figure 5.20 RVIZ screen output during real-world experiment in Istanbul Technopark

Using the same platform setup, indoor data is also collected by walking around the office in Technopark building. In the same way, indoor data processed with the Vins-Mono Qin et al. (2017) algorithm. Figure 5.19, shows the RVIZ user interface while processing collected outdoor data. In Figure 5.20, terminal outputs that logs the errors about the initialization and the example image that the extracted feature points are represented. There are additional challenges for the visual perception system when the indoor data used. It is observed that in indoor experiment extracted features get lowered due to the low light and textureless surfaces.

5.2.1.3 Using SQPNP in VINS-MONO

In Qin et al. (2017) PnP algorithm is used in both initialization and the visual-inertial estimation sections. In the sliding window approach, in the initialization section, the extracted features are tracked with the KLT optical flow algorithm between the frames in the sliding window. After that, by using the feature matches between the consecutive frames with a known camera matrix, the essential matrix is calculated. By decomposing the essential matrix, up to scale relative transformation between the corresponding frames is found. With known relative transformation with respect to the reference frame, extracted features are triangulated in order to get the 3D locations of the points. In the end, with known 2D-3D correspondences, using the iterative PnP method, which is based on the Levenberg-Marquardt algorithm Levenberg (1944), poses of the rest of the frames in the sliding window are found.

There are cases that the iterative method does not work correctly for the pose estimation. The most common case is for all of the points located in the same plane in the 3D space. That results in inconsistent, non-accurate pose estimation calculation with the PnP. In recent PnP solutions, that problem tried to be solved with various approaches. One of the recent works about the PnP solution called SQPNP Terzakis & Lourakis (2020) claimed to be robust in such planar cases.

In this thesis study, I have used the SQPNP instead of the iterative PnP used in the VINS-MONO. I have compared the absolute pose error results with using different PnP methods in VINS-MONO. In comparison, there is a parameter used about whether to use the initial pose guess in the PnP solution or not.

In sequences V1-03 and V1-03, there are edge cases in the initialization part for the SQPNP. It calculates jumps in the initialization. For the rest of the trajectory and in the other sequences, SQPNP seems to outperform iterative PnP without initial pose guess. When using initial pose guess, iterative PnP solution resulted in more accuracy in all sequences.

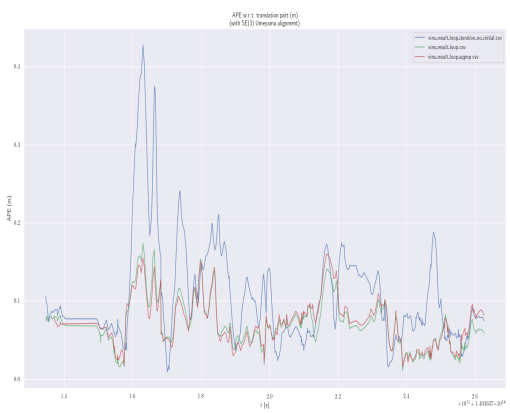
(a) APE vs Time in MH01-Easy Dataset in EuRoC MAV

(b) Histogram of APE in MH01-Easy Dataset in EuRoC MAV

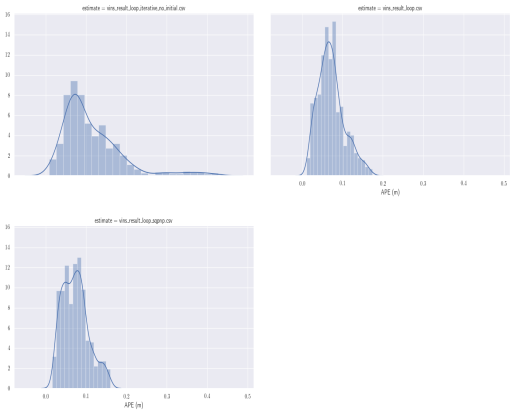
(c) Statistical Values on APE in MH01-Easy Dataset in EuRoC MAV

Figure 5.21 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in MH01-Easy Dataset in EuRoC MAV

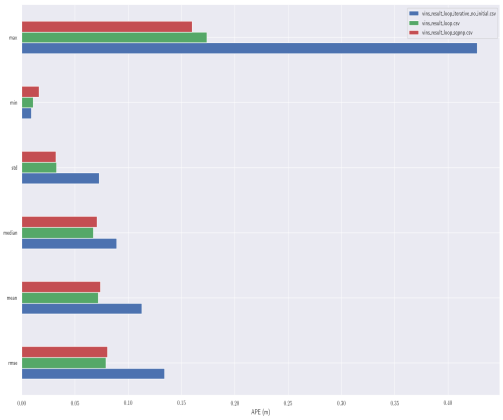
In Figure 5.21 (a), it can be observed when iterative perspective-n-point algorithm used without initial guess, in initial part of trajectory pose couldn't be estimated. Moreover overall APE results are also higher that the method with initial guess. In Figure 5.21 (c), mean APE results indicates that tested methods to the perspective-n-point problem give similar results but with a slightly difference SQPNP overperforms iterative PnP in terms of overall accuracy in the pose estimation.



(a) APE vs Time in MH03-Medium Dataset in EuRoC MAV



(b) Histogram of APE in MH03-Medium Dataset in EuRoC MAV



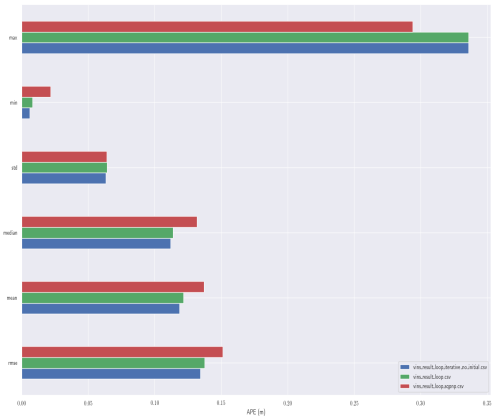
(c) Statistical Values on APE in MH03-Medium Dataset in EuRoC MAV

Figure 5.22 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in MH03-Medium Dataset in EuRoC MAV

In the experiment with medium difficulty sequence of the Machine Hall dataset, iterative PnP solution without initial guess seems suffer from scale issue in most of the parts of the trajectory. There are huge jumps in estimations shown in Figure 5.22 (a). Also in APE histogram of the experiments in Figure 5.22 (b), it can be observed that there are pose estimation that have APE above 12. Moreover, in terms of the overall performance, mean APE results in Figure 5.22 (c) shows that SQPNP overperforms iterative PnP in sequence Machine Hall 03.

(a) APE vs Time in MH05-Difficult Dataset in EuRoC MAV

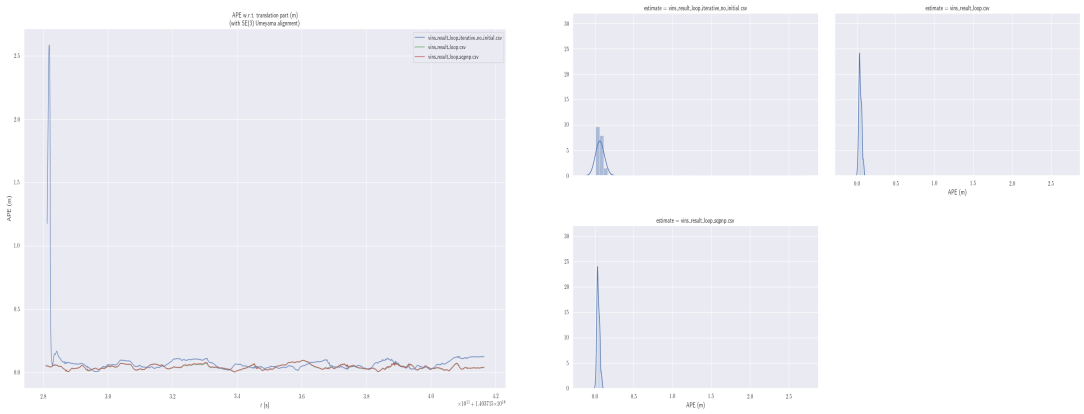
(b) Histogram of APE in MH05-Difficult Dataset in EuRoC MAV



(c) Statistical Values on APE in MH05-Difficult Dataset in EuRoC MAV

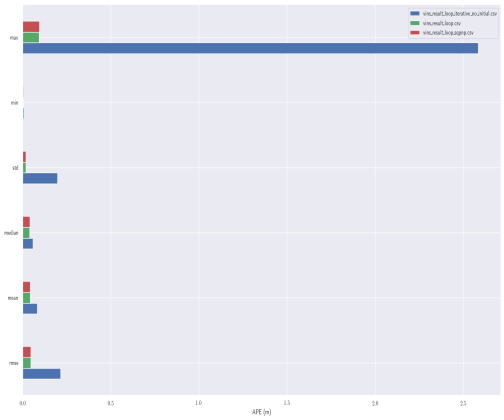
Figure 5.23 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in MH05-Difficult Dataset in EuRoC MAV

In the most environmentally difficut conditioned Machine Hall dataset, all three solutions to the PnP problem have the least accuracy among Machine Hall datasets. It can be observed from the mean APE results in both Figure 5.22 (c) and Figure 5.23 (c), APE results are almost doubled in that sequence for both SQPNP and iterative PnP. Besides that, there is not much difference between the method that initial guess used and the method that it is not used in the PnP solution as shown in the trajectory in Figure 5.23 (a).



(a) APE vs Time in V101-Easy Dataset in EuRoC MAV

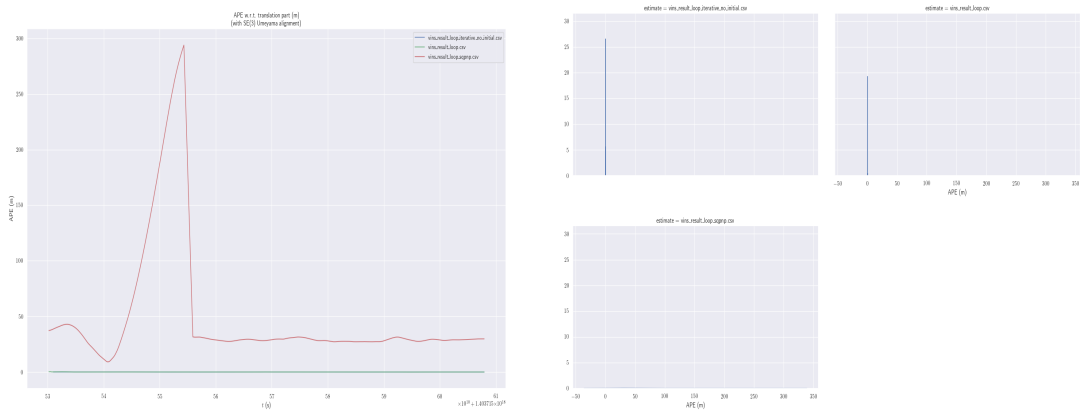
(b) Histogram of APE in V101-Easy Dataset in EuRoC MAV



(c) Statistical Values on APE in V101-Easy Dataset in EuRoC MAV

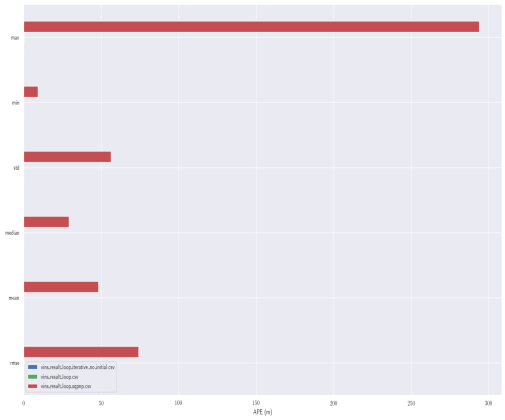
Figure 5.24 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in V101-Easy Dataset in EuRoC MAV

In vicon room 101 dataset, iterative PnP solution without initial guess has a huge jump in the initial pose estimation as it can be observed in Figure 5.24 (a). The reason behind this could be the solution could stuck in the local minimum. Even the max value for the APE seems very high in Figure 5.24 (c) for that solution, the rest of the trajectory have similar results with the SQPNP and iterative PnP with initial guess.



(a) APE vs Time in V102-Medium Dataset in EuRoC MAV

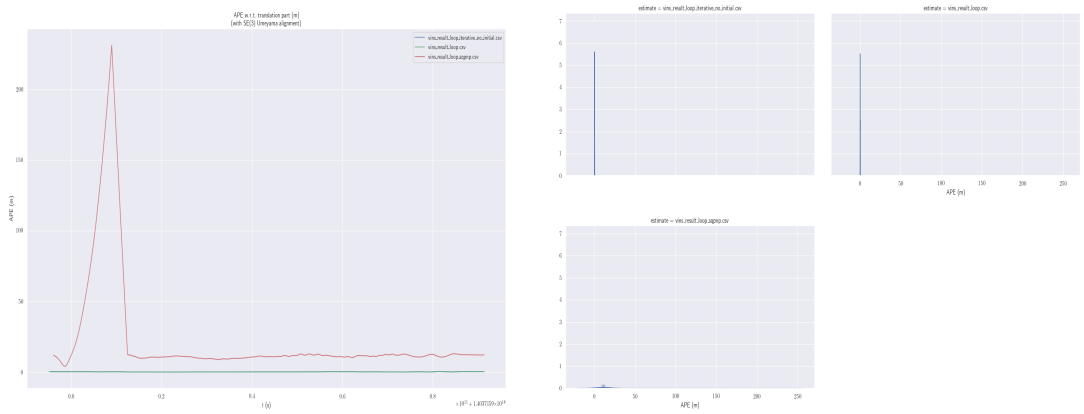
(b) Histogram of APE in V102-Medium Dataset in EuRoC MAV



(c) Statistical Values on APE in V102-Medium Dataset in EuRoC MAV

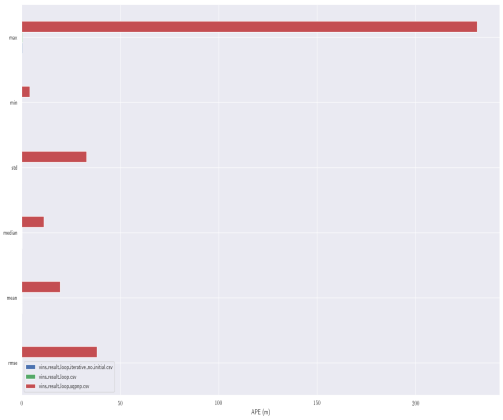
Figure 5.25 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in V102-Medium Dataset in EuRoC MAV

In vicon room 102 dataset, SQPNP implementation withhout initial guess fails in the whole trajectory as it can be observed in Figure 5.25. SQPNP APE is around 74 while the mean of the rest is around 0.05. PnP solutions could be effected by issues when there is feature points that only from a single plane, single line. In that sequence there is an issue about the feature point representations that makes SQPNP fails in total. But both iterative PnP solutions are well performed in vicon room 102 dataset.



(a) APE vs Time in V103-Difficult Dataset in EuRoC MAV

(b) Histogram of APE in V103-Difficult Dataset in EuRoC MAV



(c) Statistical Values on APE in V103-Difficult Dataset in EuRoC MAV

Figure 5.26 Results: SQPNP: result_loop_sqpnp, Iterative without initial guess: result_loop_iterative_no_initial, Iterative with initial guess: result_loop in V103-Difficult Dataset in EuRoC MAV

In vicon room 103 dataset, same with the vicon room 102 dataset SQPNP fails in the experiment. In Figure 5.26 (a), it can be observed that SQPNP failed in the begininning of the trajectory, even the rest of the trajectory has low APE, the huge APE in initial section effects the overall mean APE in Figure 5.26 (c). That means overall performance of SQPNP is not that much low in that sequence, initial huge jumps affects the statistical results.

Table 5.6 APE - RMSE of PnP methods used in VINS-MONO

PnP Methods	MH01 EASY	MH03 MEDIUM	MH05 DIFFICULT	V101 EASY	V102 MEDIUM	V103 DIFFICULT
SQPNP(no initial guess)	0.075653	0.08046	0.15126	0.04630	74.19366	38.25552
Iterative PnP(no initial guess)	0.07539	0.13408	0.13436	0.21445	0.05911	0.18130
Iterative PnP(with initial guess)	0.07617	0.07890	0.13757	0.04549	0.07265	0.18138

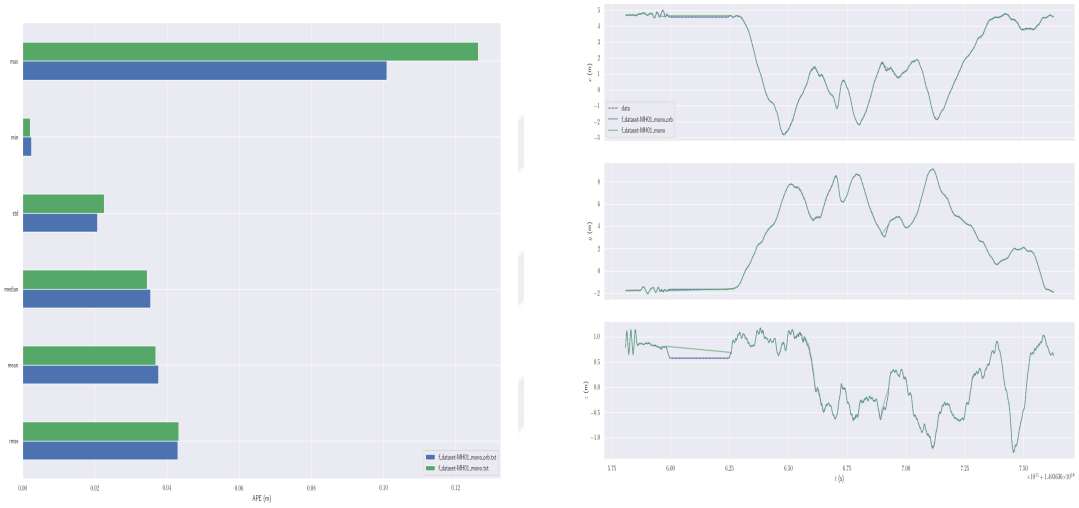
5.2.1.4 Using Superpoint Features in ORB-SLAM

In this experiment, Superpoint features that are presented in DeTone et al. (2018) are used instead of the ORB features in ORB-SLAM. Superpoint is one of the recent works about learning-based interest point extractors. There are many other learning-based feature detectors as Revaud et al. (2019), Mishchuk et al. (2017). Among those studies, the reason behind the selection of the Superpoint for this experiment is its score in the CVPR image matching challenge Trulls et al. (2020). In that benchmark, the algorithms that got the highest scores for the 2k feature point category use Superpoint interest points with various matchers.

Moreover, in that challenge, there is a category where the number of features is not restricted. Instead of a non-restricted number of feature point categories, a 2k interest point category is considered because more than the 2k points is not suitable for real-time applications. Even with 2k interest points for each frame, there is a need for high-performance hardware to maintain the system running in near real-time. Superpoint descriptors matched with knn matcher in the experiment, and the distance metric is used as L2-Norm between Superpoint feature descriptor arrays. The pre-trained weights are used for Superpoint, which is trained with MSCOCO Lin et al. (2014). Implementation is based on the work of the Deng et al. (2019) and the parameters of the Superpoint features are tuned with several iterations, which are based on the APE results of the sequences in the EuRoC MAV dataset Burri et al. (2016). As a result of the parameter tuning, Superpoint configuration is set as; 2000 features per image, confidence threshold of the points are set as 0.07, and NMS is set to 4. The ratio test is applied for the matching part, and the ratio threshold is set to 0.7. In order to use python implementation of the Superpoint in C++ for orb slam, C++ API of the PyTorch Paszke et al. (2019) library is used. The algorithm runs on the GPU device Nvidia 2060 super with six GB memory.

The APE results of the Superpoint features seem to outperform the ORB features in most of the sequences. However, in most sequences, the number of frames that the

pose is calculated is lower in Superpoint. The calculated ones are mostly better in APE scores, but there are missing frames in most sequences. Especially in sequences V102 and V103, most of the trajectory could not be calculated in Superpoint. The output of the absolute pose error does not represent the robustness of the algorithm. In a recent study and the most comprehensive dataset for SLAM algorithms Wang et al. (2020), there is another metric called success rate described for evaluating the SLAM systems in terms of robustness. They are calculating the ratio of missing frames to the number of all frames, and that ratio represents the success rate. Using ORB features in terms of the SR(success rate) resulted in much better rates than using Superpoint features in ORB-SLAM.

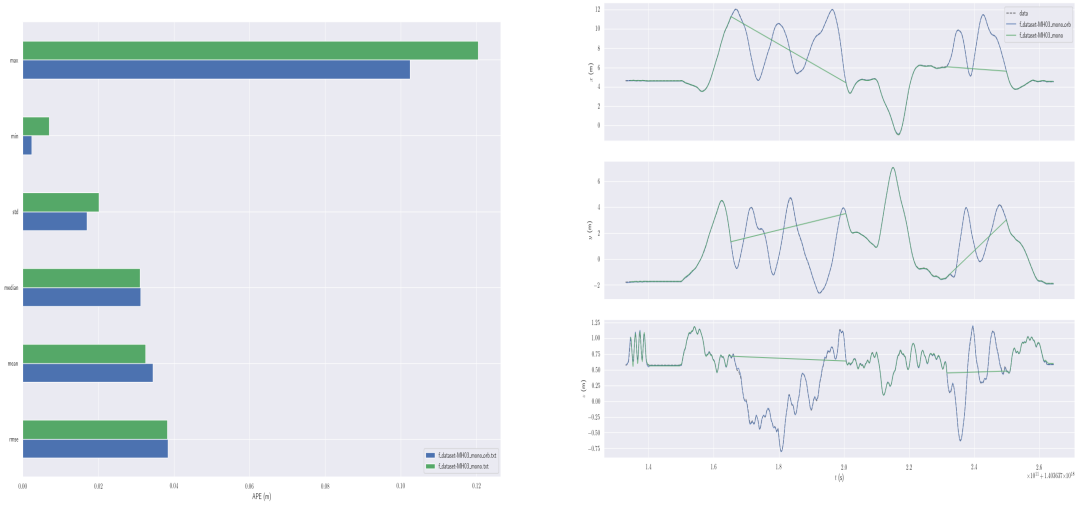


(a) Stats of Superpoint and ORB in MH01-EASY - EuRoC MAV Dataset

(b) Trajectories in x-y-z of Superpoint and ORB in MH01-EASY - EuRoC MAV Dataset

Figure 5.27 Results: Superpoint features: mono, ORB features: mono_orb in MH01-EASY Dataset in EuRoC MAV

In Machine Hall 01 dataset, the results are very similar in terms of the APE as it can be observed from the Figure 5.27 (a) . In Figure 5.27 (b), the trajectoris are plotted and it is shown that there are some intervals that couldn't be estimated in the SuperPoint implementation. The reason behind this could be the implementation in the SuperPoint features in ORB-SLAM3. There could be some missing cases that didn't covered in the implementation. In the rest of the trajectory, it is observed that both feature are performing similarly in Machine Hall 01 dataset.

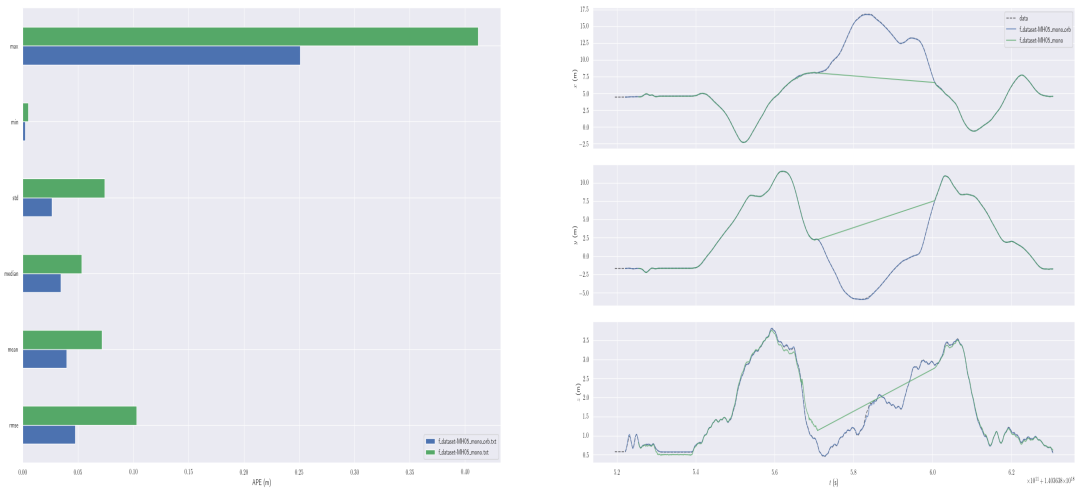


(a) Stats of Superpoint and ORB in MH03-MEDIUM - EuRoC MAV Dataset

(b) Trajectories in x-y-z of Superpoint and ORB in MH03-MEDIUM - EuRoC MAV Dataset

Figure 5.28 Results: Superpoint features: mono, ORB features: mono_orb in MH03-Medium Dataset in EuRoC MAV

In medium difficulty dataset in the Machine Hall sequence, same with the MH 01 dataset there are poses that couldn't be estimated in the SuperPoint implementation in Figure 5.28 (b). In terms of the APE results of the estimated poses, in that seqeunce SuperPoint implementation is slightly better that the ORB implementation.

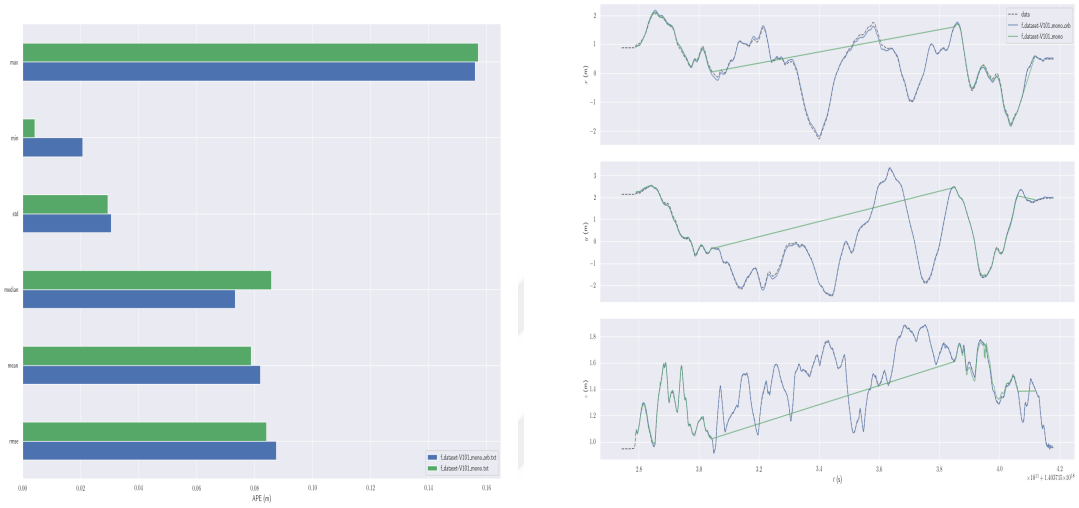


(a) Stats of Superpoint and ORB in MH05-DIFFICULT - EuRoC MAV Dataset

(b) Trajectories in x-y-z of Superpoint and ORB in MH05-DIFFICULT - EuRoC MAV Dataset

Figure 5.29 Results: Superpoint features: mono, ORB features: mono_orb in MH05-Difficult Dataset in EuRoC MAV

In the most difficult conditioned sequence in the Machine Hall dataset, SuperPoint implementation overperforms the ORB implementation as it can be observed from the Figure 5.29 (a). Still there are poses that couln't be estimated in SuperPoint implementation but the results are better than ORB features. It can be said that SuperPoint give more reliable features in the low light, low textured environments.

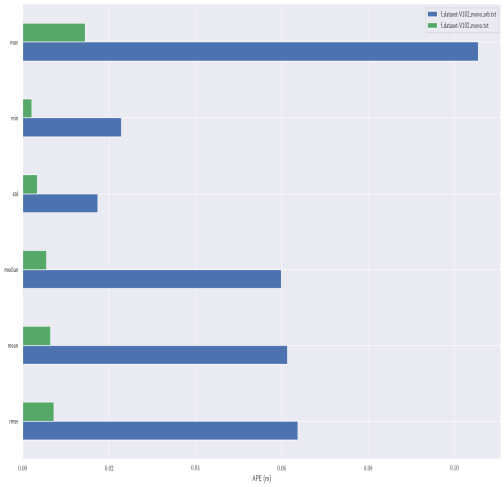


(a) Stats of Superpoint and ORB in V101-EASY - EuRoC MAV Dataset

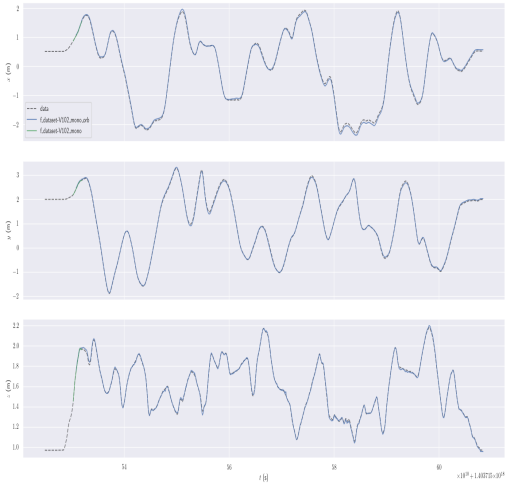
(b) Trajectories in x-y-z of Superpoint and ORB in V101-EASY - EuRoC MAV Dataset

Figure 5.30 Results: Superpoint features: mono, ORB features: mono_orb in V101-EASY Dataset in EuRoC MAV

In vicon room 101 dataset, results are similar in terms of APE as in the easiest sequence of the Machine Hall dataset. It can be said that, the difference appears more in the difficult environments for the visual perception. In vicon room 101 dataset, the poses that couldn't be estimated in SuperPoint imlementation is higher that the Machine Hall sequences that observed in Figure 5.30 (b).



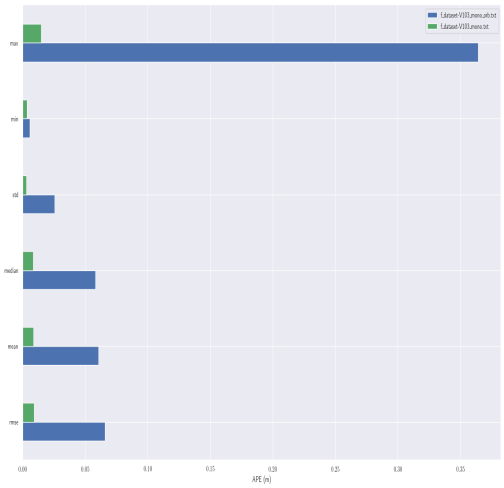
(a) Stats of Superpoint and ORB in V102-MEDIUM - EuRoC MAV Dataset



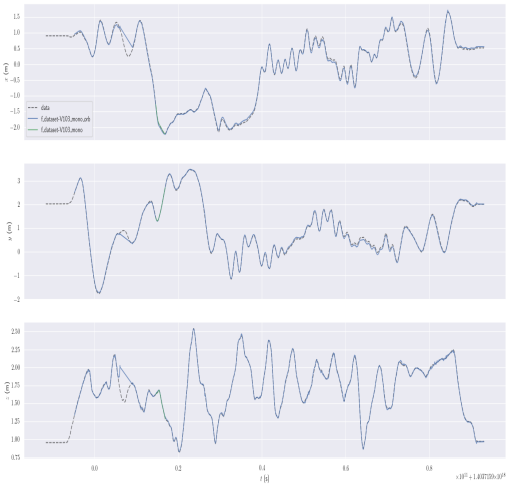
(b) Trajectories in x-y-z of Superpoint and ORB in V102-MEDIUM - EuRoC MAV Dataset

Figure 5.31 Results: Superpoint features: mono, ORB features: mono_orb in V102-Medium Dataset in EuRoC MAV

In medium version of the Vicon dataset, Superpoint overperforms Orb implementation as it can be observed in Figure 5.31 (a). When the condition gets more difficult in the environment, difference between the SuperPoint and the ORB implementation gets bigger in terms of the APE.



(a) Stats of Superpoint and ORB in V103-DIFFICULT - EuRoC MAV Dataset



(b) Trajectories in x-y-z of Superpoint and ORB in V103-DIFFICULT - EuRoC MAV Dataset

Figure 5.32 Results: Superpoint features: mono, ORB features: mono_orb in V103-Difficult Dataset in EuRoC MAV

In the most difficult sequence in the vicon room dataset, Superpoint overperforms the ORB implementation. The difference in the APE shown in Figure 5.32 (a) not totally represent the real performance. Because in SuperPoint implementation there are missing pose estimations in the trajectory as it can be observed in Figure 5.32 (b).

Table 5.7 APE - RMSE of Superpoint and ORB features in ORB-SLAM

Feature Extractors	MH01 EASY	MH03 MEDIUM	MH05 DIFFICULT	V101 EASY	V102 MEDIUM	V103 DIFFICULT
Superpoint	0.04321	0.03822	0.10309	0.08405	0.00723	0.00912
ORB	0.04297	0.03842	0.04757	0.08742	0.06370	0.06576

6. CONCLUSION & FUTURE WORK

Visual-inertial SLAM systems are getting more attention in the literature and industry. Vehicles are becoming more autonomous, virtuality in our lives is increasing with augmented reality applications and visual - inertial SLAM is in the core of these promising application areas. In this thesis, literature on both visual and visual-inertial SLAM systems are reviewed. Moreover modules of the visual and inertial structures are also discussed. In that discussion, different configurations in visual modules, sensors, smoothing approaches are expressed and compared in detail. In addition, fusion methods in visual and inertial data, tightly and loosely coupled fusion, are discussed.

Furthermore, an end-to-end visual inertial pipeline is proposed and implemented. In terms of the visual modules, for the feature extraction SuperPoint DeTone et al. (2018) is used. SuperPoint features are matched with faiss Johnson et al. (2017). Based on feature matches essential matrix is calculated by five point algorithm and MAGSAC Barath & Matas (2018) which described as advanced RANSAC. In order to track features KLT Lucas & Kanade (1981) is used and for estimating the pose from 2D-3D correspondences SQPNP Terzakis & Lourakis (2020) is used. In inertial side, GTSAM Kaess (2015) library is used for IMU-preintegration Forster et al. (2015) and pose estimation from IMU. Implemented visual inertial SLAM pipeline is tested on one of the most popular public dataset Euroc Mav Burri et al. (2016). Moreover in the scope of the thesis, state of the art algorithms are studied by configuring modules and comparing the results with the configured implementations. For that purpose, in ORB-SLAM3, instead of ORB features that are used in the pipeline originally, SuperPoint features are tested. In that experiment it is observed that, when the environment gets difficult for the visual perception by getting darker, less detailed textures, blurred images, SuperPoint seems more reliable in that conditions. The calculated APE difference between the orb implementation and the SuperPoint implementation is gradually increased when the environment gets difficult to perceive. Furthermore, in another implementation, one of the most promising perspective-n-point solution SQPNP is implemented within the Vins-Mono instead of the originally used iterative PnP solution. The results showed that even there is no initial guess in SQPNP implementation, in some sequences it overperforms iterative PnP solution. But it is observed that there is edge cases for the SQPNP

solution where there are huge pose error in some timestamps in Vicon Room sequence. In addition to those experiments, in order to experience visual inertial SLAM in real-time, hardware platform is set up and data collected from the environment separately as indoor and outdoor. Collected real world data is processed and results are discussed. Finally, state of the art algorithms are studied, compared and the results are discussed in terms of both the algorithmic logic behind their structure and used visual and inertial modules.

As a future work, I will be more focused on the visual inertial alignment and solving the minimization problem to get the scale. In the current pipeline visual and inertial measurements are well aligned but not being solved in an equation yet. Furthermore, based on visual modules there are promising studies based on the deep learning such as Yang et al. (2020). I am planning to focus on comparing learning based methods with traditional approaches by benchmarking both on public datasets.

BIBLIOGRAPHY

Barath, D. & Matas, J. (2018). MAGSAC: marginalizing sample consensus. *CoRR*, *abs/1803.07469*.

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M., & Siegwart, R. (2016). The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, *35*.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., & Leonard, J. (2016). Simultaneous localization and mapping: Present, future, and the robust-perception age. *IEEE Transactions on Robotics*, *32*.

Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M. M., & Tardós, J. D. (2020). Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam.

Delmerico, J. & Scaramuzza, D. (2018). A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 2502–2509).

Deng, C., Qiu, K., Xiong, R., & Zhou, C. (2019). Comparative study of deep learning based features in slam. In *2019 4th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, (pp. 250–254). IEEE.

DeTone, D., Malisiewicz, T., & Rabinovich, A. (2018). Superpoint: Self-supervised interest point detection and description.

Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2015). Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. *Robotics: Science and Systems (RSS) conference*.

Hartley, R. & Zisserman, A. (2003). *Multiple View Geometry in Computer Vision* (2 ed.). USA: Cambridge University Press.

Heng, L., Bürki, M., Lee, G. H., Furgale, P., Siegwart, R., & Pollefeys, M. (2014). Infrastructure-based calibration of a multi-camera rig. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 4912–4919).

Heng, L., Furgale, P., & Pollefeys, M. (2014). Leveraging image-based localization for infrastructure-based calibration of a multi-camera rig. *Journal of Field Robotics*, *32*.

Heng, L., Li, B., & Pollefeys, M. (2013). Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 1793–1800).

Huang, G. (2019). Visual-inertial navigation: A concise review. *CoRR*, *abs/1906.02650*.

Jin, Y., Mishkin, D., Mishchuk, A., Matas, J., Fua, P., Yi, K. M., & Trulls, E. (2020). Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision*, *129*(2), 517–547.

Johnson, J., Douze, M., & Jégou, H. (2017). Billion-scale similarity search with gpus.

Kaess, M. (2015). Gtsam library.

Levenberg, K. (1944). A method for the solution of certain non – linear problems in least squares. *Quarterly of Applied Mathematics*, *2*, 164–168.

Lichtsteiner, P., Posch, C., & Delbruck, T. (2008). A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, *43*(2), 566–576.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014). Microsoft coco: Common objects in context. cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*, 91–110.

Lucas, B. & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision (ijcai). volume 81.

Mishchuk, A., Mishkin, D., Radenovic, F., & Matas, J. (2017). Working hard to know your neighbor's margins: Local descriptor learning loss. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., & Garnett, R. (Eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc.

Qin, T., Li, P., & Shen, S. (2017). Vins-mono: A robust and versatile monocular visual-inertial state estimator. *CoRR*, *abs/1708.03852*.

Revaud, J., Weinzaepfel, P., de Souza, C. R., Pion, N., Csurka, G., Cabon, Y., & Humenberger, M. (2019). R2D2: repeatable and reliable detector and descriptor. *CoRR*, *abs/1906.06195*.

Rosinol, A., Abate, M., Chang, Y., & Carlone, L. (2020). Kimera: an open-source library for real-time metric-semantic localization and mapping.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, (pp. 2564–2571).

Sarlin, P., DeTone, D., Malisiewicz, T., & Rabinovich, A. (2019). Superglue: Learning feature matching with graph neural networks. *CoRR*, *abs/1911.11763*.

Shen, S., Michael, N., & Kumar, V. (2015). Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft mavs. *Proceedings - IEEE International Conference on Robotics and Automation*, *2015*, 5303–5310.

Teed, Z. & Deng, J. (2020). Raft: Recurrent all-pairs field transforms for optical flow.

Terzakis, G. & Lourakis, M. (2020). A consistently fast and globally optimal solution to the perspective-n-point problem.

Trulls, E., Fua, P., Matas, J., Miskhin, D., Yi, K., & Jin, Y. (2020). Imw 2020: Leaderboard.

Vidal, A. R., Rebecq, H., Horstschaefer, T., & Scaramuzza, D. (2018). Ultimate

slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios. *IEEE Robotics and Automation Letters*, *3*(2), 994–1001.

Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A., & Scherer, S. A. (2020). Tartanair: A dataset to push the limits of visual SLAM. *CoRR*, *abs/2003.14338*.

Yang, N., von Stumberg, L., Wang, R., & Cremers, D. (2020). D3VO: deep depth, deep pose and deep uncertainty for monocular visual odometry. *CoRR*, *abs/2003.01060*.

Zubizarreta, J., Aguinaga, I., & Montiel, J. M. M. (2020). Direct sparse mapping. *IEEE Transactions on Robotics*, *36*(4), 1363–1370.