ORB-SLAM EXAMINATION IN THE INDOOR TEXTURELESS ENVIRONMENT

An Undergraduate Research Scholars Thesis

by

YUAN-PENG YU

Submitted to the Undergraduate Research Scholars program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Dezhen Song

May 2017

Major: Computer Engineering

TABLE OF CONTENTS

ABSTRACT

ORB-SLAM Examination in the Indoor Textureless Environment

Yuan-Peng Yu
Department of Computer Science & Engineering
Texas A&M University

Research Advisor: Dr. Dezhen Song
Department of Computer Science & Engineering
Texas A&M University

Visual SLAM is a challenging topic because the algorithm needs to be run in real-time and give a precise estimation of camera pose. The field of research become popular after Davison [1] introduced computer vision methods to SLAM. The current state-of-the-art SLAM algorithms: LSD-SLAM [2] and ORB-SLAM [3] can provide camera trajectories and build a precise map. However, these research have few data and experiments for the indoors. The research would focus on applying SLAM methods in the indoor environment because of the rising demands of indoor robots. The purpose of the research is to evaluate the performance when ORB-SLAM algorithm works in a textureless indoor environment. ORB-SLAM is believed to be one of the best among state-of-the-art visual SLAM algorithms, where it adopts a heuristic model selection to tackle with different scene scenario in their experiments. However, the researcher is interested in whether repetitive patterns, textureless walls, and planar structures in the indoors affect the robustness of a visual SLAM algorithm. The research conduct experiments in the HRBB 4th floor to examine the performance of ORB-SLAM in a textureless indoor environment.

DEDICATION

I dedicate the thesis to my beloved family.

I would like to express the deepest gratitude to my parents for their nursing, patience, and love.

NOMENCLATURE

SLAM	Simultaneous Localization And Mapping.
VO	Visual Odometry.
EKF	Extended Kalman Filter.
PTAM	Parallel Tracking and Mapping.
LSD-SLAM	Large-Scale Direct Monocular SLAM.
ORB	Oriented FAST and Rotated BRIEF. An image feature descriptor.
ORB-SLAM	A SLAM system that uses the ORB feature descriptor.
F	Fundamental matrix.
H	Homography matrix. The relationship between two images if the observing object is located in a planar surface.
K	The intrinsic camera calibration matrix
P	The camera matrix

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

Recently, the development of autonomous vehicles has caught many people's attention. Autonomous cars can detect the surrounding environment and navigate itself to the destination without the inputs from humans. Among the key technologies applied in autonomous cars, the Simultaneous Localization and Mapping (SLAM) is to study how an ego-motion robot explore an unknown environment. While navigating, the robot can record its corresponding position in the world coordinates and incrementally build a map of the environment at the same time. The SLAM problem is complex because the robot needs to compute the localization and mapping concurrently and complete these tasks in real-time.

Theoretical Foundation

Simultaneous Localization and Mapping (SLAM) is a crucial process for a robot to explore a previously unknown environment. The tasks of identifying the current position of the robot, generating the coordinates of the robot from inputs of cameras, and creating a global map of the environment are full of the mathematical insights and domain knowledge. The theoretical framework of the research is based on the Probabilistic Robotics [4] and Multiple View Geometry In Computer Vision [5]. Many different algorithms and building blocks have been proposed in the field of SLAM research. The Bayesian filtering models were applied to SLAM in the 1990s. Thrun, Burgard, and Fox proposed a probabilistic approach to SLAM [6]. In this way, the map building problem becomes a problem that estimates the most likelihood of the robot's action. A map can be divided into small blocks, and the probabilities of each block that the robot will appear in the location are not entirely equal. The hardware constraints on the robot's motion

and the continuity of time and space limited the robot's locations in a map. Although the research [6] could not allow the robot to identify landmarks autonomously, the research contribute a fundamental framework in the SLAM. Important theories and concerns are discussed thoroughly in the book, Probabilistic Robotics [4].

Based on the perception devices of a robot, different SLAM methodologies can be adopted to measure the range between the robot and the surroundings. There are three commonly used devices: light detection and ranging (LIDAR), sonar, and camera vision. LIDAR can sense the distances from targets precisely and efficiently. However, the cost of using LIDAR in robots is expensive. Compared to LIDAR, the price of the sonar is much cheaper. Nevertheless, the measurement of sonar is not precise enough due to the wide emitting angle of ultrasound waves. The popularity of vision based methods is rising in the recent research.

Vision-based perceptions become important because the price of cameras is much lower than that of the LIDAR. Cameras are also close to the way that human sense the world and images provide rich information. Monocular cameras, stereo cameras, and RGB-D cameras are most likely applied in the SLAM techniques. Cameras have different characteristics. Monocular cameras are cheap and are widely used, but they lack the depth information. The operating mechanism of stereo cameras are similar to human eyes and it can provide the depth information. RGB-D cameras can retrieve color and depth information at the same time, which is largely applied in the recent research. Although vision based methods require a plenty of processing due to high data rates, the advanced hardware and algorithms relieve the bottleneck. Because monocular cameras are ubiquitously built in mobile phones, the paper will mainly focus on using monocular cameras to solve the SLAM problem.

Visual Odometry

Computer vision was introduced to the robotics SLAM research around the 2000s. One of computer vision algorithms that are closely related to the SLAM problem is visual odometry (VO). The goal of VO is to estimate the ego-motion of a robot by the camera inputs. Researchers can track the camera pose from images to observe the robot's movement. The term is suggested by Nister [7] and is analogous to wheel odometry. In order to estimate the robot's movement, wheel odometry calculates the number of turns of wheels in a time period whereas VO observes the changes between a set of images. VO is better than wheel odometry because of the immunity against the skidding of wheels and non-uniform terrains. There is no prior knowledge of the environment needed in VO, which meets the scenario in the SLAM problem.

The basic flow in visual odometry contains five steps.

1. Cameras generate a set of sequential images. While a robot is moving, the cameras attached to the robot record the environment periodically.
2. A set of images are selected as keyframes because they contain important feature points, which are also called as landmarks. Feature-based methods extract the feature points in keyframes. Corners, conspicuous points, and repeatable textures are considered as good features in an image. The characteristics of feature points are that they contain a high variation when the image is moved. The commonly used feature descriptors are Harris Corner, Scale-Invariant Feature Transform (SIFT) [8], and Speeded Up Robust Features (SURF) [9].
3. Feature-based methods track feature points and compare the difference between two consecutive keyframes.

4. Obtain the robot's motion based on the feature correspondence. This step usually cost the most computation.
5. A local camera-pose optimization is performed. Bundle adjustment (BA) is generally regarded as a good way to estimate camera localization.

To correctly track the camera trajectories, Scaramuzza [10] suggests four important premises of using visual odometry. First, the environment has a proper illumination. Dark camera images could not offer sufficient information to analyze. Second, the static scene should be the major part in the images than moving objects. If the moving objects dominate the input images, VO cannot estimate the correct position of the robot. Third, images should contain sufficient feature points or patterns so that these patterns can be recognized as landmarks. By matching the feature points between images, VO estimates the motion of the robot. Lastly, the exploring scene should be overlapped between the two consecutive images.

Visual SLAM

Visual SLAM combines the SLAM research in robotics with the research of computer vision. Visual SLAM uses the visual-based perception devices to conduct the SLAM research. Instead of LIDAR and sonar sensors, cameras are adopted because images provide much more information about the explored scene. The downside of visual SLAM is that the computing cost is high and advanced algorithms are needed. In fact, the field of visual SLAM become fast developed because the computing power of CPU has improved and many novel solutions have proposed.

Visual SLAM and visual odometry are slightly different because of the application purpose. The most significant difference is that visual SLAM includes a loop closure. The goal of visual SLAM is to build a map and a full trajectory for the camera (or the robot) whereas the

goal of visual odometry is to incrementally obtain the next pose for the camera (or the robot). According to Scaramuzza [10], visual SLAM is targeting on a global consistent trajectory and map whereas visual odometry is targeting on a local consistent trajectory. Before the loop closure, visual SLAM has the same process flow in visual odometry: keyframes collecting, feature extracting and matching, camera pose estimation, and local camera-pose optimization.

A visual SLAM system should include the following building blocks: features extraction, feature matching, motion estimation, and bundle adjustment. In the section, we will introduce the function of each system block.

Related Work

Visual SLAM related work could be divided into three categories based on their methodologies: filtering methods, feature-based methods, and direct methods. Important visual SLAM efforts are discussed in the following section.

Extended Kalman Filter Based SLAM

One of the most fundamental work of Visual SLAM is Davison's work [1, 11]. A monocular camera is used to track the environment. Landmarks (features) are extracted from images by the Shi and Tomasi operator. Davison adopted a Bayesian approach and used Extended Kalman Filter (EKF) for the camera pose estimation, which become the framework for the later SLAM work. However, the drawback is that the computation cost is high because the covariance matrix grows quickly while the map is large.

PTAM

To solve the high demanding of camera-pose optimization, Klein [12] proposes that the tracking and mapping task are separate into two parallel tasks and the tasks are executed in concurrent threads. The algorithm is commonly used in the modern visual SLAM systems.

LSD-SLAM

LSD-SLAM is significantly different from the feature-based SLAM methods. Using the skills of photo alignment, Engel [2] directly compared the pixels between two keyframes in the LSD-SLAM. Without extracting features, LSD-SLAM preserves more details in the images and creates a point cloud map that is more meaningful to humans. However, the computation cost is higher than feature-based methods.

ORB-SLAM

The ORB-SLAM [3] algorithm is categorized into the feature-based method. So far, the ORB-SLAM system presented by Mur-Artal has the best performance. The name of the algorithm comes from the feature descriptor: ORB (Oriented FAST and Rotated BRIEF). ORB feature descriptor is evolved from the FAST [13] and BRIEF [14] descriptor. Rublee [15] proposed the ORB descriptor to replace the traditional SIFT (Scale-Invariant Feature Transform) feature detector to improve the performance of a SLAM system. The main advantages of adopting the ORB descriptor include rotation invariance and high noise intolerance. ORB descriptor is faster and more accurate than the original FAST descriptor. When compared to BRIEF, ORB has a lower consumption on computing. The efficiency of ORB feature descriptor enables a better image-matching ability and fast computation. Thus, the benefit of ORB descriptor allows ORB-SLAM to have a good performance.

ORB-SLAM is the current state of the art visual SLAM algorithm. The algorithm followed the framework of PTAM, and included modules that effectively improved the performance of SLAM system. It performs well in the KITTI dataset benchmark [3].

CHAPTER II

CHALLENGES IN MONOCULAR VISUAL SLAM

System Diagram of Visual SLAM

Before we start to illustrate the difficulties and challenges in monocular visual SLAM, we want to introduce the overall system of visual SLAM. Figure 2.1 is the system diagram of ORB-SLAM. ORB-SLAM is the state-of-the-art in visual SLAM area. We will explain how visual SLAM works with the assistance of ORB-SLAM system diagram. Visual SLAM consists of three main parts: (1) Local tracking and robot pose estimation, (2) Optimization engine, and (3) Map construction. When the robots achieve another new frame, the visual SLAM system will start. After the system determines that the new frame contains sufficient interest points for recognition, it estimates the relation robot pose from those interest points.

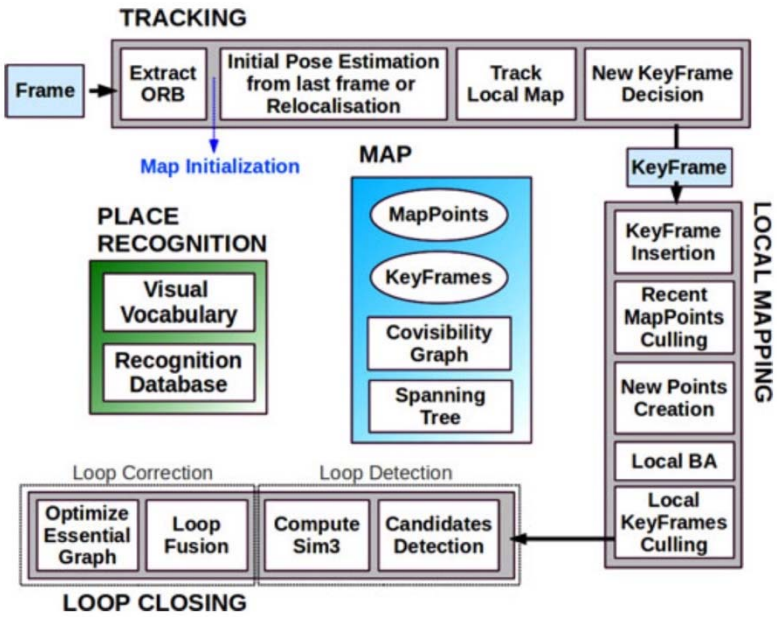


Figure 2.1: System diagram of ORB-SLAM (adapted from [3])

In Figure 2.1, local tracking and robot pose estimation correspond to the tracking and local mapping except for local bundle adjustment (BA). In this part, the robots will utilize the extracted interesting points to estimate their local poses. However, since there exists uncertainty from sensors, the local robot poses are not accurate enough from the long-term trajectory perspective.

To raise the correctness in pose estimation, it relies on the optimization engine in each visual SLAM system. In ORB-SLAM, it utilizes local bundle adjustment (LBA) to optimize the robots poses within a consecutive fixed number of frames to achieve the scenes consistence. At the same time, the robot poses are able to help the robots to estimate the environment outlook. Map construction keeps updating the map from the relation between robots poses and the scenes.

Visual SLAM system is a robust system. Each building block corresponds to a different role in assisting robots to understand and localize the surroundings. Therefore, if the part of local tracking and the part of robot pose estimation generates wrong robot pose, it will result in the following optimization and map construction with wrong result. No matter how well the optimization engine is, the wrong local robot pose is not able to recover. Hence, we are interested in studying in the parts of local tracking and robot pose estimation.

Visual SLAM Challenges

In addition to the difficulty in local tracking and robot pose estimation, different sensors have different challenges. Here we use a monocular camera in our project since the monocular camera is the easiest accessible sensor these days. Nowadays almost everyone owns a smartphone that is mounted a monocular camera. We will address the challenges we face when we use monocular cameras in visual SLAM.

Monocular cameras are precise bearing sensors, contain rich information in the projected images. However, cameras are sensitive to the light condition. When the environment is too dark, or too bright, it will be hard for cameras to recognize the environment and lose its advantage in containing rich environment information on its image plane. Therefore, it is necessary for us to aware how well the brightness is, during the process of visual SLAM.

Moreover, monocular cameras lack of absolute depth and scale information. The cameras only can accumulate the scale estimated from the previous frames. Therefore, if there are scenes that monocular cameras are able to register in the consecutive frames, the monocular cameras can recover the relative scale among them. However, once the current scene does not contain any similar scene compared with the previous frame, the monocular camera will lose the relative scale. It only can re-estimate a scale that is not related to the previous consecutive frames. This problem often happens when the robots face a turn in a corridor. If the robot changes its view with a direct 90-degree turn, the robots will lose the recognizable scene and the connection with the accumulated scale. Hence, the estimated trajectory will exist different scale on different segments. To tackle this problem, we usually make robots turn with small translation movement rather than a direct 90-degree turn.

Above problems are able to handle whenever we are careful with the brightness condition and follow the turning policy. Nevertheless, the discussion of these two types of problems is based on the assumption that the environment has recognizable scenes. If there are no recognizable scenes or exist scenes which are hard to recognize like repetitive, or textureless patterns, the robot will be confused during the navigation. In the human-made environment, there are lots of rectangular structures that are repetitive. Those rectangular structures are similar for robots to recognize. Additionally, textureless patterns result in no referable information for

robots. Without any recognizable or referable scenes, robots will lose the ability to induct where it is and what the outlook of the environment is. Hence, the human-made environments like corridor are always challenging for visual SLAM. We are interested in how the performance of ORB-SLAM will be if we adopt ORB-SLAM in human-made environments, and focus on the result of tracking and local mapping part.

CHAPTER III

PROBLEM STATEMENT

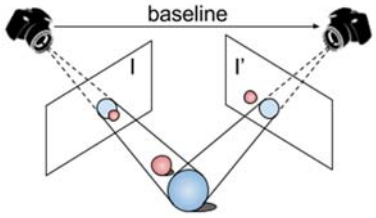


Figure 3.1: Two-view geometry

Problem Statement

Assumption

Consider that a mobile robot equipped with a monocular camera explores an indoor environment, such as a corridor or a passageway. In two-view geometry (Figure 3.1), we compare two sequential keyframes each time. We assume the following conditions are satisfied.

- The scene is mainly surrounded with planar surfaces, such as ceilings, floors, and walls. Either textureless surfaces or repetitive patterns dominate the scene.
- The monocular camera on the robot remains the same height while the robot is moving. The camera has been calibrated, and the lens distortion has been corrected in the input images.

Notation

In the thesis, we use the superscript ′ to denote the corresponding relation in the second view. The coordinate systems here are the right-handed coordinate system. The notations used in the thesis are defined as the following.

- *I:* the first input image frame

- I': the second input image frame
- $\{W\}$: 3D world coordinate system
- $\{C\}$: 3D camera coordinate system
- $\{I\}$: 2D image coordinate system
- x: feature point in the image I
- X: feature point in the 3D world coordinate system
- R, t: the rotation matrix and translation vector
- H: the 2D homography matrix
- F: the fundamental matrix
- K: the intrinsic camera calibration matrix

Problem Definition

Given two consecutive image I and I' in the textureless indoor environment, verify camera poses (R, t), find the camera model, and compare the trajectory of camera poses with ground truth.

To solve the problem, mathematical background about two-view geometry will be introduced. In the following sections, we will show how 3D world object is represented in a 2D image in the camera model section. In the two-view geometry, the relationship between the two images can be represented by a fundamental matrix. If the observing objects are located on a planar surface, the relation between the two image frames can be simplified as the homography matrix.

Camera Model

The behavior of a monocular camera can be represented by a pinhole camera model. Figure 3.2 shows a pinhole camera geometry. A pinhole camera is located in the camera

center C that is the origin in the 3D camera coordinate system. The pinhole camera observes an object X in the 3D world coordinate, and project X to x in a 2D image plane. The projection is a linear mapping from the point $(X, Y, Z)^T$ in 3D world space to the point $(x, y)^T$ in image space. We can acquire the position of x by the property of symmetry. The linear mapping can be described as

$$(X, Y, Z)^T \rightarrow \ (x, y)^T = \left(\frac{fX}{Z}, \frac{fY}{Z}\right)^T \tag{3.1}$$

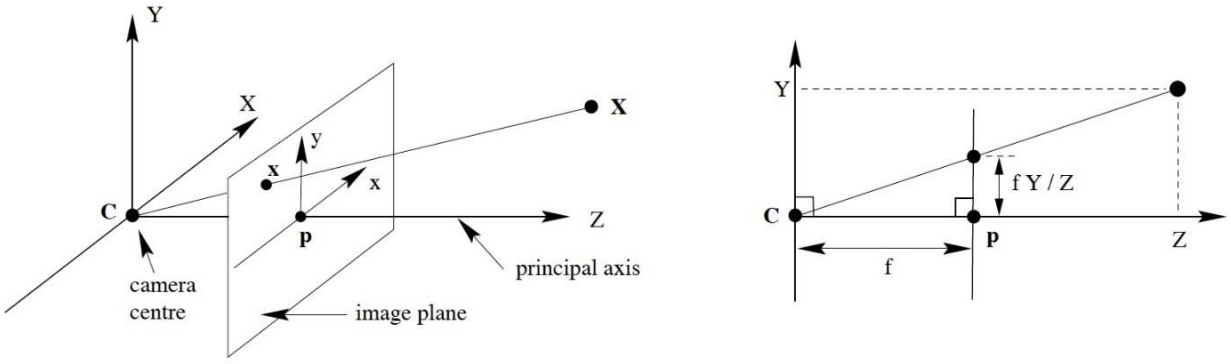


Figure 3.2: Pinhole camera geometry (a) the projection from 3D world to 2D image (b) the symmetry property (adapted from [5])

We use homogeneous vectors to represent the world point X and the image point x, and write the linear mapping in terms of matrix multiplication. Equation 3.1 may be written as

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.2}$$

In Figure 3.2, the principal axis is the normal vector to the image plane, and it connects to the camera center C. The point that the principal axis goes through the image plane is the principal point p. In Equation 3.2, we assume that the principal point is the origin of the image plane. In practice, the assumption may not be true, and there may be an offset (p_x, p_y) to the origin of the image plane.

Considering the principal point offset, Equation 3.2 becomes

$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.3}$$

$$K \equiv \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

The matrix K is defined as camera calibration matrix, which records the intrinsic parameters of the camera. Thus, the linear mapping from the 3D world coordinate to the 2D image coordinate can be described as the following equation.

$$x_{\{I\}} = K[I \mid 0]X_{\{C\}} \tag{3.5}$$

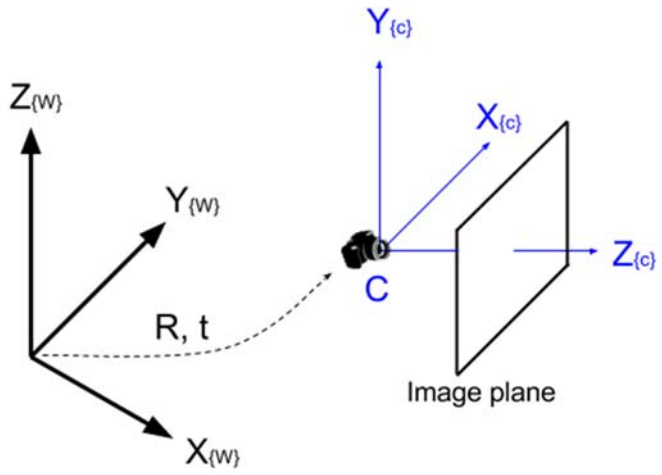


Figure 3.3: Transfer from the world coordinate system to the camera coordinate system.

In practice, the camera coordinate system will not equal to the world coordinate system. (See an example in Figure 3.3.) The two coordinate systems can be related through an Euclidean transformation, a rotation matrix R and a translation vector t. If $C_{\{W\}}$ represents the coordinates of the camera center in the world coordinate system, then $X_{\{C\}} = R(X_{\{W\}} - C_{\{W\}})$.

Thus, the Euclidean transformation can be written in homogeneous form.

$$X_{\{C\}} = \begin{bmatrix} R & -RC_{\{W\}} \\ 0 & 1 \end{bmatrix} X_{\{W\}} \tag{3.6}$$

Equation 3.5 can be written as

$$x_{\{I\}} = K[I \mid 0] \begin{bmatrix} R & -RC_{\{W\}} \\ 0 & 1 \end{bmatrix} X_{\{W\}} = KR\left[I \mid -C_{\{W\}}\right] X_{\{W\}} \tag{3.7}$$

Finally, a camera matrix P is defined to describe the linear mapping between the image coordinate system and world coordinate system.

$$x_{\{I\}} = PX_{\{C\}} \tag{3.8}$$

where

$$P = KR\left[I \mid -C_{\{W\}}\right] = K[R \mid t] \tag{3.9}$$

For added generality, non-ideal intrinsic factors are considered in camera calibration matrix K. First, image coordinate systems use pixels as units. The number of pixels in a camera may be different in the x direction and y direction. Scale factors (m_x, m_y) are introduced. The focal length of the camera becomes (α_x, α_y) where $\alpha_x = fm_x$ and $\alpha_y = fm_y$. The principal point of the camera will also turn into (x_0, y_0) where $x_0 = p_x m_x$ and $y_0 = p_y m_y$. Second, a skew parameter is adopted to check whether the x-axis is perpendicular to the y-axis. In most cases, the skew parameter should be zero. Thus, the intrinsic camera matrix

$$K \equiv \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

Homography Matrix

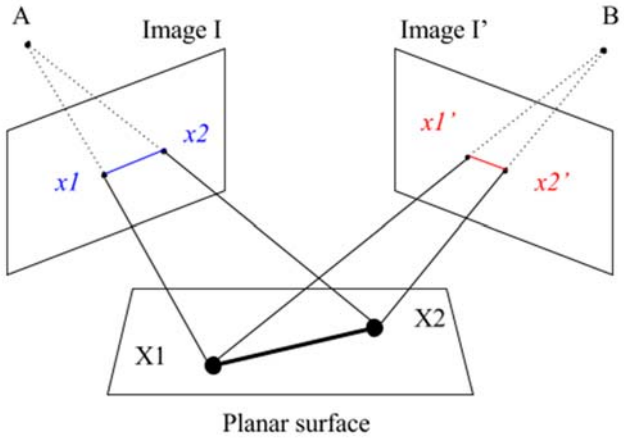


Figure 3.4: The homography relation between two views

Figure 3.4 shows the relationship between two consecutive images taken by the monocular camera on a mobile robot. A set of points x_i' in the second image I' can be regard as a projective transformation of a set of corresponding points x_i in the first image I. According to Hartley and Zisserman [5], each image is in the projective space P^2. A linear transformation mapping from P^2 to P^2 can be represented by a non-singular 3×3 matrix.

$$\begin{pmatrix} x_x' \\ x_y' \\ x_z' \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_x \\ x_y \\ x_z \end{pmatrix} \tag{3.11}$$

The matrix multiplication can be written as the following equation.

$$x' = Hx \tag{3.12}$$

The 2D homography matrix H contains nine elements. Because h_{33} is defined up to scale, it is generally set as one. The number of degrees of freedom becomes eight for the projective transformation. For a point in the image, the number of degrees is two. To solve the homography matrix, we need to find at least four corresponding points to provide eight equations. Thus, an exact solution for the homography matrix requires four points correspondences in two consecutive images.

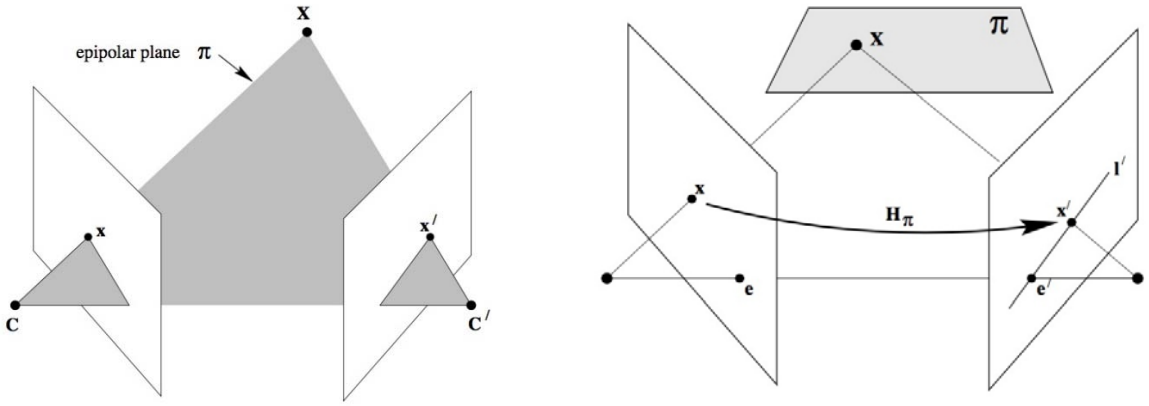
In reality, we calculate an approximate solution for the 2D homography matrix because the measurement of points contains noise. If there are more than four feature points correspondences selected from images, a homography matrix solution may not be satisfied with all of the feature points. The system of linear equations is overdetermined. An approximate solution is expected by minimizing an appropriate cost function. The goal becomes to find the best solution with the minimized errors. The error can be measured by different cost functions. Commonly used cost functions include algebraic distance, symmetric transfer error, reprojection error, and Sampson error. By minimizing the cost function, an approximate solution for the homography matrix H can be determined.

According to Hartley and Zisserman [5], the Gold Standard algorithm is used to compute the 2D homography matrix. First, four corresponding points can determine the homography matrix H. An initial estimate of H can be computed by the normalized Direct Linear Transformation (DLT) algorithm, or the Random Sample Consensus (RANSAC) algorithm. Second, an error vector is evaluated by the selected cost function. Iterative minimization methods, such as Newton algorithm or Levenberg-Marquardt algorithm, will be adopted to minimize the error vector. When the cost function obtains a minimized error vector, the optimal solution for H is acquired.

Fundamental Matrix

According to Hartley and Zisserman [5], the epipolar geometry is a geometry between two perspective cameras. The fundamental matrix F is used to describe the two-view relationship in the epipolar geometry. When an image is taken by a pinhole camera, the image loses the depth information. In Figure 3.5 (a), we take photos of an object in 3D world coordinates system from

two different positions. If we would like to know the depth of the object, we will have to recover depth information from the two images.



(a) Epipolar geometry (b) The fundamental matrix

Figure 3.5: The epipolar geometry and the fundamental matrix (adapted from [5])

In Figure 3.5 (a), the plane composed of X, C and C' is called epipolar plane. Baseline connects the two camera centers C and C'. The baseline will be contained in the epipolar plane. In Figure 3.5 (b), the epipoles are defined as the intersection between baseline and the image plane. Also, camera center C can project to the other image to form the epipole e'. When the epipolar plane intersects with the image plane, the epipolar line l' are generated. From Figure 3.5 (b), the line from image point x to camera center C can project to the the epipolar line l'. Thus, the fundamental matrix F are used to represent the linear mapping from x to l'. For any corresponding pair from x to x', the fundamental matrix has a property that it satisfies

$$x'^T F x = 0 \tag{3.13}$$

Same as the homography matrix, the fundamental matrix can be computed by its Gold Standard algorithm. First, a normalized 8-point algorithm will be used to obtain an initial value. Second, the initial value of the fundamental matrix F can be used to solve the camera matrix P and P' where $x = PX, x' = P'X$. Third, the fundamental matrix F and X can be jointly solved by

Equation 3.13 while a cost function is iteratively minimized. The cost function is a geometric distance, which is defined as

$$\sum d(x, \hat{x})^2 + d(x', \hat{x}')^2 \tag{3.14}$$

where the $\hat{x}$ and $\hat{x}'$ are true values that satisfy $\hat{x}'^T F \hat{x} = 0$ and the rank of F is two. The minimization can be computed iteratively by using the Levenberg-Marquardt algorithm. When the minimization converges, we can obtain the fundamental matrix F.

CHAPTER IV

EXPERIMENTS AND RESULTS

Although the ORB-SLAM has a good performance in the outdoor setting, no sufficient experiments and data support that ORB-SLAM also performs well in the indoor environment. We conduct a series of experiments to verify the indoor performance of ORB-SLAM. The challenge to apply SLAM in the indoor environment is that there are plenty of planar structures in the indoors. The purpose of the experiments is to examine whether ORB-slam could be correctly applied in the indoor textureless environment.

ORB-SLAM adopts a model selection method based on the characteristics of input images while other SLAM algorithms generally use one fixed model based on their main purpose. Therefore, ORB-SLAM can select a better model for scene description. Whether planar structures are dominated in the scene will affect which model should be adopted. If planar structures dominate the scene, a homography matrix will be used. If the input scene is not planar, a fundamental matrix will be selected.

Textureless Environment Examination

The characteristics of features

To begin with, we would like to examine what a textureless environment is. A textureless environment means that the number of feature points in the environment is low. The simplified definition of features is that features are unique, specific patterns, or distinct from other elements in the image. Features can be easily identified and tracked during the comparison of images. Features normally distributed in the regions that have the maximum variation or gradient.

Image features are important. The reason is that feature points are used as landmarks references, and the characteristics could help the robots to accurately determine the camera poses. The baseline distance, the rotation matrix, and the translation vector between two image frames can be estimated from the relationship between feature points. If there are plenty of features in the scene, more landmarks can be used to compare the difference between the two images. Take Figure 4.1 as an example, we try to match two corresponding key frames. In the left image, we can still see a chair. In the right image, the chair cannot be found. We can find that most features appear at the unique black and white regions. These matched features can help us estimate the robot pose. Because the monocular camera remains the same height to the ground, only the lines that are parallel to moving direction of the mobile robot provide correct matching results.

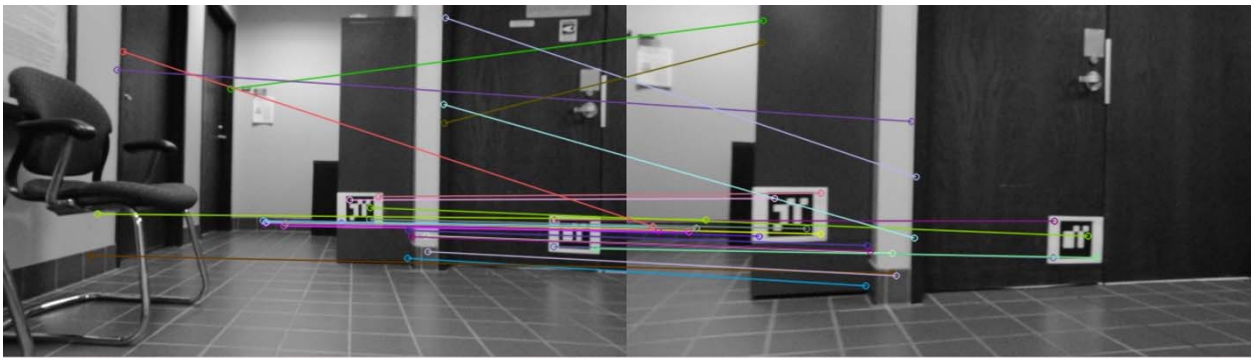


Figure 4.1: Features matching between two sequential key frames.

Experiment 1

A good feature descriptor should be rotation-invariant. That is, features can still be identified even after an image rotation. The Scale-Invariant Feature Transform (SIFT) [8] feature descriptor not only is rotation-invariant, but also is scale invariant. We use the SIFT feature descriptor to detect features in the dataset. The goal of the experiment is to help us identify the number of feature points and how features are distributed in the image dataset.

(a) Features detection in an office (b) Features detection in a corridor

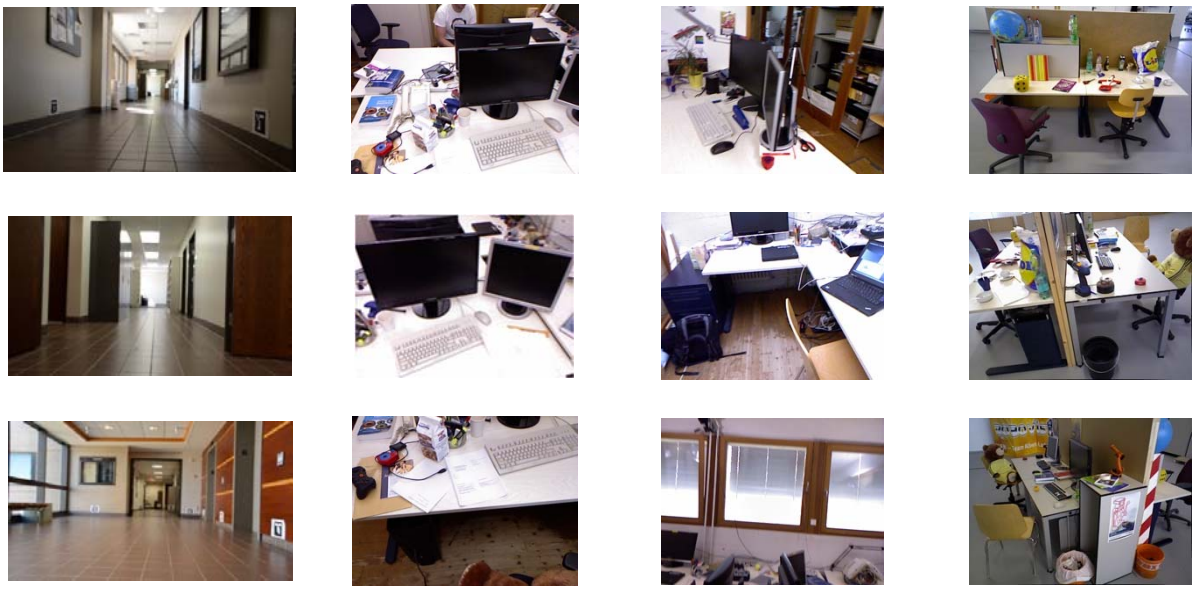
Figure 4.2: Features Detection in two different environment (a) an office (b) a corridor

Figure 4.2 (a) is one of the images in the ORB-SLAM dataset. The image describes a typical office desk environment. The colored circles show the keypoints that are identified as features by the SIFT features detector. Figure 4.2 (b) belongs to our image dataset, HRBB 4th floor, and it shows a typical corridor environment. Although both images are taken in the indoor environment, we can find the following difference.

- Features are seldom located in the white desk, white walls, and dark regions. Also, features do not often appear at repetitive patterns, such as floor tiles and ceiling.
- The corners of rectangles are often picked as features.

Experiment 2

In the second experiment, we would like to compare the number of features in the different dataset. In Figure 4.3 (a), the HRBB 4th floor dataset is our image dataset. Our mobile robot took the photos in the corridors of Computer Science & Engineering department. Figure 4.3 (b), (c), and (d) are the image dataset used in the ORB-SLAM to test the indoor performance.



(a) HRBB 4th floor (b) freiburg1_xyz (c) freiburg1_room (d) freiburg3_long office_household

Figure 4.3: Examples of images dataset. We use dataset (a) in the experiment. ORB-SLAM uses dataset (b), (c), and (d) to test.

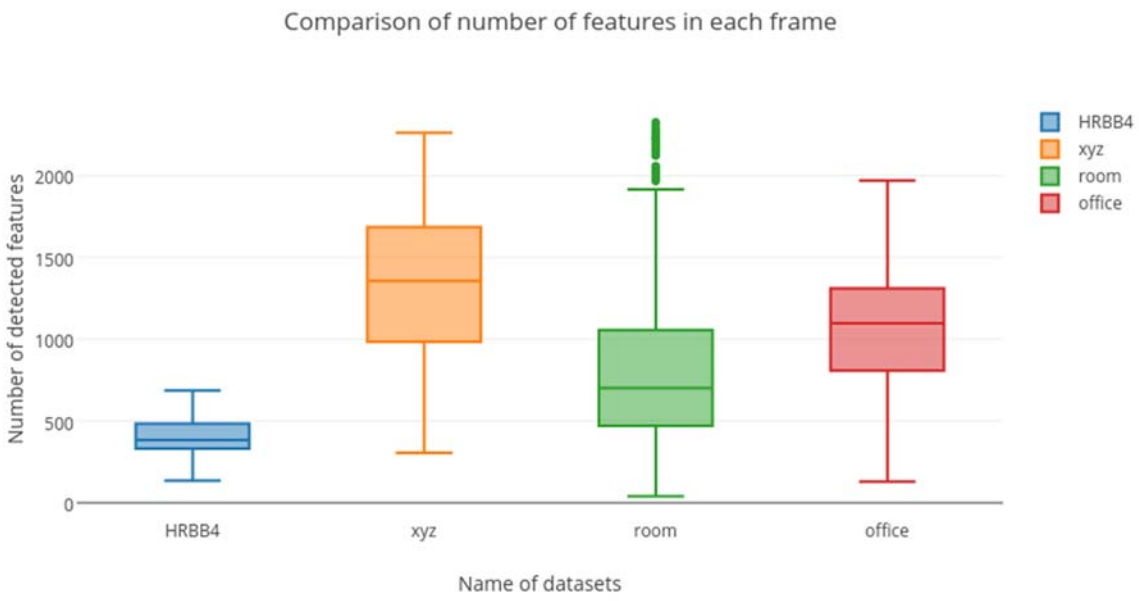


Figure 4.4: Comparison of number of features in each frame between different datasets

# of features	Our dataset	ORB dataset		
in each frame	HRBB4	xyz	room	office
Mean	**399.8**	1328.4	813.6	1054.3
Standard deviation	**104.7**	435.4	461.6	346.7
Total frames	239	798	1362	2585

Figure 4.5: The mean and standard deviation of the number of features in different dataset

Our image dataset, HRBB 4th floor, is surrounded by textureless walls and floor tiles. The image datasets in ORB-SLAM are taken mainly in the office environment. We would like to compare whether the input scene affects the number of features. In Figure 4.4, we can compare the number of features in each frame between the four image datasets. We use a boxplot to show the distribution of the number of features in each dataset. The boxplot contains five important parameters: minimum, maximum, first quartile, third quartile, and median. We consider that most of the inliers are distributed in the interquartile range, the range between the first quartile to the third quartile. Also, we calculated the mean value and standard deviation in Figure 4.5. We can conclude the following points from Figure 4.4 and Figure 4.5.

- The average number of feature points in HRBB 4th floor dataset is much lower than other datasets, and it meets our understanding that human-made environment is textureless.
- The low average tells us that human-made environment like HRBB 4th floor dataset generally lacks of feature points.
- The probability of features mismatching in a human-made environment like HRBB 4th floor dataset is high (See Figure 4.1) since there are less extracted feature points, and repetitive pattern are hard to match with matching algorithm from the features descriptor.

Recover Trajectory of Camera Poses

Experiment 3

One of the goals of visual SLAM algorithm is to track the position of a mobile robot. The position of robot equals to the camera pose. By the two-view geometry, we can compare two input images and find the camera pose by the rotation matrix and translation vector. Figure 4.6 on page 30 are the screenshots during the executing of ORB-SLAM. The left image window shows the current input whereas the right window displays the map and the trajectory of camera poses. After executing ORB-SLAM, we compare the trajectory computed by ORB-SLAM to the ground truth. Figure 4.7 (a) on page 31 is the trajectory that is based on the ground truth in HRBB 4th floor dataset. Figure 4.7 (b) on page 31 is the trajectory computed by ORB-SLAM. From the experiment, we can conclude the following points.

- The shapes of the two trajectories are similar, which is close to the shape of the corridor in HRBB 4th floor dataset.
- However, the scales of the two figures are not equal to each other since the chosen keyframes are different between ORB-SLAM and our dataset. Currently, we cannot verify the details of trajectory without the correct scale yet. Further work is required to recover the correct scale from ORB-SLAM.

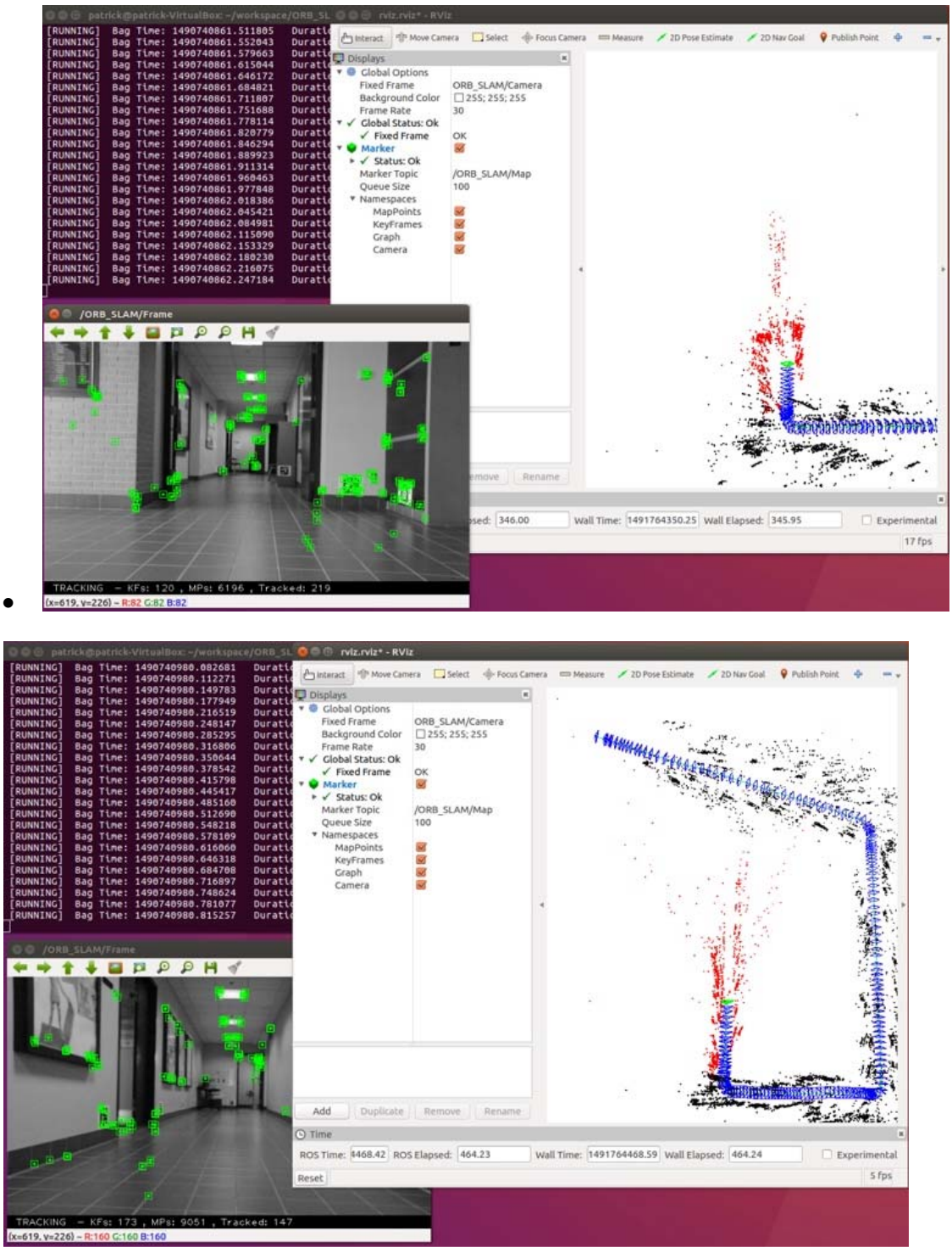
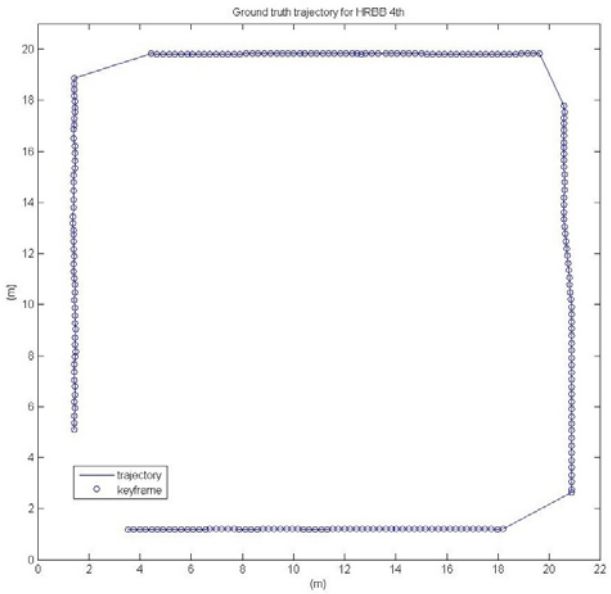
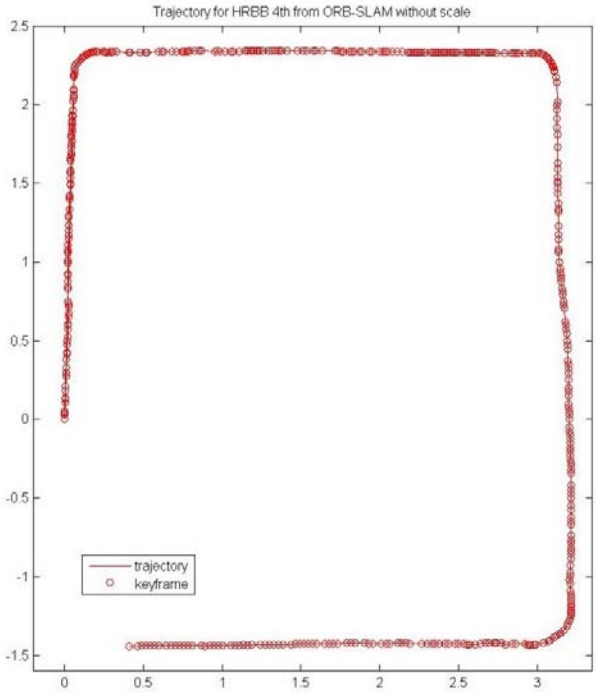


Figure 4.6: Execution screenshot of ORB-SLAM



(a) The trajectory of ground truth



(b) The trajectory of ORB-SLAM

Figure 4.7: The trajectories of camera poses

CHAPTER IV

CONCLUSION

Conclusion

In the research, we have verified that the HRBB 4th-floor dataset is a textureless environment that has a low number of features. We executed ORB-SLAM with the HRBB 4th-floor dataset. Since there is a scale issue, we only can compare the shape of the trajectory with the ground truth of trajectory. Because of the similar shape of trajectory, it raises researchers' interest in the absolute trajectory error between the two trajectories after researchers solve the scale issue.

Future work

To conduct a further evaluation of ORB-SLAM, we believe that the following items are worth to do in the near future.

- In a textureless environment, we can examine the validity of model selection method in ORB-SLAM, and compare the selected models with the scenes.
- Since ORB-SLAM uses the ORB feature descriptor to find features, it is interesting to compare the SLAM performance among different feature detectors such as SIFT, SURF or ORB.
- It is interesting to validate the absolute trajectory error to measure the performance of ORB SLAM working in a textureless environment.

REFERENCES

[1] A. J. Davison, "Real-Time Simultaneous Localisation and Mapping with a Single Camera," in *ICCV*, 2003, pp. 1403-1410.

[2] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 1935-1942.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics,* vol. 31, pp. 1147-1163, 2015.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*: MIT press, 2005.

[5] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*: Cambridge university press, 2003.

[6] S. Thrun, W. Burgard, and D. Fox, "A probabilistic approach to concurrent mapping and localization for mobile robots," *Autonomous Robots,* vol. 5, pp. 253-271, 1998.

[7] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. I-I.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision,* vol. 60, pp. 91-110, 2004.

[9] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, 2006, pp. 404-417.

[10] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine,* vol. 18, pp. 80-92, 2011.

[11] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE transactions on pattern analysis and machine intelligence,* vol. 29, 2007.

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

[13] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*, 2006, pp. 430-443.

[14] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*, 2010, pp. 778-792.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 2564-2571.