# Monocular Vision-Based Localization in Metric Space for Autonomous Vehicles

Muhammad Adi Puspo SUJIWO

## Abstract

Localization is one of the vital parts of any robotic systems, along with motion planner. Currently established motion planners work in the metric search space, which in turn necessitate the localization to be metric-based. In autonomous vehicle research community, this requirement is fulfilled by LiDAR-based systems that are bulky and expensive. An alternative to LiDAR systems is visible–spectrum camera, with advantages in cost, power consumption and size. However, vision-based localization systems are not able to work with motion planners due to scale ambiguities that are inherent in the solution of Structure–from–Motion in monocular case, and cannot be avoided. Vision-based systems also suffer from reliability problems in the long run from variability in environment lighting.

This research addresses these two problems and presents testing results in two different situations: mobile robots in pedestrian environment and an autonomous car in an urban setting. The proposed solutions are developed from the concept of separation between mapping and localization processes. This separation enables utilization of advanced sensors in mapping phase but maintains use of low-cost sensors for localization. To measure effectiveness of this system, two performance measurements are defined: accuracy of estimated positions as the deviation from ground truth measured by LiDAR-based localization in metric, and coverage (percentage of the track that could be covered by visual localization alone)

To solve scale ambiguities and provide metric workspace, visual maps are created separately by recording actual positions of the camera in real-world by employing external localization from a LiDAR-based system, which is also used to measure ground truth. These visual maps can be utilized to provide metric pose with only using a monocular camera by scaling of true positions of map keyframes. By working in metric space, it is also possible to combine results from vision-based localization with odometry data to cover situations where camera tracking is lost by an implementation of a particle filter. In order to improve the reliability of visual localization in long run, this research also proposed automatic gamma control and custom vocabulary for each scene.

Testing in pedestrian environment addressed two issues: performance of the visual localization system in real-world setting and reliability for long-term localization. In general, average errors of visual localization in pedestrian settings tend to be low (under 1 meter) but some error spikes larger than 3 meters had been measured. Coverage in the pedestrian environment was also good (approaching 99 %), especially after applying automatic gamma control. Meanwhile, testing in urban road settings with higher velocity revealed lower coverage than pedestrian settings which corresponds to the inability of the place recognition system to keep pace with vehicle movement. This testing also shows higher errors compared to the pedestrian setting. However, these errors were measured to be lower than GPS system.

These results provide proof that autonomous vehicle navigation using vision sensors and other inexpensive sensors is possible. The key ingredients are precise visual metric maps created by third parties equipped with a camera and accurate localization sensors.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Since the DARPA Grand and Urban Challenges were held [1, 2], public has begun to realize the possibilities of autonomous vehicles. Automation techniques, which were previously only known in robotic research communities, are now closely within grasp for public consumption in the street. The most often cited benefits of vehicle automation are reduced accident rates, increased traffic flows and fuel efficiency, and possibilities for performing other tasks inside the car [3].

One of the vital tasks for robots and autonomous cars is localization as part of greater scopes of navigation [4]. The localization problem is defined as finding current robot position in order to determine the way it should be going. Therefore, the localization problem has a close relationship with cognition (deciding how to reach the goal) and motion control (modulating motive powers to achieve the desired trajectory). In order for navigation to work, it is important for these three parts to use same metric space[1] with centimeter-level accuracy.

The current state-of-the-art for localization solution is by using LiDAR as shown by Google [5]. However, this solution incurs high cost in part of manufacturers; not to mention its bulk and power requirement. Therefore it is desirable to have low-cost sensors for localization with similar capability for integration with existing motion planning and control systems.

This research proposes a vision-based localization system with a capability for integration with existing motion planners inside metric space (Figure 1.1). The main concept of this system is the principle of separate mapping and localization process, similar to most LiDAR-based systems explained in [6]. Separate mapping and localization process will allow system builders to employ high-end sensors to build the maps but kept using camera-only in the localization, as required for consumer-level devices.

In addition to cost and size, another benefit of visual localization in technical level

[1] Topological space, at which distances are not defined, is used in some combined vision-based localization and motion control systems.

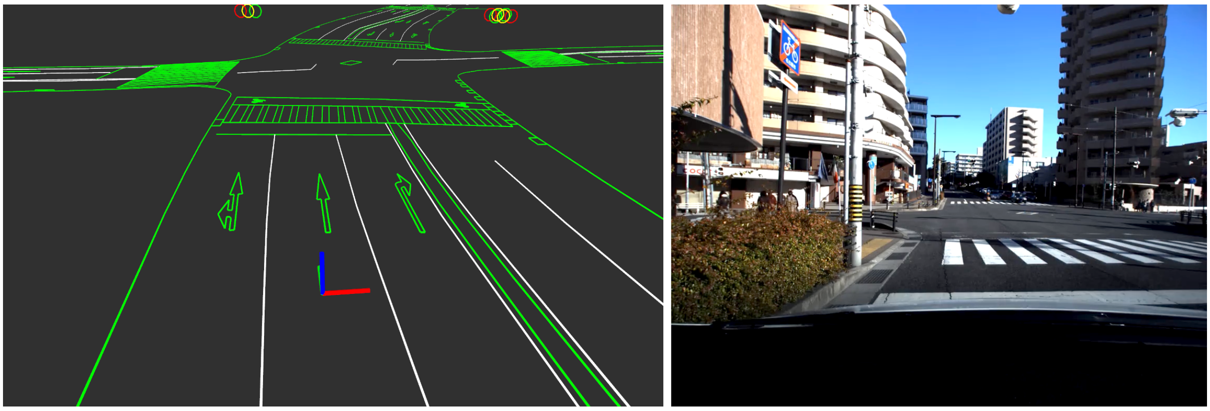


Figure 1.1: Determining vehicle localization using only a monocular camera (right) in metric space allows integration with motion planners and existing infrastructure maps (left).

is its capability to perform global localization by using built-in place recognition. This term refers to robot capability to initialize position without any other aids, something that previously can only be performed by GPS. Meanwhile, a camera does not suffer from signal disturbances as much as satellite-based systems may have. Therefore, vision systems have potential as location initializer for LiDAR-based localization instead of GPS as currently practiced.

Obviously, the visual localization system still has drawbacks, which are also identified in this research. Most important one is its sensitivity to environmental changes such as lighting and occlusions. Therefore, we also proposed some changes in localization workflow that attempts to address this shortcoming. These changes become important, especially for longer time intervals between mapping and localization.

## 1.1 Problem Statements

The main objective of this research is to develop monocular vision-based localization system in metric space for autonomous vehicles that can be integrated to currently developed motion planners. In order to accomplish this task, we start from existing visual SLAM methods and add a capability to estimate metric positioning. Obviously, this metric positioning is not achieved solely by using vision but using multiple sensor modes.

In addition to solving the visual-metric problem, reliability problems could also hamper system deployments in consumer vehicles. Therefore, these problems must be identified in a safe environment (i.e. not vehicle in urban settings). Therefore, system tests must be performed using mobile robots in a pedestrian environment. The annual Tsukuba Challenge gives excellent opportunities for this pedestrian environment. Having multiple years data from similar location also provide an opportunity to develop capabilities for lifelong localization, which will be useful in urban settings.

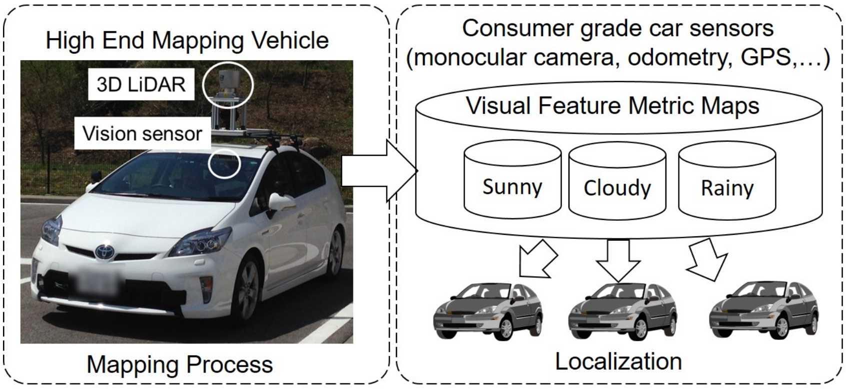

High End Mapping Vehicle
3D LiDAR
Vision sensor
Mapping Process
Consumer grade car sensors
(monocular camera, odometry, GPS,...)
Visual Feature Metric Maps
Sunny
Cloudy
Rainy
Localization

Figure 1.2: System Concept

Lastly, real tests in urban settings are conducted to identify problems in visual localization system under development. In this regard, we would like to evaluate performance in terms of coverage/availability and accuracy in metric. In addition, we need a comparison with GPS as a major sensor type currently installed in consumer vehicles.

## 1.2 Contributions

The main contribution of this research is a monocular vision-based localization system for consumer-grade autonomous vehicles. To realize this system, the concept of separate mapping and localization process are introduced, as shown in Figure 1.2. This concept is further broken down into three parts.

1. **Solution to monocular vision-based localization in metric space using augmented maps**. As shown in Figure 1.2, the localization system separates mapping and localization phase. Maps for localization are built using visual information from a monocular camera and augmented with accurate external localization system. This augmentation is used to convert camera coordinate from visual space into metric one that is suitable for motion planners in localization phase. It must be emphasized that this external localization is only used for mapping but not for localization. By working in metric space, it is possible to combine results from our method with other metric sensors such as odometry and GNSS. In this research, the visual localization system is expanded to utilize multiple maps with a single camera and fuse them with odometry using simple particle filter in 2D space.

2. **Identification and solutions for reliability problems in lifelong visual localization system**. After solving scale problem, the next problem to be tackled is reliability problem, which generally consists of low availability and inaccurate results. Solutions for reliability problems are separated into three parts:

   (a) Utilization of custom vocabulary of the particular scene in order to increase place recognition performance and improve localization availability.

   (b) Automatic gamma control for adjusting image appearance during high contrast situations.

   (c) Expanded policies of keyframe search in order to accelerate relocalization after lost events.

3. **Experimental results of the method in two different settings in the real environment: mobile robots in pedestrian environment and a passenger car in urban roads**. Pedestrian environment testings were performed in both Tsukuba Challenge 2015 and 2016. The 2015 test was intended as a proof-of-concept for our scale correction and sensor fusion methods. The 2016 test was aimed at improving availability and accuracy of our method while investigating the possibility for long-life localization operations. Meanwhile, the urban road test was performed with a similar method with 2015 test using a faster passenger vehicle and aimed to find problems in high-speed operation.

## 1.3 Thesis Structure

This thesis is divided into six chapters. In Chapter 1, research background and contributions are stated. Next, Chapter 2 describes general issues of localization for autonomous vehicles and followed by specific issues of vision-based localization.

In Chapter 3, solution for visual localization using a monocular camera in metric space are described. Also, sensor fusion and utilization of multiple maps are described in detail here, along with testing in 2015 Tsukuba Challenge.

Following the first Tsukuba Challenge results, improvements of our method are described in Chapter 4. The listed improvements are the utilization of custom vocabulary of the particular scene, automatic gamma control and expanded relocalization policies. This chapter also features results of testing of lifelong localization, whereby one-year-old map data are leveraged for the same location.

Testing of our visual localization method in public road environment is described in Chapter 5, using mostly the same method from Chapter 3. Various situations for mapping and localization are described in here, along with accuracy comparison with a GNSS-based localization system.

Finally, conclusions of this research are listed in Chapter 6 along with directions of future works.

# Chapter 2

# Literature Studies

In the robotics and autonomous vehicle literatures [7, 8, 9], navigation usually has three principal components:

1. To accurately determine position and velocity

2. To plan and execute necessary motions towards its destination

3. To detect and avoid dynamic obstacles along the path

The first and last components are necessary to achieve the second one accurately and safely. However, the second component is usually restricted to motion planning domain while the last one is for collision avoidance domain. For the rest of this work, navigation and localization will be used interchangeably.

In this thesis, the term SLAM (Simultaneous Localization and Mapping) will be broadly mentioned. As the name suggests, the SLAM problem consists of estimating position of the robot while incrementally constructing a map of the environment around it in motion [10]. Therefore the results of SLAM solutions will be used as ingredients for map-based localization systems; this is reflected in Figure 1.2. However, this thesis also discusses map-less localization systems, eg. dead reckoning and GNSS.

## 2.1 Representing Position and Orientation

The objective of robot localization system is to provide position and orientation (*pose*) of the robot. Before stepping further, this section formalizes general representation of position and orientation for next sections. As a convention, this thesis uses right-handed Cartesian coordinate system [7, 11].

In 3D space, the position of a robot is stated as a translation from a fixed point and represented numerically as $\mathbf{t} = (t_x, t_y, t_z)$. Orientation of the robot is described as rotations along its three fixed axes (roll, pitch, and yaw). Therefore, pose of a robot has

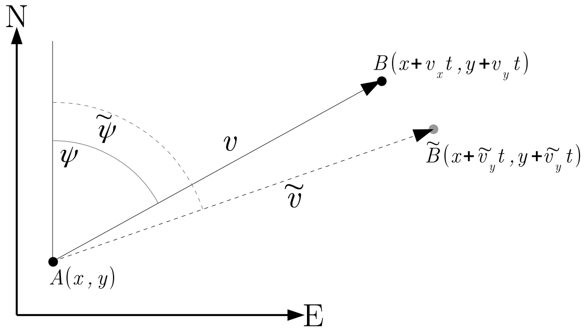


Figure 2.1: Dead Reckoning Principle

six degrees of freedom. However, calculations using rotation angles are prone to gimbal lock (loss of one degree of freedom). A popular solution to store orientation is by using quaternion $\mathbf{q} = (q_x, q_y, q_z, q_w)$ where $q_{x..w}$ are real numbers, and convert it to rotation matrix $R_{3\times 3}$ [11].

## 2.2 Non-Visual Localization Methods

This section briefly explains some methods to determine robot positions using methods other than visible electromagnetic waves. Dead reckoning and GNSS are two methods that do not require stored maps, while LiDAR localization is based on previously created maps.

### 2.2.1 Dead Reckoning

Dead reckoning is the oldest method of navigation known by human, having been practised for millennia in maritime areas and remains widely used as a backup for radio navigation [9]. It refers to a process of calculating location based on estimated speed, direction and time of travel with respect to a previously determined location (Figure 2.1).

The modern form of dead reckoning is INS (*Inertial Navigation System*), that employs inertial sensors that measure linear and angular acceleration (accelerometer, gyroscope). This system is completely self-contained and does not depend on external electromagnetic waves. Therefore, INS is highly reliable but may accumulate significant drift over a long time, primarily from instrumentation and environmental errors [9]. As shown in Figure 2.1, errors in velocity and attitude may cause localization errors.

In this research, the principle of dead reckoning is implemented by using data from IMU (Inertial Measurement Unit) instruments that are embedded in the robotic platform under use. These instruments supply 2D linear velocity and rate of rotation of the robot or vehicle. The data are employed in the calculation of odometry inside particle filter (Algorithm 3) to estimate robot position when visual localization is not available.

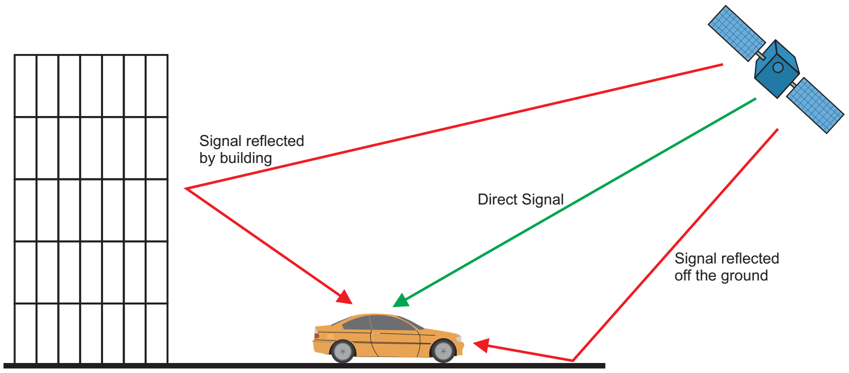


Figure 2.2: Multipath Interference

### 2.2.2 Global Navigation Satellite System (GNSS)

A GNSS[1] receiver determines its position by measuring ranges between its antenna and a set of satellites that transmit time signals at precisely known locations and then performing a trilateration algorithm to compute receiver's position [12]. This constellation of satellites circle the Earth approximately twice a day, and consists of 20 to 30 satellites in geosynchronous orbits. Adequate position measurements require minimum four different satellites; three for antenna positions in 3D space and one for solving internal clock bias.

Because of its dependency to transmitters, GNSS positioning suffers from signal disturbances that may occur during its trip. The most prominent disturbance in urban environments is multipath interference (Figure 2.2), caused by multiple reception of the same satellite signal. This effect corrupts the propagation time for calculating ranges to satellites by creating time-varying bias [13].

Currently, there are two solutions for solving general interference problems in GNSS: Differential and Real-Time Kinematics (RTK). Differential GNSS systems leverage complementary fixed networks of terrestrial transmitters in addition to satellite ones. RTK GNSS basically use tracking of wave phase; however, this system is only appropriate for static receivers.

### 2.2.3 LiDAR-Based Mapping and Localization

The LiDAR-based SLAM is a popular method for autonomous vehicle applications, and is capable to provide highly accurate maps and localization. Basic LiDAR devices are instruments that measure distances from transmitter to reflecting objects using laser [14]. To provide 3D measurements, multiple transmitters/detectors are stacked and rotated around an axis to scan the space around the device at a rapid rate (usually 10 Hz).

LiDAR-based SLAM is explained first in [15], and works principally by searching for

[1] There are a number of GNSS implementations; the most famous one is Global Positioning System (GPS) owned by US Government. Other equivalent systems are GLONASS of Russia and Quasi-Zenith Satellite System (QZSS) of Japan.

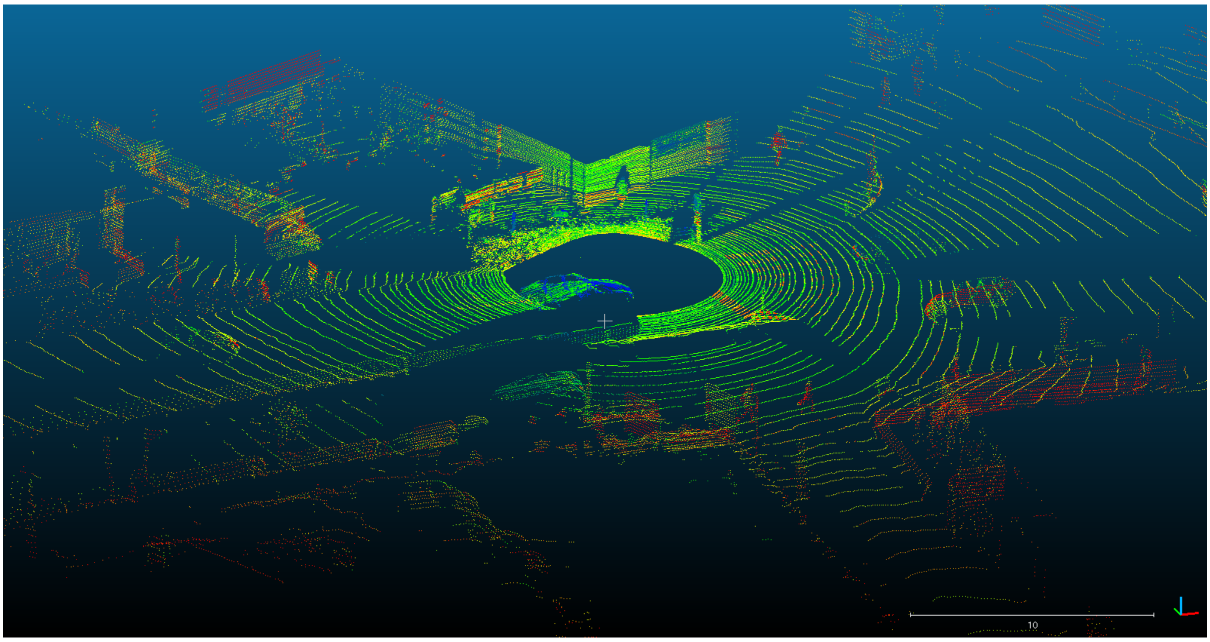


Figure 2.3: An Example of LiDAR Scan

transformation between two consecutive scans. However, this method is unable to directly provide absolute positions within a given map as it is unable to do global map search. In other words, LiDAR is unable to perform global localization[2] without assistance.

Figure 2.3 shows an example of 3D point cloud from single LiDAR scan out of Velodyne HDL–64. In the figure, object features (walls, humans, poles) around device can be easily identified, and all distances between them can be inferred in metric sense. Searching the transformation (that means rotation and translation) between two scans is performed using scan registration algorithms; the most commonly used methods are Iterative Closest Points (ICP) [16] and Normal Distribution Transform (NDT) [17] with extension in [18].

## 2.3 Visual SLAM

Vision as the only sensor mode for mapping and localization has been studied extensively in the last 10 years, starting from visual odometry by Nister et. al [19], PTAM [20], and others [21, 22]. The main reasons for this direction are possibility to obtain range information (after solving Structure-from-Motion problem) concurrently with environmental information such as color and texture, thus giving the robot capability for integration with other tasks such as object detection and recognition. Unlike previous sensors such as laser scanners and GNSS, vision cameras are passive as they do not emit energies or depend on transmitter networks, less expensive, lighter and consume less power. However, as we shall see, inferring range information from visual data will incur errors caused

[2] Global localization refers to a problem of determining the position of a robot within a map without being given any prior estimate.

by an intrinsic factor (scale ambiguity) and extrinsic factors: lighting variability, lack of texture, motion blurs and others.

Previous solutions for visual navigation usually employ stereo camera rigs, as demonstrated in [19]. This setup refers to multiple camera sensors and lenses that have partially overlapped field of vision. The benefit of this system is that depth information of the scene can be deduced using triangulation. However, this configuration has several drawbacks: difficulty in multiple camera calibration and time synchronization between both cameras. In addition, for larger distances the depth and scale information from stereo cameras degenerate into similar case of monocular cameras ([22, 23]).

Most monocular vision-based SLAM methods rely on a 3D reconstruction based on multiple views of a scene [24], which in turn is based on *structure from motion* (SfM). The SfM technique refers to the process of estimating 3D structures from 2D image sequences while inferring motion of the camera. As stated in [25] and [26], all monocular SfM methods inherit common scale ambiguities, i.e. the recovered 3D structures and camera motion are defined up to an unknown scale factor which cannot be determined from image streams alone. This is because, if the scene and cameras are scaled together, this change will be indistinguishable in the captured images (Figure 2.11). This fact results in difficulties to provide metric position of the camera, which is very important for autonomous vehicle navigation and control. Overall flow of the visual SLAM process is illustrated in Figure 2.4.

Depending on portions of pixels utilized for reconstruction, solutions for visual SLAM are classified as either *dense–type* reconstruction or *sparse–type* reconstruction. Dense reconstructions use all pixels of the image streams, with advantages of robustness against image artifacts such as noise, blur and high-frequency textures (for example, trees and asphalt). An example of this class of method is LSD–SLAM [22], whose an example result is shown in Figure 2.5. However, drawbacks of using all pixels are high CPU time per frame. In addition, relocalization accuracies have been observed to be lower than sparse reconstructions when dynamic objects are present in the scene [27, 20].

On the other hand, sparse reconstructions only track and use a limited subset of pixels, concentrating only on “features” of the image streams. These features come from image pixels that are selected by salient feature detectors (Subsection 2.3.2). Examples of this approach are PTAM [20] and ORB–SLAM [27], whose result is shown in Figure 2.10. Compared to dense approach, sparse reconstructions generate relatively small pointclouds that leads to lower computational cost. Other advantage from resulting sparse pointcloud is that image query to keyframe list becomes feasible due to fewer feature points for matching; this enables visual SLAM performs global localization using place recognition (Subsection 2.3.5).

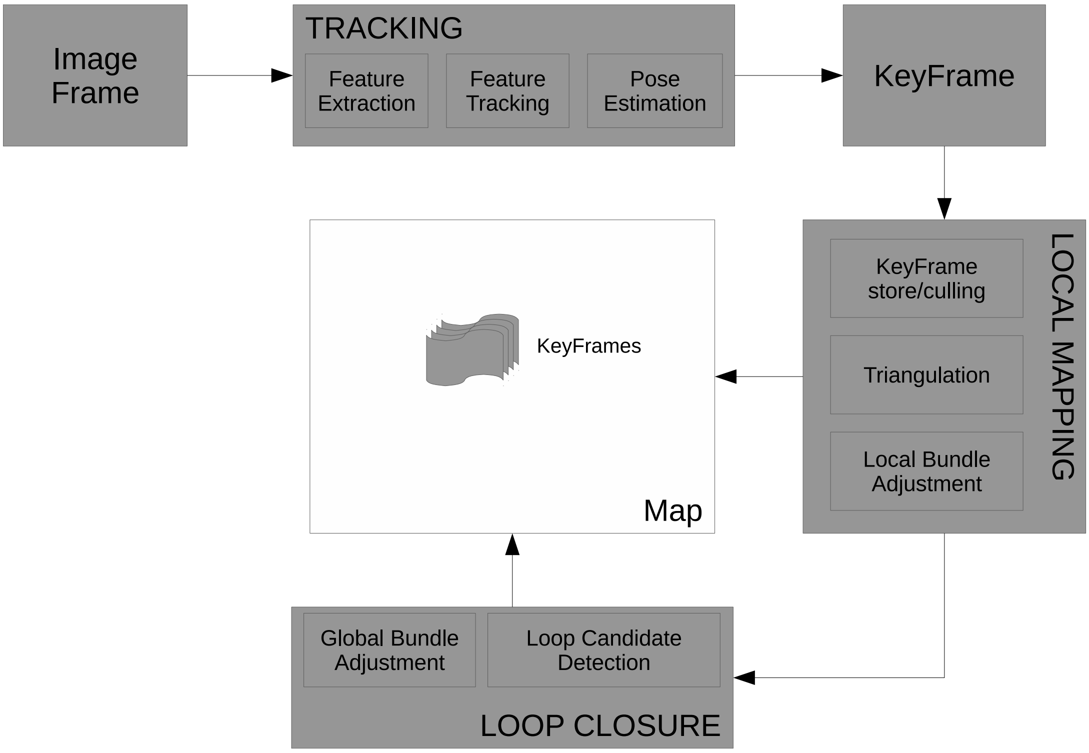
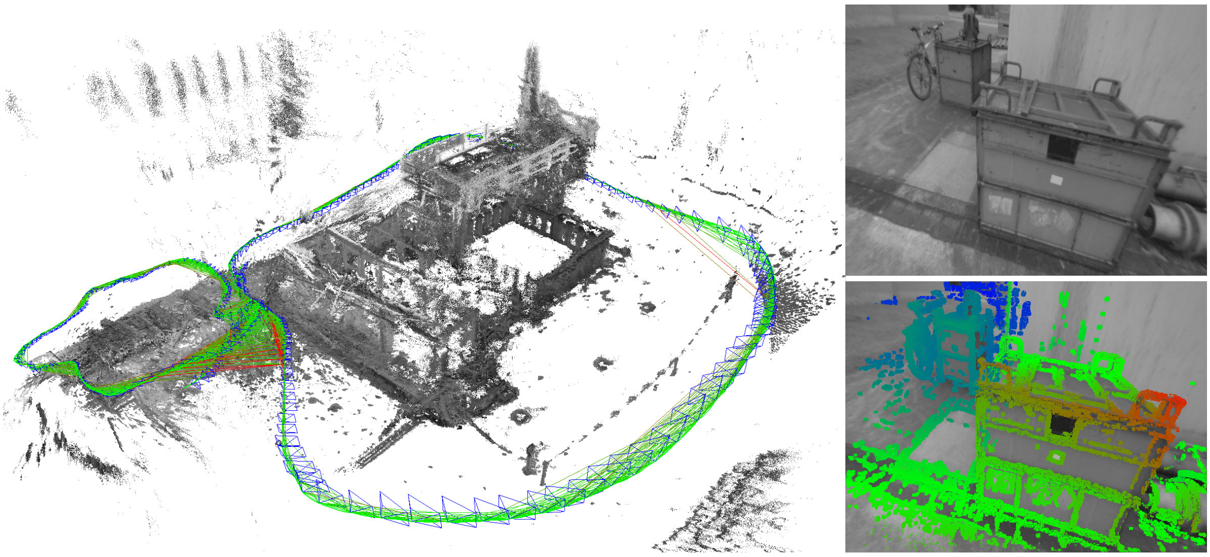


Figure 2.4: General Visual SLAM Workflow



(a) Accumulated Pointcloud

(b) Selected Keyframe with Color-Coded Depth Map

Figure 2.5: An Example Result of LSD–SLAM

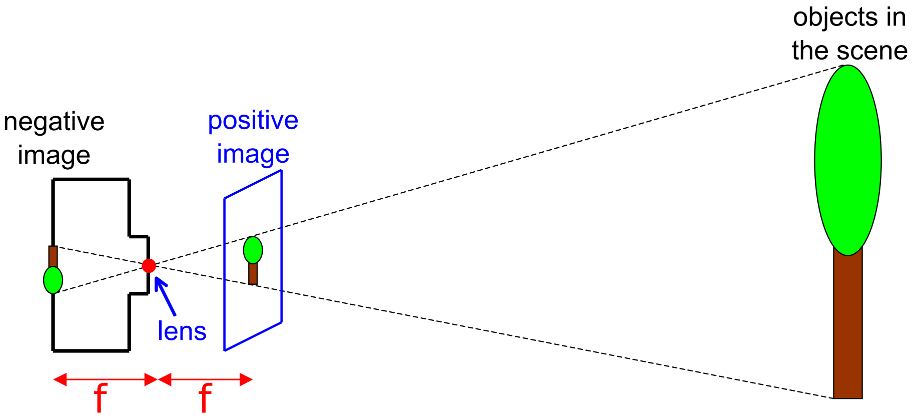


Figure 2.6: A Model of Camera Obscura [28]

### 2.3.1 Digital Image Formation

3D reconstruction in visual SLAM is an inverse process of how 2D images in a camera produced from 3D objects. In its simplest form, a digital camera can be represented by a rectangular matrix of photodetectors (image sensors) and a lens for focusing lights. This simple model, called camera obscura, is pictured in Figure 2.6. Image the from lens in the back sensor is inverted, and straight image that user usually observes in screen actually corresponds to the projection of the scene onto a hypothetical plane situated in front of the camera at the same distance from the lens as the sensor. This distance is called focal length of the camera.

Currently, the most widely used technologies for imaging sensors are CCD and CMOS. There are some fundamental differences between these type of sensors, but their advantages and disadvantages that are relevant in this work are highlighted in Table 2.1 [29, 30]. In this research, all the cameras are equipped with CCD sensors with consideration of eliminating distortion, especially in high-velocity vehicles.

Image formation in camera follows projective transformation of 3D world onto 2D image surface. Using this transformation, depth information is lost; thus it is impossible to distinguish in the image between large objects in distant place or small ones in near place. This transformation can be formulated as follows (Figure 2.7). For a point $\mathbf{X}(x, y, z)$ in 3D world space, we would like to know its projection using camera C in the 2D image plane as $\mathbf{x}(x_s, y_s)$. The camera is specified as $3 \times 4$ parameter matrix

$$K = \begin{bmatrix} f_x & s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

where the values of $(f_x, f_y)$ are focal lengths, $(c_x, c_y)$ is coordinate of principal point in image plane, and $s$ is image plane skew. These values are usually known beforehand,

[3] As of January 2018, there were no CMOS-type digital cameras equipped with global shutter

| **CCD** | **CMOS** |
|---|---|
| Advantages<br>1. Resistance to image noise<br>2. Global shutter: all rows in sensor can be captured simultaneously, therefore there is no object distortion in frame | Advantages<br>1. Low cost and power due to integration of various signal and image processing blocks (ADC, compression, etc.)<br>2. Resistance to blooming and smears |
| Disadvantages<br>1. Vulnerability to blooming and smears<br>2. High cost and power | Disadvantages<br>1. Low sensitivity<br>2. Must use rolling shutter, that causes distortion when moving objects exist in frame[3] |

Table 2.1: Comparison of CCD and CMOS Image Sensors

or calculated using camera calibration process. The camera pose in world coordinate is $\mathbf{C}(R, \mathbf{t})$ where $R$ is $3 \times 3$ camera rotation matrix and $\mathbf{t} = [x_c, y_c, z_c]$ is camera coordinate in world coordinate. Then, the projection $\mathbf{x}$ is calculated as

$$\begin{aligned} \begin{bmatrix} x_f \\ y_f \\ f \end{bmatrix} &= K \begin{bmatrix} R & -R\mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \\ x_s &= \frac{x_f}{f} \\ y_s &= \frac{y_f}{f} \end{aligned}$$

## 2.3.2 Image Features Extraction and Representation

The first step in SfM is the detection of distinct feature points in image frames. A "feature" is defined as an interesting part of the image which will be used for subsequent analysis as required ([31]). For example, one may wish to align two images to be seamlessly stitched into a composite mosaic. Another relevant application for this research is for establishing a dense set of correspondences so that 3D model may be constructed (such as shown in Figure 2.8).

The desirable properties of feature detectors are low computational requirements, robustness against changes in scale, viewpoints, illumination, and noise. Because feature detectors are modeled after human visual stimuli, these detectors are usually classified

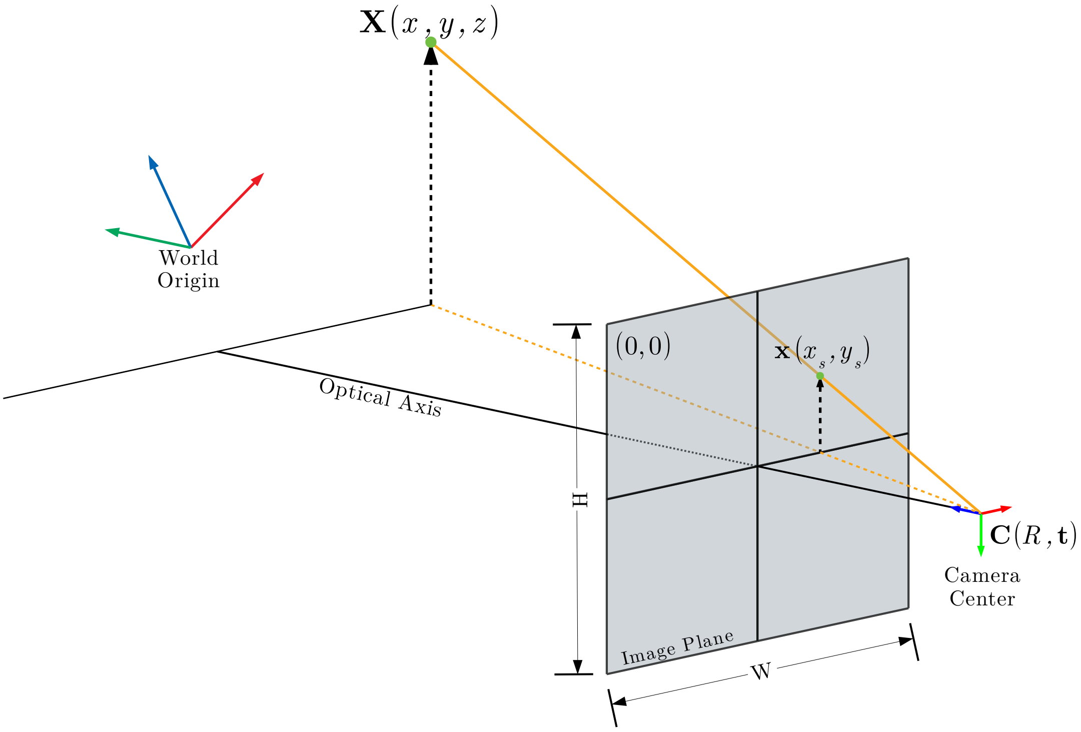


Figure 2.7: Projective Transformation

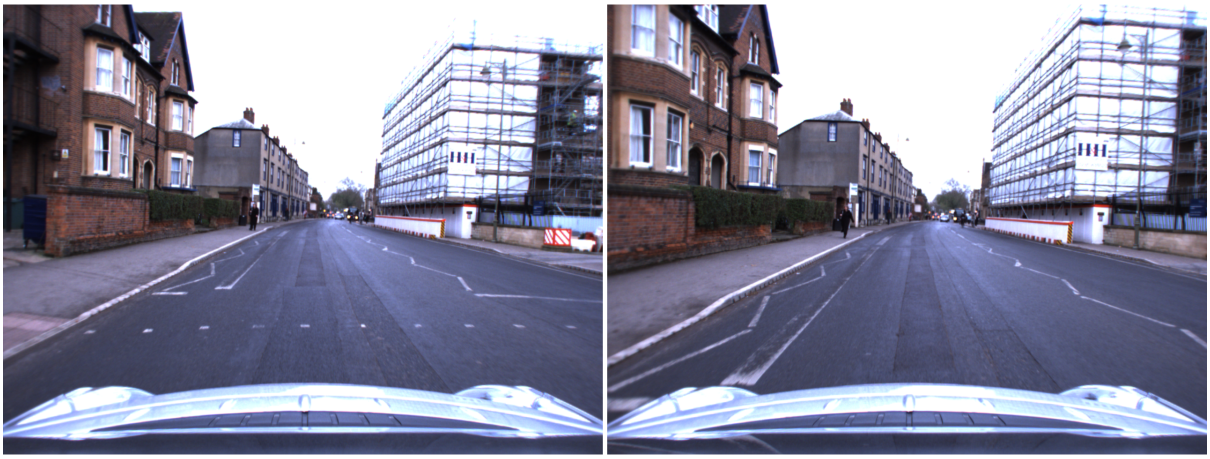


Figure 2.8: Two similar images to be matched from same scene. In the right, the camera experienced translational movement (taken from Oxford RobotCar Dataset [32]).

into three groups [33]:

- *Corner*, refers to the point at which two different edge directions occur in the local neighborhood.
- *Edge*, refers to pixels at which the image intensities change abruptly. Image pixels are discontinuous at different sides of edges.
- *Blob/Region*, refers to distinct parts of images that are bounded by lines or curves from different segmented regions

For the purpose of 3D structure recovery, the most widely used detectors are corner detectors. Corner detectors are usually selected because of its robustness against camera movements (translation and rotation). The gold-standard [34] for corner detectors is SIFT (Scale-Invariant Feature Transform) described in [35], which is known to be invariant to scale, rotation, and illumination changes (in smaller degrees). However, SIFT's high computational times and patent status make it difficult for applications in commercial products.

In this thesis, the ORB feature detector and descriptor [36] is extensively used. This detector is known to be invariant against scale and rotation, small storage requirements, in addition to being fast. However, as described in [37], this detector is not invariant against illumination changes. Therefore, as reported in Chapter 4, the ORB feature matching is subject to failure when facing high appearance changes.

The ORB consists of two parts: the FAST corner detector augmented with orientation and scale, and BRIEF descriptor with rotation identifier. First, corners are detected by using particular FAST patterns ([38]) to check for abrupt intensity changes between a center pixel and those around it; usually the radius is fixed as 9 pixels. These corners must be filtered using Harris corner measure to discard bogus responses along edges. To make the corners scale-invariant, these FAST features are computed for all scale levels of the image pyramid[4]. Next, each FAST points are converted to BRIEF bit vector with length 256 bits [39], and added with an angle indicator of increment 12°.

### 2.3.3 3D Structure Recovery from Motion

Being able to track image features from frame-to-frame, relative pose between two frames and 3D point structures can be reconstructed (Figure 2.9). General workflow of SfM is described as follows [40]. These processes only involve information from 2D points in frames, so no external information (eg. odometry) is involved.

[4] Image pyramid is defined as representation of image that are down-resized multiple times

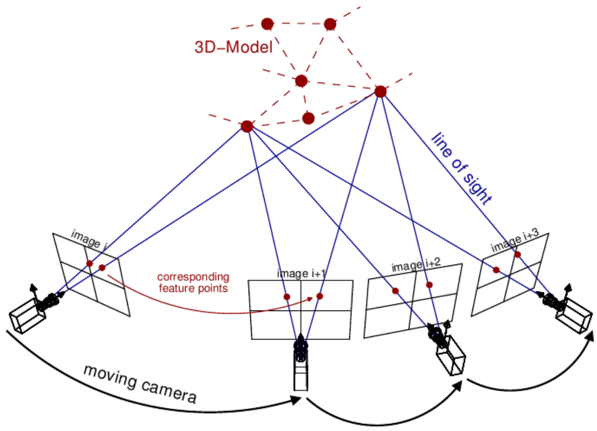


Figure 2.9: General Structure-from-Motion (courtesy of OpenMVG)

1. Find matches of feature points between two frames. This process is specific to image features being employed; for ORB features, matching is performed using nearest neighbor search and bit set counting [41].
2. Compute the Fundamental Matrix $F$, that relates any pair of matching 2D feature points $\mathbf{x} \longleftrightarrow \acute{\mathbf{x}}$ in two images. The fundamental matrix is defined by equation

$$\acute{\mathbf{x}}^T F \mathbf{x} = 0$$

3. Camera pose (position and orientation) of current image frame relative to previous one is derived from $F$ via essential matrix $E$. This pose is not unique, so one must check the solutions by using reprojection errors. The relation of fundamental matrix $F$ and essential matrix $E$ is[5]

$$E = K^T F K$$

4. Calculate 3D points using triangulation.

The result of an SfM session is usually stored as a map. This map minimally consists of keyframes and reconstructed map points. However, for full utilization, this map usually contains other data, such as:

- Each keyframe usually contains its position relative from mapping's start point (which usually fixed at the origin) and list of visible map points.
- Each map point usually stores feature descriptor and position in 3D space.
- Parent-child relationships between each keyframe (usually form a tree, with the first keyframe (starting map point) as root.

An example of SfM session is shown in Figure 2.10. Here, estimated camera poses are shown in blue rectangles, while computed 3D structure points (that resemble buildings)

[5] In this research, camera intrinsic parameters ($K$) is assumed to be constant.

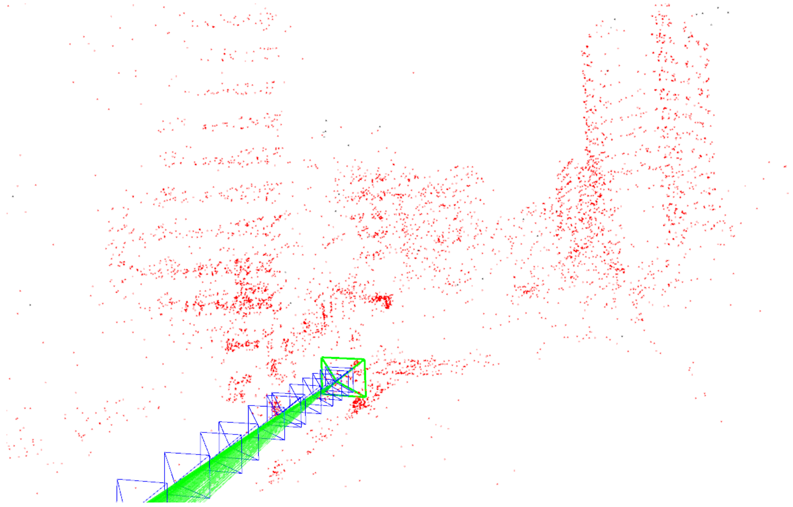


Figure 2.10: A Sample Result of Structure-from-Motion from ORB–SLAM

are drawn as red dots. It must be noted that the above workflow must be done using *calibrated* camera; i.e. internal parameters (focal length, principal point and distortions) have been known.

A major drawback of monocular SfM is that the scale of the reconstructed 3D point is not known. This brings consequence that any information derived from them (most prominently, camera pose) cannot be determined. Therefore, this method is not usable directly in robot localization and motion control. The scale problem can be stated formally as follows [40]. Let $\mathbf{X}_i$ as set of points and two camera poses $Q_1(R_1, \mathbf{t}_1)$, $Q_2(R_2, \mathbf{t}_2)$. Their camera matrices are defined, with $K$ is camera calibration parameters, as

$$P_i = K \begin{bmatrix} R_i & \mathbf{t}_i \\ 0 & 1 \end{bmatrix}$$

Now let

$$H = \begin{bmatrix} R & \mathbf{t} \\ 0 & \lambda \end{bmatrix}$$

and its inverse

$$H^{-1} = \begin{bmatrix} R^T & -\frac{1}{\lambda} R^T \mathbf{t} \\ 0 & {}^1/_\lambda \end{bmatrix}$$

as any similarity transform where $R$ is camera rotation, $\mathbf{t}$ a translation and $\lambda^{-1}$ represents scaling. Replacing points $\mathbf{X}_i$ by transformed $H\mathbf{X}_i$ and cameras $P_1$ and $P_2$ by $P_1 H^{-1}$ and $P_2 H^{-1}$ does not change the observed points, since $P\mathbf{X}_i = (PH^{-1})(H\mathbf{X}_i)$. This means that camera translations can only be known as $\rho\mathbf{t} \in \mathbb{R}^3$, where direction of $\mathbf{t}$ is known but $\rho$ and its sign are not. In other words, camera coordinates and map points in 3D space have scale ambiguities and their distances are not defined, thus not in metric space. As

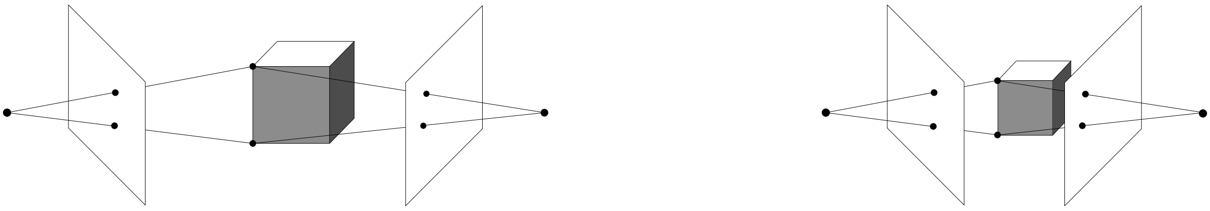


Figure 2.11: Reconstruction Ambiguity [40]

an example of this result, Figure 2.11 illustrates that resizing the cube and moving the cameras do not change projections in the frame.

One possible solution for scale ambiguity is by using stereo camera rig. Here, depth information of the objects is calculated by using triangulation. However, for large-distance scales in open space, stereo camera rig degenerates into monocular case [22], further reducing its usability.

### 2.3.4 Bundle Adjustment

In real-world cases, image measurements and reconstructed 3D coordinates will contain noise. Bundle adjustment is defined as a process of refinement concerning estimated camera poses $\hat{P}_i(\hat{R}_i, \hat{\mathbf{t}}_i)$ and 3D map points $\hat{\mathbf{X}}_j$ after 3D structure recovery described in previous subsection. This problem is reduced to minimizing reprojection errors, measured as distance from projection of projection $\hat{\mathbf{X}}_j$ in all related camera poses $\hat{P}_i$ to their 2D position $\mathbf{x}_i^j$, or

$$\min_{\hat{P}_i, \hat{\mathbf{X}}_j} \sum_{i,j} d(Q(\hat{P}_i, \hat{\mathbf{X}}_j), \mathbf{x}_i^j)$$

This minimization problem is usually solved using Levenberg–Marquardt algorithm (LMA). An example of open-source LMA implementation for solving bundle adjustment and employed by ORB–SLAM is g2o [42].

### 2.3.5 Place Recognition

Visual place recognition is a vital component of visual SLAM. Its main task is to decide whether or not a particular frame is already seen; usually, the system is equipped with a visual map that may be subject to modification. In visual localization, the purposes of place recognition are two-fold. First, it searches for initial position before visual tracking part is performed. In this regard, place recognition serves as global localization, similar to GNSS. Second, place recognition is executed when visual tracking is lost due to visual disturbances (restart). Performing visual place recognition is challenging due to many factors: severe change of appearances (see Figure 4.8 for example), perceptual aliasing (multiple places may look very similar), and viewpoints discrepancies of visited places.

This research employs bag-of-words method [43] for place recognition, which is based on a database of image feature descriptors. The database is composed of two parts: vocabulary tree built from a collection of image feature descriptors ("words") and direct and inverted indices. Basically, the vocabulary is a list of image features arranged into a tree data structure similar to Huffman tree. In ORB–SLAM, the implementation of image database leverages pre-made generic vocabulary tree from an unspecified dataset. This research proposes a custom–made vocabulary tree for any particular place instead of generic one, and described in Chapter 4.

### 2.3.6 Visual–Inertial Navigation

One method to recover correct camera poses in metric space is by associating keyframes with external references [26]. This approach combines 3D structure-from-motion of the camera and relative pose from metric positioning such as IMU [44, 45]. Because of accumulated errors in IMUs, usually filtering or graph-based optimizations have to be employed.

In general, there are two approaches for visual–inertial navigation. The tightly-coupled approach integrates metric positioning into the whole process of visual SLAM (mainly, SfM and bundle adjustment) to resolve scale ambiguities; examples of this approach are [46] and [47]. The drawback of this approach is that its implementations must be tailored specifically for the problem at hand.

Another approach is loosely-coupled visual–inertial; here, the vision part is considered as "black box" and only position results are used. An example of this approach is [48], which integrates ORB–SLAM results and odometry information to obtain scaling coefficient information using Kalman filter. In contrast to our approach, this method silently assumes that this scaling is constant throughout run.

This research takes the loosely-coupled approach for resolving scale in visual SLAM, which means that scale is determined after camera pose is calculated. To assist scale calculation, absolute keyframe positions are supplied from accurate LiDAR positioning when building the map; this virtually eliminates error accumulation in visual maps created by ORB–SLAM.

# Chapter 3

# Vision-Based Localization in Metric Space

## 3.1 Objectives

This chapter proposes a positioning method based on ORB–SLAM [27] using a monocular camera with LIDAR-aided mapping. The ORB–SLAM is one of the most recent monocular vision–based SLAM methods with an open-source implementation. This method estimates camera positions and generates map from an image sequence in real-time. Originally, ORB–SLAM was designed to solve visual SLAM problem. As mentioned in the beginning of Chapter 2, there is a distinction between SLAM and localization. This distinction also relates to system concept defined in Figure 1.2.

However, as previously explained in Subsection 2.3.3, any visual SLAM method that only relies on visual data is not capable to work in metric space. Therefore, vision-based methods incur several problems when adopted for localization problems such as robustness and metric consistency. Our proposed method has two key points:

1. The estimation of metric position in localization by using ORB–SLAM with LIDAR-aided mapping

2. The solution of robustness problem using sensor fusion between multiple maps and odometry data

In general, this chapter discusses:

1. Benchmark tests related to ORB–SLAM conducted in the Tsukuba Challenge environment;

2. Description of ORB–SLAM with LIDAR-aided mapping to solve problems of metric consistency between multiple maps;

3. Experimental results and evaluation in Tsukuba Challenge environment. Finally, this chapter illustrates the capability of sensor fusion method of vision-based localization method with odometer to provide continuous localization over the course.

## 3.2 Overview of Tsukuba Challenge

The Real–World Robot Challenge (RWRC) is a real-world autonomous navigation challenge held in City of Tsukuba, Japan[1]. The robots are required to autonomously run over a 1 km route. Therefore, an accurate localization system plays a critical role here. The robots are required to maintain their position accuracy over the course despite changes in the environment such as dynamic obstacles (people and other robots), differences in the illumination, weather and other factors. Therefore, the RWRC is an ideal event to identify and solve problems with monocular visual localization in a low-velocity, real-world setting.

For most of the course of Tsukuba Challenge, almost all teams use LiDAR-based localization methods [49]. These methods use sensor fusion approach with LiDAR and odometry, which can compensate for weaknesses in the characteristics of individual sensors. Essential to this approach is the use of a good dead reckoning method such as a calibrated gyroscope.

On the other hand, vision-based localizations have received little attention in RWRC. One of the efforts for developing visual localization is [50]. This method is basically a type of topological localization by following an image sequence and estimating pose by using feature points. Compared to this method, our method sets out to obtain metric positioning, similar to LIDAR-based navigation.

## 3.3 Overview of ORB–SLAM

### 3.3.1 Description

To repeatedly perform localization, our implementation of ORB-SLAM consists of two parts: mapping and localization-only. The mapping process runs similar to the original implementation, with an addition of map storage at the end of the mapping run. Meanwhile, the localization process starts with map restoration using data saved previously in the mapping process. The localization step proceeds in much the same way as in the mapping stage. However, map modification is disabled in the relocalization part.

The ORB–SLAM main routine creates an environmental map which consists of keyframes and map points. Each keyframe stores its position in ORB–SLAM coordinates and a list

[1] Within this thesis, the terms “RWRC” and “Tsukuba Challenge” will be used interchangeably

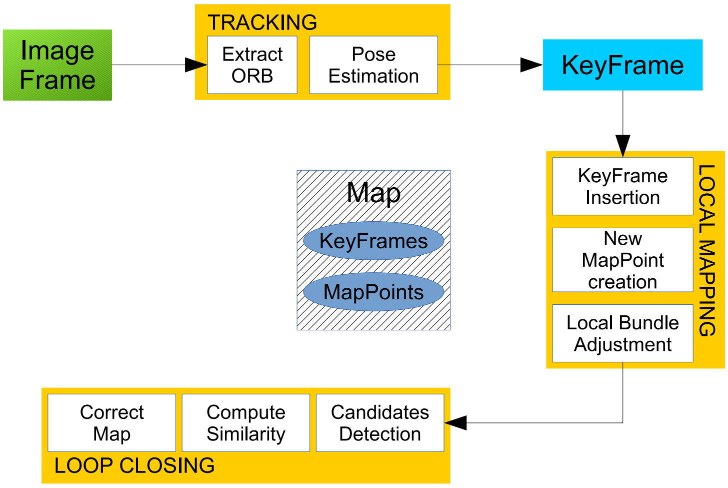


Figure 3.1: ORB–SLAM System Overview

of 2D feature points. The entire ORB–SLAM process consists of three parallel threads: tracking, local mapping and loop closing. The relationship of these threads is shown in Figure 3.1.

### Feature Detection

The first step in all 3D reconstruction is to identify feature points in each frame. ORB-SLAM uses ORB features, described in [36]. The ORB feature offers advantages such as faster computation and lower storage requirement (32 bytes), in addition to a degree of resistance to rotation and noise.

### Map Initialization

Before inserting more keyframes and map points, the map must be initialized by computing relative pose between two initial frames to triangulate an initial set of map points, which are then used for tracking. ORB-SLAM uses a combination of homography and fundamental matrices inside a RANSAC scheme to build motion and structure recovery as described in [51]. When this stage is successful, the system will have an initial set of keyframes and map points with which tracking may proceed. However, tracking may fail shortly after the initial map is built; if this occurs, the initial map is reset and started over.

### Tracking and Local Mapping

The tracking thread is responsible for providing localization and map building. After ORB corners are detected, the tracking thread develops a map incrementally over the recovered 3D map points, while computing camera poses. To accelerate this process, the tracking operates in a smaller subset of the overall map, called the local map, that covers currently visible keyframes and some connected ones. The tracking thread also performs map “clean-up”, which involves culling bad map points and keyframes.

There are three tracking modes that may be used. First is relocalization by searching all keyframes by *bag-of-words*; this is the slowest but indispensable when recovering from lost tracking. The second choice involves tracking the local map, as described above. Alternatively, the third choice involves tracking using constant velocity model which is equivalent to visual odometry [19]. This choice is fastest and may be the most frequently used mode. However, it may be inaccurate.

#### Loop Closing

Loop-closure detection is crucial for enhancing the accuracy of SLAM algorithms, both topological and metrical. This problem consists of detecting when the robot has returned to a former location after having discovered new terrain. Such detection makes it possible to increase the precision of the actual pose estimation.

Essentially, loop closing in ORB-SLAM uses image-to-map approach [52]. First, it takes the most recently processed keyframe and searches for a loop candidate keyframe in the local map using the bag-of-words method [43]. Next, the similarity transformation is computed. Loop correction is performed by inserting new edges into the covisibility graph and fixing connectivity between loop candidate and surrounding keyframes. Next, ORB-SLAM performs pose graph optimization, whereby loop closing errors are distributed by moving the candidate and its connected keyframes.

### 3.3.2 Problems of ORB–SLAM

#### Scale Ambiguity

ORB–SLAM outputs localization results based on maps in its own coordinate system (which is not metrically correct), that are not free from distortion due to scale ambiguities. In some cases, result maps may exhibit heavy deformation, as illustrated in Figure 3.2. This deformation is a result of scale ambiguities over multiple image frames in large scale such as Tsukuba Challenge, as predicted in Subsection 2.3.3.

#### Lack of Support for Lifelong Mapping

The original design of ORB–SLAM involved a single run for both localization and mapping, so there are no features for storing in-memory map to disk. This means that the system must be initialized with empty map prior to navigation tasks. However, it is desirable for distinct mapping and localization processes to be done multiple times with the same path, so map saving and restoration is essential. This feature is also useful for improving the map robustness when faced with changing condition [53]. In practical

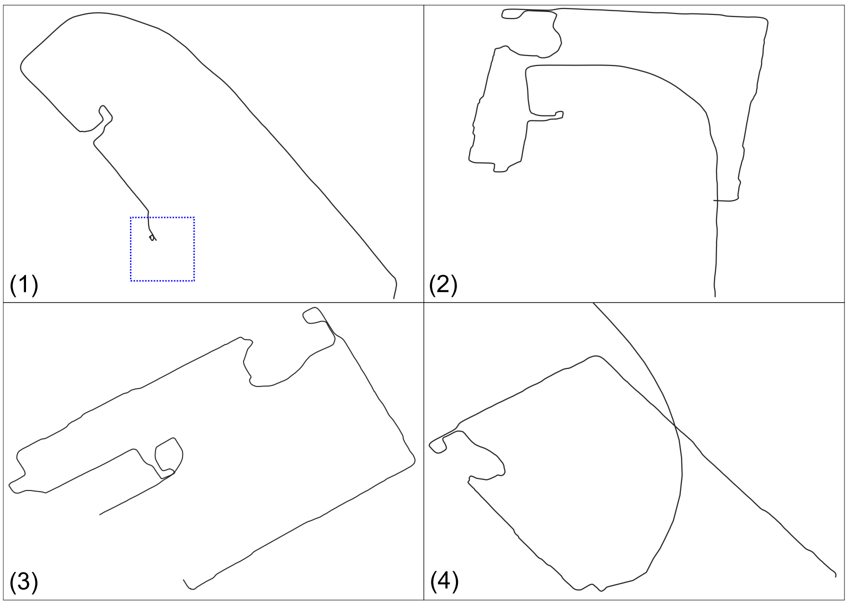


Figure 3.2: Scale distortion in the generated ORB-SLAM map trajectory from Tsukuba Challenge 2015.

applications, this will enable autonomous vehicles to localize positions despite changes in weather, time of day, and other conditions. The operation of multiple maps, discussed in subsection 3.4.3, also requires stored maps.

### Visual Disturbances

Any disturbances in the camera vision that make it unable to view previously tracked feature points may lead to the failure of ORB–SLAM tracking. These disturbances include vision occlusion on the part of the camera and fast rotation of the robot. Other forms of visual disturbances are lens flares and smears, which happen when the camera is faced to the sun. The effects range from lost tracking to straying off the path, which may negatively affect the usability of the ORB–SLAM.

## 3.4 Methods

This section explains how to realize metrically correct monocular visual localization for solving main problems of ORB–SLAM explained in the previous subsection. First, LIDAR-aided mapping is employed to solve scale problem of keyframe distances and provide metrically correct positioning. Next, map storage and restoration process to allow for separate mapping and localization processes are described. Lastly, multiple observations from distinct ORB–SLAM maps may be employed simultaneously with odometry data to derive accurate position and orientation of the robot.

### 3.4.1 Map Building and Restoration

The main output of ORB–SLAM mapping is a set of keyframes. As explained above, our main goal is to take this map and compute the localization during robot runs as guidance for the navigation system, be it a robot or an autonomous car. Therefore, it is necessary to get localization results that are metrically acurate. However as illustrated in Figure 3.2, it would be very difficult to accurately obtain the position using deformed maps from ORB–SLAM.

To formalize the map and localization process, the following notations are introduced. An ORB–SLAM map is a set of poses (included in keyframes): $M = \{P_i | 0 \leq i \leq N-1\}$, where $N$ is the number of keyframes until the mapping process is stopped.

As stated in [26] and [54], the scale ambiguity in visual SLAM can be solved in mapping phase by associating each keyframe to an external reference with true position (in metric sense). In a longer run, this association must be done correctly so that error accumulation in the scale correction is eliminated. Therefore, the external reference must have a high level of accuracy. In 2015 Tsukuba Challenge, LiDAR-based localization was chosen as the reference due to its immediate availability. Other positioning methods may also be used, such as GPS or odometry, as long as their error corrections are provided [55]. However, GPS usage in Tsukuba Challenge is generally limited due to heavy vegetations.

This research used the 3D Normal Distribution Transform (NDT) scan-matching method with 3D LIDAR to obtain accurate positions [56] as keyframes' external references. Figure 3.3 shows a visualization of 3D map of the Tsukuba Challenge track. This map was built by applying the 3D NDT scan-matching method using the Velodyne HDL–32 LIDAR.

Map storage consists of three main parts: keyframes, map points, and keyframe relationships. Each keyframe stores the camera pose $P_k$ in ORB–SLAM coordinates, the camera intrinsic parameters, all of the ORB feature points recorded at keyframe creation, and external reference pose $P_n$ in metric coordinates, recorded at keyframe creation. For localization phase, the system will reconstruct the following data structures:

1. List of keyframes and their relationships
2. Map point list
3. Octree of keyframe position in ORB–SLAM coordinates. This tree will be used for fast searching of the keyframes during localization using augmented positioning.

By default, ORB–SLAM will try to find the position against the last keyframe whenever it loses tracking. However, for situations after map restoration, the last keyframe will be unknown. Instead, ORB–SLAM is modified to execute place recognition by searching for appropriate keyframe using the bag-of-words method. Place recognition is also applied

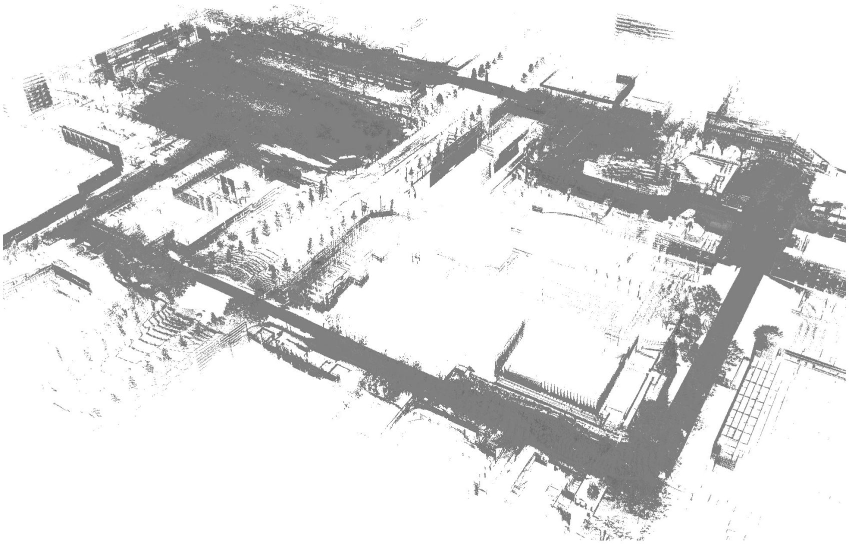


Figure 3.3: 3D View of Tsukuba Challenge map generated by NDT scan matching from Velodyne scans

when the system loses track; this is done to ensure that the system always gets the keyframe as the basis for tracking. The drawback is that keyframe search using bag-of-words method is slower than tracking using the last keyframe.

### 3.4.2 Metric Localization

In the localization process, the system depends solely on visual data. Therefore, external methods such as LiDAR-based localization are not required. In order to assist metric pose computation, keyframes' real poses in metric space are recorded during mapping phase along with their computed positions in ORB–SLAM frame as $P_n$ and $P_k$, respectively. In localization phase, robot pose in metric space as $P_c$ is estimated as scaling-up from ORB–SLAM pose $P_o$. This process is described in Algorithm 1 and illustrated in Figure 3.4.

### 3.4.3 Using Multiple Maps

During our experiments, maps of the same location but created at different times were observed to deliver varying results in terms of coverage (Figure 3.12). Therefore, it is reasonable to combine the results from two maps in order to: 1) alternately provide localization whenever one of the maps fails; 2) reduce errors from all the maps. In this regard, any method for sensor fusion may be used. It must be stressed that, after scale correction, all of the maps will provide consistent results that are metrically correct and in the same coordinate system.

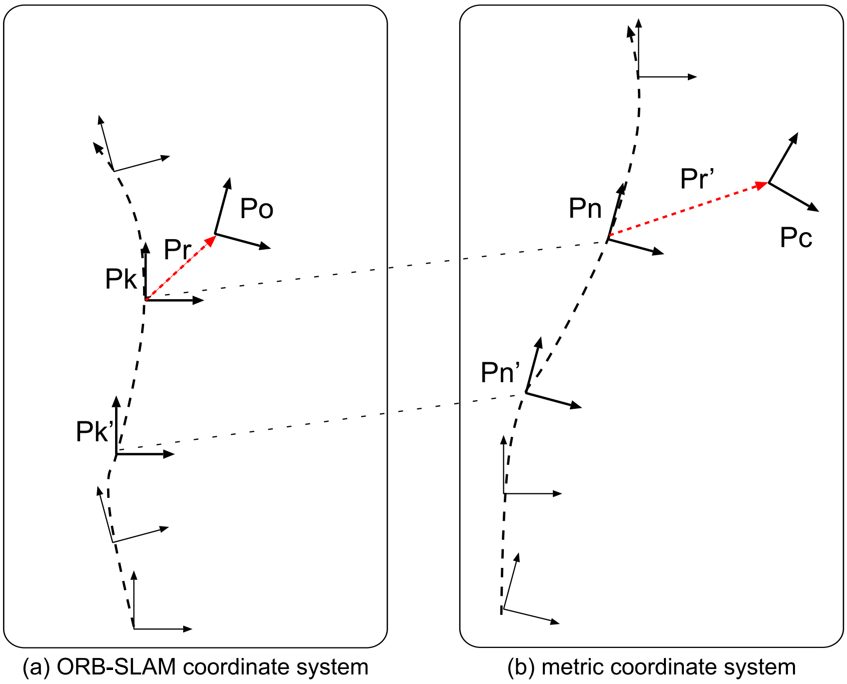


Figure 3.4: Metric transformation

In its original version, the ORB–SLAM does not allow using multiple maps. However, it is possible to run multiple processes of ORB–SLAM with the same input data; each one utilized different maps built from different times. Hence, multiple results can be produced simultaneously from single input camera. Observations from these distinct maps may be combined together with odometry as discussed in the next subsection.

### 3.4.4 Particle Filter with Odometry Data

As an approach for sensor fusion, this subsection provides a formulation derived from particle filter as described by [57]. This formula basically estimates position and orientation from velocity and rotation speed measured by odometry, while correcting these values as ORB–SLAM localization supplies position and orientation updates.

To simplify formulation, the robot is assumed to move in 2D plane; robot state at time $t$ is represented by its position and orientation in 2D plane as $\mathbf{x}_t = (x_t, y_t, \theta_t)$. Control data from odometry come as linear velocity and rotation speed and represented respectively, as $\mathbf{u}_t = (v_t, \omega_t)$. The particle filter takes a sample of $M$ number of "particles"; each particle represents a possible state of the robot. As the motion proceeds, all particles are updated by control variables $\mathbf{u}_t$ and ORB–SLAM measurements $\mathbf{z}$ from all maps if available. The particle filter then selects particles proportional to their fitness against $\mathbf{z}$ as weight $w$. The complete particle filter is described in Algorithm 2, complemented with its motion model and measurement model in Algorithm 3 and 4.

The motion and measurement models have simple working assumptions. Inside the

**Algorithm 1:** Position correction from ORB-SLAM to metric coordinate

```
Search the nearest keyframe in the map from $P_o$ in the octree as $P_k$. From here, we
    take the corresponding external reference position $P_n$.
Find the previous offset keyframe $P_k^{'}$ and its corresponding external reference
    position $P_n^{'}$.
Compute the scale correction factor. This factor is a ratio of magnitude of
    translation between external reference $P_n^{'}$ to $P_n$ and translation between keyframe
    $P_k^{'}$ to $P_k$
        $s = \frac{\left\|\mathbf{t}_n^{'} - \mathbf{t}_n\right\|}{\left\|\mathbf{t}_k^{'} - \mathbf{t}_k\right\|}$
Apply the distance scale
        $P_r = P_k^{-1} P_o$
        $P_r^{'} = (s\mathbf{t}_r, \mathbf{q}_r)$
        $P_c = P_n P_r^{'}$
Return $P_c$
```

motion model, noises are introduced to $v_t$ and $\omega_t$ in order to account for errors in them. The noises are assumed to be Gaussian with standard deviation $\alpha_1$ and $\alpha_2$, which are device-specific and must be determined by experiment. Meanwhile in the measurement model, each particle's weight is determined from its distances to all ORB–SLAM measurements. Here, each ORB–SLAM measurement is assumed to be independent and may contain noises (for example, see Figure 3.15). Therefore, the nearest measurement to the particular particle is selected as predictor, resulting in largest weight from all measurement as described in algorithm 4. This measurement model is easily expandable to include more than two ORB–SLAM results.

## 3.5 Experimental Settings

To evaluate our localization system, we conducted four runs whereby the robot traversed the trajectory mandated by the 2015 Tsukuba Challenge. In each run, we recorded camera images and performed localization using Velodyne LiDAR. From these runs, we created two maps for localization process. The LiDAR-based localization results would be used as ground truth for comparison. To reduce computation, camera resolution was reduced to 800 × 600 before processing. Time and conditions of each run are described in Table 3.1.

The robot for 2015 Tsukuba Challenge was derived from a Segway RMP–200 platform, using a PointGrey Grasshopper3 camera and Velodyne HDL–32 LIDAR. The robot ran through the mandated course at a speed less than 1 m/s. In the course of the run, the robot

**Algorithm 2:** Particle filter localization

```
Input: $\mathbf{x}_{t-1}$, $\mathbf{u}_t$, $\mathbf{z}_t$
$\overline{\mathbf{x}}_t = x_t = \emptyset$
for $n = 1$ to $M$ do
    $\mathbf{x}_t^{[n]} = \text{motion\_model}\left(\mathbf{u}_t, \mathbf{x}_{t-1}^{[n]}\right)$
    $\mathbf{w}_t^{[n]} = \text{measurement\_model}\left(\mathbf{z}_t, \mathbf{x}_t^{[n]}\right)$
    $\overline{\mathbf{x}}_t = \overline{\mathbf{x}}_t + \langle \mathbf{x}_t^{[n]}, \mathbf{w}_t^{[n]} \rangle$
end for
for $n = 1$ to $M$ do
    draw $x_t$ from $\overline{\mathbf{x}}_t$ with probability $p \propto w_t$
    add $x_t^{[n]}$ to $\mathbf{x}_t$
end for
Return: $\mathbf{x}_t$
```

**Algorithm 3:** Computing poses $X_t = (x^{'}_t, y^{'}_t, \theta^{'}_t)$ from pose $X_{t-1} = (x_t, y_t, \theta_t)$ and control $\mathbf{u}_t = (v_t, \omega_t)$

```
**motion_model**$(\mathbf{u}_t, \mathbf{x}_{t-1})$:
$\hat{v} = v + \text{rand}\,(\alpha_2)$
$\hat{\omega} = \omega + \text{rand}\,(\alpha_2)$
$x^{'} = x + \hat{v}\cos(\theta)\Delta t$
$y^{'} = y + \hat{v}\sin(\theta)\Delta t$
$\theta^{'} = \theta + \hat{\omega}\Delta t$
Return: $X_t = (x^{'}, y^{'}, \theta^{'})$
```

**Algorithm 4:** Particle weighting $w$ of state $X_t = (x^{'}_t, y^{'}_t, \theta^{'}_t)$ against metric ORB-SLAM measurements $Z_1 = (x_1, y_1, \theta_1)$ and $Z_2 = (x_2, y_2, \theta_2)$. Here, $\Sigma$ is covariance matrix which represents error measurements of ORB-SLAM in lateral, longitudinal and yaw.

```
**measurement_model**$(X, Z_1, Z_2)$:
$\Sigma = \begin{bmatrix} \sigma_x & & \\ & \sigma_y & \\ & & \vartheta \end{bmatrix}$
$w_1 = \exp\left\{-\frac{1}{2}(X_t - Z_1)^T \Sigma^{-1} (X_t - Z_1)\right\}$
$w_2 = \exp\left\{-\frac{1}{2}(X_t - Z_2)^T \Sigma^{-1} (X_t - Z_2)\right\}$
Return: $w = \max\,(w_1, w_2)$
```

| Run | Date & Time<br>(Nov. 2015) | Weather<br>Condition | Lighting<br>Contrast | Human<br>Presence |
|---|---|---|---|---|
| Map 1 | 6th, 13:44 | Clear | High | Low |
| Map 2 | 7th, 11:30 | Overcast | Medium | Low |
| Test 1 | 3rd, 14:55 | Clear | High | High |
| Test 2 | 7th, 14:20 | Overcast | Low | Low |

Table 3.1: Time and Condition for mapping and testing runs

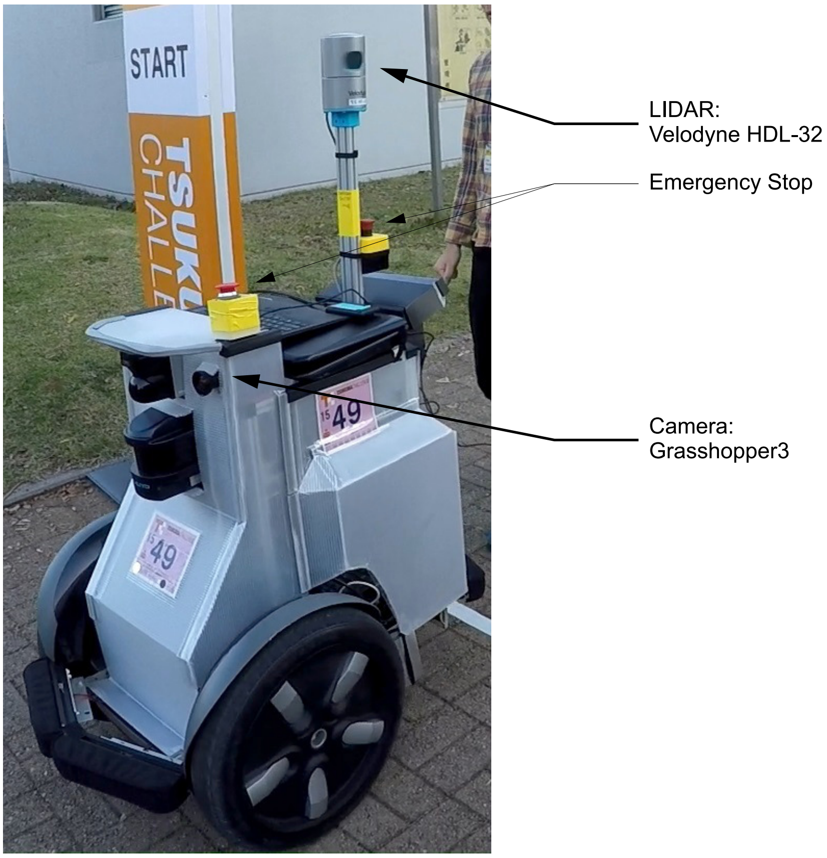


Figure 3.5: Robot vehicle for system evaluation in Tsukuba Challenge 2015

would often encounter dynamic obstacles such as human or bicycle, which necessitated decision by the operator to either stop or maneuver the robot. Our robot setup is shown in Figure 3.5.

The track to be covered in the Tsukuba Challenge was very different from that used for the original ORB-SLAM paper evaluation, which primarily used New College dataset [58]. To simplify the discussion, the track is roughly divided into five major areas; each had distinct visual features and its own challenges. These areas are shown in Figure 3.6, with descriptions as follows.

1. Area 1 was a public park area, with many trees as main features and occasional building background (Figure 3.7).

2. Area 2 was a checking pit in front of a large hall building. When passed in after-

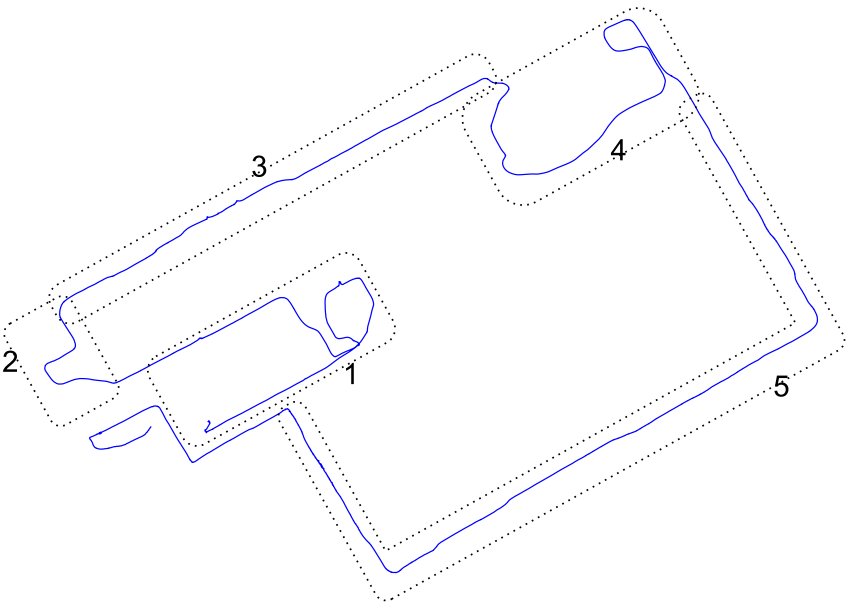


Figure 3.6: Breakdown of Tsukuba Challenge 2015 track by visual features

noon, this area may feature high contrast due to setting sun; most lens flares were encountered here.

3. Area 3 was a pedestrian footpath covered by paved blocks and surrounded by trees and autumnal leaf drops on the ground. There might be some encounters with curious pedestrians that approached the robot; these people were registered on the map (Figure 3.8).

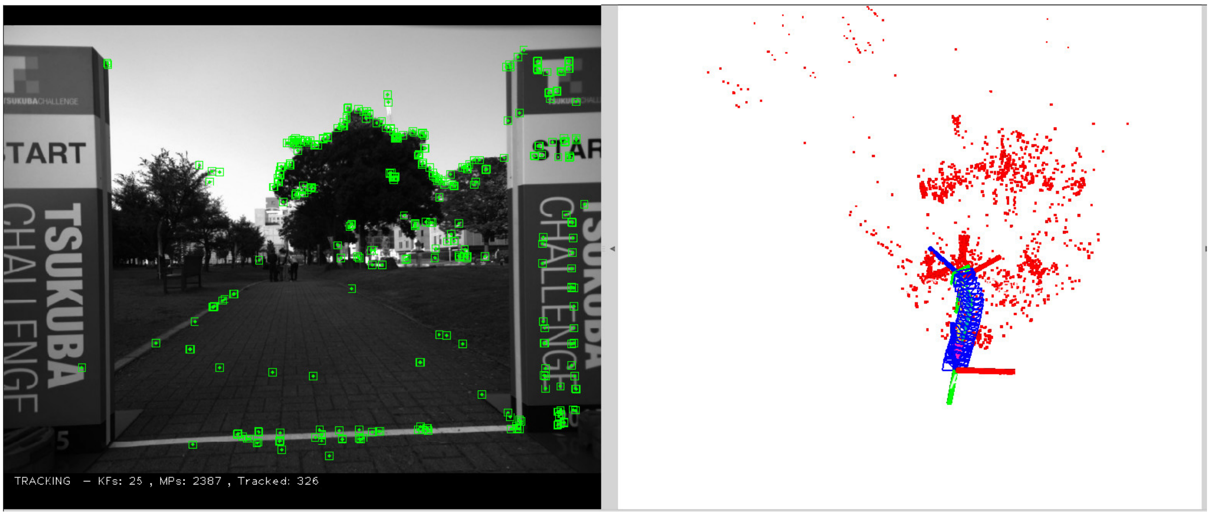
4. Area 4 was an outdoor scene with many buildings as background. This area featured quite strong contrast, as shown in Figure 3.9. A situation like this can confuse automatic exposure system of the camera, and makes it difficult to detect feature points.

5. Area 5 had mostly the same situation as area 3, but encounters with dynamic objects were rare.

## 3.6 Results and Discussions

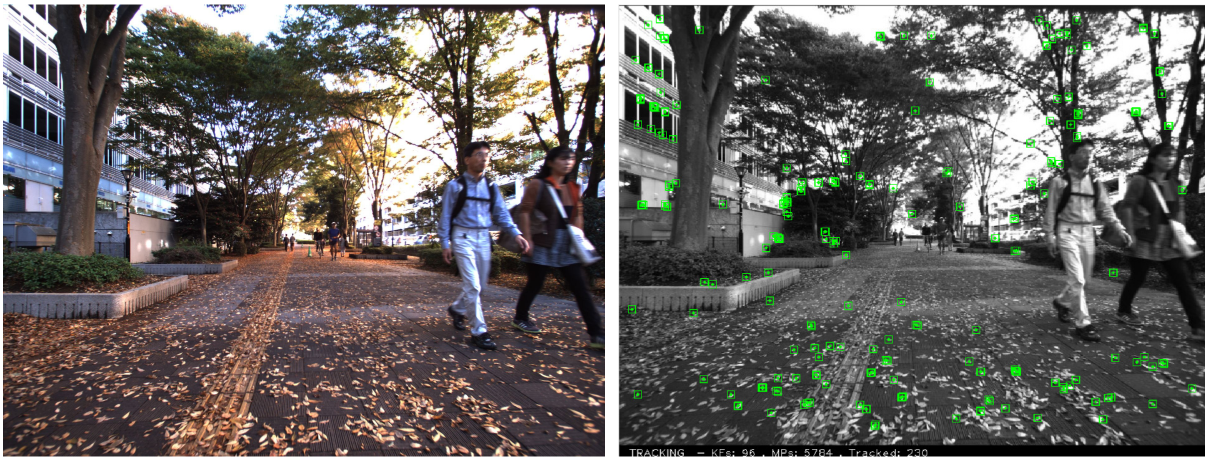
### 3.6.1 Map Saving and Restoration

By using the developed map storage routine, map data structures of ORB–SLAM can now be saved and restored at any time. From our experience, map saving and restoration do not affect ORB–SLAM performance. In fact, the system gains useful capability, i.e. map building can now be done incrementally using the same location but different times. This is useful when building a lifelong map from different situations such as in varying



```
TRACKING  – KFs: 25 , MPs: 2387 , Tracked: 326
```

Figure 3.7: Starting point



```
TRACKING  – KFs: 96 , MPs: 5784 , Tracked: 230
```

Figure 3.8: Typical situation in Tsukuba Challenge: pedestrian tracks covered with leaf drops

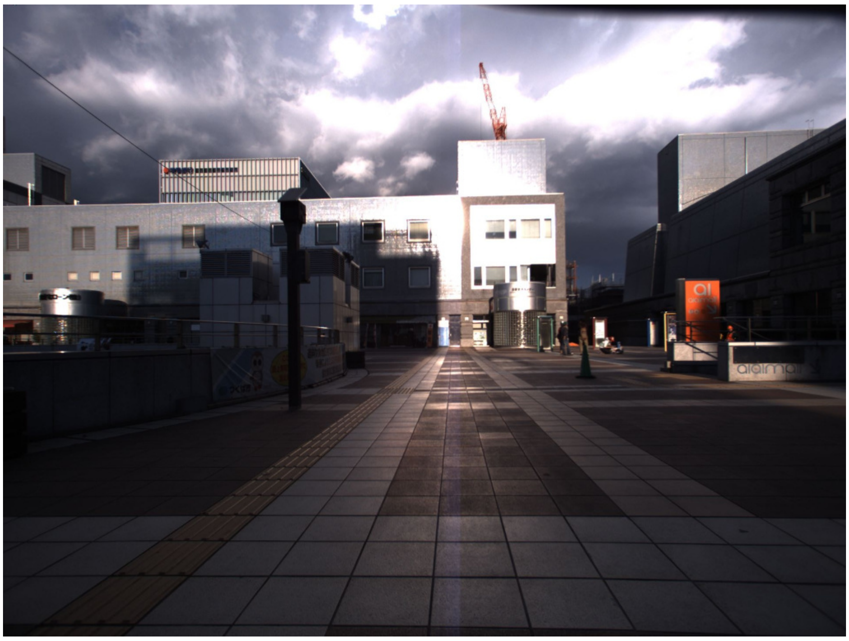


Figure 3.9: Many areas have strong contrasts

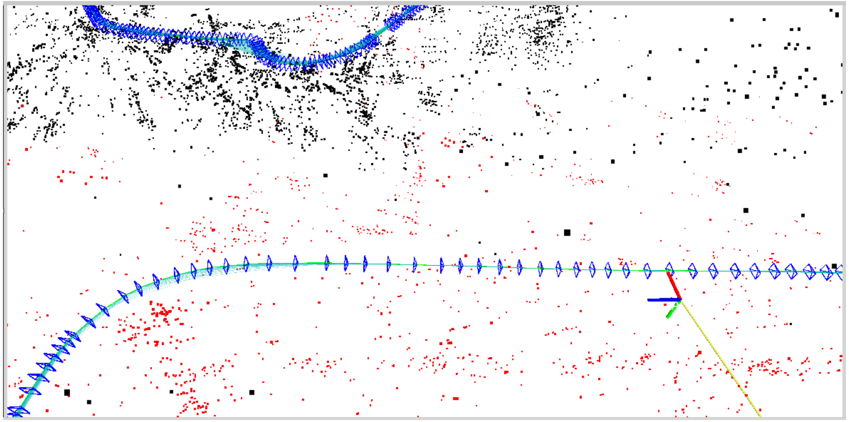


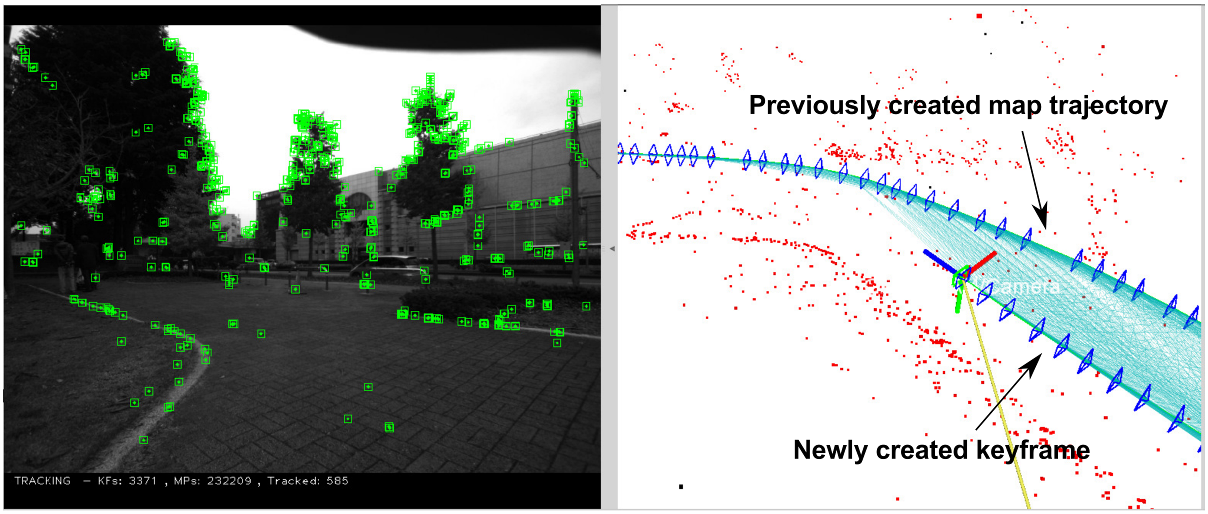
Figure 3.10: Robot traversing previously created map

weather and during the day/night. An example of the relocalization after map restoration is illustrated in Figure 3.10. Example of incremental map building is shown in Figure 3.11.

In this experiment, a result map of the whole trajectory of 1.5 km requires approximately 2 GB of disk space and RAM. This map consists of 3100 keymaps and 400000 map points, and require 5 to 7 minutes to load from disk. Long loading times are certainly caused by serialization process of storing maps; improvements could use memory-mapped I/O that is faster but may require larger disk space.

### 3.6.2 First Position Fix

To get an initial position fix for relocalization, ORB–SLAM performs a keyframe search based on the appearance of feature points. This search may return more than one can-



TRACKING - KFs: 3371 , MPs: 232209 , Tracked: 585

Figure 3.11: ORB–SLAM created a new map based on old map

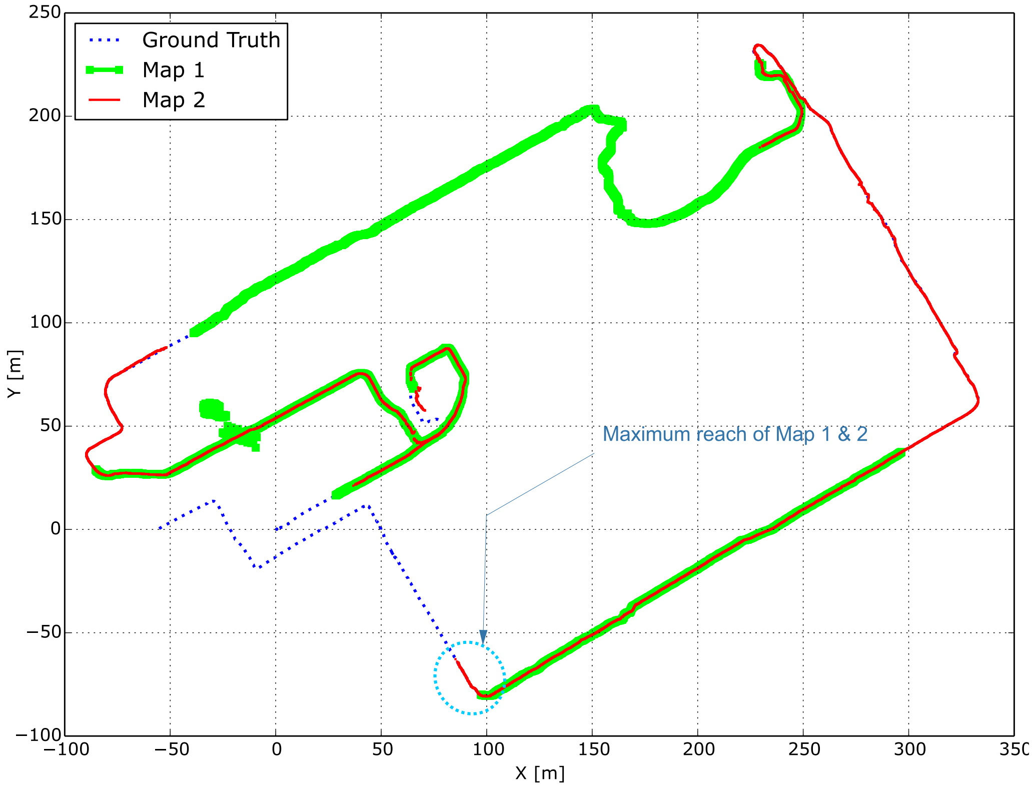
didate, which will be evaluated according to the reprojection error. Only one candidate is accepted, and it must have at least 15 map points that match the feature points in the current frame. In the evaluation run, obtaining the fix was slow due to insufficient matches. One possible enhancement that would enable quicker initial fix is to increase the number of feature points from the ORB computation. However, this approach greatly slows the search process and does not always correlate to a quicker fix.

Another problem related to position fix is when initializing ORB-SLAM map. During the experiment, we found that initialization will succeed (without getting false initialization) whenever the robot is moved, both in rotation and translation. In our experiences, false map initialization and slow position fix can be solved by increasing number of extracted ORB points (by default the number is 1000) to 2500. The drawback is higher computation times per frame. However, another benefit from increasing this number is better resistance to visual disturbances due to increasing number of map points.

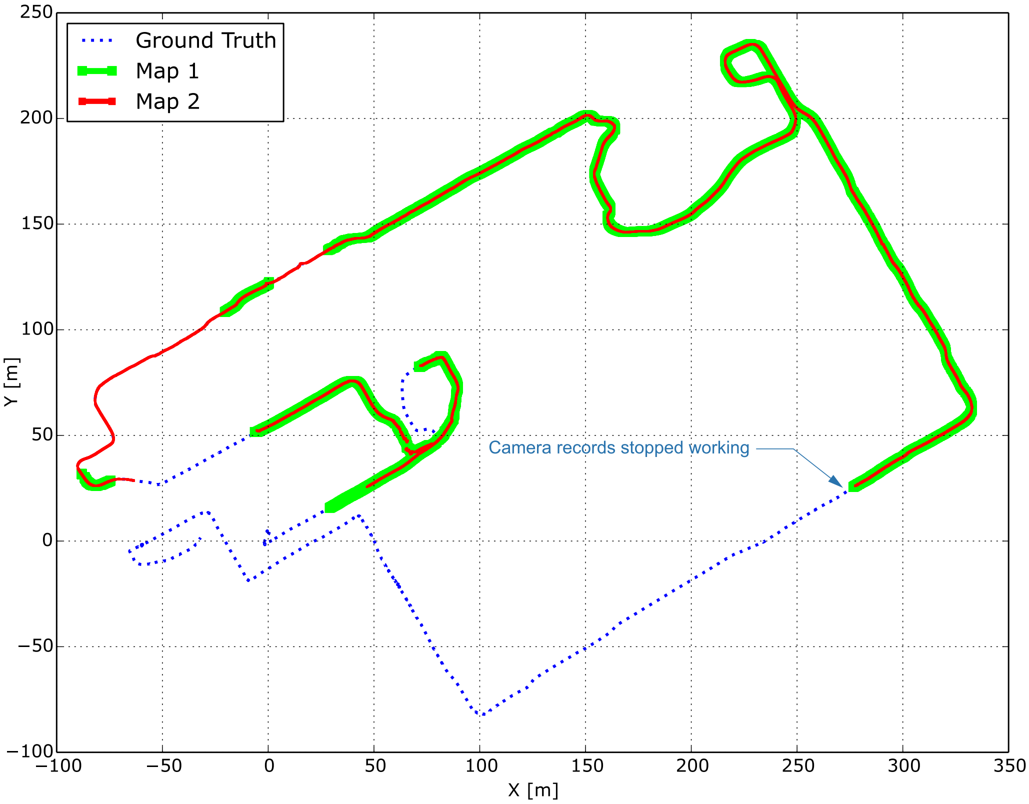
### 3.6.3 Relocalization and Tracking

During this experiment, ORB–SLAM is found to be resistant to occasional and partial vision occlusion. Partial occlusion includes lens flares and people moving in front of the background images. Total occlusion however, may cause the tracking failure, which may be difficult to recover. This lost tracking explains the existence of blank areas in Figure 3.12 (part of the trajectory that has no bold parts).

Common situation and tracking of ORB–SLAM are depicted in Figure 3.13. The figure illustrates a frame, taken in the Oshimizu park area, with a background of buildings in the distance and some trees in the foreground. There were also some people in the scene. Most of the ORB points (and map points, shown in green dots) fell in the trees and ground, but very few of those were in background.

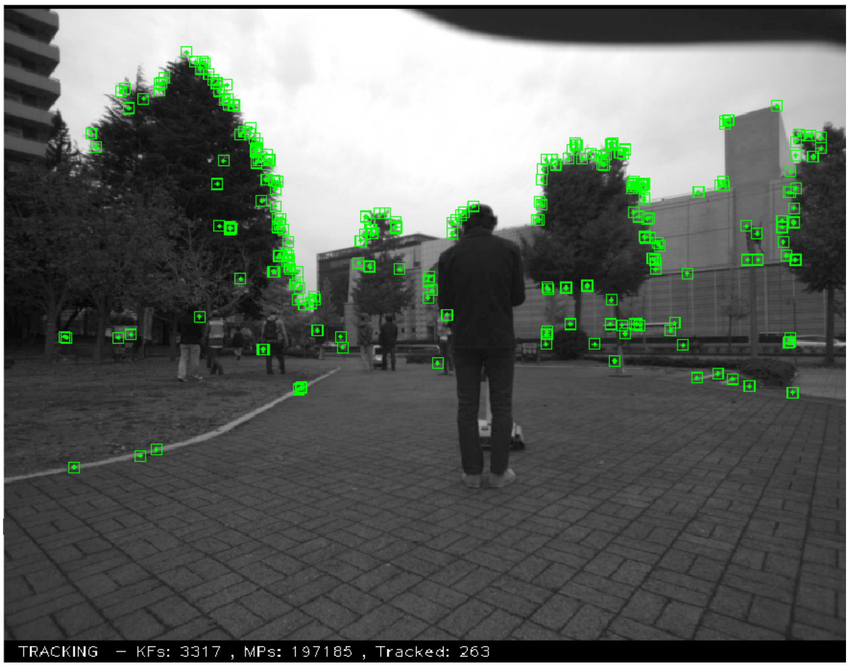


(a) Test Run 1



(b) Test Run 2

Figure 3.12: Coverage Plots of 2015 Tsukuba Challenge



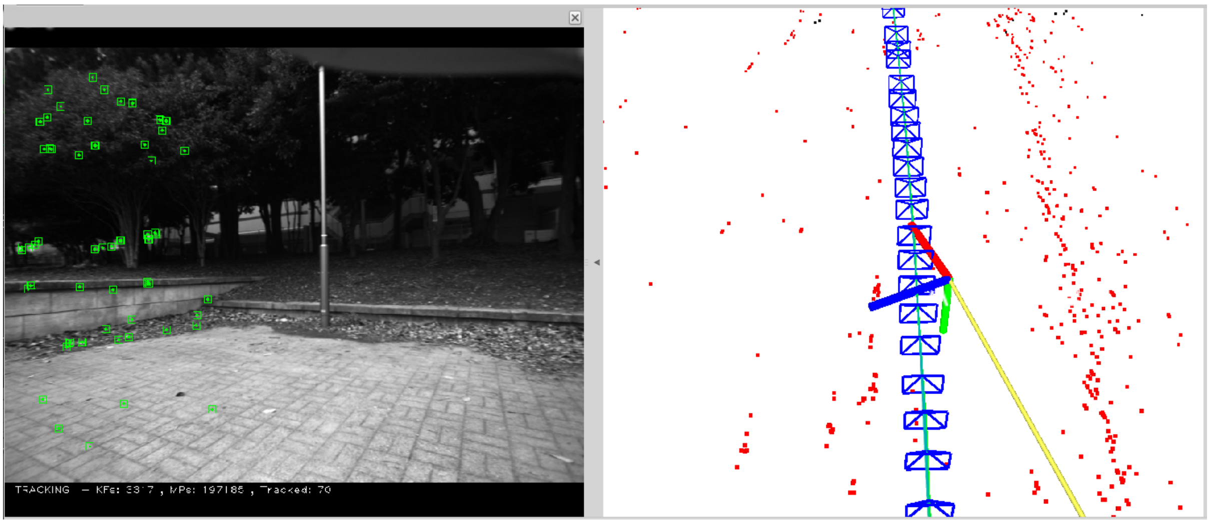
TRACKING - KFs: 3317 , MPs: 197185 , Tracked: 263

Figure 3.13: ORB–SLAM performed tracking

Figure 3.14 depicts a situation in which the robot performed a violent rotation such that lost tracking was imminent. Note the absence of ORB feature points in the right portion of the image frame. On the right, the axis shows that the robot was on the right track, but robot was oriented towards a place with very few map points. The blue axis represents the front.

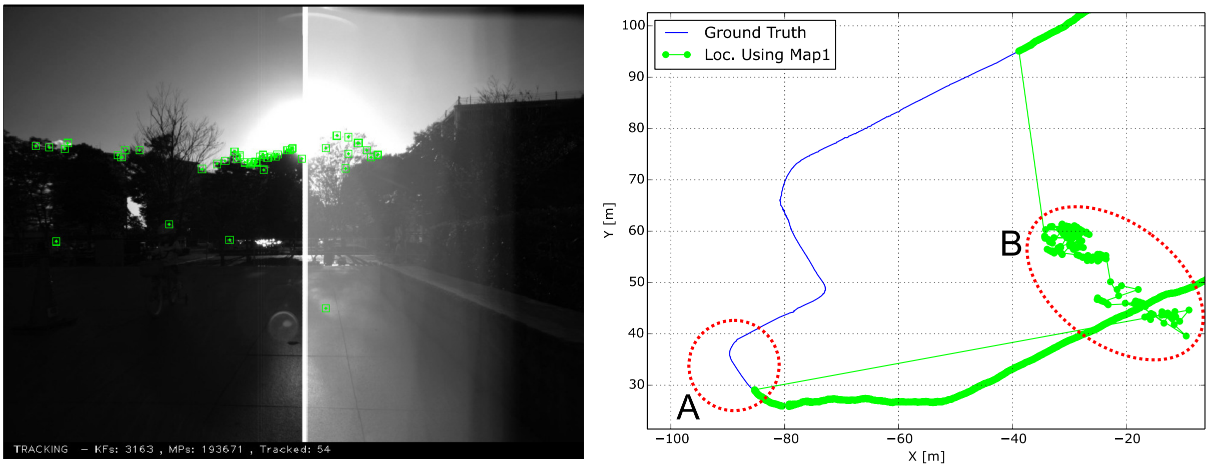
In both test runs, each map delivered a different level of performance regarding the track coverage. In Figure 3.12a for test run 1, both maps are essentially complementary to cover tracking for the whole track. However, area 1 is particularly must be concerned where ORB-SLAM loses the tracking even when using both maps. This area is deemed critical because the robot had to perform many turns successively. Also notable are some stray trajectory points from map 1; these points were traced to instances of lens flares due to the camera facing south-west while the sun was low. In the test run 2 as depicted in Figure 3.12b, both maps also provided complementary coverage. There was also significant time delay from the start of the motion to the initial position fix when using both maps.

In both test runs localization system was unable to cover the whole ground truth; the reasons were technically unrelated to ORB–SLAM capability. At all mapping runs and test runs except test run 1, camera recording stops early before reaching the finish line. Thus, map 1 and 2 were unable to cover the whole Tsukuba Challenge track (ORB–SLAM is unable to localize too long from the last keyframe in the map). Also, the camera stopped working too early in the test run 2, rendering ORB–SLAM stopped working. Percentage of tracks covered by all maps are listed in Table 3.2.



```
TRACKING  – KFs: 3317 , MPs: 197185 , Tracked: 70
```

Figure 3.14: Lost tracking is imminent after quick rotation



```
TRACKING  – KFs: 3163 , MPs: 193671 , Tracked: 54
```

Figure 3.15: At right, a part of robot trajectory is shown. Circle A shows location where disturbance took place; B shows localization results at that time. At left, camera image at corresponding time.

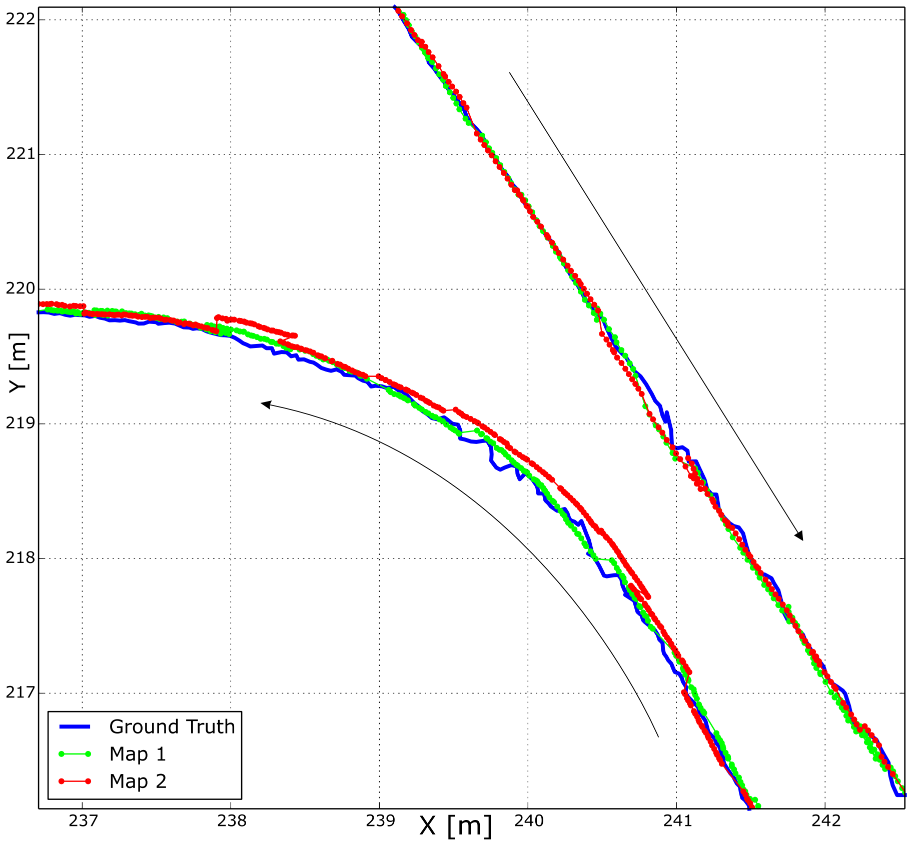


Figure 3.16: Visual comparison of Modified ORB–SLAM trajectory and ground truth in a turning situation

In general, there are two main reasons for the robot losing tracking: visual disturbances (including, but not limited to, lens smears and complete vision occlusion), and rapid rotation in part of the robot due to the appearance of dynamic obstacles. An example of visual disturbances (in form of lens smear) causing a loss of tracking and a high number of errors in track run 1 using map 1 is shown at Figure 3.15, where spurious points from localization are present.

In Tsukuba Challenge, average computation times of each frame were around 58ms. This number equals to about 19 Hz, which is lower than original ORB–SLAM that delivers around 25-30 Hz.

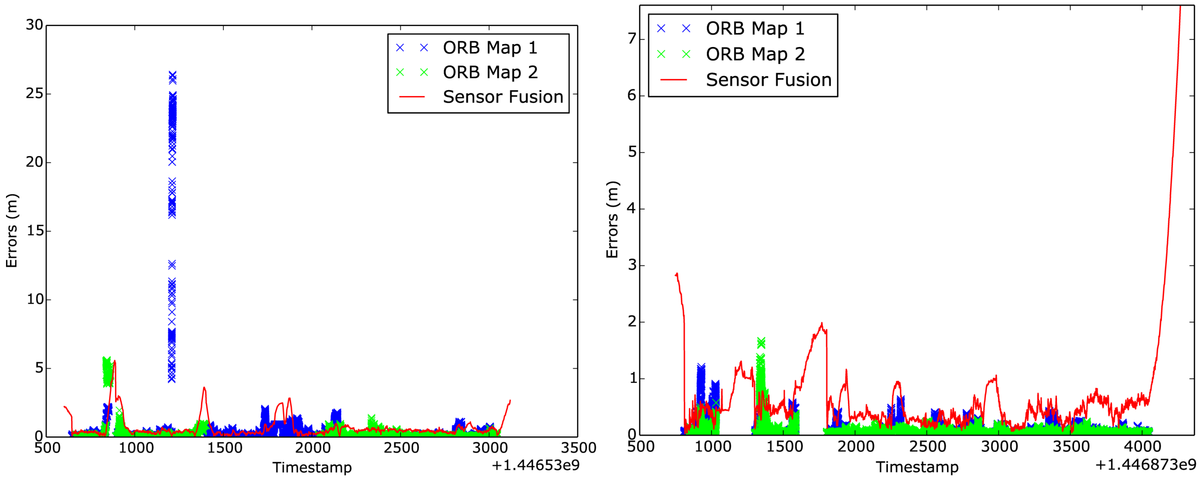
### 3.6.4 Localization Accuracy

Figure 3.16 depicts a situation in which the robot enters and exits from a turning. In tthis turning, the modified ORB–SLAM exhibits large deviations compared to ground truth, while straight path exhibits less deviation. It is also clear that each map produces different results, despite localizing in the same path and time.

Table 3.2 summarizes the performance of ORB–SLAM when covering the Tsukuba Challenge track. On average, the accuracy of ORB–SLAM is quite good when considering

| Map | Errors (in m) | | | % Coverage |
|---|---|---|---|---|
| | Average | Std. Dev | Maximum | |
| Test Run 1 | | | | |
| Map 1 | 0.38 | 1.60 | 26.41 | 68.3 |
| Map 2 | 0.19 | 0.53 | 5.62 | 70.1 |
| Joint | | | | 95.1 |
| Test Run 2 | | | | |
| Map 1 | 0.08 | 0.11 | 1.21 | 68.4 |
| Map 2 | 0.06 | 0.09 | 1.67 | 80.9 |
| Joint | | | | 82.4 |

Table 3.2: Summary of ORB–SLAM performance compared to ground truth



(a) Test Run 1

(b) Test Run 2

Figure 3.17: Error Graphs From 2015 Tsukuba Challenge

that errors in the order of 25 cm are within the range of robot's camera tracking. However, there may be some concern when this errors greatly increases, especially during test run 1. These errors may, however, be regarded as a deviation from the norms, as suggested in Figure 3.15. In particular, this problem may be solved by using a more robust camera.

### 3.6.5 Multiple Maps and Odometry

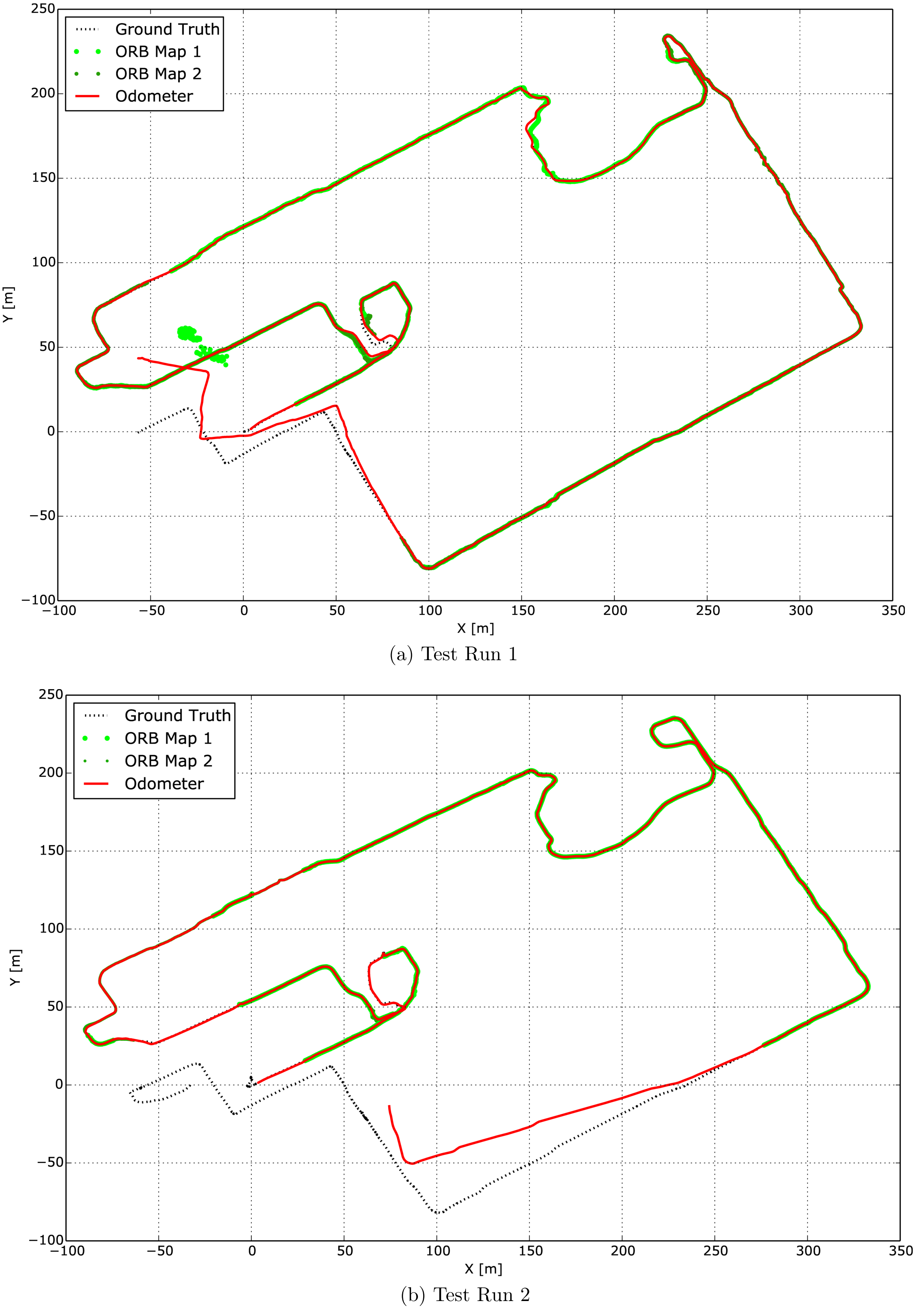
Despite attaining a good level of accuracy across the test runs in 2015 Tsukuba Challenge, ORB–SLAM was unable to maintain localization for the entire track. By recapitulating the performance summary in Table 3.2 and Figure 3.17, it is reasonable to say that larger part of the track can be covered using joint map. This subsection discusses results of sensor fusion between ORB–SLAM and odometry as formulated in subsection 3.4.4. Algorithm 2 basically outputs a distribution of possible robot pose; definitive pose for the purpose of robot control is taken by averaging this distribution.

Figure 3.18 depicts trajectories of robot in both test runs as computed by sensor fusion of odometry and ORB–SLAM using both maps. In the left figure, we can see that the sensor fusion method is capable to combine the measurement from both maps and remove noise (that came from Map 1 due to lens flare in area 3). The sensor fusion method also succeeds in covering areas where ORB–SLAM missed the tracking. A similar situation is also present in the test Run 2 whose trajectory is shown in the right. By relating the coverage graph (Figure 3.12) and error graphs (Figure 3.17), most of the spikes in sensor fusion errors can be attributed to ORB–SLAM losing tracks in area 1 and 2.

## 3.7 Summary

This chapter explains solution of monocular visual localization with an application to pedestrian environment in the 2015 Tsukuba Challenge that will be used for the next two chapters. Within the limitations of our system, the experiments confirmed that vision-based localization using augmented maps obtained from vision and LIDAR-based methods are capable of providing localization that is quite accurate for controlling the robot. Unlike the original results, ORB–SLAM was unable to produce acceptable results in dynamic environment such as Tsukuba Challenge in terms of coverage.

By using sensor fusion method between ORB–SLAM and odometer, continuous coverage of the track can be achieved. However, due to accuracy problem of the odometer, the localization may give large errors when correction from ORB–SLAM results are absent. In these results, navigation using odometer and ORB–SLAM localization has been shown as possible with good accuracy, as long as ORB–SLAM tracking is maintained.



(a) Test Run 1

(b) Test Run 2

Figure 3.18: Trajectory of fusion of ORB–SLAM and Odometry for each Test Run

# Chapter 4

# Improving Availability and Lifelong Run for Visual Localization

## 4.1 Motivations

Previous chapter has shown a possibility of vision-based localization method using a monocular camera in metric space. The method allows image-based fusion of monocular camera-based localization results and external metric-based localizations (e.g., GPS and odometry) by using particle filtering algorithm. Despite the position estimate was accurate in most areas of the Tsukuba Challenge 2015, the coverage was not enough (approximately 68 %). Therefore, the next objective of this research is towards improving coverage of our system.

Other interesting aspect of our system is possibility of lifelong localization, which refers to the capability to perform localization over a long span of time. By having multiple datasets of approximately same locations from multi-years, we would like to study the performance of the system for localizing the robot using map data from a prior year.

### 4.1.1 Contributions

The main differences of this work towards previous one (chapter 3) are an improved localization coverage based on custom vocabulary and improved global localization routine. Furthermore, visual feature maps with accurate placements of keyframes are built with prior image preprocessing and specific vocabulary for place recognition. These modifications improve localization accuracy and coverage of our visual localization method. Experiments using log data taken at 2015 and 2016 Tsukuba Challenge are used to demonstrate the effectiveness of this proposal.

The contributions in this chapter are twofold:

- Experimental proof of coverage improvements from custom vocabulary for specific scenes.

- Enhanced relocalization routine which is key for global localization.

### 4.1.2 Related Works

There are some works which address autonomous navigation in pedestrian paths ([49], [59]) which show that long-range navigation is feasible. Yet, outdoor navigation is a complicated task to achieve. Robot localization modules usually rely on prior environmental maps. As environments change with time, map maintenance is necessary. Map update is a hard task given that consistent map building with the same coordinate frame is necessary. There are also existing works regarding visual map maintenance and update (e.g., [53]).

An interesting study of lifelong vision-based localization can be found in [60]. They referred a summary map that is built from several localization trials. This means that they tried localization experiments in the same place and updated a visual feature map. By the summary map, they succeeded in lifelong visual localization over 16 months. Chapter 1 of this research also proposed similar idea which uses multiple visual maps to cope with a problem of appearance changes.

Most of the current method in robotic motion planning depends on accurate geometry of the vehicle and its environment [61]. In this regard, motion planner algorithms usually search for most optimum paths that are subject to the presence of obstacles and vehicle motion constraints. Due to this nature, planner algorithms require that both localization and obstacle detector working in metric space. However, as stated in Subsection 2.3.3, current vision SLAM methods (and thus localization) are not free from scale drift due to their inherent limitations [40]. An example of a solution of combined vision-based navigation that works in topological space is devised in [62] and [63].

A similar method to ours can be found in [64]. In this work, 3D reconstructed feature points from local bundle adjustment are matched with 3D LiDAR maps. The proposed method here is a geometric-based matching method because 3D reconstructed features are directly matched with the LiDAR maps in a metric frame. In contrast, our proposal is an appearance-based matching method because ORB–SLAM estimates own pose by comparing ORB features. These methods have different advantages over each other.

Current progress of visual place recognition for SLAM purposes have been surveyed in [65]. As mentioned in the paper, handling variable illumination conditions is critical for place recognition performance. One possible solution is by tweaking color-to-grayscale conversion; as shown in [66], this commonly ignored process has a significant contribution to the accuracy of image recognition. An interesting possibility instead of simple color conversion is to change image colors to illumination-invariant color space [67], which can be used to handle shadows and light changes throughout times of the day.

## 4.2 Proposed Methods

This chapter describes additional improvements that aim to increase coverage. Main features of current addition are:

1. Custom vocabulary for place recognition;
2. Automatic gamma control; and
3. Non-strict keyframe selection in place recognition.

Framework of our localization system is shown in Figure 4.1 and 4.2. These figures describe the map building and localization processes.

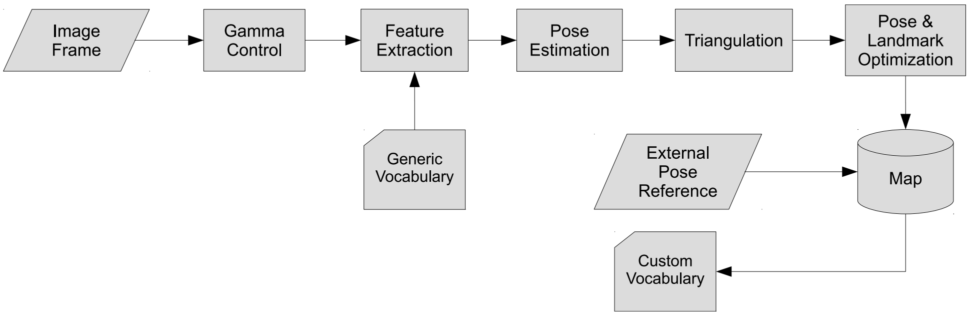


Figure 4.1: Flowchart of the proposed mapping method. Construction of custom vocabulary from the final visual map is new addition from the previous method. Gamma control is inserted as image preprocessing prior to feature extraction.

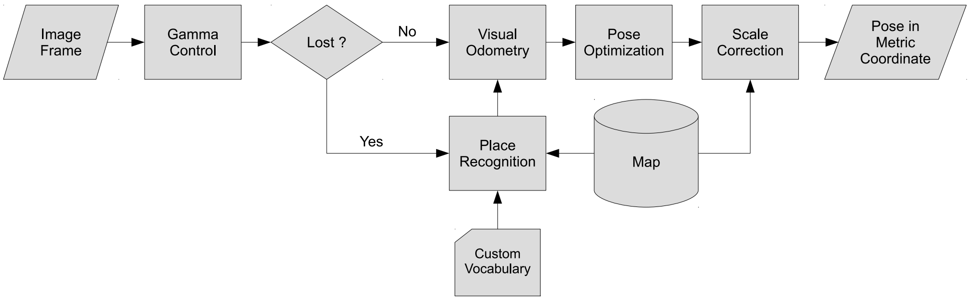


Figure 4.2: Flowchart of the proposed localization method. Place recognition subprocess uses custom vocabulary instead of generic vocabulary. Similar to mapping, localization also uses gamma control for frame preprocessing.

### 4.2.1 Custom Vocabulary

Original ORB-SLAM employs a generic vocabulary extracted from an unspecified image training sequences [27], which was noted to work well for a number of publicly available datasets. As described in [43], this vocabulary is used for transforming detected features of the image onto a sparse numerical vector (hence the name "bag-of-words"). In 2015 and 2016 Tsukuba Challenge, we found that the place recognition using generic vocabulary often failed to work. One solution from image retrieval field to increase probability of matching query image against the image database is by using vocabulary extracted specifically for particular image database as explained in [68]. Therefore, the first proposed addition for ORB-SLAM is to utilize custom vocabulary for any specific location (in this one, the Tsukuba Challenge track). The preliminary ORB–SLAM map in here is regarded as image “database”, and vocabulary is extracted after the mapping process.

Constructing image vocabulary is basically a form of vector quantization [69, 70], in which the vocabulary is arranged as a tree. The process of extracting vocabulary is performed by collecting a rich set of feature descriptors from training images. As described in [43], the extracted descriptors are discretized and clustered using k-means and inversely weighted according to its relevance in the training sequence. The whole vocabulary construction are processed by DBoW2 library [43].

### 4.2.2 Automatic Gamma Control

As described in [71], there are no feature detectors and descriptors that are truly illumination-invariant; hence feature matching may not work under varying brightness. We apply gamma correction to handle high contrast situation in daytime lighting as often encountered in the Tsukuba Challenge track. In this regard, gamma control works as a type of providing illumination invariance prior to ORB feature extraction. The gamma correction basically works by applying exponential correction for pixel value: $I_i \leftarrow I_i^\gamma$ where $I_i\,(0 \leq I \leq 255)$ is a pixel value of $i$-th pixel.

To automatically decide the value of $\gamma$, we first compute a histogram of pixel values, $h(I)$, inside a masked region on the image, A, denoted as:

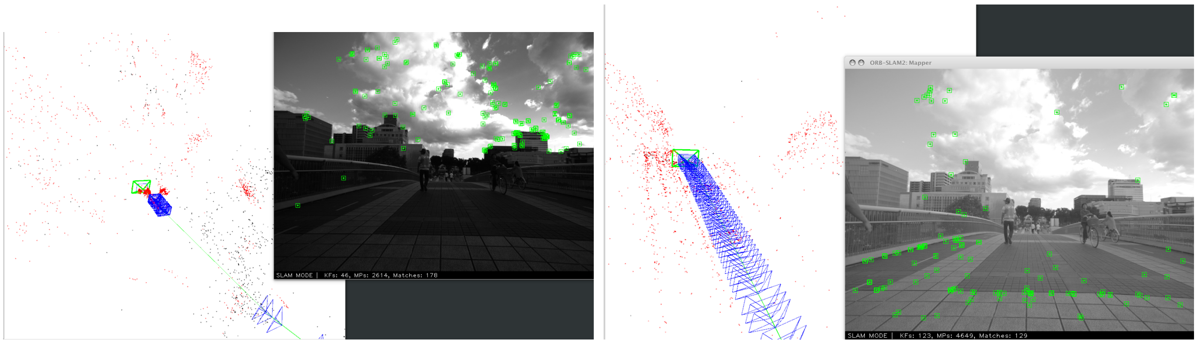
$$h(I) = \sum_{i \in A} \delta_{I,I_i}$$

where $\delta$ is Kronecker delta. Normalized cumulative distribution function (CDF), $c(I)$, is then calculated from the histogram as:

$$c(I) = \frac{\sum_0^I h(I)}{\sum_0^0 h(I)}$$

Then we compute the value of $\gamma$ to adjust for the midtone that aims to simulate human visual response against strong backlight [72] as:

$$\begin{aligned} I_{50} &= c^{-1}(0.5) \\ \gamma &= -\frac{\ln I_{50}}{\ln 2} \end{aligned}$$



SLAM MODE | KFs: 46, MPs: 2614, Matches: 178

SLAM MODE | KFs: 123, MPs: 4649, Matches: 129

Figure 4.3: Mapping without (left) and with (right) gamma correction. In left figure, almost all tracked ORB features fell in the sky and clouds, resulting in closely spaced keyframes but sparse map points; indicating relatively little motion (pyramid markers depict keyframes). In right figure; using gamma correction, result keyframes are uniformly spaced, and more map points fell in the ground with distinctive patterns following their placement in the ground.

The $\gamma$ value is calculated from masked region which represents the midtone intensity of that region; however, we the gamma correction is applied to the whole image. The masked region may be determined arbitrarily; but the best results are obtained when it is taken from lower half of image, as this region is subject to be dark when the camera is facing high contrast scenes. Effect of this gamma correction is to add brightness and contrasts in shadow areas, while reducing contrast in highlighted ones. In turn, there are more ORB features to detect and track in the ground (closer to camera). This is shown in Figure 4.3.

### 4.2.3 Non-Strict Prediction for Relocalization

Original ORB-SLAM implementation stipulates relocalization by bag-of-words (BoW) search in the internal database for looking up keyframe candidates. This set of candidates are then filtered by discarding similar keyframes, with a preference to keyframes that have a history of previous match with prior queries. The candidate filtering acts to reduce computation time, because the next step (scoring and geometry check) is quite expensive. In practice, this method often fails because either the number of candidates is too few, or the candidates do not match with geometry check. To increase success probability of relocalization, we propose a modification of candidate selection by removing the candidate filtering. Instead, we compute scores of all candidates and select 25% best keyframes. Obviously, this necessitates a trade-off between CPU usage and coverage.

To accelerate position finding, we also add searching nearest keyframes that share visible map points with last good keyframes. Consequently, this method is not usable for initializing global localization when the system starts, as no prior information of keyframes exists. The idea of searching nearest keyframes is not new, but inspired by PTAM [20]. Here, we combine this method with BoW search as a fallback. Complete

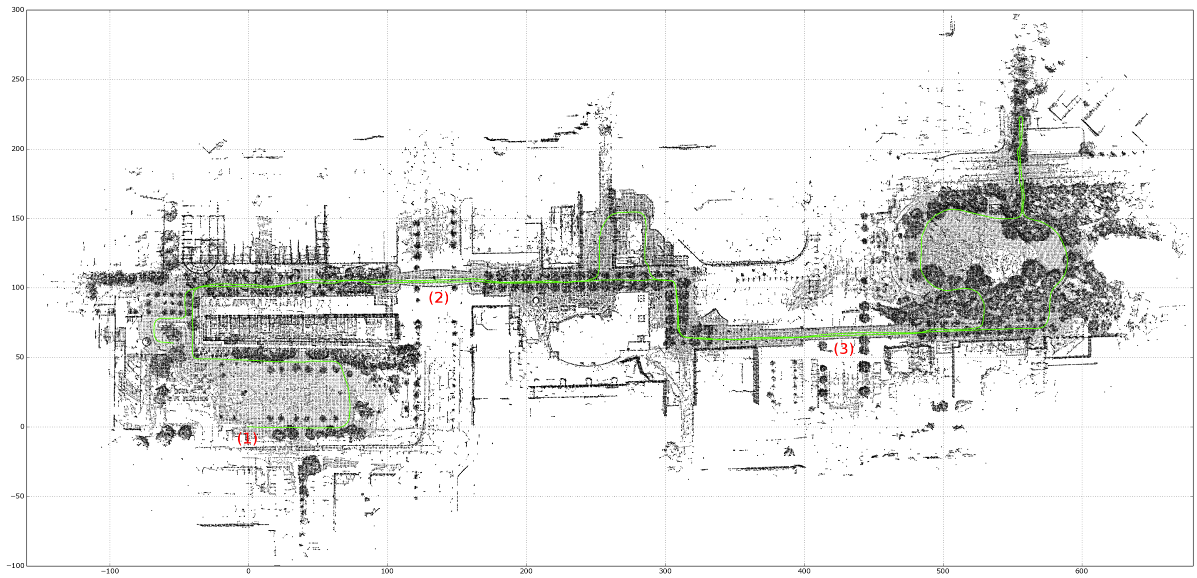


Figure 4.4: Trajectory of 2016 Tsukuba Challenge with Overlay of Top-Down Point Cloud Map Projection. (1) is starting point; (2) and (3) are the bridge area.

algorithm for relocalization is listed in Algorithm 5.

**Algorithm 5:** Relocalization after Lost Occurrence

```
Data: Image frame with feature descriptors
Result: Keyframe candidate $c$ or $\emptyset$
**if** prior keyframe is not found **then**
    Find keyframes from database that share descriptors with image frame as set $K$
**else**
    Search keyframes that have topological relation with last good keyframe as set $K$
**end if**
Compute scores for all elements in $K$
Select candidate $c$ with maximum score
Geometry check:
    Project all keypoints in $c$
    **if** Keypoints in $c$ match with image frame **then**
        return $c$
    **else**
        return $\emptyset$
```

# 4.3 Experiments

## 4.3.1 Settings

Three experiments were conducted to examine effectivity of our method. The first experiment is used to validate performance characteristics in terms of accuracy and coverage against 2016 Tsukuba Challenge track. In addition, it is also important to identify situations that may cause failure to our method. This information will be important for further deployment of vision-based localization in public road cases. The trajectory of 2016 Tsukuba Challenge is shown in Figure 4.4. One mapping run and four localization runs were conducted separately for this experiment; all runs were from the same timeframe. We also took ground truth measurements in both mapping and localization runs using results from LiDAR-based localization.

The second experiment involved previous (2015) Tsukuba Challenge dataset, on which results have been reported in Chapter 2. For this experiment, one mapping run and two localization runs were performed within the same timeframe. Similar to 2016 experiment, ground truths were established from LIDAR-based measurements.

Third experiment is by using map from 2015 dataset but applied to 2016 dataset. The objective of this experiment is to find out if the developed vision-based localization method is applicable for lifelong usage.

### 4.3.2 System Setup

To evaluate our method, we collected two types of datasets, which consists of 2015 and 2016 Tsukuba Challenge track. All datasets consist of image streams from PointGrey Grasshopper3 camera and LIDAR scans from Velodyne HDL–32. The LIDAR scans were used for establishing ground truths in both mapping and localization. All computations for mapping and localization were run in a gaming-grade laptop using Intel Core i7-6700HQ, 64GB RAM with HDD as storage.

## 4.4 Results and Discussion

| Runs | Coverage (%) | | Current Errors (m) | | | Previous Errors (m) | | |
|---|---|---|---|---|---|---|---|---|
| | Current | Previous | Avg. | Max. | St.Dev | Avg. | Max. | St.Dev |
| 11-03 | 96.7 | 68.3 | 0.74 | 13.36 | 0.91 | 0.38 | 26.41 | 1.60 |
| 11-07 | 97.3 | 68.4 | 0.68 | 3.08 | 0.49 | 0.08 | 1.21 | 0.11 |

Table 4.1: Coverage and Accuracy from 2015 Experiment

Results from the three experiments are summarized in table 4.1 for 2015 experiment, table 4.2 for 2016 experiment, and table 4.3 for long-term localization experiment. In general, our method shows improvements in term of coverage; previously, our method recorded coverage about 68 % in 2015 datasets using single map. With current modifica-

[1] This column represents ratio of length of covered track in 2016 against length of identical tracks in 2015 and 2016.

| Runs | Coverage (%) | Errors (m) | | |
|---|---|---|---|---|
| | | Avg. | Max. | St.Dev |
| 10-15 13:56 | 90.8 | 0.13 | 3.05 | 0.10 |
| 10-15 15:14 | 98.5 | 0.14 | 1.86 | 0.12 |
| 10-16 13:32 | 97.2 | 0.16 | 3.72 | 0.15 |
| 10-16 14:36 | 98.4 | 0.16 | 2.21 | 0.17 |

Table 4.2: Coverage and Accuracy from 2016 Experiment

| Runs | Coverage (%) | Cross-year Coverage (%)[1] |
|---|---|---|
| 10-15 13:56 | 19.7 | 75.8 |
| 10-15 15:14 | 16.7 | 64.0 |
| 10-16 13:32 | 25.7 | 98.5 |
| 10-16 14:36 | 16.5 | 64.0 |

Table 4.3: Coverage of Localization in 2016 experiment using 2015 Map

tion, our single-map vision-based localization shows a high percentage (90 % at minimum) when using maps created from corresponding date and time (ie. same year).

Table 4.3 shows coverage performance of our vision-based localization in 2016 experiment using a map created from 2015 as a type of lifelong localization. Overall, map created from previous year does not perform well due to low overlap between each trajectory (26.1%). However, a relative comparison only for overlap areas results in favorable results of lifelong localization. Due to significantly different ground truths between two years, we could not report on accuracy of lifelong localization.

### 4.4.1 Coverage

As shown in Table 4.1, the coverage for first experiment (2015 datasets) shows a high level of coverage; 96.7 % and 97.3 % for first and second runs respectively. Compared to previous results in Table 3.2, there have been significant improvements in term of localization coverage; previous results recorded coverage of 68 % at worst. In current results, coverage improves to more than 97 %. This means that the vision-based localization has seen improvement in term of speed to recover from lost occurrence. The potential areas of lost are shown in Figure 4.5. We found that these lost events mostly took place either in the turnings or strong intensities (and smears).

For the second experiment, our visual localization method showed more variations, but still exhibited a high rate of coverage. Lowest coverage came from the first run of 2016 (15th October 13:56) that amounts to 91 %. In this dataset, as shown in Figure 4.6, we encountered long part of lost occurrence that happened after hard bump prior to entering the bridge. Another significant part of unrecoverable vision localization was in the forest area, in which the camera was facing the sun, thus getting frequent lens smears.

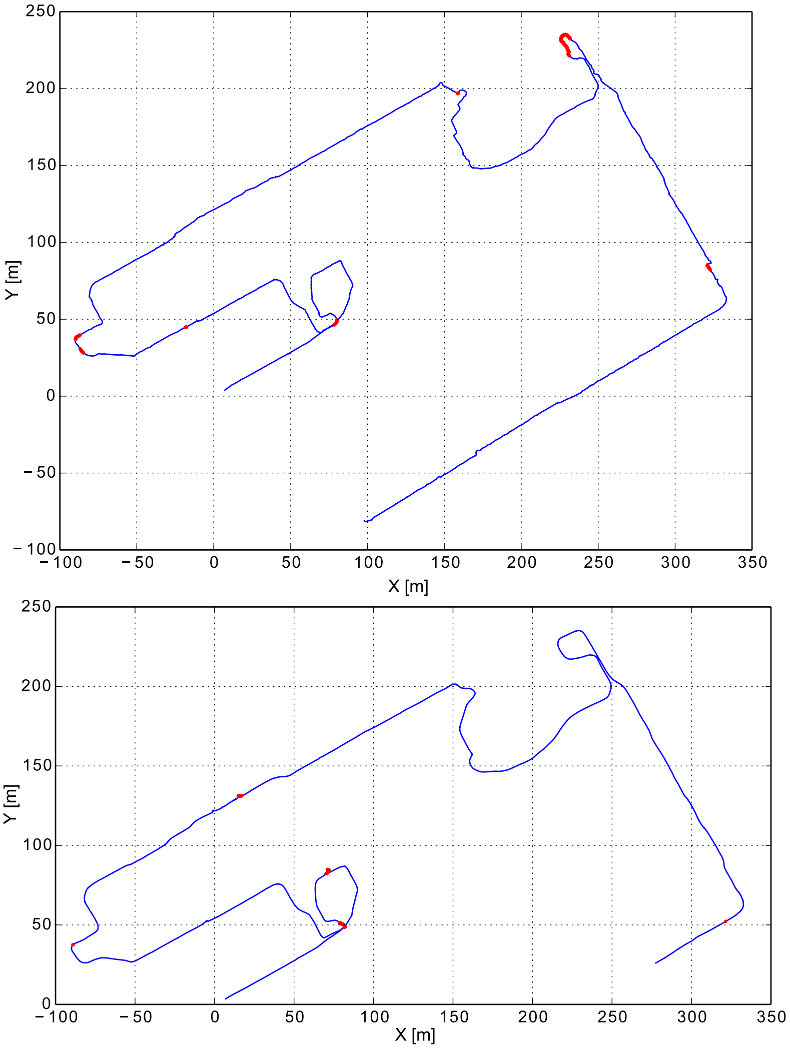
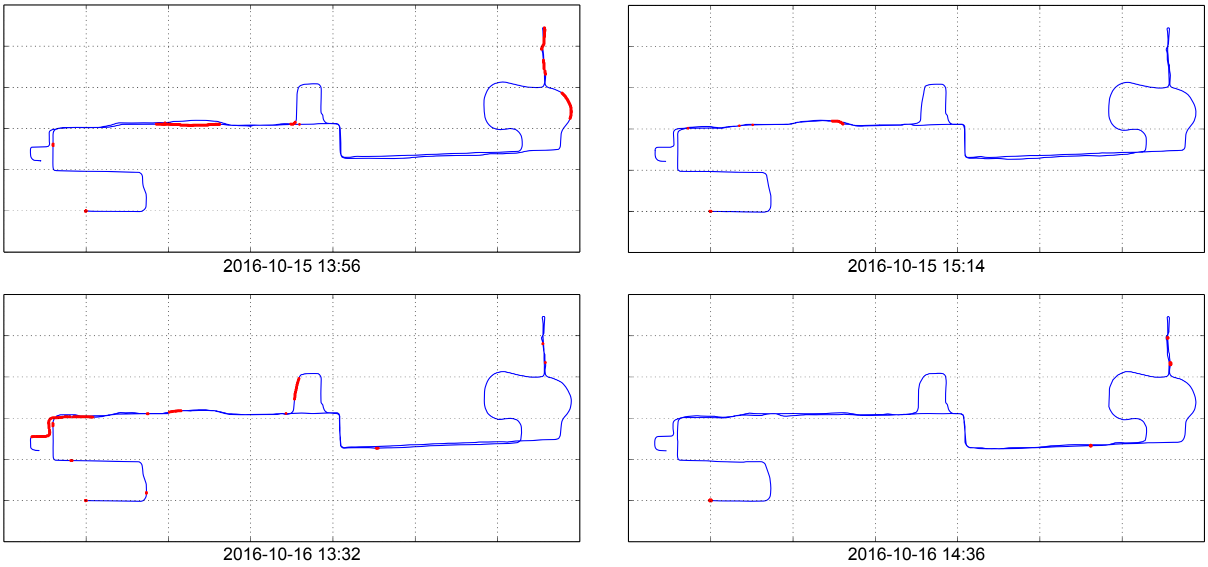


Figure 4.5: Lost events positions according to ground truth in the 2015 datasets experiment as pointed by bold points. Top: lost occurences for 2015-11-03. Bottom: 2015-11-07.


2016-10-15 13:56

2016-10-15 15:14

2016-10-16 13:32

2016-10-16 14:36

Figure 4.6: Positions of lost occurences in the 2016 datasets as pointed by red markers

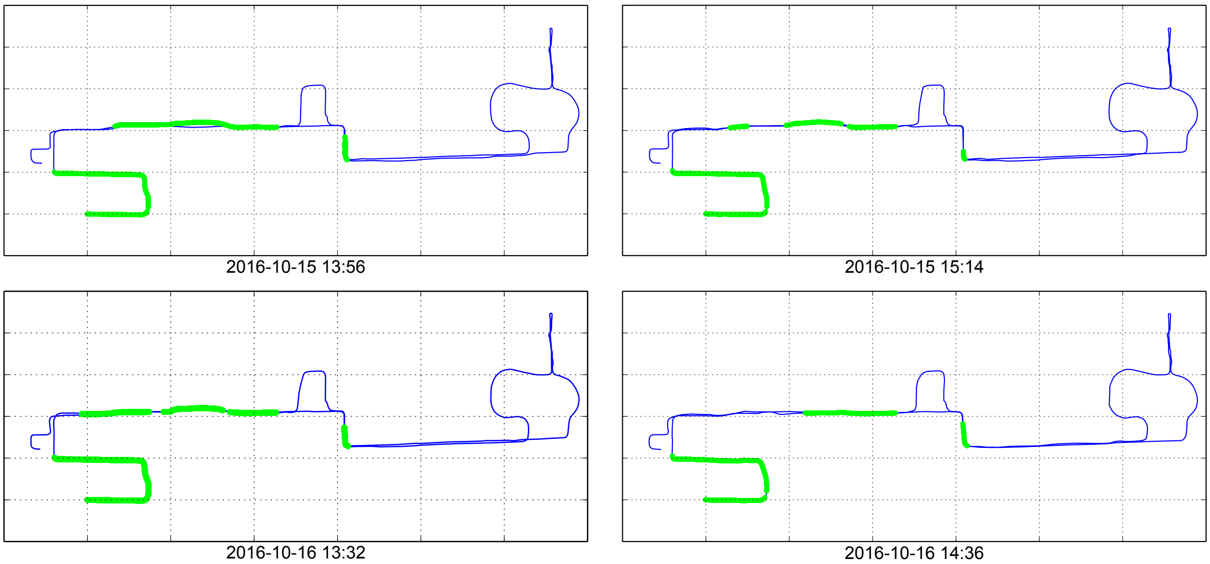


Figure 4.7: Coverage plots of 2016 track using 2015 map; green markers point to navigable areas.

In the third experiment, we performed localization in the 2016 datasets using a map created from 2015 (Table 4.3). Overall, 2015 map could only cover less than 26 % of the 2016 track. This is understandable, since only 26.1 % of 2016 trajectory that overlaps with 2015 one.

A particular part of Tsukuba Challenge that is difficult for lifelong localization is the paved pedestrian area covered by large trees. In 2015 datasets, a large amount of this area was littered with falling leaves; however, this cover was almost non-existent in 2016. Therefore, the changes between both years were substantial; making the place recognition subsystem failed. Figure 4.8 shows an example of this situation. Inversely, prominent places where static image features are dominant and highly visible make the place recognition easier. Examples of these places are the starting point and the bridge area (pointed by (1) and (2) in Figure 4.4). Overall, coverages plot for the third experiment is shown in Figure 4.7.

### 4.4.2 Accuracy

Tables 4.1 and 4.2 list errors of our vision-based localization method in all experiments. For the 2015 datasets, the new method recorded lower accuracy than previous one, as shown by the average errors for both testing runs. However, in the first test runs of 2015 dataset, the large maximum errors was improved to 13.4 m from previous error of 26.4 m. This improvement did not take place in the second run, as maximum error had increased to 3.1 m.

The current method registered much better accuracy in the 2016 experiment. The maximum average errors are now below 20 cm, while the maximum errors are significantly

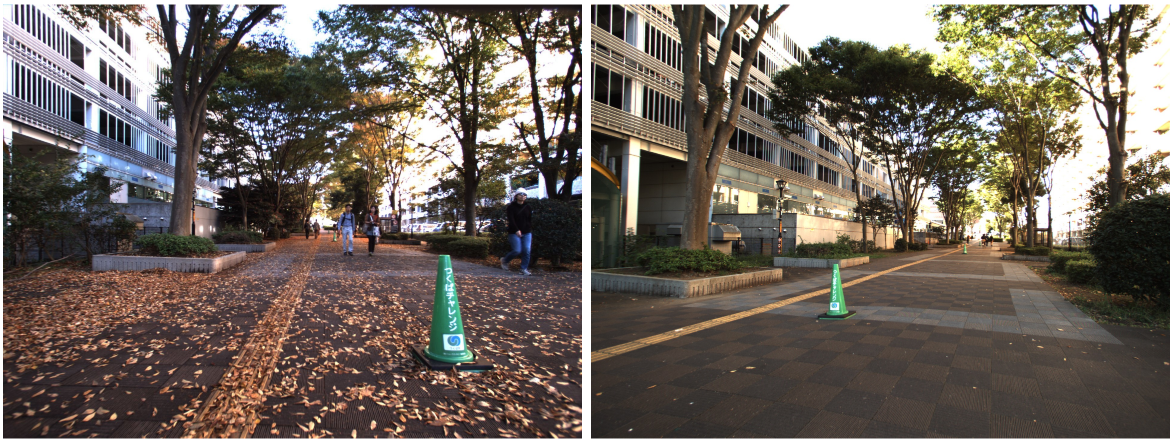


Figure 4.8: Same place, unrecognized: left is 2015 situation, while right is from 2016 in the same place

reduced below 4 m. Also, overall maximum errors have dropped significantly below in the order of below 4 m. To identify sources of localization errors and how they develop, the size of errors are plotted as circles in their respective locations for each experiment. For the sake of brevity, only one dataset is plotted from each experiment as all of the datasets behave similarly in term of error distributions. This error distribution relative to locations are plotted in Figure 4.9. From both experiments, most of large errors occurred in three location types:

- Before and/or after recovery from lost
- Hard turns
- Open space, where image features fall in far places

Relationships between accuracy and coverage for all datasets are shown in Figure 4.10. Here, for 2016 experiment, most of the time (above 90 %) the localization system was able to provide positions within errors below 50 cm, which is adequate for most purpose of navigation. For the rest of time, sensor fusion with odometry will be able to cover the localization requirement. This sensor fusion is also able to mask the large “jumps” that occasionally appears. For 2015 experiment, during 80 % of time the localization system could only provide accuracy within 1.2 m, which is not enough for navigation. This was caused by time discrepancies between image stream and LiDAR.

### 4.4.3 Computational Time

Figure 4.11 plotted fluctuation of per-frame computational time of typical localization test. On average, per-frame time amounts to 83.4 milliseconds, that equals to 12 frame per seconds. This is slightly lower to 15 fps of camera image rate that we use, but usable for real-time (as comparison, typical LiDAR-based localization methods make for 10 Hz due to hardware scan rate). However, when localization is lost, the system will perform

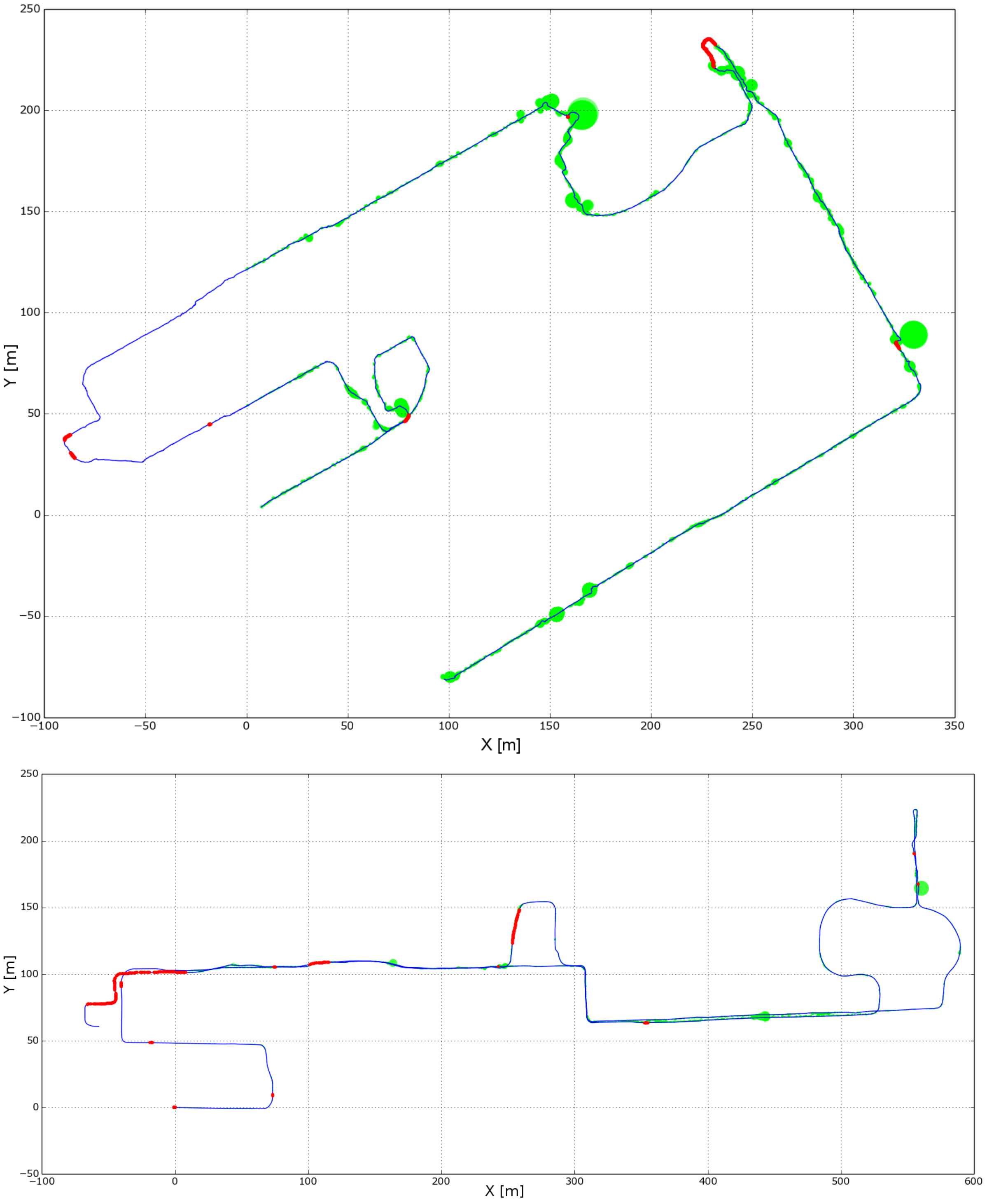


Figure 4.9: Error Distribution by Position for 2015 (top) and 2016 (bottom) Experiment. Lost occurrences are marked by red points.

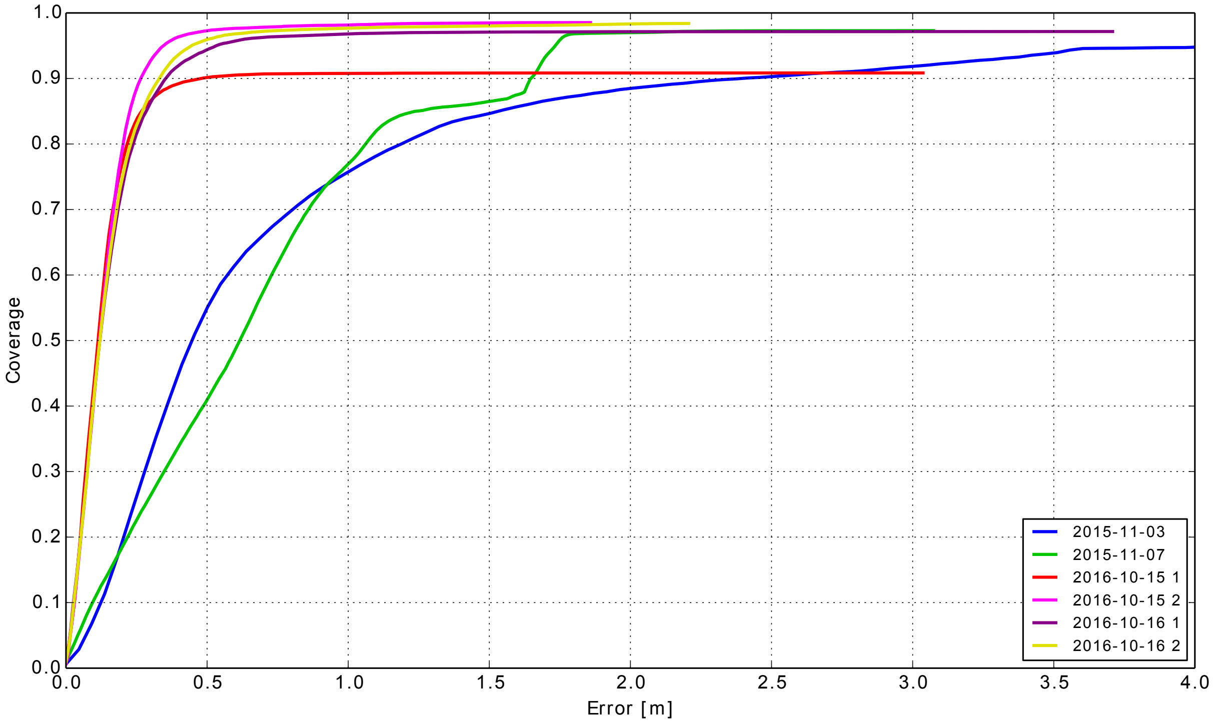


Figure 4.10: Cumulative Distribution Function of Errors in Each Dataset

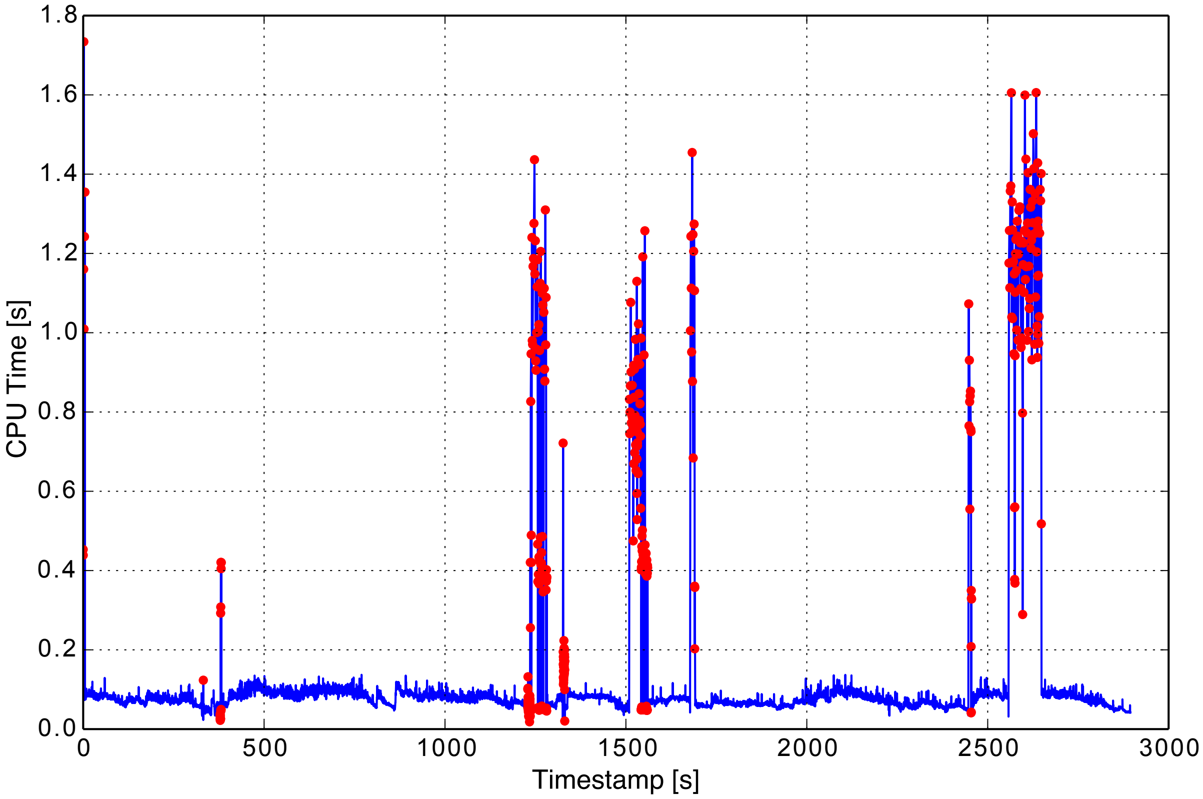


Figure 4.11: Fluctuation of CPU Time per Frame; lost occurences are marked by red points.

global relocalization by place recognition that increases computation time significantly (occasionally exceeding 1 second). Compared to the original ORB–SLAM, these surges of CPU usage are not good signs for real-time usage. The increase of time amount is proportional to the number of keyframe candidates for place recognition.

### 4.4.4 Discussions

From these three experiments and by looking at three parameters (coverage, accuracy and computational time), this chapter is concluded with the following findings:

1. *Modified keyframe search for place recognition with custom vocabulary works correctly.*

   Compared to our previous results, the localization system has successfully addressed lack of coverage. In the 2015 datasets experiment and 2016 datasets experiment as shown in Figure 4.5 and Figure 4.6, lost occurrences have dropped significantly and the system is now able to recover quickly.

2. *Image features from both far and near places are required.*

   Features from prominent landmarks (ie. buildings) could help for place recognition. However, presences of this type of features without features from near places may cause visual odometry subsystem to deduce very small motion. In contrast, near-place features (eg. from trees and paving blocks) are not quite useful for landmarks as shown by the third experiment because they are prone to changes. In this regard, gamma control to regain brightness in dark areas has a contribution for accuracy of localization as it could help to recover features from near places.

3. *Lifelong localization is possible.*

   In the third experiment, localization performed successfully in the places that had not change considerably. Also, as shown from second findings above, prominent landmarks in the frame will help for global localization.

4. *There is trade-off between robust place recognition and CPU usage.*

   Compared to the original ORB-SLAM, the current keyframe search method basically performs brute-force search against all candidates rather than filters just the most likely ones. As a consequence, this increases CPU time as the complexity of scoring function of keyframe matches is linear against the number of features.

# Chapter 5

# Evaluation of Visual Localization in Public Road

## 5.1 Backgrounds

In Chapter 3, the applicability of vision-based localization in metric space in a pedestrian environment has been proposed and tested with mobile robots. In this chapter, the same method is exercised in an urban setting with a real passenger car. This environment presents different challenges compared to previous one. First, vehicle (and camera) velocity is higher than mobile robots in Tsukuba Challenge; therefore image frames for input will present difficulties in visual odometry routines. Next, there is a higher degree of environmental changes, in addition to appearance variations. Prominent examples are temporary road obstructions, background clutter, motion blur from moving vehicles, pedestrians and weather variations.

Early efforts for vision-based mapping and localization in autonomous vehicle area were centered in topological methods, such as the FAB–MAP [73]. Other notable work is [74] that proposed a method for topological global localization based on adaptive mapping with an omnidirectional camera where the system performance is measured in terms of correct localization percentage. Later, the work in [53] presents a visual mapping system using stereo camera which updates a metric map. This work shows the importance of robust visual place recognition to update the map and recover from localization errors.

## 5.2 Experimental Settings

### 5.2.1 Vehicle Settings

Figure 5.1 shows the experimental vehicle. This research uses a LiDAR (Velodyne HDL64E–S2), a monocular camera (Point Grey grasshopper3), Differential GNSS-IMU

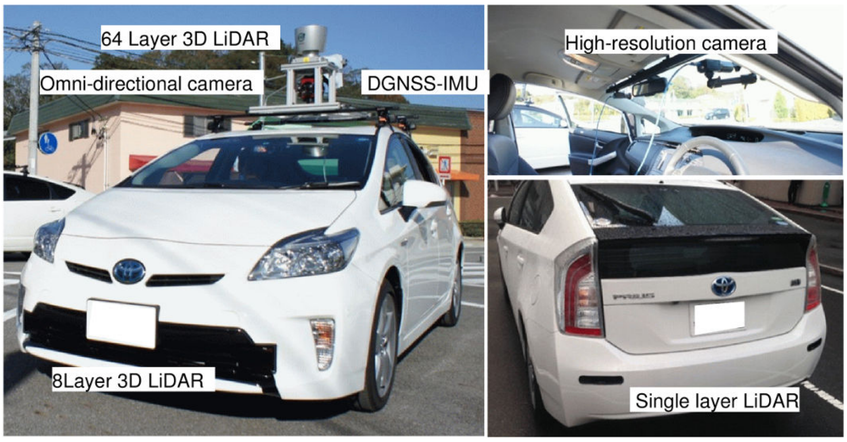


Figure 5.1: Experiment car and its on-board sensors

| Runs | Start | Stop | Weather |
|---|---|---|---|
| 01-21 | 13:40 | 15:00 | Overcast |
| 01-21 N | 15:10 | 17:00 | Overcast |
| 01-26 | 09:00 | 11:00 | Sunny |
| 01-29 | 11:10 | 12:30 | Rainy |
| 02-03 | 09:00 | 11:00 | Sunny |
| 02-05 | 13:40 | 15:00 | Sunny |
| 02-09 | 13:40 | 15:00 | Rainy |
| 02-12 | 08:30 | 09:50 | Cloudy |
| 02-12 N | 13:20 | 14:20 | Cloudy |
| 02-24 | 10:30 | 11:50 | Sunny |

Table 5.1: Time and Condition for Mapping and Testing Runs Around Nagoya University Campus

(JAVAD DELTA-G3T with IMU) and the CAN information coming from the vehicle. The Velodyne HDL64E–S2 is mounted on the top, with measurement range around 120 m, the vertical angular range is 26.9° and measurement period is 10 Hz. The GNSS is used to obtain the initial position for NDT localization.

### 5.2.2 Datasets

Datasets for evaluation of our system were collected by running the vehicle around Nagoya University campus for multiple days. The trajectory is shown in Figure 5.2. For each run, image sequence for localization tests and LiDAR scans for ground truth were collected simultaneously. Images were captured in full HD resolution at a rate of 20 Hz, but downscaled to 800 × 600 pixels for processing. We also captured raw CAN[1] data for processing into odometry information, which will be used for sensor fusion.

In general, the vehicle traversed different loop patterns of trajectories that vary from

[1] CAN (Controller Area Network) bus is a data communcation standard for communications between microcontrollers inside a vehicle.

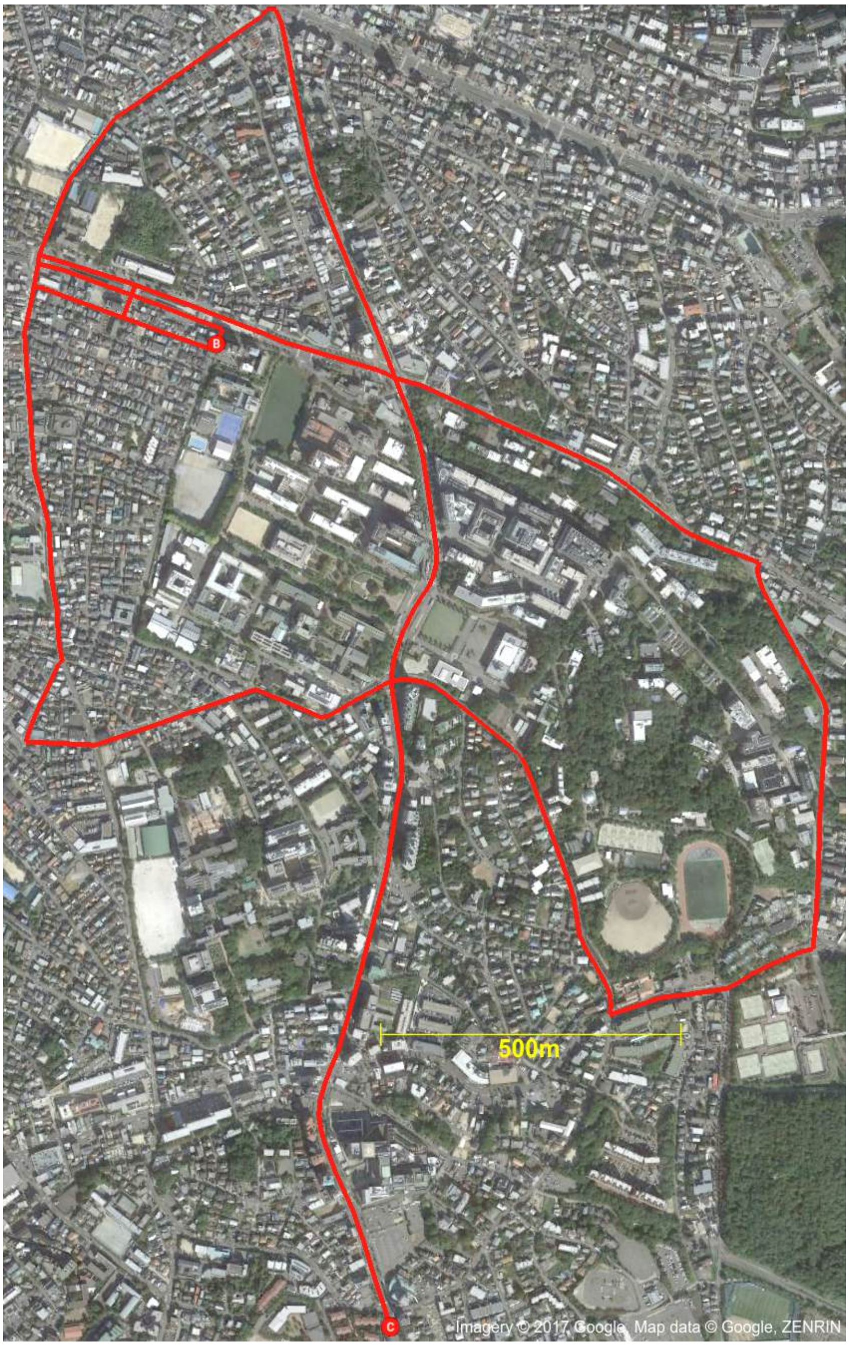


Figure 5.2: Vehicle path used for evaluation

9 to 30 km. The track is dominated by suburban environment, with occasional high-rise buildings along the road. Time and weather condition of each mapping and testing runs are recorded in Table 5.1. From these runs, three were selected to be fetched into mapping process (marked in yellow). All other runs were designated as localization runs using these three maps.

### 5.2.3 Evaluation Criteria

To evaluate our system, two numerical criteria are used to gauge quality of localization: *coverage* and *accuracy.*

1. *Coverage*: Coverage of a single map in one vehicle run is defined as percentage of time that the map is capable to localize without being lost. This measurement could provide a rough description the of capability of the maps and localization system when coping with changing condition.

2. *Accuracy*: Quality of localization results are evaluated by metric errors, which represent distances from predicted vehicle positions against ground truths provided by the NDT scan matching. These errors are further separated into lateral and longitudinal ones, in order to understand the coupling between the visual localization system and motion planner. For each run, means and maximum values were taken to illustrate the capability of this system.

## 5.3 Results and Discussions

Our algorithm is implemented under the ROS (Robotic Operating System) framework, that runs under middle-range PC. On average, a single map requires around 3–4 GB memory.

### 5.3.1 Mapping Process

As described in Chapter 3, the mapping process requires the vehicle to record image sequences and true camera poses in the world. Next, these image sequences are processed through ORB–SLAM to produce keyframes and map points, and corresponding vehicle poses are appended to each keyframe. These augmented keyframes and map points are then saved to map files in a disk for subsequent localization.

Two mapping runs were conducted on the same day with cloudy weather, with another one in a different day with sunny weather.As shown in Figure 5.5, maps from cloudy day provided better coverage than a sunny one. We suppose that this low coverage was caused by prevalent visual disturbances that occur both in mapping and localization. Figure 5.3

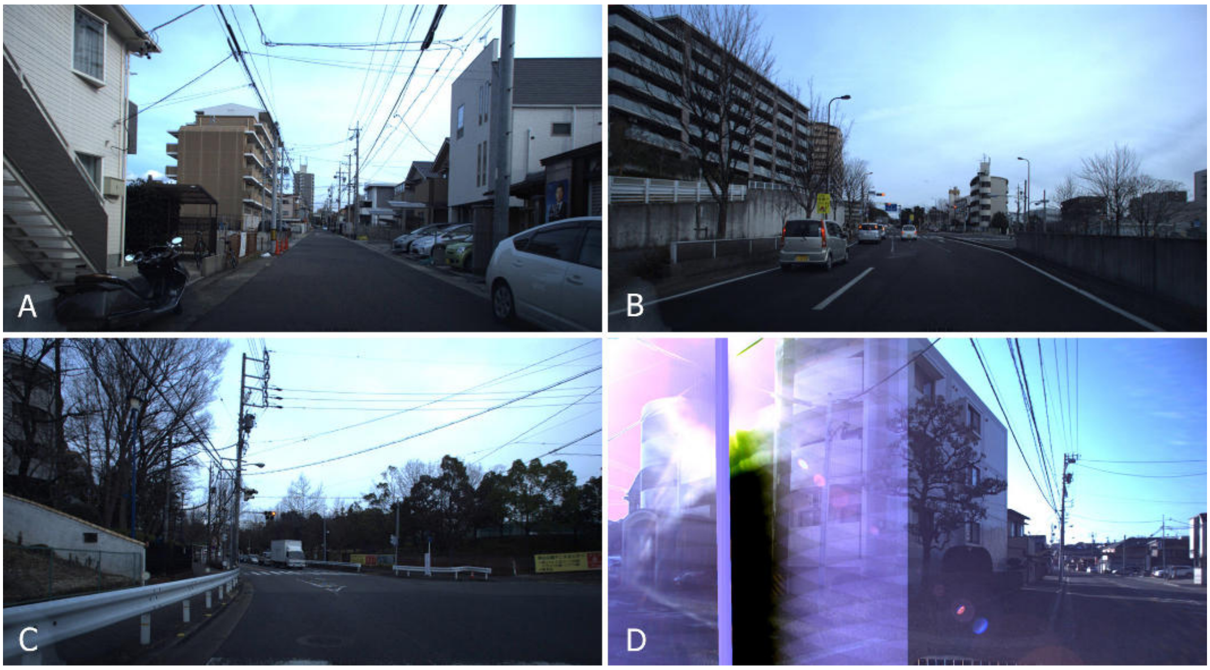


Figure 5.3: Mapping situations around Nagoya University campus

gives common situations that occurred in mapping runs. Most part of the track covers suburban areas (A) and wide road (B), with a forest as other substantial parts (C). As expected in a sunny day, the vehicle may encounter visual disturbance such as lens smear and flares (D).

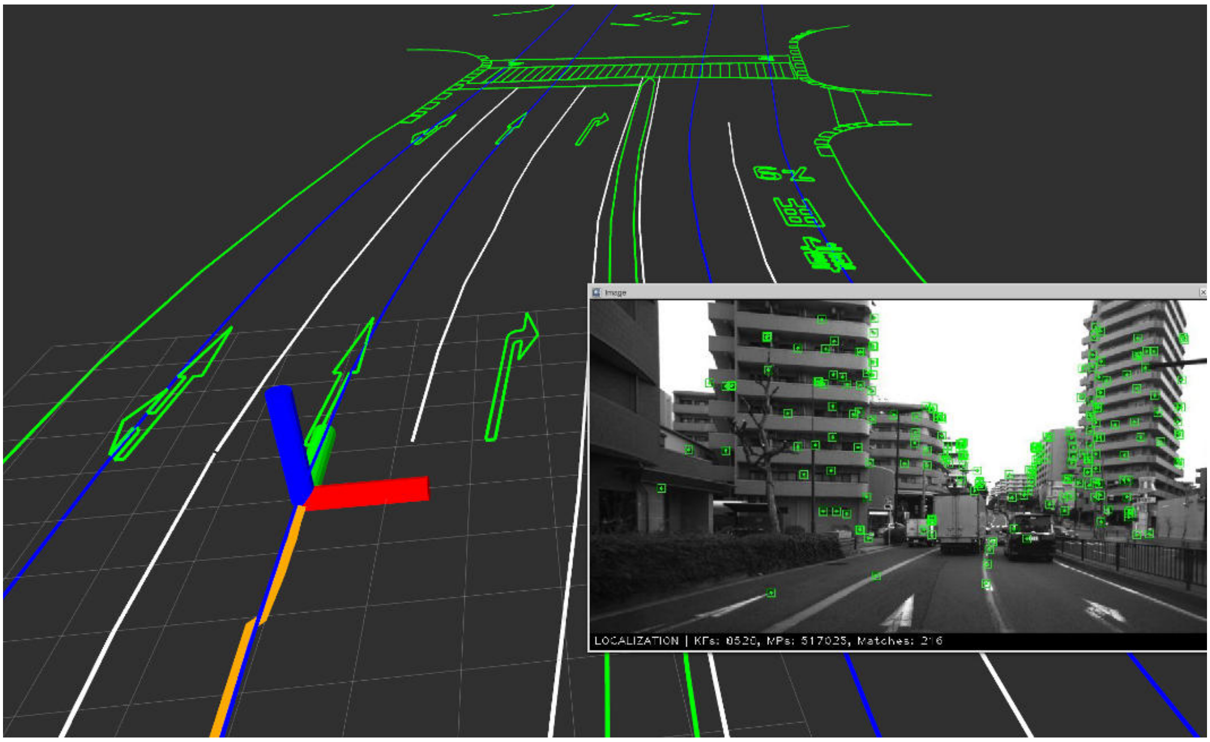
### 5.3.2 Localization Results Overview

Figure 5.4 depicted visualization of localization process. The colored axis shows vehicle position from sensor fusion of visual localization and odometry, overlaid with metric vector map that covers road marks. In the inset, visual feature tracking is being performed by ORB–SLAM, projecting map points currently visible in the image.

### 5.3.3 Coverage

Each map gives different performance regarding the coverage for localization. Figure 5.5 shows how coverage of each map changes in every run, at which map 3 gave the lowest coverage. During its creation, we noticed that the camera experienced heavy lens smears and flares during the run. This condition also occurred during localization run 02–03. Meanwhile, map 1 and 3 that were created on same day delivered similar coverage.

For an extreme example, we plot the coverage of all maps in runs 02–03 (lowest coverage from all maps) at Figure 5.6. In this particular run, there are significant parts of the track in which all maps were unable to provide localization. From this observation, it is interesting that there is a possibility to improve coverage by jointly using results



LOCALIZATION | KFs: 0520, MPs: 517025, Matches: 216

Figure 5.4: Visualization of localization at work

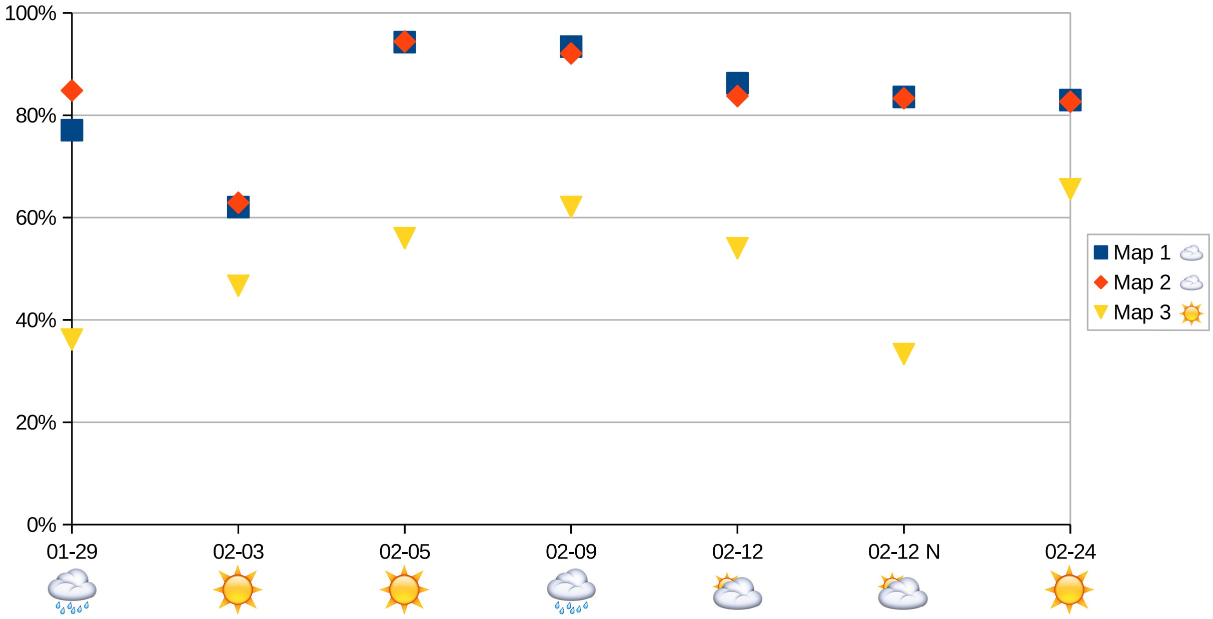


Figure 5.5: Comparison of coverage from each map for each of vehicle runs

| Runs | Lateral | | | | | | Longitudinal | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Map 1 | | Map 2 | | Map 3 | | Map 1 | | Map 2 | | Map 3 | |
| | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. |
| 01-29 | 0.21 | 0.33 | 0.17 | 0.30 | 0.28 | 1.06 | 0.51 | 1.39 | 0.40 | 0.79 | 1.45 | 12.04 |
| 02-03 | 0.22 | 0.46 | 0.21 | 0.63 | 0.20 | 0.47 | 0.51 | 1.51 | 0.43 | 3.82 | 0.67 | 2.09 |
| 02-05 | 0.28 | 0.57 | 0.16 | 0.32 | 0.17 | 0.27 | 0.43 | 1.66 | 0.36 | 0.81 | 0.54 | 2.08 |
| 02-09 | 0.31 | 0.68 | 0.16 | 0.26 | 0.24 | 0.94 | 0.49 | 1.50 | 0.41 | 0.96 | 1.13 | 10.17 |
| 02-12 | 0.30 | 0.64 | 0.18 | 0.27 | 0.21 | 0.51 | 0.46 | 1.21 | 0.49 | 0.99 | 0.76 | 5.27 |
| 02-12 N | 0.28 | 0.50 | 0.23 | 0.46 | 0.20 | 0.42 | 0.51 | 1.28 | 0.44 | 0.79 | 0.70 | 4.90 |
| 02-24 | 0.28 | 0.58 | 0.24 | 0.41 | 0.50 | 1.28 | 0.74 | 2.62 | 0.24 | 0.71 | 2.06 | 11.36 |

Table 5.2: Accuracy of Vision-Only Localization (errors in meter)

| Runs | Lateral | | | Longitudinal | | |
|---|---|---|---|---|---|---|
| | Average | Maximum | Std.Dev. | Average | Maximum | Std.Dev. |
| 01-29 | 0.20 | 2.98 | 0.25 | 1.08 | 5.01 | 0.86 |
| 02-03 | 0.54 | 48.50 | 2.78 | 1.46 | 48.34 | 2.58 |
| 02-05 | 0.21 | 3.08 | 0.26 | 0.86 | 3.80 | 0.61 |
| 02-09 | 0.27 | 4.73 | 0.38 | 1.09 | 4.60 | 0.88 |
| 02-12 | 0.22 | 3.73 | 0.26 | 1.04 | 4.73 | 0.88 |
| 02-12 N | 0.25 | 6.48 | 0.32 | 1.27 | 4.36 | 0.87 |
| 02-24 | 0.22 | 4.55 | 0.27 | 0.97 | 5.73 | 0.83 |

Table 5.3: Accuracy of Sensor Fusion of Multi-Map Visual Localization and Odometry (errors in meter)

from different maps together in a consistent manner.

### 5.3.4 Accuracy of Vision-Only Localization

From the coverage graph, we expect that good map coverage will deliver accurate metric localization. Unfortunately, this is not always the case. Figure 5.7 depicted a scene where some large but very brief "jumps" had taken place. These localization errors came from map 3, that has been described to have problems during its creation. However, other maps may also exhibit similar behavior randomly. All statistics of error rate from vision-only localization are shown in Table 5.2. In general, average errors, both laterally and longitudinally are small. However, for map 3 we found that it has substantially large standard deviation compared to other maps in the longitudinal direction.

### 5.3.5 Accuracy of Sensor Fusion

Having knowledge that single map is unable to cover the whole track, we also performed sensor fusion experiments to test the viability of results of combining multiple maps with odometry from CAN data and provide single localization results. Table 5.3 shows average,

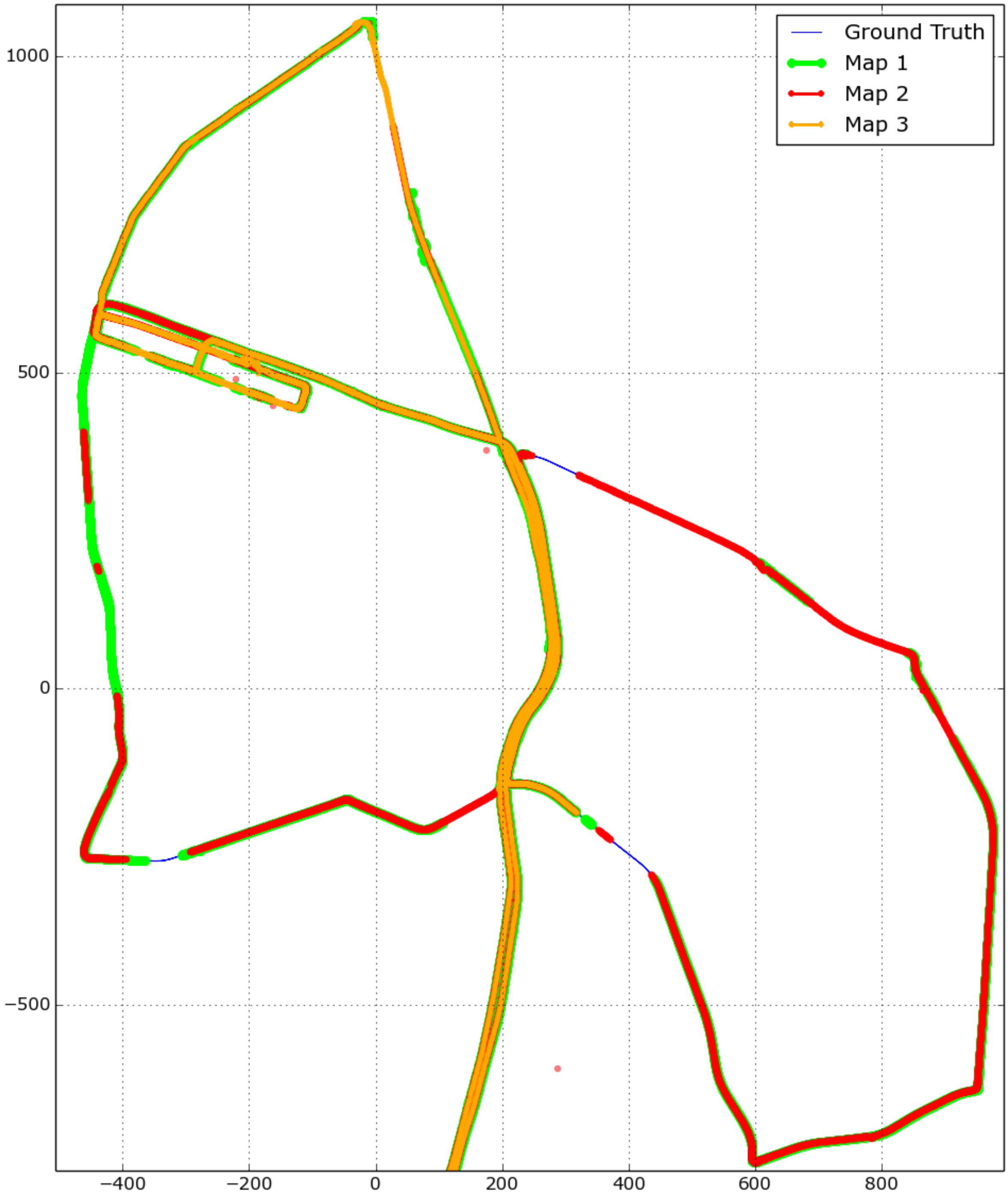


Figure 5.6: Trajectory coverage for 02-03 dataset

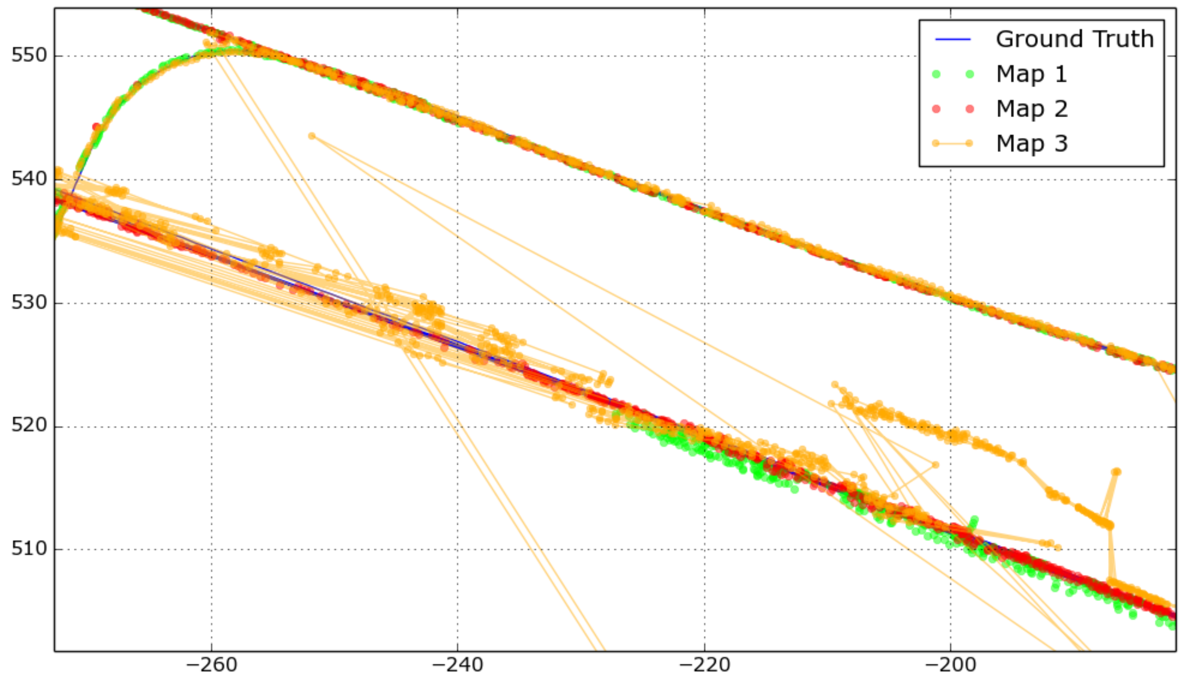


Figure 5.7: Severe case of large lateral and longitudinal errors in run 02-03

maximum and standard deviation of our sensor fusion approach during all localization test runs. In that table, multiple maps and sensor fusion approach is capable to jointly suppress large error that may result from single localization results.

### 5.3.6 Performance Comparison of Sensor Fusion and Vision-Only Localization

To compare performance between sensor fusion and vision-only localization, we take an example from run 02-03 that will be representative of large errors from both methods and plot cumulative distributive function (CDF) of lateral and longitudinal errors in Figure

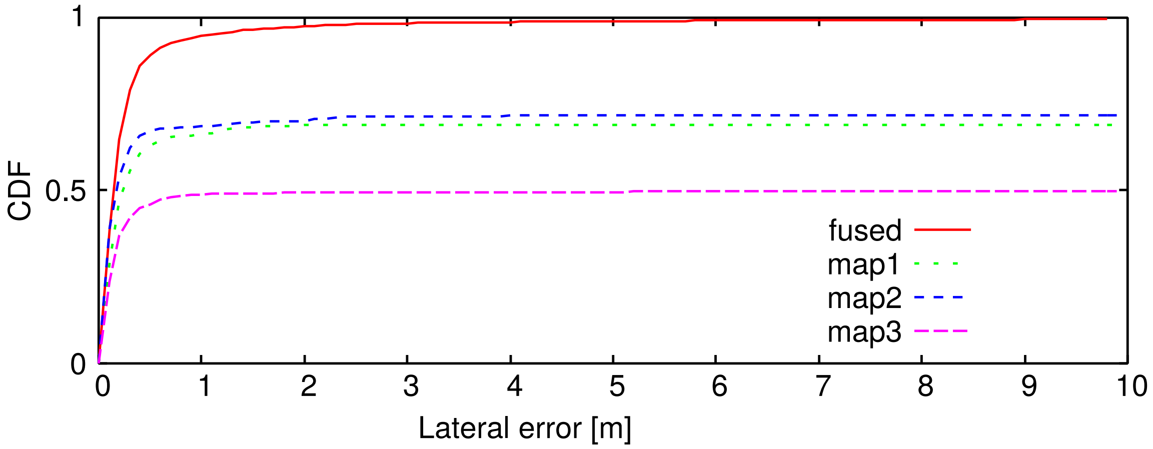


Figure 5.8: Lateral Error CDF Plot for Run 02-03

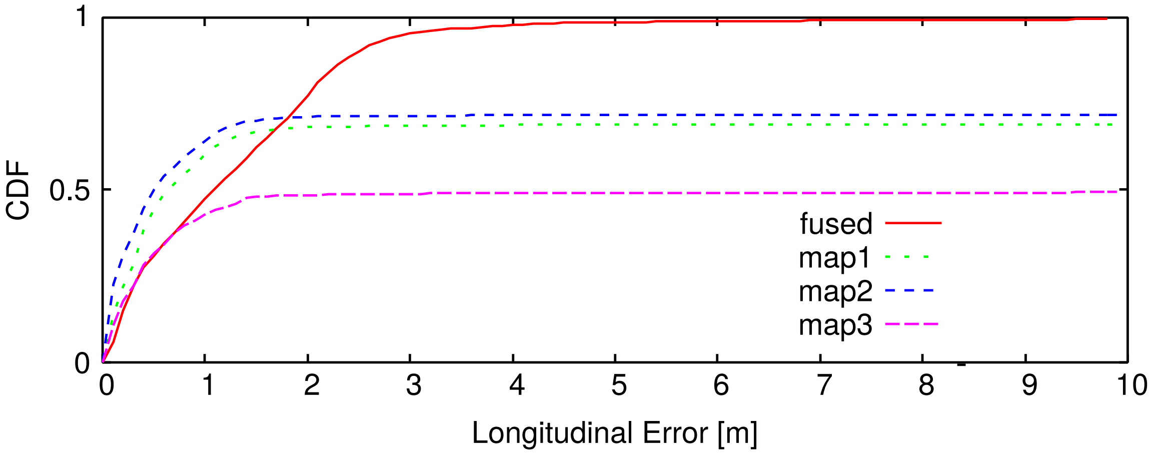


Figure 5.9: Longitudinal Error CDF Plot for Run 02-03

| Runs | Lateral | | | Longitudinal | | |
|---|---|---|---|---|---|---|
| | Average | Maximum | Std.Dev. | Average | Maximum | Std.Dev. |
| 01-29 | 1.64 | 15.70 | 1.47 | 1.12 | 14.85 | 1.23 |
| 02-03 | N/A | N/A | N/A | N/A | N/A | N/A |
| 02-05 | 1.09 | 9.64 | 0.87 | 0.88 | 8.22 | 0.67 |
| 02-09 | 0.97 | 8.76 | 0.62 | 0.79 | 7.10 | 0.55 |
| 02-12 | 1.03 | 3.81 | 0.52 | 0.77 | 4.88 | 0.52 |
| 02-12 N | 1.11 | 8.49 | 0.68 | 0.88 | 5.03 | 0.57 |
| 02-24 | 1.23 | 13.06 | 0.93 | 0.95 | 13.07 | 0.77 |

Table 5.4: GNSS Localization Errors

5.8 and 5.9, respectively. In each figure, the horizontal axis represents size of errors in meters, while vertical one represents the percentage of error data covered into the corresponding error size.

Figure 5.8 shows that lateral error distribution of sensor fusion converges to unity faster than vision-only localization. In other words, the sensor fusion localization of multiple maps results in better accuracy than any single maps used for vision-only localization. A similar situation also occurs in Figure 5.9, which shows cumulative distribution of longitudinal errors from all localization methods. From these plots, it is clear that sensor fusion of multiple vision maps and odometry localization is capable to suppress large errors that may occur from vision-only localization. This error suppression has its limit, that is when vision localization lost for a quite long time.

Table 5.4 shows GNSS localization results using same datasets (N/A=not available). Average lateral errors estimated by our vision-based localization method are smaller than that of GNSS localization results. Although longitudinal error average of our method is close to that of GNSS results, standard deviations of GNSS longitudinal error is smaller than that of our method. These results suggest that accuracy of longitudinal estimation would be improved by fusing GNSS data in the particle filter.

# Chapter 6

# Conclusions and Future Works

## 6.1 Conclusions

This research aims to develop and investigate a vision-based localization system that only uses monocular camera for the purpose of autonomous vehicles. One major requirement of this system is a capability to provide vehicle position in metric space as input for currently developed motion planner. Another requirement is reliability under environmental challenges, which is important for deployment in the real world. Therefore, issues covering the usability of this system in real situations must be identified.

For the first part, the basics of monocular localization in metric space are developed and tested in a real, convenient environment (i.e. Tsukuba Challenge). Notwithstanding the deviation that occurred in extreme situations, the experiments confirmed that vision-based localization using augmented maps obtained from vision and LIDAR-based methods are capable of providing localization within an order of centimeters, which is quite accurate for controlling small robots in outdoor environments. Unlike the original results, ORB-SLAM was unable to produce acceptable results in dynamic environments such as Tsukuba Challenge. Meanwhile, coverage of the vision-only localization system was quite low (below 70 %). The low coverage indicates that the place recognition subsystem as part of ORB-SLAM still has a problem of recovering from lost events.

By using sensor fusion method between ORB–SLAM and odometer, we can achieve continuous coverage of the track. However, due to accuracy problem of the odometer, the localization may give large errors when updates from ORB–SLAM results are not acquired quickly. In these results, navigation using odometer and ORB–SLAM localization has been shown as possible with good accuracy, as long as ORB–SLAM tracking is maintained. The sensor fusion approach, combined with multiple maps in vision side has also been shown to improve the coverage.

In the second part that covered 2016 Tsukuba Challenge event, the goal is to improve the coverage of vision-only localization system from the previous year. In addition, testing of possibility for lifelong localization from our visual localization system is required. Solving coverage problems require three parts. First, utilization of custom vocabulary for the particular track are introduced, including how to build and restore it in mapping and localization phase. Second, automatic gamma control is added as an attempt to control highly variable illumination. Lastly, keyframe searching policy is extended during localization to increase success probability of keyframe search. From the point of view of coverage, the localization system is capable to provide high availability, higher than 90 % for all the log data tested in the Tsukuba track. It also could cope with data from different years providing coverage of 75 %, as long as environmental changes were low.

The capability to support fully vision-based navigation is still limited due to a concern of large errors that may occur in some areas. It has been shown previously that it is possible to combine this method with other metric localization systems such as odometry with particle filter which can cope with these issues. It is left for future work the exploration for methods that strive to increase accuracy for any general situations.

In the third part, the possibility for operations of visual localization in urban road environment has been shown. The vision-only localization provided coverage only in good visual conditions. Vision-odometry sensor fusion and combined with utilization of multiple maps maintained full coverage and provided smoother pose estimated results. As a comparison, our vision-based localization has been shown to have better accuracy than high-end GNSS. However, truly autonomous navigation based on low-cost sensors (in this case, monocular vision and IMU) is still difficult because of reliability issues regarding large errors and low coverage that may occur.

As part of the contribution to autonomous vehicle research community, the implementation of ORB–SLAM developed for this research has been included as part of Autoware. Autoware is a suite of open-source software for urban autonomous driving that includes a number of functionalities such as vehicle control, localization and object detection [75].

## 6.2 Future Works

The area of visual-kinematic navigation is a fast-moving research field, in which this research is targeted to. Therefore, the future works are aimed to solve general problems of usability and reliability, especially for handling extreme situations.

1. The scale correction method in Algorithm 1 requires nearest keyframe search after initial pose computation by ORB–SLAM. Our investigation in Chapter 5 shows that this process is prone to errors, especially in high contrast situations. Therefore, the first priority is to remove the scale correction in order to derive metric positioning. Especially, pose estimation by the camera (in mapping phase) should be eliminated and replaced by accurate pose from NDT (the tightly–coupled visual–inertial map-

ping) as mentioned in subsection 2.3.6. By doing this, triangulation process will potentially result in metric landmarks and map points. This will lead to the elimination of scale correction because pose estimation has already been in metric space. Another effect is that particle filter may be evaluated directly in metric coordinate, as particle position is now able to use map point projection for scoring.

2. Operations in low-light and high velocity conditions often cause image matching failure because of motion blur. The root cause of this failure is point-type feature detectors, which are not invariant against image blur. Application of region or blob-type feature detectors such as MSER [76] could potentially alleviate matching failure. However, replacing feature detectors will require significant modifications in SLAM workflow.

3. Another area worth investigating is *Long-Term Mapping*; which would provide a capability to build and "grow" a map of the area from different times [60, 53]. It has to be acknowledged that multiple maps approach is actually not scalable, as CPU and storage requirements grow linearly with the number of maps. Instead of multiple maps, the approach of "growing" single map will increase a degree of autonomy in part of consumer vehicles.

4. Visual place recognition as a major component of this system should be developed further because it can be decoupled from other components and combined with LiDAR-based systems as GNSS replacement. In this regard, deep learning-based place recognition deserves to get further attention as better candidate [65, 77].

# Acknowledgements

All praise to Allah, Lord of the Universe, who has given me the power against all odds to finish this research.

This thesis is presented to the people of Indonesia and Japan, as a symbol of cooperation and prosperity among them. Financial and technical support in this research come from MEXT COI stream program, A Diverse and Individualized Social Innovation Hub - The “Mobility Society” for the Elderly: Lead to an Active and Joyful Lifestyle.

First thanks are addressed to Prof. Shinpei Kato, Prof. Yoshiki Ninomiya and Eijiro Takeuchi, who gave me an opportunity to continue higher education in Japan beyond my home country. In addition, I thank my advisors (Prof. Masato Edahiro and Prof. Hiroaki Takada) for helping me to polish this thesis. I also expressed gratitude to all lab members, especially Luis Yoichi Morales, PhD and Naoki Akai, PhD for helping me through this research. Also, special thanks are given to Mobility Group members and Autoware developers who have given much feedback in this research.

I would like to express deepest gratitude for my family and parents, who have given their relentless support both from Indonesia and Japan. May Allah gives His greatest grace for them.

And lastly, great thanks are expressed to Indonesian Ministry of Research, Technology and Higher Education who had gave me permission and opportunity to pursuit overseas education.

# List of Publications

## Journals

1. Adi Sujiwo, Tomohito Ando, Eijiro Takeuchi, Yoshiki Ninomiya, Masato Edahiro. "Monocular Vision-based Localization using ORB–SLAM with LiDAR-aided Mapping in Real-World Robot Challenge", *Journal of Robotics and Mechatronics*, Vol 28 No. 4, pp. 479-490, 2016. Corresponds to Chapter 3.

2. Adi Sujiwo, Eijiro Takeuchi, Luis Yoichi Morales, Naoki Akai, Hatem Darweesh, Yoshiki Ninomiya, Masato Edahiro. "Robust and Accurate Monocular Vision-Based Localization in Outdoor Environments of Real-World Robot Challenge", *Journal of Robotics and Mechatronics*, Vol.29 No.4, pp. 685-696, 2017. Corresponds to Chapter 4.

## Conference

Adi Sujiwo, Eijiro Takeuchi, Luis Yoichi Morales, Naoki Akai, Yoshiki Ninomiya, Masato Edahiro. "Localization Based on Multiple Visual-Metric Map", in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2017)*, Daegu, Korea. Awarded Best Conference Paper, corresponds to Chapter 5.

# Bibliography

[1] R. Behringer, S. Sundareswaran, B. Gregory, R. Elsley, B. Addison, W. Guthmiller, R. Daily, and D. Bevly, "The DARPA grand challenge - development of an autonomous vehicle," in *2004 IEEE Intelligent Vehicles Symposium*, pp. 226–231, June 2004.

[2] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic.* No. 56 in Springer Tracts in Advanced Robotics, Springer Science & Business Media, Nov. 2009.

[3] H. Lipson and M. Kurman, *Driverless: Intelligent Cars and the Road Ahead.* MIT Press, Sept. 2016.

[4] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots.* MIT Press, 2 ed., Feb. 2011.

[5] T. Vanderbilt, "Let the robot drive: The autonomous car of the future is here," Jan. 2012.

[6] F. Moosmann, *Interlacing Self-Localization, Moving Object Tracking and Mapping for 3D Range Sensors.* PhD thesis, Institut für Mess- und Regelungstechnik mit Maschinenlaboratorium (MRT), Karlsruhe, 2013. ISBN:978-3-86644-977-0.

[7] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB.* Springer Science & Business Media, Nov. 2011.

[8] Ü. Özgüner, T. Acarman, and K. A. Redmill, *Autonomous Ground Vehicles.* Artech House, 2011.

[9] J. A. Farrell, *The Global Positioning System & Inertial Navigation.* McGraw Hill Professional, Jan. 1999.

[10] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE Robotics Automation Magazine*, vol. 13, pp. 99–110, June 2006.

[11] J. M. V. Verth and L. M. Bishop, *Essential Mathematics for Games and Interactive Applications, Third Edition.* CRC Press, Sept. 2015.

[12] D. M. Bevly and S. Cobb, *GNSS for Vehicle Control.* GNSS technology and applications series, Artech House, 2010.

[13] S. Miller, X. Zhang, and A. Spanias, *Multipath Effects in GPS Receivers: A Primer.* Morgan & Claypool, Dec. 2015.

[14] P. McManamon, *Field Guide to Lidar.* SPIE, 2015.

[15] A. Nüchter, *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom.* Springer, Dec. 2008.

[16] E. Mendes, P. Koch, and S. Lacroix, "ICP-based pose-graph SLAM," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 195–200, Oct. 2016.

[17] P. Biber and W. Strasser, "The normal distributions transform: a new approach to laser scan matching," in *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings*, vol. 3, pp. 2743–2748 vol.3, Oct. 2003.

[18] M. Magnusson, *The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection.* Örebro: Örebro universitet, 2009.

[19] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, vol. 23, pp. 3–20, Jan. 2006.

[20] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007. ISMAR 2007*, pp. 225–234, Nov. 2007.

[21] L. M. Paz, P. PiniÉs, J. D. TardÓs, and J. Neira, "Large-Scale 6-DOF SLAM With Stereo-in-Hand," *IEEE Transactions on Robotics*, vol. 24, pp. 946–957, Oct. 2008.

[22] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), no. 8690 in Lecture Notes in Computer Science, pp. 834–849, Springer International Publishing, Jan. 2014.

[23] O. Faugeras, Q.-T. Luong, and T. Papadopoulo, *The Geometry of Multiple Images: The Laws that Govern the Formation of Multiple Images of a Scene and Some of Their Applications.* MIT Press, 2004.

[24] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[25] R. Szeliski and S. B. Kang, "Shape ambiguities in structure from motion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 5, pp. 506–512, 1997.

[26] M. Lourakis and X. Zabulis, "Accurate scale factor estimation in 3d reconstruction," in *Computer Analysis of Images and Patterns*, pp. 498–506, Springer, 2013.

[27] 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, Oct. 2015.

[28] T. Moons, L. v. Gool, and M. Vergauwen, *3D Reconstruction from Multiple Images, Part 1: Principles.* Now Publishers Inc, Oct. 2009.

[29] M. Bigas, E. Cabruja, J. Forest, and J. Salvi, "Review of CMOS image sensors," *Microelectronics Journal*, vol. 37, pp. 433–451, May 2006.

[30] J. Nakamura, *Image Sensors and Signal Processing for Digital Still Cameras.* Taylor & Francis, 2005.

[31] R. Szeliski, *Computer Vision: Algorithms and Applications.* Springer Science & Business Media, Sept. 2010.

[32] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *The International Journal of Robotics Research*, p. 0278364916679498, 2016.

[33] Y. Li, S. Wang, Q. Tian, and X. Ding, "A survey of recent advances in visual feature detection," *Neurocomputing*, vol. 149, pp. 736–751, 2015.

[34] S. Krig, *Computer Vision Metrics: Textbook Edition.* Springer, Sept. 2016.

[35] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, Nov. 2004.

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

[37] T. Tuytelaars and K. Mikolajczyk, “Local invariant feature detectors: a survey,” *Foundations and trends® in computer graphics and vision*, vol. 3, no. 3, pp. 177–280, 2008.

[38] E. Rosten, R. Porter, and T. Drummond, “Faster and Better: A Machine Learning Approach to Corner Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 105–119, Jan. 2010.

[39] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” *Computer Vision–ECCV 2010*, pp. 778–792, 2010.

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

[41] Sean Eron Anderson, “Bit Twiddling Hacks.”

[42] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g2o: A general framework for graph optimization,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3607–3613, IEEE, 2011.

[43] D. Galvez-López and J. Tardos, “Bags of Binary Words for Fast Place Recognition in Image Sequences,” *IEEE Transactions on Robotics*, vol. 28, pp. 1188–1197, Oct. 2012.

[44] A. I. Mourikis and S. I. Roumeliotis, “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, Apr. 2007.

[45] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, “Robust visual inertial odometry using a direct EKF-based approach,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 298–304, Sept. 2015.

[46] A. Concha, G. Loianno, V. Kumar, and J. Civera, “Visual-inertial direct SLAM,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1331–1338, May 2016.

[47] R. Mur-Artal and J. D. Tardós, “Visual-Inertial Monocular SLAM With Map Reuse,” *IEEE Robotics and Automation Letters*, vol. 2, pp. 796–803, Apr. 2017.

[48] A. Spaenlehauer, V. Fremont, Y. A. Sekercioglu, and I. Fantoni, “A Loosely-Coupled Approach for Metric Scale Estimation in Monocular Vision-Inertial Systems,” (Daegu), IEEE, July 2017. arXiv: 1707.07518.

[49] Y. Morales, A. Carballo, E. Takeuchi, A. Aburadani, and T. Tsubouchi, "Autonomous robot navigation in outdoor cluttered pedestrian walkways," *Journal of Field Robotics*, vol. 26, no. 8, p. 609, 2009.

[50] A. Ohshima and S. Yuta, "Teaching-Playback Navigation by Vision Geometry for Tsukuba Challenge 2008," tech. rep.

[51] O. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 02, pp. 485–508, Sept. 1988.

[52] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, "A comparison of loop closing techniques in monocular SLAM," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1188–1197, 2009.

[53] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 1156–1163, IEEE, 2009.

[54] H. Lategahn, *Mapping and Localization in Urban Environments Using Cameras*, vol. 28. KIT Scientific Publishing, 2014.

[55] D. Burschka and G. D. Hager, "V-GPS (SLAM): Vision-based inertial system for mobile robots," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 1, pp. 409–415, IEEE, 2004.

[56] E. Takeuchi and T. Tsubouchi, "A 3-D Scan Matching using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3068–3073, Oct. 2006.

[57] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics.* MIT Press, Aug. 2005.

[58] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The New College Vision and Laser Data Set," *The International Journal of Robotics Research*, vol. 28, pp. 595–599, May 2009.

[59] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous Robot Navigation in Highly Populated Pedestrian Zones," *Journal of Field Robotics*, vol. 32, pp. 565–589, June 2015.

[60] P. Mühlfellner, M. Bürki, M. Bosse, W. Derendarz, R. Philippsen, and P. Furgale, "Summary Maps for Lifelong Visual Localization," *Journal of Field Robotics*, vol. 33, pp. 561–590, Aug. 2016.

[61] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles," *arXiv:1604.07446 [cs]*, Apr. 2016. arXiv: 1604.07446.

[62] C. Siagian and L. Itti, "Biologically Inspired Mobile Robot Vision Localization," *IEEE Transactions on Robotics*, vol. 25, pp. 861–873, Aug. 2009.

[63] C.-K. Chang, C. Siagian, and L. Itti, "Mobile robot vision navigation & localization using gist and saliency," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 4147–4154, IEEE, 2010.

[64] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular Camera Localization in 3d LiDAR Maps," *Proc. ˜ of the IEEE.*

[65] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, pp. 1–19, Feb. 2016.

[66] C. Kanan and G. W. Cottrell, "Color-to-Grayscale: Does the Method Matter in Image Recognition?," *PLOS ONE*, vol. 7, p. e29740, Jan. 2012.

[67] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill, and P. Newman, "Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles," in *Proceedings of the Visual Place Recognition in Changing Environments Workshop, IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China*, vol. 2, p. 3, 2014.

[68] J. Sivic and A. Zisserman, "Efficient Visual Search of Videos Cast as Text Retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 591–606, Apr. 2009.

[69] J. Sivic and A. Zisserman, "Video Google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 1470–1477 vol.2, Oct. 2003.

[70] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, pp. 2161–2168, 2006.

[71] J. Heinly, E. Dunn, and J.-M. Frahm, "Comparative evaluation of binary features," in *Computer Vision–ECCV 2012*, pp. 759–773, Springer, 2012.

[72] A. Adams, *The Negative.* New York Graphic Society, 1976.

[73] M. Cummins and P. Newman, “FAB-MAP: Probabilistic localization and mapping in the space of appearance,” *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[74] F. Dayoub and T. Duckett, “An adaptive appearance-based map for long-term topological localization of mobile robots,” in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3364–3369, Sept. 2008.

[75] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, “An Open Approach to Autonomous Vehicles,” *IEEE Micro*, vol. 35, pp. 60–68, Nov. 2015.

[76] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, pp. 761–767, Sept. 2004.

[77] Z. Chen, A. Jacobson, N. Sunderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, “Deep Learning Features at Scale for Visual Place Recognition,” in *arXiv:1701.05105 [cs]*, Jan. 2017. arXiv: 1701.05105.