Scale Estimation for Monocular SLAM using Depth from Defocus

by

Tomoyuki Shiozaki

A thesis submitted in partial fulfilment of the
requirements for the degree of Master of Engineering (Research)

at the

Centre for Autonomous Systems
Faculty of Engineering and Information Technology
University of Technology Sydney

March 2018

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signed: _____

Date: _____

Scale Estimation for Monocular SLAM using Depth from Defocus

by

Tomoyuki Shiozaki

A thesis submitted in partial fulfilment of the requirements for the degree of Master of Engineering (Research)

Abstract

An autonomous robot must map its environment and estimate its egomotion to perform effectively. Monocular simultaneous localization and mapping (SLAM) can generate maps of the robot's environment, except for the absolute scale. Alternatives based on stereo or RGB-D camera based SLAM systems can obtain the metric scale but have disadvantages in terms of the cost, size and power requirements. This thesis is focused on the development of an absolute metric scale monocular SLAM system for autonomous robots. A depth from defocus (DfD) technique that relies on image blur is used to estimate the metric scale. However, existing methods for DfD suffer from ambiguities caused by texture, motion blur, and the location of the focal plane. The novelty of this research is combining DfD with camera motion to resolve estimation errors caused by these ambiguities and compute a reliable measure of metric scale. Monocular SLAM algorithms are also prone to scale drift, where the scale gradually changes while mapping. It is demonstrated that integrating DfD into monocular SLAM eliminates scale drift and results in accurate metric scale maps.

Acknowledgements

This work would not have been accomplished without the support and encouragement of many others around me. In the following lines, I would like to take this opportunity to show my appreciation to those people who have helped me in the realization of this thesis.

Especially, my deepest gratitude and appreciation goes to my primary supervisor, Professor Gamini Dissanayake for accepting me as a Master student and for providing the opportunity to work on this topic. He has been supervising me with his immense patience, motivation, enthusiasm, and expertise. Thanks for the countless hours of thought-provoking discussions.

I am deeply grateful to my alternate supervisor Doctor Ravindra Ranasinghe for his continuous support which was indispensable for the accomplishment of this thesis.

I would like to express my greatest appreciation to Professor Tomonari Furukawa for giving me the initial idea of performing this study and for his helpful advice on this research.

I would like to offer my special thank to the funding received through Canon Inc. and Canon Australia Pty. Ltd. to undertake my Master's degree. I would never have been able to reach where I am today, without their support.

My gratitude also goes to Mr. John Hazelton for proofreading my thesis and providing me with insightful comments and suggestions.

I would like to thank Mr. Kuranage Asok Aravinda Perera and Mr. Clyde Webster for reviewing my thesis and providing valuable advice. I have had the support and encouragement of them.

I am appreciative to all my colleagues at Centre for Autonomous Systems (CAS), I find it to have been an exciting opportunity for me to work with such an intelligent and motivated group of people.

Finally, and most importantly, I would like to thank my wife Hiromi and my two little children, Shogo and Mizuho. Their support, encouragement, patience, and love were undeniably the bedrock upon which the past two years of my life in Sydney have been built.

Contents

List of Figures

List of Tables

Acronyms & Abbreviations

1D	One-Dimensional
2D	Two-Dimensional
3D	Three-Dimensional
BRIEF	Binary Robust Independent Elementary Features
CAS	Centre for Autonomous Systems
CoC	Circle of Confusion
CPU	Central Processing Unit
DfD	Depth from Defocus
DfF	Depth from Focus
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
IMU	Inertial Measurement Unit
KLT	Kanade-Lucas-Tomasi
PSF	Point Spread Function
PTAM	Parallel Tracking and Mapping
ORB	Oriented FAST and Rotated BRIEF
RGB-D	Red, Green, Blue, and Depth

RMSE Root mean square error

ROI Region of Interest

SfM Structure from Motion

SLAM Simultaneous Localization and Mapping

UKF Unscented Kalman Filter

UTS University of Technology Sydney

Nomenclature

	General Notations
A	Aperture diameter of a lens
A_m	Amplitude of a step function
$\square^*$	Unconstrained $\square$ in the two-step projection method
B	Unknown offset
$\boldsymbol{b}$	Motion blur vector
b_δ	Distance from the image plane to a virtual plane where the rays from an out-of-focus point converge
b_f	Distance from the lens center to the image plane along the optical axis
c	Diameter of the circle of confusion
$\boldsymbol{c}[\cdot]$	Constraint function for equality state constraints
d	Distance from the lens center to a point along the optical axis
$D(\cdot)$	Depth-Defocus function
d_f	Distance from the lens center to a focal plane along the optical axis
Δt	Length of time between discrete steps
e_{thl}, e_{thh}	Threshold values to select features with strong edges
$\boldsymbol{\epsilon}_k = [\epsilon_{\Lambda_k}\ \epsilon_{\lambda_k^i}\ \epsilon_{\sigma_k^i}]^T$	Process noise vector of EKF
$\boldsymbol{\eta}_k = [\eta_k^i]^T$	Observation noise vector of EKF
$\boldsymbol{F}$	State transition matrix
f	Focal length of a lens
$G(\cdot,\cdot)$	Gaussian-shaped point spread function

γ	Camera specific constant to approximate c with σ
$\boldsymbol{H}$	Observation matrix
$\boldsymbol{H_c}$	Constrained matrix
$\hat{\square}$	Prediction value of $\square$
$I(\cdot,\cdot)$	Blurred image
$I_f(\cdot,\cdot)$	Sharp image
I_x	Gradient magnitude along x_e axis of $I(x_e, y_e)$
I_y	Gradient magnitude along y_e axis of $I(x_e, y_e)$
$\square^i$	i-th feature point
$\square^{i,j}$	i-th feature point seen from j-th keyframe
k	Time (discrete step)
Λ	Scale factor which defines the relationship between the metric map and the estimated geometry
λ	Texture correction factor for DfD
Λ_{ini}	Initial guess of Λ
λ_{ini}	Initial guess of λ
m	Number of keyframes
N	Number of observed points
N_c	F-number of a lens
ν	Innovation vector
$\boldsymbol{P}$	State covariance matrix
$\boldsymbol{p} = [x\ y\ z]^T$	Non-scaled feature location in camera coordinates
$\boldsymbol{p_w} = [x_w\ y_w\ z_w]^T$	Non-scaled feature location in world coordinates
$\phi_\square$	Calibration parameters for Depth-Defocus function
$\square^+$	First-step constrained $\square$ in the two-step projection method
$\boldsymbol{Q}$	Covariance matrix of the process noise
R	Gradient ratio of input image and reblurred image
$\boldsymbol{R}$	Covariance matrix of the observation noise
r	Index to evaluate the constancy of λ
$\boldsymbol{R_c}$	Covariance matrix of the noise or the extent of constraint violations

r_{thl}, r_{thh}	Threshold values to select features with constant λ
$\boldsymbol{S}$	Innovation covariance matrix
$\boldsymbol{S_c}$	Constrained innovation covariance matrix
σ	Standard deviation of Gaussian-shaped PSF
σ_b	σ for motion blur
σ_m	Measured σ for defocus blur
σ_{mb}	Composite σ of σ_m and σ_b
σ_r	Standard deviation of reblurred Gaussian-shaped PSF
smg	Index to evaluate edge strength
t	Time (continuous)
θ	Edge direction angle
$u(\cdot)$	Step function
$\boldsymbol{u} = [f_u \ f_v]^T$	Optical flow vector
v	Image velocity
$\boldsymbol{W}$	Filter gain vector
$\boldsymbol{W_c}$	Constrained filter gain vector
x_e	X-axis on an image where the edge is placed at $x_e = 0$
$\boldsymbol{x}_k = [\Lambda_k \ \lambda_k^i \ \sigma_{m,k}^i]^T$	State vector of EKF
y_e	Y-axis on an image where the edge is placed at $y_e = 0$
z	Non-scaled distance from the lens center to a point along the optical axis
$\boldsymbol{Z}_k = [z_k^i]^T$	Observation vector of EKF
z_{th}	Threshold value to select features nearby camera
$\boldsymbol{\zeta}_k = [\zeta_k^i]^T$	Noise vector for the extent of constraint violations

Operations

$\square * \square$	Convolution operator
$\lvert \cdot \rvert$	Absolute value
$\sqrt{\cdot}$	Square root
$\exp(\cdot)$	Exponential
$\square^2$	Square of $\square$

$\bigtriangledown$	Gradient operator
$\|\| \cdot \|\|$	Norm
$\dot{\square}$	First derivative of $\square$
$E[\cdot]$	Expectation
$\Sigma(\cdot)$	Summation
argmin$[\cdot]$	Argument of the minimum

State Transitions

$\square_{k-1\|k-1}$	Previous state
$\square_{k\|k-1}$	Predicted current state
$\square_{k\|k}$	Updated current state

Transforms

$\square^T$	Transpose
$\square^{-1}$	Inverse
$\boldsymbol{r}$	Rotation matrix of keyframe pose
$\boldsymbol{t}$	Translation vector of keyframe pose
$\boldsymbol{t}_t$	Ground truth of $\boldsymbol{t}$
$\boldsymbol{T} = [\boldsymbol{r}\|\boldsymbol{t}]$	Transformation matrix of keyframe pose

Chapter 1

Introduction

1.1 Background

There exists a growing demand for autonomous robots in a broad range of fields: mobile platforms, construction, cleaning, inspection, underwater, rescue and security [1]. In order to do tasks in an unknown environment unsupervised, an autonomous robot must perceive the environment, build a map, and localize itself in the map. This problem is called simultaneous localization and mapping (SLAM) [2]. To perform SLAM effectively, it is crucial that sensors are selected to work with the intended algorithms. One of the key sensors for an autonomous robot is the visual sensor, as it can provide rich information about the environment. Single camera-based systems, known as monocular SLAM [3], are attractive because of their size, cost, and versatility. However, a monocular camera cannot be used to estimate the scale of a scene directly since the projection of the scene onto a two-dimensional (2D) image plane causes the loss of the depth information. Ideally, the scale, which defines the relationship between the estimated geometry and the metric map, while unknown, should stay constant. However, due to the loss of the depth information, monocular SLAM algorithms are prone to scale drift, where the scale gradually changes while mapping [4]. For this reason, stereo camera systems based on similar parallax methods as human eyes have been typically used for vision-based SLAM systems. The disadvantage of the stereo camera is its cost and size due to using two cameras. Alternatively, the depth sensor such as an RGB-D camera is now available and popular. Although its size is

getting smaller recently and this makes it possible to mount it even on small products, it is still quite a special camera, and thus costly. In the highly competitive robot market, a monocular SLAM approach which can overcome the scale drift problem is demanded.

In the computer vision research area, several approaches have been proposed to estimate the metric scale from a monocular camera. One of the typical approaches is Depth from Defocus (DfD) [5] which is a technique to estimate the metric depth of a scene from image blur. The amount of defocus blur can be characterized by the diameter of the blur circle on the image. Although conventional defocus estimation methods require multiple images of the same scene with changes to camera settings such as aperture and focal length, it has recently been demonstrated that the amount of defocus blur can be estimated even from a single image [6], [7]. It therefore has the potential to be used for monocular cameras even with motion. The motivation of this research is that DfD has a possibility to resolve the scale drift problem in monocular SLAM. However, DfD has ambiguities caused by texture, motion blur, and focal plane. First, defocus from a single image is not straightforward to distinguish between blur caused by defocus and texture [6]. Second, motion blur caused by camera motion or target object motion effects the accuracy of defocus blur estimation. Third, an object in front and behind the focal plane may be viewed with the same amount of defocus blur [8]. The absolute depth estimation from defocus blur is still a challenging problem [9].

This thesis attempts to develop a DfD-based metric scale monocular SLAM system. The novelty of this research is that the proposed framework offers effective methodologies for solving the three ambiguities in DfD and providing accurate metric scale information of an environment with only a monocular camera. To the best of our knowledge, although methods for solving each problem exist, none of them resolve those ambiguities in one framework. This research reveals that metric scale estimated by DfD is effective for resolving the scale drift problem in monocular SLAM. The proposed approach using DfD does not require any additional sensors or camera modifications. It therefore retains all the advantages of using a monocular camera. Furthermore, the approach using DfD does not need any prior knowledge of the environment and thus has a potential to enhance the performance of monocular SLAM in a broad range of applications.

1.2 Aim, Objective, and Significance

The target stakeholder in this thesis is the robot manufacturing industry. The thesis aims to provide a small, inexpensive vision-based autonomous robot system. To achieve it, this thesis set the following two objectives:

1. to deliver a monocular scale estimation method
2. to deliver a scale drift-free monocular SLAM system

The significance of the first objective is to enable a monocular camera to replace the conventional 3D vision system such as a stereo camera and a depth sensor for autonomous mobile robots. If the monocular scale estimation has a suitable accuracy for some applications, the large and costly conventional 3D vision systems are no longer necessary for them. As a result, it helps to reduce the size and cost of those applications.

The significance of the second objective is to enable a monocular autonomous mobile robot to build a map of an unknown environment and estimate its egomotion precisely, which are essential abilities for autonomous robots. The monocular based SLAM system which can create an accurate metric scale map will facilitate downsizing and reduce the cost of autonomous robots.

1.3 Research Method

1.3.1 Objective 1: Monocular Scale Estimation

The methodology is based on fusing DfD with changes in images induced by 3D relative motion between the camera and the scene. To begin with, the ambiguity caused by texture in DfD is solved because it is the main cause of the scale estimation error. Srikakulapu et al. [10] pointed out that the scale estimation error due to texture in DfD can be corrected by using texture information such as contrast of edges on the texture, although their method cannot estimate the metric scale. Based on this idea, experiments are conducted

to measure the amounts of defocus blur at edges with different levels of contrast on some test charts. The results show that low-contrast edges mainly contribute to the defocus estimation error in DfD. Also, it is demonstrated that a depth-invariant correction factor makes it possible to correct the estimation error. Here, the motivation of this research is that the image motion will help to estimate this correction factor and result in accurate depth estimation. The proposed method uses a sequence of images taken by a moving monocular camera using a lens with a fixed focal length and a finite aperture. An algorithm to correct the scale estimation error caused by blur due to texture is designed in Extended Kalman Filter (EKF) framework which is a recursive optimal estimator for unknown states of a non-linear system. This method is implemented in MATLAB® and evaluated in an experiment to estimate the absolute metric depth of a set of points on a cluttered desk environment. The accuracy of the scale estimation is evaluated by measuring the root means square error (RMSE) of the distances between the camera and the target point locations. Also, the estimation results of the correction factors are compared with their truth.

1.3.2 Objective 2: Scale drift-free monocular SLAM system

In practical implementations of monocular SLAM, the generated map and camera poses accumulate scale drift [11]. Here, the motivation of this research is that DfD will be able to resolve the scale drift problem of the monocular SLAM. Monocular SLAM algorithms currently available use feature points [12], which are specific points in the image to represent the environment, and select a set of image frames called keyframes for representing the camera poses. The proposed method uses DfD to estimate the metric scale to the locations of feature points observed from a given keyframe gathered through monocular SLAM. The algorithm can be selectively applied to local regions of the map to correct the monocular SLAM output to minimize the impact of scale drift. However, to apply for autonomous mobile robots which move quickly on a large field, it is necessary to resolve the remaining two ambiguities caused by motion blur and focal plane in DfD. Dai and Wu [13] pointed out that motion blur is related to image motion. Alexander et al. [14] demonstrated that the focal flow sensor which uses defocus blur and differential motion of images does not have the focal plane ambiguity problem. Referring to these ideas, this thesis proposes an

algorithm to resolve all of three ambiguities in DfD. To evaluate the proposed method, DfD is integrated into ORB-SLAM [15], which is one of the best monocular SLAM systems currently available. This research conducts a real moving camera localization and mapping experiment in a corridor environment over a long distance causing the monocular SLAM algorithm to accumulate scale drift. The code is implemented in MATLAB®. The accuracy of the scale estimation is evaluated by measuring the RMSE of the distances between the camera and the map points reconstructed by the estimated scale. Also, it is shown that the proposed method is effective even on a small phone camera in an indoor office environment.

1.4 Contribution

The main contributions of this thesis are as follows:

- A correction factor for scale error caused by blur due to texture in DfD
- An EKF framework with DfD for producing an accurate metric scale map of a scene
- A method for eliminating the impact of motion blur in DfD
- A technique to avoid the focal plane ambiguity in DfD
- A non-linear optimization with DfD to post-process output of SLAM for eliminating scale drift

1.5 Publications Related to this Thesis

- **T. Shiozaki**, G. Dissanayake. Monocular 3D Metric Scale Reconstruction using Depth from Defocus and Image Velocity. In *Proc. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, 2017, pp. 6723-6728.

- **T. Shiozaki**, G. Dissanayake. Eliminating Scale Drift in Monocular SLAM using Depth from Defocus. *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 1, pp. 581-587, Jan. 2018.

1.6 Thesis Outline

This thesis is organized into five chapters and structured as below.

In Chapter 2, a literature review about monocular depth estimation, depth from defocus, and monocular SLAM is given. Also, ambiguity problems in DfD are introduced, and strategies for resolving the ambiguities are shown by referring to related works. Finally, the scale drift problem in monocular SLAM systems is introduced, and the novelty of the methodology proposed in this thesis is described.

In Chapter 3, a monocular 3D metric scale estimation method using DfD is presented. The DfD method is introduced, and the scale error caused by low contrast texture is formulated. Then, an EKF approach based on the relationship between changes in defocus blur and image motion induced by camera motion is proposed. The experimental results are shown to demonstrate the proposed method.

In Chapter 4, a novel approach to eliminate scale drift in monocular SLAM by using DfD is presented. A non-linear least squares optimization problem is formulated to integrate depth estimates from defocus to monocular SLAM. An algorithm to process the output generated by a monocular SLAM algorithm to correct for scale drift at selected local regions of the environment is proposed. The proposed algorithm is experimentally evaluated by processing the output of ORB-SLAM.

In Chapter 5, conclusions from work presented in this thesis are described together with the future work. One of the limitations of the DfD based approach is the short effective range depending on the lens, making the proposed method unsuitable for being used in a large scale environment. Some possible solutions combining additional scale estimation approach are presented for future work.

Chapter 2

Literature Review

This chapter contains a review of the literature in the three main areas of relevance to the work presented in this thesis: monocular depth estimation, depth from defocus, and monocular SLAM. In Section 2.1, monocular depth estimation methods are surveyed, and the reason why DfD is suitable for autonomous mobile robots is shown. In Section 2.2, DfD methods are reviewed, and the impact of ambiguities due to texture, motion blur, and focal plane in DfD are described. Finally, in Section 2.3, related works of monocular SLAM are reviewed from the aspect of the scale drift problem.

2.1 Monocular Depth Estimation

Observing a set of corresponding points of a scene from multiple frames captured at different camera positions makes it possible to recover the 3D locations of the points relative to the camera except for the absolute scale (see Fig. 2.1). This is the basis for estimating environmental structure from motion (SfM) [16]. In order to recover the absolute metric scale of a scene, this method needs information about the 3D relative motion between the camera and the scene by using additional sensors such as an inertial measurement unit (IMU) [17]. Depth from motion blur [18], [19] using image blur amount induced by camera motion to estimate the scene depth also cannot recover the absolute scale (see Fig. 2.2). Another method for monocular depth estimation is active stereo with structured

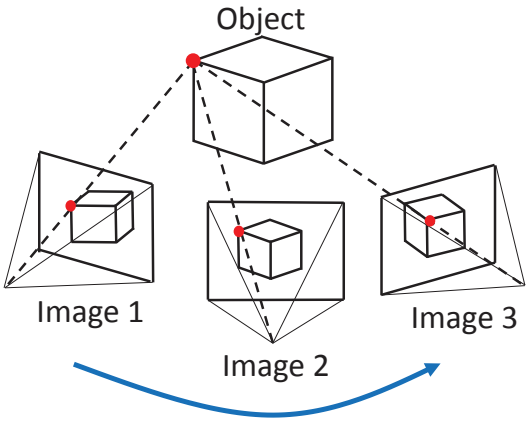


FIGURE 2.1: Illustration of Structure from Motion. The red point shows a corresponding point of a scene observed from different camera positions.

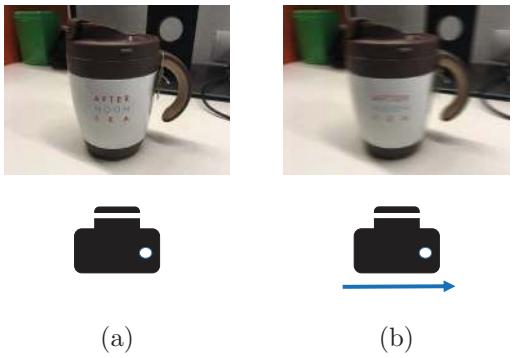


FIGURE 2.2: Illustration of Motion Blur. (a) is a focused image captured by a stationary camera and (b) is a blurred image captured by a moving camera. The size of the motion blur is one of the monocular depth cues.

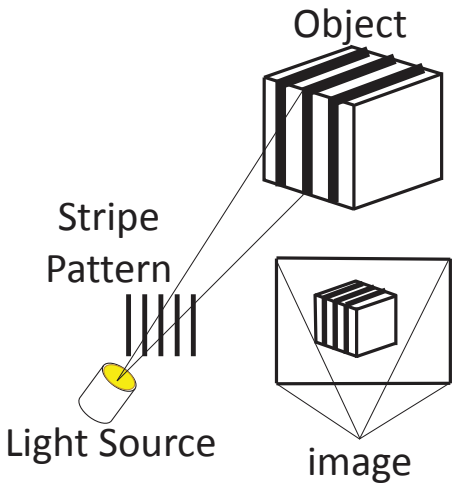


FIGURE 2.3: Illustration of Active Stereo. In this example, a stripe pattern is projected onto the surface of the object. The effective measuring range depends on the light source power.

light used in RGB-D cameras [20]. As shown in Fig. 2.3, the projection of a known structured pattern onto the surface of an object allows a monocular camera to calculate the depth information of the object with absolute scale. Either way, using additional sensors or devices compromise the main advantage of monocular cameras: their versatility. Also, these methods, especially using light sources, have the disadvantage of consuming greater power.

When used with a SfM or SLAM algorithm, the geometric constraints can be used to

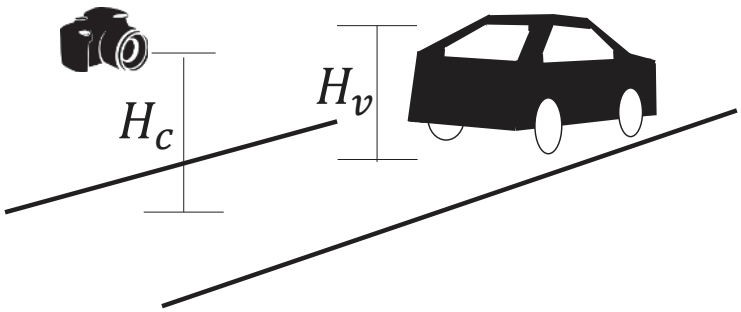


FIGURE 2.4: Illustration of geometric constraints. Under the assumption that the ground is flat, the known fixed camera height above the ground plane (H_c) allows calculation of the metric scale in a SfM or SLAM system. The size of an object such as a vehicle running in front of the camera (H_v) is also used to recover the metric scale.

estimate the scale. For example, a known fixed height of the camera above the ground plane is used to estimate scale in [21], [22], [23], [24]. In Fig. 2.4, the camera height H_c allows calculation of the metric scale in a SfM or SLAM algorithm under the assumption that the ground plane is flat. Alternatively, the size of objects such as cars, buildings, and pedestrians in the environment can be used as depth cues [25], [26]. Typically, these methods are based on supervised learning to identify the size of objects, and currently this is one of the popular research areas. The main drawback of these methods is that they are effective in only limited scenes: on roads or environments populated with known objects.

Use of image intensities is a different class of scene depth estimation methods. For example, shape from shading [27] can compute the surface shape of an object from gradual variations of shading in a single image. Photometric stereo [28], [29] using several images of an object captured from the same viewpoint but under different light directions is another technique (see Fig. 2.5). Illumination from different directions creates different shading onto the surface of the target object and enables the computation of the 3D shape of the object. However, both these methods require sufficient information about the illumination conditions and the surface reflectance. Therefore, these methods are not suitable for autonomous robots which do not have any prior information about environments.

The approaches using behavior of image blur caused by an optical system of the camera are depth from focus (DfF) and depth from defocus (DfD). DfF searches the focal plane position where a point on the object is focused and estimates the distance to the point [30]. This method requires many images of the same scene with different focus settings to

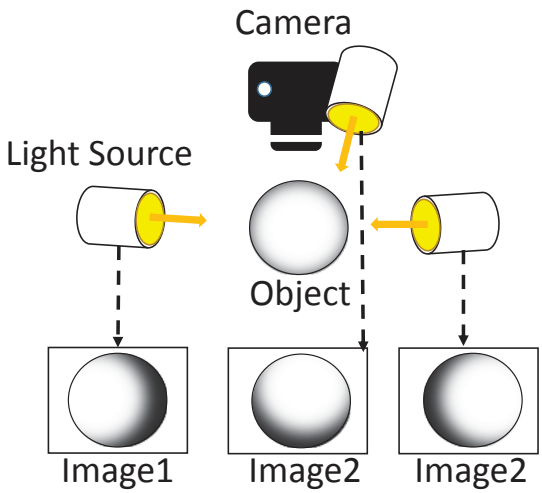


FIGURE 2.5: Illustration of photometric stereo. Illumination from different directions makes different shading onto the surface of an object. The changes in the intensities on the images makes it possible to compute the 3D shape of the object.

obtain the maximum sharpness image at the target point, thus is not suitable for mobile robots with dynamics. On the other hand, DfD relies on the amount of defocus blur which depends on the distance between the object and the focal plane (see Fig. 2.6) [31]. It has been demonstrated that the defocus blur can be estimated even from a single image [6], [7]. Therefore, DfD has the potential to be used for autonomous robots with dynamics. Also, DfD does not require any additional sensors or prior knowledge about the scene. Use of DfD to recover the metric scale can retain all the advantages of using a monocular camera for autonomous mobile robots.

In the next section, a review of DfD methods is presented.

2.2 Depth from Defocus

The literature on DfD is divided into two categories: defocus estimation and depth estimation. Defocus estimation has been widely researched due to its usefulness for many applications in addition to depth estimation: image quality assessment, image deblurring, and refocusing [6]. Conventional defocus estimation methods require multiple images with changes to camera settings such as aperture and focal length to obtain different amounts

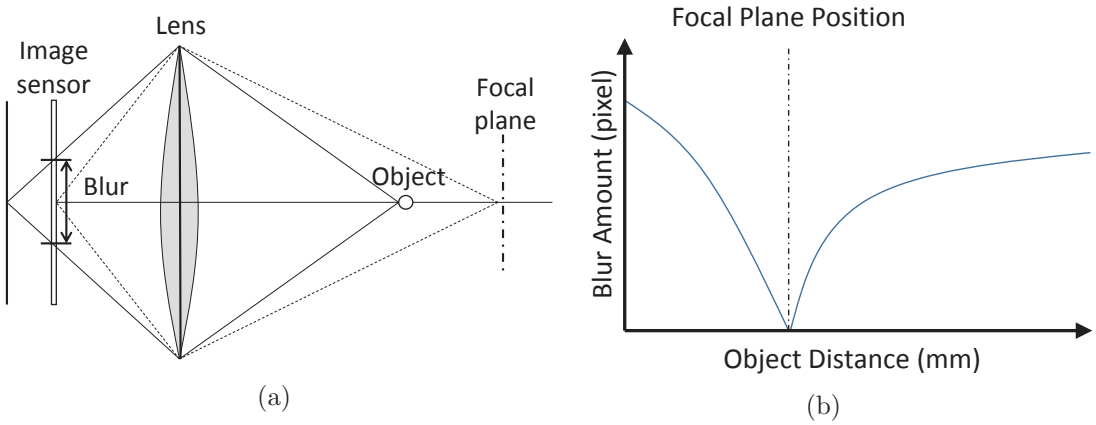
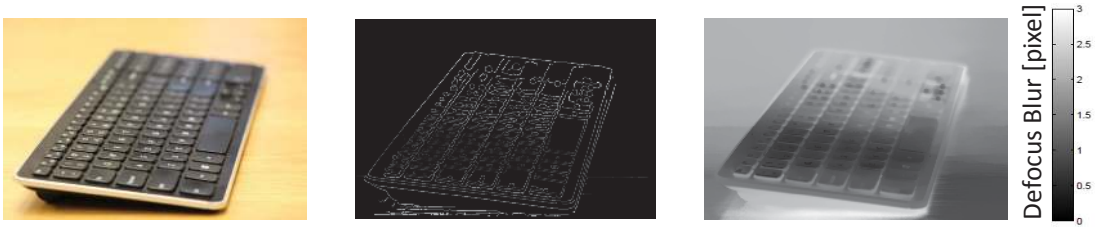


FIGURE 2.6: Illustration of Depth from Defocus. (a) shows the image formation of the thin lens model and (b) shows Depth-Defocus curve. The defocus blur amount depends on the distance between the object and the focal plane as shown in (b).

of defocus blur on the same scene [32], [33]. Taking images of the same scene with different camera settings is complex and requires matching between corresponding points in the different images [34] and therefore is not particularly attractive in many applications. Pentland [35] pointed out that defocus blur can be extracted at edge locations even on a single image. Elder [36] used the first-order and second-order derivatives of the input image to find the edge locations and estimate amounts of defocus blur on them. Zhuo and Sim [6] proposed a method based on the Gaussian gradient ratio of input and reblurred images. It has been known that image blur due to defocus can be modeled by Gaussian-shaped point spread function (PSF) [35]. As shown in Fig. 2.7, convolving an image with Gaussian PSF yields a blurred image. The size of PSF can approximate the blur amount. However, the previous methods [35], [36] available to estimate the blur amount based on Gaussian PSF were prone to image noise. Using the Gaussian gradient ratio makes the method in [6] more robust to image noise than those available in the literature. Another advantage of this method is that it can generate a sparse defocus map at edge locations of the image efficiently because it does not need the image deconvolution procedure used in conventional methods [5], [33]. Furthermore, it can propagate the sparse defocus blur map to the entire image including non-edge pixels. This is illustrated in Fig. 2.8 where (a) is the input image, (b) is the sparse defocus map at detected edge locations, and (c) is the full defocus map generated from (b). However, the process to create the full defocus map takes time due to the propagation process [37]. Here, for applications to estimate the metric scale of a scene, it is not required to obtain the pixel-level defocus information. A

FIGURE 2.7: Illustration of image convolution. $*$ means the convolution operator.

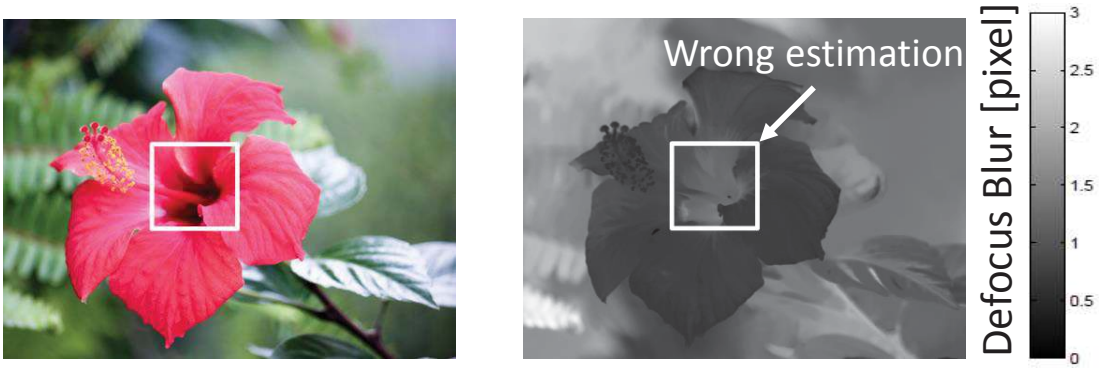


(a) (b) (c)

FIGURE 2.8: Demonstration of the defocus map estimation method proposed by [6]. (a) is the input image, (b) is the sparse defocus map at edge locations of (a), and (c) is the full defocus map generated by propagating the defocus blur amount at edge locations of (b) to the entire image. In (b) and (c), the grayscale indicates the amount of defocus blur.

small number of points of depth information will be adequate to recover the metric scale of a scene. Due to the computational efficiency and the robustness to image noise, the sparse defocus estimation method proposed in [6] is utilized in this research.

Pentland [35] is one of the pioneers in the field of depth estimation from defocus blur. He used the geometry of the thin lens model and showed how to calculate the depth to an image point. Subbarao and Surya [5] proposed a spatial-domain convolution/deconvolution transform method under the assumption that two images taken with different camera settings such as focal length and aperture size were obtained. However, the convolution and deconvolution transformation procedure is computationally costly. Wöhler et al. [31] proposed the Depth-Defocus function which expresses the relationship between the depth to a point and its defocus blur amount. Also, they proposed a calibration method for obtaining accurate depth estimates. Their Depth-Defocus function does not require multiple images of the same scene taken with different camera settings. In addition, it is computationally efficient due to the pre-process calibration. Therefore, the method proposed in [31] is used in this research.



(a) (b)

FIGURE 2.9: Illustration of the blur texture ambiguity, adapted from [6]. (a) is the input image and (b) is the full defocus map. In the white boxed region, a wrong defocus estimation occurred due to the texture of the flower.

However, to be effective, single image DfD methods require strategies to resolve ambiguities due to texture, motion blur, and focal plane. The literature for each category is surveyed in the following.

2.2.1 Blur Texture Ambiguity

Given a single image, DfD methods cannot differentiate whether a blur is caused by defocus or texture [6]. This is illustrated in Fig. 2.9 where (a) is the input image and (b) is the full defocus map generated by the method proposed in [6]. In the white boxed region on the image, the wrong defocus estimation occurred due to the blur caused by the texture of the flower. This problem is called the blur texture ambiguity problem. To eliminate the blur texture ambiguity, DfD methods typically require multiple images of the same scene [38], [39] or structured lighting onto the object surface [40]. Zhou et al. [41] used the coded aperture pairs of the lens. Although this method enables more robust depth estimation even on weak texture, it needs a camera modification. Use of multiple images of the same scene, additional devices, or a modified camera negates the advantages of a monocular camera. Wöhler et al. [31] searched the focused scene for points of interest observed in a sequence of images to solve the blur texture ambiguity. A point of interest in an image is called a feature point. If an image where a feature point is focused on can be obtained, the blur effect caused by texture on it can be calculated and eliminated from the measurements

(a) (b)

FIGURE 2.10: Illustration of the difference between defocus blur (a) and motion blur (b). Although the defocus blur is a non-directional blur, the motion blur is in the same direction as the camera or the object motion.

in the other images where the feature point is out-of-focused. However, in practice, it is difficult for a moving robot to observe a set of feature points for a long time, and thus it is not feasible to search the focused scene for each feature point in robotic applications. Srikakulapu et al. [10] proposed a method to correct the depth map by using texture information such as edge sharpness, spot energy, and contrast, although their method could not estimate the metric scale of the scene. Referring to the idea of [10], this research focuses on evaluating the amounts of defocus blur on edges on textures with different contrast and then reveals that one of the main causes of the blur texture ambiguity is the difference of the contrast at edge locations. Also, this research demonstrates that the defocus estimation error caused by low-contrast edges can be corrected using a sequence of images taken by a moving camera. The proposed method does not require focused scenes used in [31] and is able to estimate metric scale even when the feature points concerned are never in focus.

2.2.2 Impact of Motion Blur

In dynamic scenes, both depth and motion contribute to the image blur. Fig. 2.10 shows the images with defocus blur (a) and motion blur (b). The difference between defocus and motion blur is that the motion blur is in the same direction as the camera or the object motion, while the defocus blur is a non-directional blur. Bascle et al. [42] proposed a method to estimate the amount of blur caused by motion and defocus from a sequence of images.

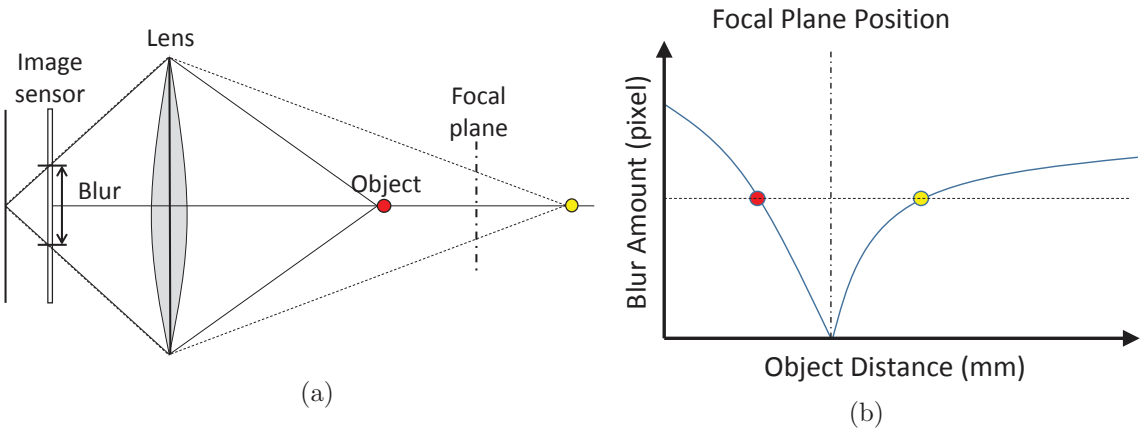


FIGURE 2.11: Illustration of focal plane ambiguity. The objects across the focal plane shown as red and yellow dots in (a) have the same amount of defocus blur as shown in (b) where the dot colors correspond to the colors of objects in (a).

They demonstrated that the estimated object motion could identify the amount of motion blur in a composite image blur of defocus and motion. Paramanand and Rajagopalan [43] proposed a method to recover the 3D structure from both motion blur and defocus blur with camera motion in an Unscented Kalman Filter (UKF) framework. Dai and Wu [13] pointed out that the motion blur constraints they proposed have mathematical similarity with the optical flow constraints [44]. Optical flow is a motion field in the image caused by relative motion between the camera and the observed scene. Referring to these ideas, a method to estimate the amount of motion blur by using optical flow induced by camera motion is proposed in this research.

2.2.3 Focal Plane Ambiguity

Objects placed in front and behind the focal plane may be viewed with exactly the same amount of defocus blur as shown in Fig. 2.11 [8]. In this figure, the points across the focal plane shown as red and yellow dots have the same amount of defocus blur. Most DfD methods assume that the focus point is put on the nearest or farthest point in the scene to locate the observed objects on one side of the focal plane [6]. However, this assumption is not suitable for robotic applications needing a wide-range observation. Sellent [8] used the coded aperture masks for the lens to make an asymmetric aperture lens which creates unique blurs for all depth from the camera. As mentioned before, such a camera modification compromises the versatility of the monocular camera. Kumar et al. [45]

demonstrated that chromatic aberration provides an effective indicator to solve the focal plane ambiguity. In many conventional cameras, however, chromatic aberration is fixed in the optical system and the image processor, and therefore is not always available. Wöhler et al. [31] pointed out that the focal plane ambiguity can be solved by using the direction of camera motion. Alexander et al. [14] demonstrated that their developed focal flow sensor using defocus blur and differential motion of images did not produce focal plane ambiguity. Referring to the ideas of [31] and [14], this research proposes an algorithm to resolve the focal plane ambiguity by using motion information generated by monocular SLAM.

In the next section, a review of monocular SLAM is presented.

2.3 Monocular SLAM

When a robot operates in an unknown environment unsupervised, the robot must make a map of the environment with respect to information gathered from sensors mounted on it and keep track of its own location based on the built map. SLAM [2] is the algorithm to achieve this procedure with various sensors such as cameras, laser range finders, GPS, and IMU. The more information available, the easier the SLAM problem. Therefore, monocular SLAM which uses only a single camera without the ability of depth perception is a challenging problem. However, monocular SLAM has many benefits to be worth the challenge in terms of the cost, size, weight, and power consumption. In addition, cameras are used in both indoor and outdoor environments, in contrast to GPS which is typically unavailable in an indoor environment.

In this section, the literature about monocular SLAM, especially focusing on the scale drift problem, is reviewed. Then, the state-of-the-art monocular SLAM systems are surveyed. Finally, ORB-SLAM [15], which is one of the best monocular SLAM algorithms currently available, is introduced.

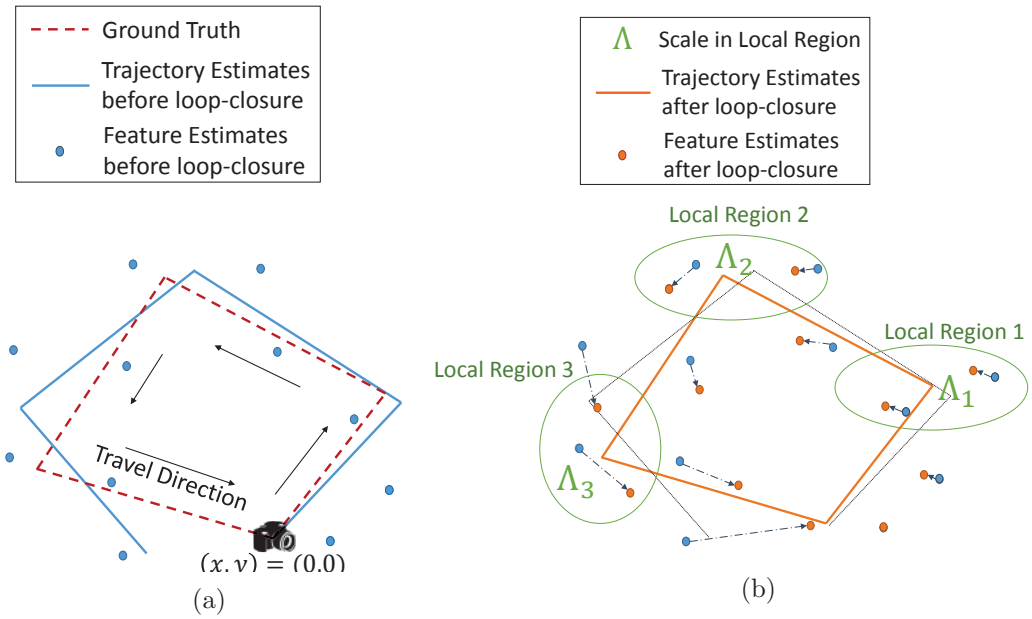
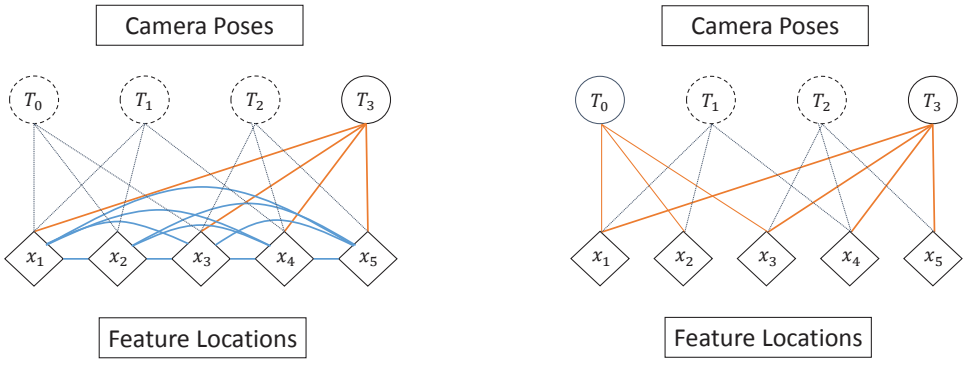


FIGURE 2.12: Illustration of scale drift. (a) shows the feature location and camera trajectory estimates before loop-closure. The blue solid-line is the trajectory estimate, the brown dot-line is the ground truth, and the blue dots are feature location estimates. Scale drift causes the mapping error. (b) shows the feature location and camera trajectory estimates after loop-closure. The orange solid-line is the trajectory estimate, the orange dots are feature location estimates. Although the loop-closure reduces the effect of scale drift, the scale error in different local regions still remains in the map, which means the scale factors Λ_1, Λ_2, and Λ_3, which are ideally the same value, become different values.

2.3.1 Scale drift in monocular SLAM

The first successful work of monocular SLAM was by Davison et al. [46], [3]. They used an EKF based SLAM technique [47], [48]. EKF is a recursive optimal estimator for unknown states of a non-linear system. EKF requires the assumption that the uncertainty associated with feature points has a Gaussian distribution. However, when the feature points exist far from the camera, the parallax induced by camera motion becomes small, and it makes the uncertainty rise sharply. In that case, the Gaussian distribution cannot approximate the uncertainty distribution in depth. Montiel et al. [49], [50] used the inverse depth parametrization to solve this problem. When using this parametrization, the Gaussian distribution can model the uncertainty in all depth range, from nearby camera to infinity. As a result, monocular SLAM could be used in large-scale environments.

When used in large-scale environments, however, monocular SLAM encounters the scale drift problem. This is illustrated in Fig. 2.12 (a) where the camera travels in the direction



(a) (b)

FIGURE 2.13: Illustration of the difference between filter-based and optimization-based SLAM systems, adapted from [52]. (a) and (b) show the filter-based and the optimization-based systems, respectively. The orange lines show the data connection between camera poses and feature locations used for the estimation. The blue lines show the tight data connection between feature location estimates. The camera poses shown with dashed-line are not used for the estimation. Note that the features still have correlations with each other as the result of the marginalization of the intermittent keyframes, although not shown in (b).

of the arrows and returns to the origin. The blue dots show the feature locations which mean the locations of the feature points in the map. Due to gradual changes in scale during the exploration, the feature location and camera pose estimates accumulate the mapping error. This is known as scale drift. The loop-closure [51], which detects the previously-visited area and updates the map to close the loop, corrects scale drift around the loop to some extent as shown in Fig. 2.12 (b). However, even with the loop-closure, the effect of scale drift still remains in the map. As a result, the scale factors in different local regions of the map, which are ideally the same value, become different. This research addresses this issue and corrects the scale error with the aid of DfD.

Next, the literature of the state-of-the-art monocular SLAM is reviewed.

2.3.2 Keyframe-based Optimization

Monocular SLAM solutions are divided into two categories: filter-based or optimization-based [52]. Fig. 2.13 shows the data links between the camera poses T_i and the feature location estimates x_i in the filter-based (a) and optimization-based (b) SLAM systems. In filter-based systems, the current pose T_i is updated in an optimal filter such as EKF

and the past poses are marginalized out after every frame. As a result, the pose-graph, which is a graphical representation of the camera poses [53], stays compact. The main drawback of this method is that feature location estimates which link with the past poses are tightly-connected to each other, and this makes the computational cost of propagating steps grow larger whenever new feature points are observed. Therefore, the number of feature points in the map is strictly limited in filter-based methods. On the other hand, in optimization-based systems, a set of selected camera poses, which are known as keyframes, is used for optimization. The other poses and all the feature locations connected to them are not used for optimization. Due to the discarding such information, the elements of the pose graph are sparsely-connected. This efficient optimization method enables SLAM to deal with a large number of feature points in the map.

Although early works in SLAM were mostly filter-type systems, Parallel Tracking and Mapping (PTAM) developed by Klein and Murray [54] changed the trend. PTAM separated the tracking task and the mapping task into two threads for running parallel in real-time. Their approach based on keyframes enabled monocular SLAM to adopt optimization using SfM algorithm with bundle adjustment [55] which can provide a much more accurate sparse map and camera poses than previous filter-based SLAM algorithms. However, PTAM was limited to small-scale operation due to the lack of the loop-closure. Strasdat et al. [4] presented a scale drift-aware loop-closure using a keyframe-based optimization with the pose-graph theory. Mur-Artal et al. [15] proposed ORB-SLAM that achieves real-time tracking, mapping, and loop-closing in large environments based on the pose-graph optimization algorithm.

Next, the detail of ORB-SLAM is introduced.

2.3.3 ORB SLAM

Many monocular SLAM algorithms currently available generate a map of point features present in the environment. The feature points observed among different camera poses are detected and corresponded by a feature detector and descriptor. ORB-SLAM [15] uses ORB [12], which is based on the FAST corner detector [56] and the BRIEF descriptor [57]. ORB incorporates oriented FAST and rotated BRIEF. Although FAST is a very

efficient corner point detector and thus is used in real-time SLAM such as PTAM [54], it cannot describe the orientation of the corner point. Although BRIEF is a very fast binary descriptor which can describe an image patch as a bit string, it lacks the rotational invariance. ORB addressed both of two issues and resolved them. Use of ORB allows ORB-SLAM to detect and match the feature points among keyframes efficiently and reliably. Also, ORB-SLAM is built on the successful previous keyframe-based optimization works of PTAM by [54] and the scale drift-aware loop-closure by [4]. The map points and keyframe poses calculated by a triangulation method based on the corresponding feature locations are optimized among the several keyframes observing a set of same feature points in real-time. When a loop is detected, the map point and keyframe pose estimates are globally optimized around the loop to close it. Using the state-of-the-art techniques for monocular SLAM makes ORB-SLAM [15] the best monocular SLAM system currently available.

However, the potential for scale drift in ORB-SLAM has been recognized. In their later work, it has been shown that using a stereo or an RGB-D camera, ORB-SLAM2 [11] provides a good solution to this problem. In [58], a Visual-Inertial ORB-SLAM that uses information from IMU to recover metric scale was presented. The focus of this research is an alternative strategy based on DfD to eliminate scale drift without additional sensors or devices.

2.4 Conclusion

In this chapter, the related literature on monocular depth estimation, depth from defocus, and monocular SLAM were reviewed. Among many methods to estimate depth from a monocular camera, the review results suggested that a single image DfD method that does not require any additional sensors, geometrical constraints, or known structure in the environment is best suited for autonomous robot applications. However, the review about DfD showed that a single image DfD has ambiguities caused by texture, motion blur, and focal plane. To the best of our knowledge, there are no previous works to resolve those ambiguities in one framework. In the review, brief strategies of the proposed methodology for resolving the ambiguities were outlined with references to some related works.

In the review on monocular SLAM, it was clear that the scale drift is still a crucial problem in the state-of-the-art monocular SLAM systems. The novelty of the methodology using DfD for monocular SLAM is to be able to estimate the metric scale of a scene without any additional devices or a prior knowledge about the scene. Therefore, it has a potential to enhance the performance of monocular SLAM in a broad range of applications.

In the next chapter, a methodology for estimating the metric scale of a scene based on DfD is presented. The blur texture ambiguity which is the main cause of the estimation error in DfD is resolved. Then, the methodology proposed in Chapter 3 is leveraged to integrate DfD into SLAM system in Chapter 4.

Chapter 3

Monocular Metric Scale Estimation

This chapter presents a methodology for 3D metric scale estimation. In Section 3.1, the DfD approach based on a formula derived using the thin lens model is introduced. The point spread function is approximated with a Gaussian filter to model the amount of defocus blur in a given image at edge locations. In Section 3.2, the impact caused by texture in DfD is formulated. In Section 3.3, using the equation which expresses the impact of texture in DfD, an optimal recursive filter based approach using EKF for scale estimation is proposed. Using a sequence of images from a monocular camera with a fixed focus lens, metric scale in a scene is estimated. In Section 3.4, the proposed algorithm is experimentally evaluated. [1]

3.1 Depth from Defocus

Fig. 3.1 shows the thin lens model [35]. All rays from a point located at the in-focus distance d_f converge to a single point on the image plane placed at the distance b_f from the lens. On the other hand, rays from an object located at any other distance d converge

[1]The work presented in this chapter was reported in "Monocular 3D metric scale reconstruction using depth from defocus and image velocity," in Proc. of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, 2017, pp. 6723-6728.

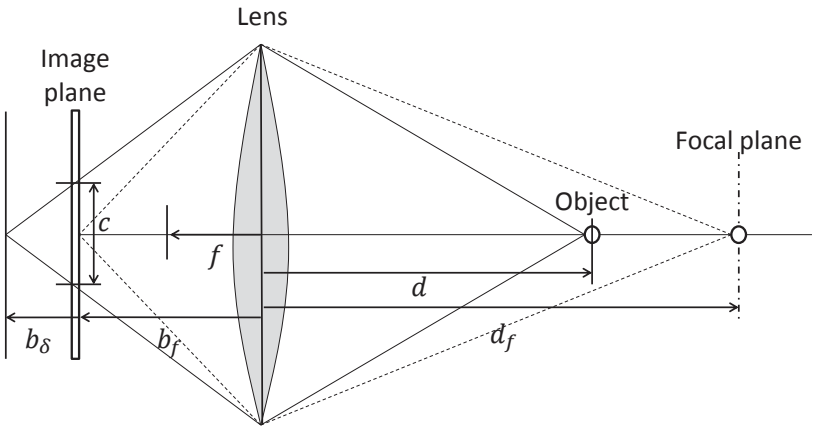


FIGURE 3.1: Thin lens model. Origin is the lens center. b_f is the distance to the image plane. d_f is the distance to the focal plane. The size of c depends on the object distance d. When the image plane is placed at $b_f + b_\delta$, the object is best focused.

to a point on a plane located at a distance $b_f + b_\delta$ from the lens and therefore will be out-of-focus when viewed at the image plane. The rays from such an object will make a blurred circle on the image plane. This is known as the circle of confusion (CoC). The diameter of this circle is given by

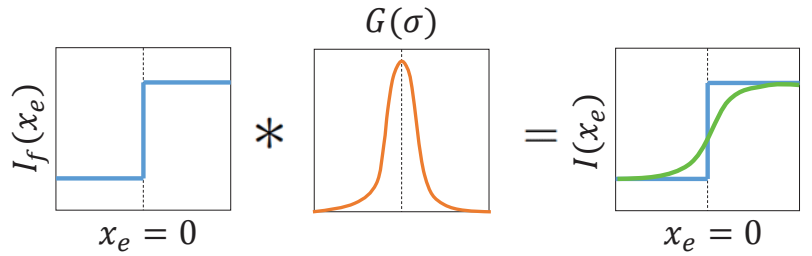
$$c = \frac{|d - d_f|}{d} \frac{f^2}{N_c \cdot (d_f - f)}, \tag{3.1}$$

where f is focal length and N_c is f-number of the camera [6]. It is seen that the larger $|d - d_f|$, the larger the CoC. To get a large amount of defocus blur, a long focal length and a large aperture are required as the f-number is $N_c = f/A$ where A is the aperture diameter of the lens.

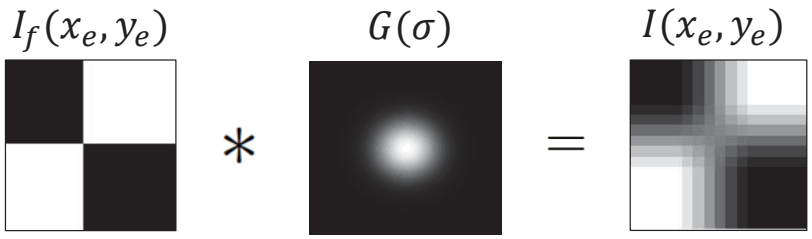
The amount of defocus blur can be estimated on edge locations on the image [35]. To simplify, consider one direction first. An ideal step edge can be modeled as

$$I_f(x_e) = A_m \cdot u(x_e) + B, \tag{3.2}$$

where A_m is an unknown amplitude, $u(x_e)$ is the step function, and B is an unknown offset. The edge is placed at $x_e = 0$. The blurred image can be approximated by a convolution of the sharp image $I_f(x_e)$ and a Gaussian-shaped point spread function (PSF) $G(x_e, \sigma)$ as



(a)



(b)

FIGURE 3.2: The illustration of image convolution. (a) shows the 1D case and (b) shows the 2D case. In (a), the blue line is a sharp edge, the orange line is the Gaussian PSF, and the green line is the blurred edge due to the image convolution.

$$\begin{aligned} I(x_e) &= G(x_e, \sigma) * I_f(x_e), \\ G(x_e, \sigma) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big(-\frac{x_e^2}{2\sigma^2}\Big), \end{aligned} \tag{3.3}$$

where $*$ means convolution and $I(x_e)$ is the blurred image [36]. This is illustrated in Fig. 3.2. Fig. 3.2(a) shows the one-dimensional (1D) case and Fig. 3.2(b) shows the 2D case. The convolution of the Gaussian-shaped PSF makes the edge blur. The larger σ results in a more blurry image. The radius of σ of the Gaussian PSF $G(x_e, \sigma)$ can model the size of c as

$$c = \gamma \cdot \sigma, \tag{3.4}$$

where γ is a camera-specific constant [59].

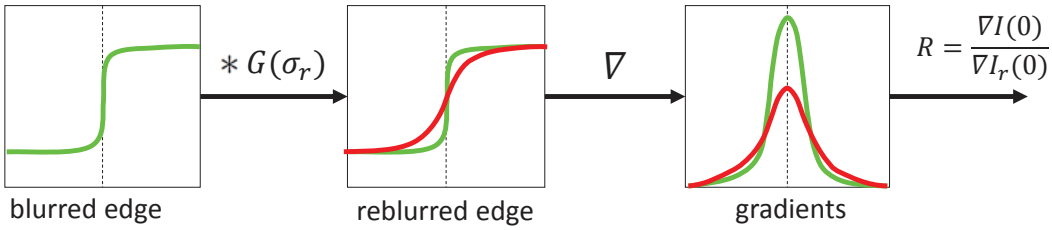


FIGURE 3.3: The overview of the blur estimation method proposed by [6]. The green lines show the blurred edges due to the defocus. The red lines show the reblurred edges by a known Gaussian PSF. The black dash lines show the edge locations. The ratio of the gradient magnitude between the blurred edge and the reblurred edge becomes maximum at the edge location and it is used to calculate the value of σ.

In [6], it is shown that the value of σ can be estimated using the gradient ratio of the input image and the reblurred image with a known Gaussian PSF. The overview of this method is shown in Fig. 3.3. In this figure, the green lines show the blurred edges due to the defocus. The red lines show the reblurred edges by a known Gaussian PSF. The gradient of the reblurred edge is written as

$$\begin{aligned} \bigtriangledown I_r(x_e) &= \bigtriangledown(G(x_e,\sigma) * I_f(x_e)) = \bigtriangledown(G(x_e,\sigma) * G(x_e,\sigma_r) * (A_m \cdot u(x_e) + B)) \\ &= \frac{A_m}{\sqrt{2\pi(\sigma^2+\sigma_r^2)}} \exp\Big(-\frac{x_e^2}{2(\sigma^2+\sigma_r^2)}\Big), \end{aligned} \tag{3.5}$$

where $G(x_e, \sigma_r)$ is the reblur Gaussian PSF. The gradient ratio of input image and reblurred image is

$$\frac{\bigtriangledown I(x_e)}{\bigtriangledown I_r(x_e)} = \sqrt{\frac{\sigma^2+\sigma_r^2}{\sigma^2}} \exp\Big(-\Big(\frac{x_e^2}{2\sigma^2} - \frac{x_e^2}{2(\sigma^2+\sigma_r^2)}\Big)\Big). \tag{3.6}$$

At the edge location ($x_e = 0$), the value of the ratio becomes maximum as

$$R = \sqrt{\frac{\sigma^2+\sigma_r^2}{\sigma^2}}. \tag{3.7}$$

Solving Eq. (3.7) gives the unknown blur amount σ as

$$\sigma = \frac{1}{\sqrt{R^2 - 1}} \cdot \sigma_r. \tag{3.8}$$

For two-dimensional images, the gradient magnitude can be calculated as

$$|| \bigtriangledown I(x_e, y_e)|| = \sqrt{\bigtriangledown I_x^2 + \bigtriangledown I_y^2}, \tag{3.9}$$

where $\bigtriangledown I_x$ is the gradient along x_e axis and $\bigtriangledown I_y$ is that along y_e axis. Furthermore, propagating the sparse defocus blur map at edge locations to the entire image can make the full defocus map. However, generating the full defocus map is computationally costly [37]. As the goal of this research is to integrate the DfD method into SLAM system, the blur amounts of only the feature points detected are used for the defocus estimation to reduce the calculation time. Feature points are specific points of interest in an image and detected using a feature detector such as ORB [12]. In this research, Canny edge detector [60] is utilized to detect edge locations in an image similar to [6] but only within the region of interest (ROI) around the feature points, and the nearest edges from the feature points are searched and used to estimate the defocus amounts. This strategy can be applied together with any feature detection algorithm.

In [31], it is shown that the metric distance can be estimated using the Depth-Defocus function which expresses the relationship between σ of Gaussian PSF and the metric distance as

$$\begin{aligned} \sigma = D(d) &= \frac{1}{\phi_1} \exp(-\frac{1}{\phi_2}(b_\delta(d))^2) + \phi_3, \\ b_\delta(d) &= \frac{d \cdot f}{d - f} - b_f, \end{aligned} \tag{3.10}$$

where ϕ_1, ϕ_2, and ϕ_3 are the calibration parameters. For a given camera setup, these parameters together with b_f and f can be estimated using a calibration process. This is illustrated in Fig. 3.4. In this calibration process, the value of σ and the metric distance are measured from the binary edge pattern and a known size of the black-and-white checkerboard shown in Fig. 3.4(a), respectively. In Fig. 3.4(b), the blue + shows

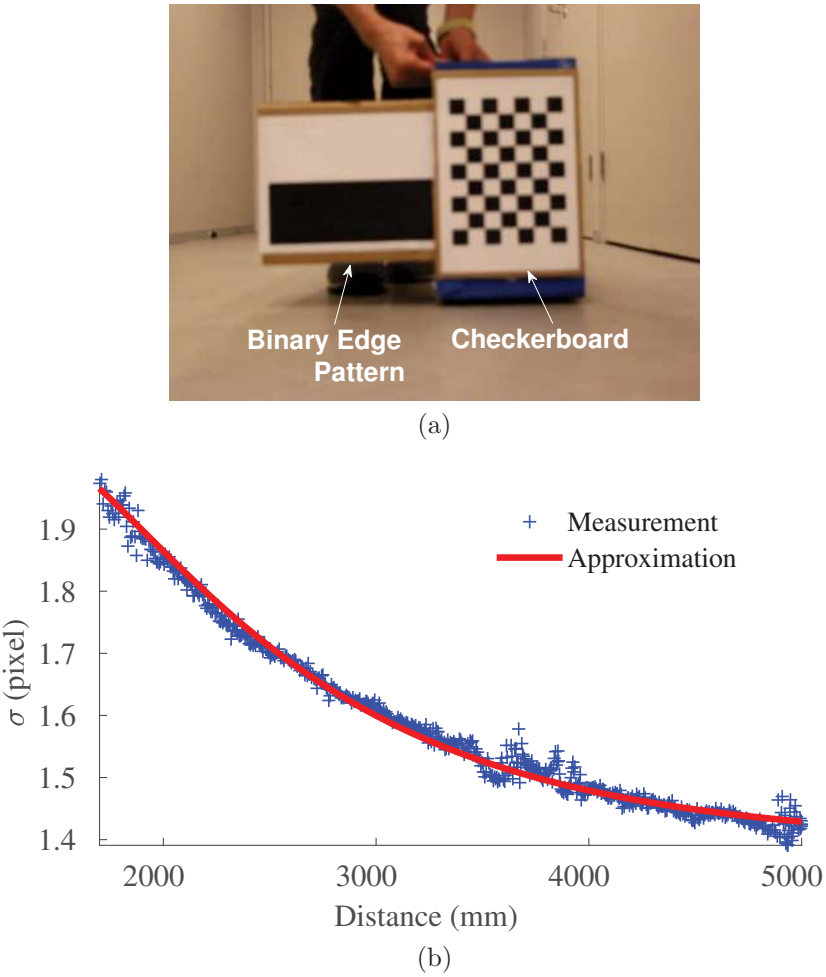


FIGURE 3.4: Calibration chart (a) and Depth-Defocus Curve (b). The σ is measured at the binary edge pattern, and the depth is measured from the known size of the checkerboard shown in (a).

the measurement σ, and the red line is the approximation by Eq. (3.10) as a result of a non-linear least squares fitting to all measurements. As expected, Eq. (3.10) could approximate the relationship between σ and the metric distance d. Solving Eq. (3.10) yields the metric distance d from the measurement σ.

However, measuring σ does not work well on weak edges due to errors caused by the blur texture ambiguity [31]. In the next section, the cause of blur texture ambiguity is analyzed, and a correction factor for the extent of blur due to texture in DfD is proposed.

3.2 Blur Texture Ambiguity

The blur texture ambiguity is due to many factors such as soft shadows, brightness and color of the object, and the illumination. The several experiments conducted in this research revealed that one of the main causes is the difference of the contrast among the ROIs. Also, it was observed that the impact of this error could be expressed empirically by the following equation:

$$\sigma_m = \lambda \cdot \sigma = \lambda \cdot D(d), \tag{3.11}$$

where σ_m is the measured blur amount, and λ describes the correction factor for the extent of blur due to texture. This is illustrated in Fig. 3.5. Fig. 3.5(a) and (b) are low contrast and high contrast edge patterns, respectively. In Fig. 3.5(c), the green $\times$ shows σ_m measured at the low contrast edge, the blue $+$ shows σ measured at the high contrast edge, the red line is the approximation of σ based on Eq. (3.10), and the black line is the approximation of σ_m based on Eq. (3.11).

As shown in Fig. 3.5(c), Eq. (3.11) could approximate the values of σ_m well. This means that λ can express the extent of blur due to texture. The contrast of texture is independent of distance d, and thus a depth-invariant gain factor λ can correct the estimation error due to texture in DfD. It can also be seen that the same is true in a more complex scene. Fig. 3.5(f) shows the results from the images of a face (Fig. 3.5(d)) and the checkerboard (Fig. 3.5(e)). In this case, the green $\times$ shows σ_m measured on the face, and the blue $+$ shows σ measured on the checkerboard. The feature points on the face were detected by Kanade-Lucas-Tomasi (KLT) algorithm [61], [62] which is a feature detection and tracking algorithm. The median of σ measured on the feature points at each distance was used as σ_m in this experiment. As expected, Eq. (3.11) could approximate the values of σ_m well. Table 3.1 shows all parameters used in the experiments. The illumination and camera parameters such as the shutter speed, the aperture size, and the sensitivity were unchanged during the experiments. In this research, λ is termed the texture correction factor.

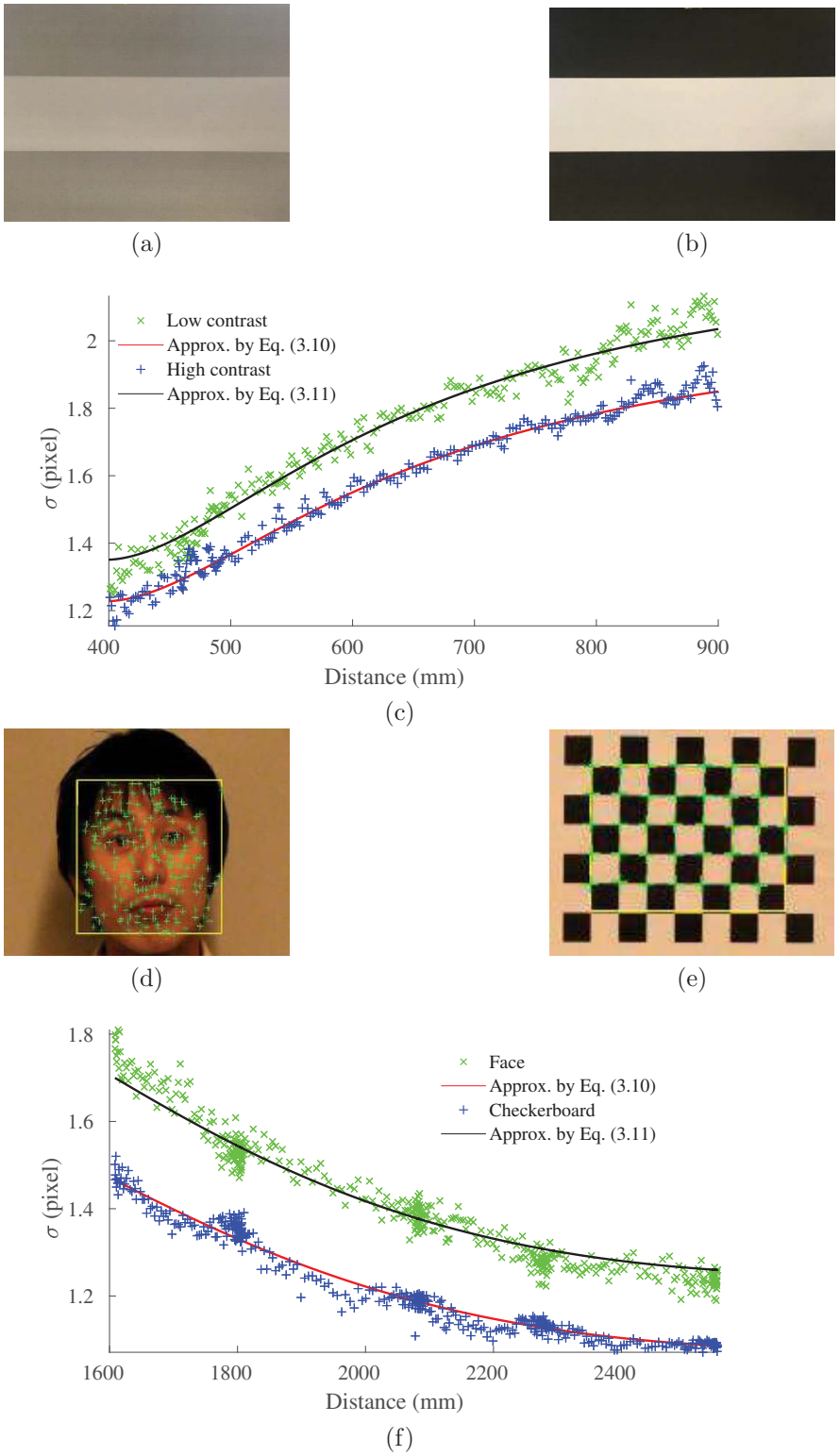


FIGURE 3.5: Demonstration of Eq. (3.11). (a) is a low contrast edge pattern with 50% and 75% gray levels. (b) is a high contrast binary edge pattern. (d) is a face and (e) is a checkerboard. In (c), the green $\times$ shows σ_m measured at the low contrast edge, the blue $+$ shows σ measured at the high contrast edge, the red line is the approximation of σ based on Eq. (3.10), and the black line is the approximation of σ_m based on Eq. (3.11). In (f), the green $\times$ is σ_m measured on the face, and the blue $+$ is σ measured on the checkerboard.

TABLE 3.1: Calibration parameters

Experiment	ϕ_1	ϕ_2	ϕ_3	b_f(mm)	f(mm)	N_c	d_f(mm)
Edge Chart	-1.13	0.275	2.13	20.4	19.4	3.5	400
Face and Checkerboard	-0.555	1.42	2.88	47.7	46.9	5.0	2800

3.3 Extended Kalman Filter

The experiments presented in Section 3.2 demonstrated that the relationship between the measurement σ_m and the metric distance d can be expressed by using Eqs. (3.10) and (3.11). This section presents an EKF framework for estimating the scale based on these relationships.

To begin with, the scale factor Λ and image velocity v^i are defined. They are illustrated in Fig. 3.6 where d^i is the metric distance to each point of a scene, z^i is its up to a scale counterpart, $\dot{z}^i$ is the derivative, and the subscript i is used to denote the i-th feature point of a scene. As shown in Fig. 3.6(a), Λ defines the scale factor of a scene as

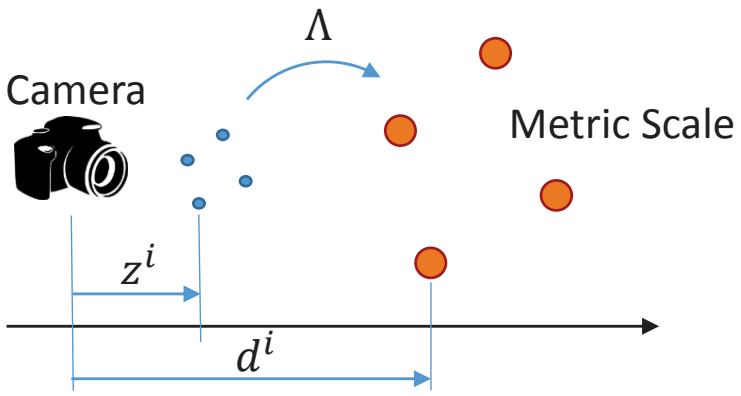
$$d^i = \Lambda \cdot z^i. \tag{3.12}$$

As shown in Fig. 3.6(b), the image velocity is the projection of the 3D relative velocity of a point onto the image plane with unit focal length $f = 1$ and is given as

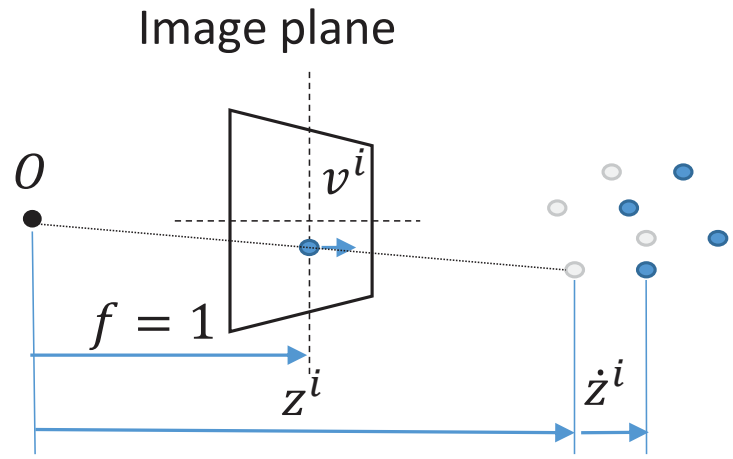
$$v^i = \frac{\dot{z}^i}{z^i}. \tag{3.13}$$

Both z^i and v^i can be obtained using a sequence of images and one of the many algorithms available in the literature, such as [63], assuming that the observed object is stationary. Given that image velocity, the time derivative of Eq. (3.12) can be expressed as

$$\dot{d^i} = \Lambda \cdot \dot{z}^i = \Lambda \cdot z^i \cdot v^i. \tag{3.14}$$



(a)



(b)

FIGURE 3.6: Illustration of the metric scale (a) and the image velocity (b)

Taking the time derivative of Eq. (3.11) and using Eqs. (3.12) and (3.14):

$$\begin{aligned}\dot{\sigma}_m^i &= \lambda^i \cdot \frac{d}{dt} D(\Lambda, z^i, v^i) \\ &= \frac{2\lambda \cdot^i b_\delta^i \cdot \Lambda \cdot z^i \cdot v^i}{\phi_1 \cdot \phi_2} \left(\frac{f}{\Lambda \cdot z^i - f}\right)^2 \exp\left(-\frac{1}{\phi_2} \cdot {b_\delta^i}^2\right),\end{aligned} \tag{3.15}$$

where $b_\delta^i = \frac{\Lambda z^i f}{\Lambda z^i - f} - b_f$. To estimate Λ and λ^i which are constants, an EKF is described in the following.

The state vector of the EKF is as follows:

$$\boldsymbol{x}_k = [\Lambda_k \ \lambda_k^i \ \sigma_{m,k}^i]^T, \tag{3.16}$$

where k is discrete time, $i = 1 \dots N$, and N is the number of observed feature points. Note that each feature point has its own texture correction factor λ_i because the texture is different in different feature points. The process equations governing the evolution of the state vector are

$$\begin{aligned} \Lambda_{k+1} &= \Lambda_k + \epsilon_{\Lambda_k}, \\ \lambda^i_{k+1} &= \lambda^i_k + \epsilon_{\lambda^i_k}, \\ \sigma^i_{m,k+1} &= \sigma^i_{m,k} + \lambda^i_k \cdot \frac{d}{dt} D(\Lambda_k, z^i_k, v^i_k) \ \cdot \Delta t + \epsilon_{\sigma^i_k} \cdot \Delta t, \end{aligned} \tag{3.17}$$

where Δt is defined as $\Delta t = t_{k+1} - t_k$ and $\boldsymbol{\epsilon}_k = [\epsilon_{\Lambda_k} \ \epsilon_{\lambda^i_k} \ \epsilon_{\sigma^i_k}]^T$ represents the process noise. Note that Eq. (3.15) is used to compute $\sigma^i_{m,k+1}$.

The observation of the amount of defocus blur is obtained at edge locations around feature points. When the observation vector is $\boldsymbol{z}_k = [z^i_k]^T$, the observation equations then become

$$z^i_k = \sigma^i_{m,k} + \eta^i_k, \tag{3.18}$$

where $\boldsymbol{\eta}_k = [\eta^i_k]^T$ is the observation noise vector. Furthermore, the constraint defined by Eq. (3.11) always needs to be satisfied. In the EKF framework, equality constraints can be imposed using the so-called projection method [64]. These constraints are rewritten as

$$c^i_k = \frac{\lambda^i_k \cdot D(\Lambda_k, z^i_k)}{\sigma^i_{m,k}} - 1 + \zeta^i_k = 0, \tag{3.19}$$

where $\boldsymbol{c}[\boldsymbol{x}_k] = [c^i_k]^T$ is the constraint vector for equality state constraints. $\boldsymbol{\zeta}_k = [\zeta^i_k]^T$ is the noise vector added to account for the possible extent of constraint violations.

It is assumed that the noises $\boldsymbol{\epsilon}_k$, $\boldsymbol{\eta}_k$, and $\boldsymbol{\zeta}_k$ are all Gaussian, temporally uncorrelated and zero-mean

$$E[\boldsymbol{\epsilon}_k] = E[\boldsymbol{\eta}_k] = E[\boldsymbol{\zeta}_k] = 0, \forall k \tag{3.20}$$

with the corresponding covariance

$$\begin{aligned} E[\boldsymbol{\epsilon}_k \cdot \boldsymbol{\epsilon}_k^T] &= \boldsymbol{Q}, \\ E[\boldsymbol{\eta}_k \cdot \boldsymbol{\eta}_k^T] &= \boldsymbol{R}, \\ E[\boldsymbol{\zeta}_k \cdot \boldsymbol{\zeta}_k^T] &= \boldsymbol{R_c}. \end{aligned} \tag{3.21}$$

Then, the following EKF procedure can be applied.

1. predict the state estimate

   $\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_k \cdot \hat{\boldsymbol{x}}_{k-1|k-1}$

2. predict the measurement

   $\hat{\boldsymbol{z}}_{k|k-1} = \boldsymbol{H}_k \cdot \hat{\boldsymbol{x}}_{k|k-1}$

3. compute the state covariance

   $\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \cdot \boldsymbol{P}_{k-1|k-1} \cdot \boldsymbol{F}_k^T + \boldsymbol{Q}_k$

4. compute the innovation covariance

   $\boldsymbol{S}_k = \boldsymbol{H}_k \cdot \boldsymbol{P}_{k|k-1} \cdot \boldsymbol{H}_k^T + \boldsymbol{R}_k$

5. compute the innovation

   $\nu_k = \boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1}$

6. compute the filter gain

   $\boldsymbol{W}_k = \boldsymbol{P}_{k|k-1} \cdot \boldsymbol{H}_k^T \cdot \boldsymbol{S}_k^{-1}$

7. update the state estimate

   $\hat{\boldsymbol{x}}_{k|k}^* = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{W}_k \cdot \nu_k$

8. update the state covariance

   $\boldsymbol{P}_{k|k}^* = \boldsymbol{P}_{k|k-1} - \boldsymbol{W}_k \cdot \boldsymbol{S}_k \cdot \boldsymbol{W}_k^T$

Here, $\boldsymbol{F}_k$ is the state transition matrix derived from the partial derivative of Eq. (3.17) with respect to the state vector. $\boldsymbol{H}_k$ is the observation matrix derived from the partial derivative of Eq. (3.18) also with respect to the state vector. $\hat{\square}$ means the prediction. $\square^*$

means the unconstraint. Next, a two-step projection method [64] to implement an EKF with non-linear equality constraints is applied. The procedure is as follows:

1. calculate the constraint matrix
   $$\boldsymbol{H}_{c,k} = \left.\frac{\partial \boldsymbol{c}[\boldsymbol{x}]}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}^{*}_{k|k}}$$
2. compute the constrained covariance
   $$\boldsymbol{S}_{c,k} = \boldsymbol{H}_{c,k} \cdot \boldsymbol{P}^{*}_{k|k} \cdot \boldsymbol{H}^{T}_{c,k} + \boldsymbol{R}_{c,k}$$
3. compute the constrained gain
   $$\boldsymbol{W}_{c,k} = \boldsymbol{P}^{*}_{k|k} \cdot \boldsymbol{H}^{T}_{c,k} \cdot \boldsymbol{S}^{-1}_{c,k}$$
4. apply the first-step constraint for the state estimate
   $$\hat{\boldsymbol{x}}^{+}_{k|k} = \hat{\boldsymbol{x}}^{*}_{k|k} - \boldsymbol{W}_{c,k} \cdot \boldsymbol{c}[\hat{\boldsymbol{x}}^{*}_{k|k}]$$
5. apply the first-step constraint for the state covariance
   $$\boldsymbol{P}^{+}_{k|k} = \boldsymbol{P}^{*}_{k|k} - \boldsymbol{W}_{c,k} \cdot \boldsymbol{S}_{c,k} \cdot \boldsymbol{W}^{T}_{c,k}$$
6. update from 1) to 3) with $\hat{\boldsymbol{x}}^{+}_{k|k}$ and $\boldsymbol{P}^{+}_{k|k}$ instead of $\hat{\boldsymbol{x}}^{*}_{k|k}$ and $\boldsymbol{P}^{*}_{k|k}$
7. apply the second-step constraint for the state estimate
   $$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}^{+}_{k|k} - \boldsymbol{W}_{c,k} \cdot \boldsymbol{c}[\hat{\boldsymbol{x}}^{+}_{k|k}]$$
8. apply the second-step constraint for the state covariance
   $$\boldsymbol{P}_{k|k} = \boldsymbol{P}^{+}_{k|k} + (\hat{\boldsymbol{x}}_{k|k} - \hat{\boldsymbol{x}}^{+}_{k|k}) \cdot (\hat{\boldsymbol{x}}_{k|k} - \hat{\boldsymbol{x}}^{+}_{k|k})^{T}$$

Here, $\square^{+}$ means the first-step constraint.

3.4 Experimental Evaluation

3.4.1 Experiment 1: Properties of the proposed method

The objective of this experiment is to evaluate the ability of the proposed method shown in Section 3.3 to estimate Λ and λ^{i}. In this experiment, the same edge patterns and camera settings used to obtain Fig. 3.5(c) were used. The chart is shown in Fig. 3.7. A sequence

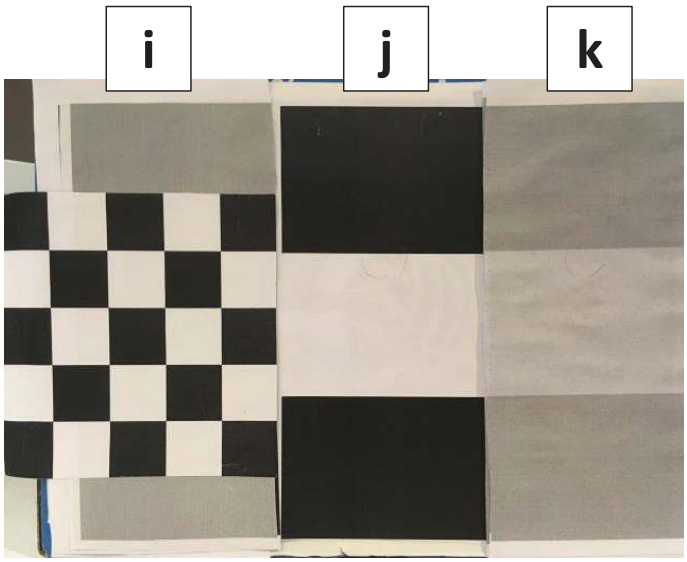


FIGURE 3.7: The chart used in Experiment 1, where (i) is the checkerboard used to compute the true metric scale, and (j) and (k) have the same edge patterns as (b) and (a) described in Fig. 3.5, respectively.

FIGURE 3.8: CANON EOS 650D (EOS Kiss X6i in Japan) camera with the EF-S 18-135mm f/3.5-5.6 IS STM lens used in the experiments.

of images with 640 × 480 pixels resolution at 30 frame-per-seconds (fps) was taken by the CANON EOS 650D with the EF-S 18-135mm f/3.5-5.6 IS STM lens shown in Fig. 3.8. Initially, the camera was positioned to face the chart at a distance $d = 1000$ mm. The camera was moved at an approximately constant speed of 55 mm/s along the optical axis until $d = 400$ mm. During the experiment, values of σ were measured at the edge locations on (k) of Fig. 3.7 and the median of them was used as σ_m^i. The true σ was measured at the edge locations on (j) of Fig. 3.7 in the same way. The true scale was calculated using a known size of the checkerboard shown in (i) of Fig. 3.7. The non-scaled distance z^i and the image velocity v^i were calculated from changes in the size of the checkerboard in the image sequence. Therefore, in this experiment, measured z^i and v^i were accurate except for some small amount of noise. In this experiment, the initial state of Λ was calculated by solving the Eq. (3.10) directly, thus it has an error caused by blur due to texture. The initial state of λ^i was calculated with initial Λ as

$$\lambda^i = \frac{\sigma_m^i}{D(\Lambda z^i)}. \tag{3.22}$$

Fig. 3.9(a), (b), and (c) show the estimates of Λ, λ^i, and σ_m^i. The blue lines show the estimates, the red lines show the ground truth, and the black line shows the measurement. The true λ^i was calculated from Eq. (3.22) with the true Λ. It can be seen that σ_m^i gradually changes as expected and Λ converges as more and more measurements are obtained. As expected, the texture correction factor λ^i stays almost constant while σ_m^i changes. Fig. 3.9(d) shows the estimated metric distance. After convergence, the final distance error between the camera and the chart was only 1.6 mm. These results illustrate that the proposed method can correctly estimate the metric scale.

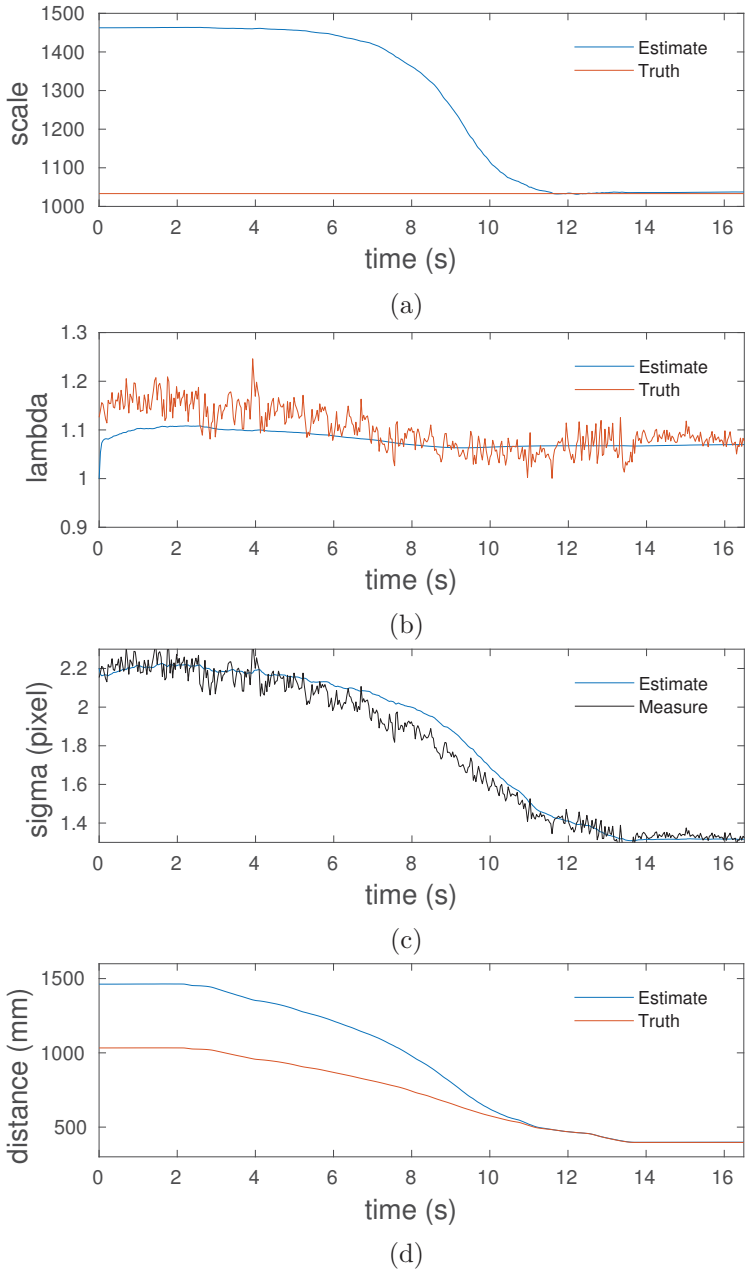


FIGURE 3.9: The estimates of Λ (a), λ^i (b), σ_m^i (c), and the metric distance d^i (d) in Experiment 1. The blue lines show the estimates, the red lines show the ground truth, and the black line shows the measurement.

3.4.2 Experiment 2: 3D metric scale estimation in a cluttered environment

This experiment is aimed at demonstrating that the proposed algorithm can estimate Λ and λ^i even in a cluttered environment. A set of feature points around a desk in Fig. 3.10 was observed by the same camera with Experiment 1. The parameters used were shown in Table 3.2. Initially, the camera was set facing to the desk at the distance of approximately 3000 mm. It was then moved at an approximately constant speed (around 230 mm/s) until about 1500 mm. The non-scaled distance z^i and the image velocity v^i were measured using the SfM algorithm with bundle adjustment [55] as implemented in Matlab®. The camera egomotion and the 3D point map obtained from the SfM algorithm were then rescaled with the metric scale estimated using the proposed EKF. In this experiment, the values of σ measured at feature points detected by KLT algorithm [61], [62] were used as σ_m^i. To avoid the focal plane ambiguity, it was assumed that all observed points were located on one side of the focal plane. As in the case with Subsection 3.4.1, the true scale was calculated with a known size of the checkerboard shown in Fig. 3.10. The initial Λ and λ^i were also calculated in the same way as in Subsection 3.4.1.

Fig. 3.11(a) shows the estimate of Λ. Fig. 3.11(b), (c), and (d) show the estimates of λ^i, σ_m^i, and the metric distance d^i at the point indicated in (m) of Fig. 3.10. It could be seen that, even in this cluttered environment, λ^i almost stayed constant while σ_m^i changed. As a result, the estimation Λ gradually converged. Note that the fluctuation of the estimates seen between one and two seconds is due to the motion blur effect.

Fig. 3.12 shows the camera poses and 3D point map with the estimated scale. The red line shows the camera trajectory reconstructed by the estimated scale. The green line shows the ground truth. The red dots show the estimated 3D locations of the observed feature points. The green dots show the 3D locations of feature points on the black-and-white corners of the checkerboard. Note that the large error in camera position at the beginning is expected as the EKF takes time to converge. The RMSE of the final distances from the camera to the reconstructed 3D points was only 0.32 mm under the assumption that the 3D point map obtained from the SfM algorithm was true. The results from this experiment demonstrated that the proposed method combined with SfM could generate

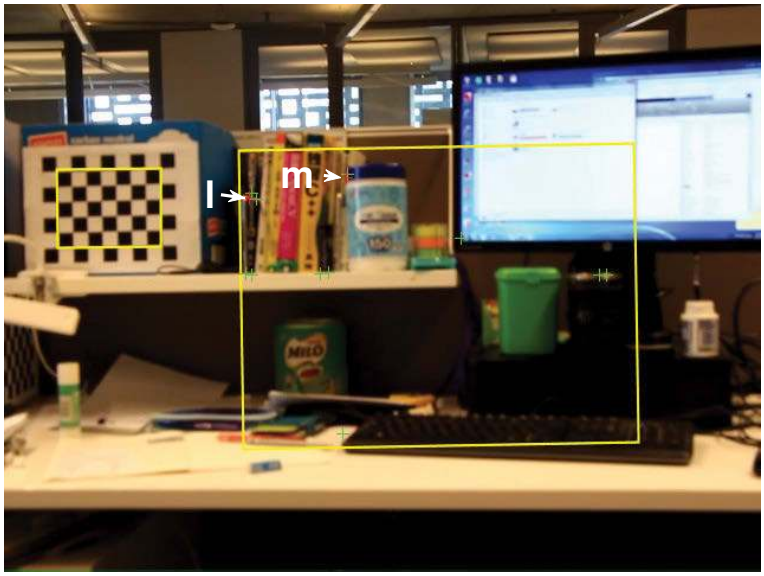


FIGURE 3.10: The desk environment used in Experiment 2. The red +'s indicated by arrows with letters 'l' and 'm' are two of the feature points where σ_m^i are measured. The green +'s show the other feature points used for the scale estimation. The checkerboard was placed to compute the true scale.

TABLE 3.2: Parameters for Experiment 2

ϕ_1	ϕ_2	ϕ_3	b_f[mm]	f[mm]	N_c	d_f [mm]
-1.14	0.086	1.53	32.3	32.2	4.0	5000

the camera poses and 3D point map to the metric scale with only a monocular camera even in a cluttered environment.

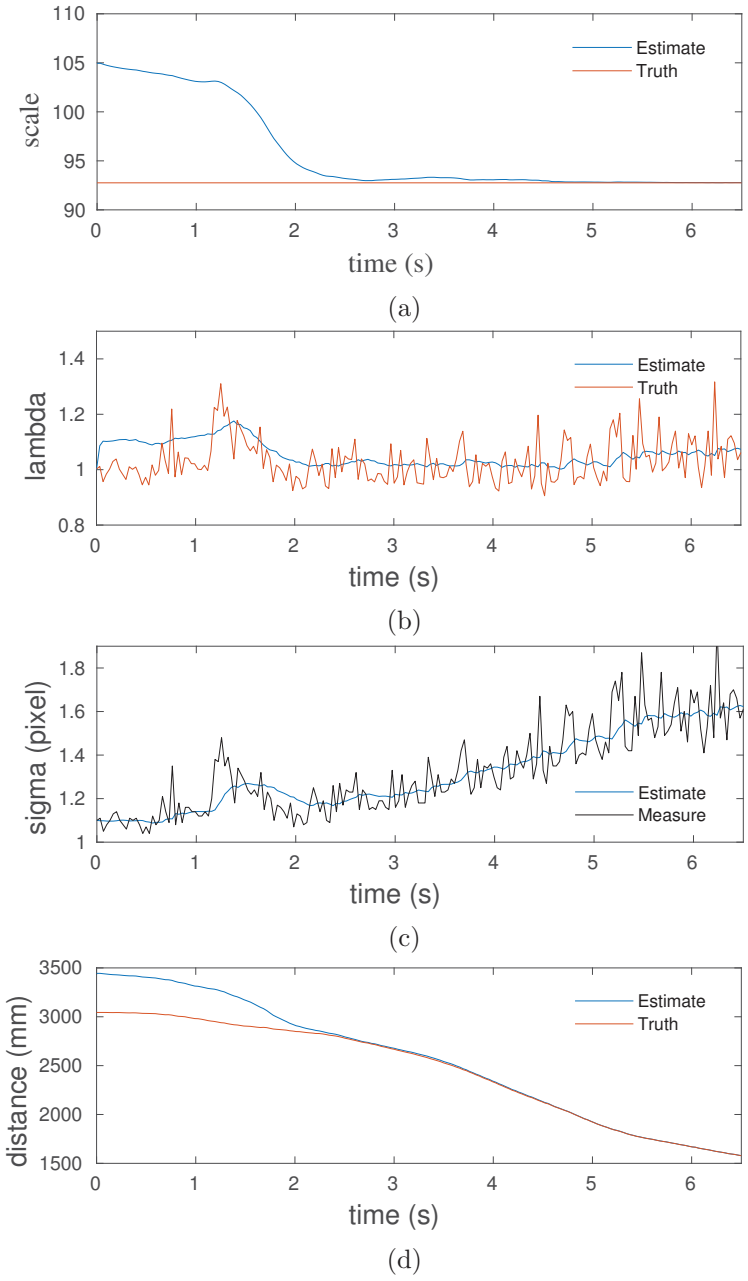


FIGURE 3.11: The estimates of Λ (a), λ^i (b), σ_m^i (c), and the metric distance d^i (d) at the point indicated in 'm' of Fig. 3.10. The blue lines show the estimates, the red lines show the ground truth, and the black line shows the measurement.

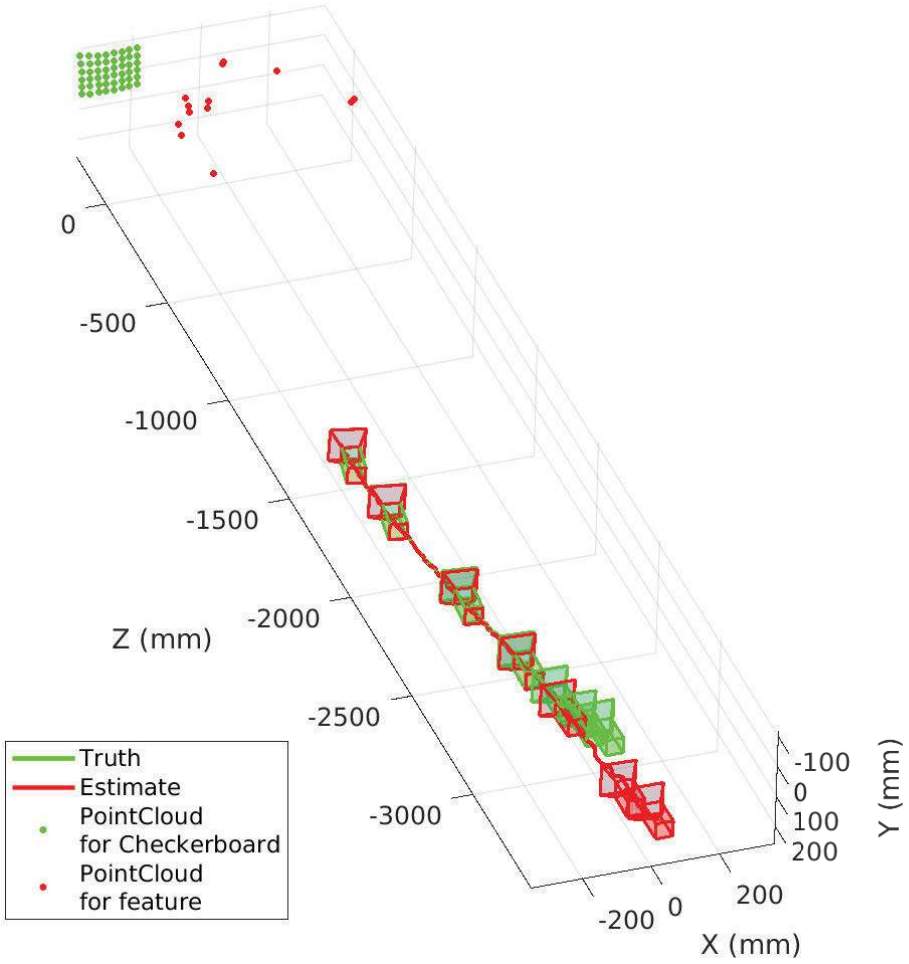


FIGURE 3.12: The camera poses and 3D point map reconstructed to the metric scale. The red line shows the camera trajectory reconstructed by the estimated scale. The green line shows the ground truth. The red dots show the estimated 3D locations of observed feature points. The green dots show the 3D locations of feature points on the black-and-white corners of the checkerboard.

3.5 Discussion

In spite of the fact that none of the feature points observed were in focus during the experiment shown in Subsection 3.4.2, the accurate metric scale of the scene was estimated. This is an advantage compared to the existing algorithm proposed in [31], which requires a focused scene for each feature point somewhere in the image sequence to resolve the blur texture ambiguity. The proposed texture correction factor λ could approximate the extension of blur due to texture on the defocus estimation and the EKF algorithm could

estimate the value of λ for each feature point. As a result, the metric scale of the scene was estimated correctly even though feature points concerned were never in focus.

The process of estimation including feature detection, tracking, and defocus estimation on an image with 640×480 pixels resolution used in the experiment shown in Subsection 3.4.2 takes about 0.1 seconds in MATLAB® with Intel® Core™ i5-6300U CPU at 2.40GHz $\times$ 4. When this algorithm is efficiently implemented in C, the calculation time is expected to be reduced within the frame period of the camera. Estimating the defocus blur amounts only at feature points of a scene in the EKF framework enables a computationally-efficient system. Therefore, the proposed method has the potential to be used in a real-time system.

The proposed method relies on the assumption that λ^i is constant. However, this assumption does not hold if there are significant changes in texture and illumination through the image sequence. For example, Fig. 3.13(a) shows the estimate of λ^i at the point indicated in (l) of Fig. 3.10. It is clear that λ^i of (l) decreases continuously due to the changes in texture. The point of (l) is positioned at the spine of the book. The spine appears as a single edge when the camera is at a distance. However, it reveals rich texture due to the letters present on it when the camera is nearby. The amount of defocus blur cannot be estimated correctly in this case as the calculated gradient is not correct when there are many intensity discontinuities in the ROI. As a result, the measurement σ^i shown in Fig. 3.13(b) does not change as expected. The assumption that λ^i is constant is no longer correct in these situations. However, the use of additive noise for the possible extent of constraint violations in EKF relaxes the constraint that λ^i is constant. Therefore, as seen from multiple results in Section 3.4, the EKF can estimate the metric scale correctly despite the fact that some of λ^i change with the camera motion.

In a dynamic scene, both motion and defocus contribute to the image blur. In the experiment shown in Subsection 3.4.2, the fluctuation of the estimates due to the motion blur effect was seen temporarily. A temporary motion blur is not a crucial problem, although it interferes with the convergence of estimates in EKF. However, continual motion blur may cause an estimation error. For a robotic application where a robot moves agilely, it is necessary to remove the motion blur effect in DfD. In Chapter 4, a method for eliminating the impact of motion blur in DfD is presented.

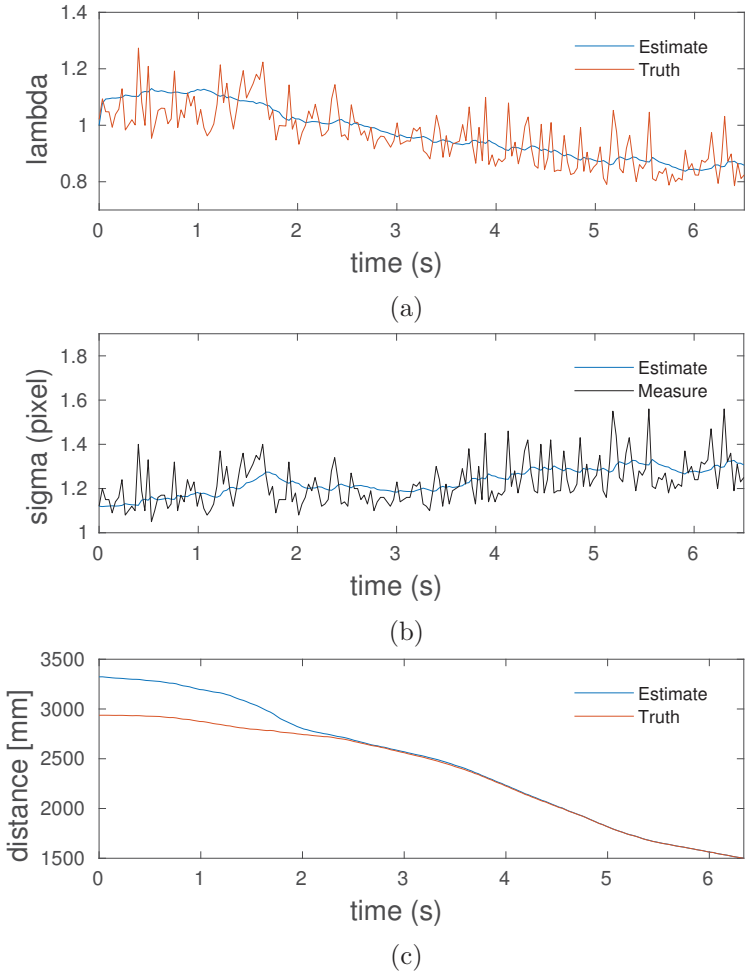


FIGURE 3.13: The estimates of λ^i (a), σ_m^i (b), and the metric distance d^i (c) at the point indicated in 'l' of Fig. 3.10. The blue lines show the estimates, the red lines show the ground truth, and the black line shows the measurement.

In the experiment in Section 3.4, it was assumed that the feature points observed were located on one side of the focal plane to avoid the focal plane ambiguity. In the proposed method, the initial scale factor of EKF is calculated from the depth derived from the inverse of the Depth-Defocus function defined by Eq. (3.10). However, this inverse function produces multiple solutions. Unless the knowledge of which side of the focal plane the feature points exist, the correct solutions cannot be selected. One of the indicators is the direction of the changes in defocus blur amount induced by camera motion [31], [14]. For example, when the camera moves toward the feature points of a scene, the blur amounts of feature points placed on the near side of the focal plane increase while the other ones placed at the far side decrease (see Fig. 3.14). This initialization procedure will help to

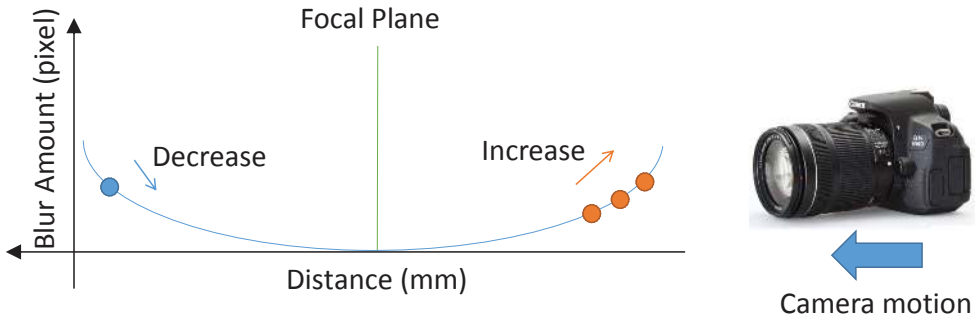


FIGURE 3.14: Illustration of a method to avoid focal plan ambiguity by using the camera motion. The orange dots show the defocus blur amounts of feature points located on the near side of the focal plane. The blue dot shows the defocus blur amount of a feature point located on the far side of the focal plane. The direction of defocus blur change induced by a camera motion is a possible indicator to resolve the focal plane ambiguity problem.

estimate the reasonable initial scale for EKF. As another approach, a method for avoiding the focal plane ambiguity problem using the output of monocular SLAM is presented in Chapter 4.

3.6 Conclusion

In this chapter, an approach for metric scale estimation of 3D environments from a sequence of monocular images was demonstrated. It was shown that blur due to texture could be represented using a depth-invariant gain factor when estimating depth from defocus. An EKF framework that incorporates information from non-scaled distances and image velocity was shown to be able to resolve blur texture ambiguity in DfD and produce accurate metric scale estimation. The first experiment shown in Subsection 3.4.1 validated the proposed method in a simple edge chart. The second experiment shown in Subsection 3.4.2 demonstrated that the proposed method was effective even in a practical scene of the cluttered desk environment. Also, this result showed that the combination of DfD and SfM in EKF could estimate an accurate metric scale of a scene. The state-of-the-art monocular SLAM systems including ORB-SLAM [15] utilize the SfM algorithm with bundle adjustment. This indicates that the proposed algorithm has a potential to be used for monocular SLAM algorithm.

In the next chapter, this method is leveraged and incorporated into monocular SLAM to resolve the scale drift problem. The next chapter also presents the methods for eliminating motion blur effect and avoiding the focal plane ambiguity by using the output of monocular SLAM.

Chapter 4

Scale Drift-free Monocular SLAM

This chapter presents a novel approach to correct errors caused by accumulated scale drift in monocular SLAM. Section 4.1 presents a brief summary of the proposed strategy for scale drift elimination. Section 4.2 presents a method for eliminating the impact of motion blur in DfD based on optical flow, which is a motion field of an image induced by camera motion. In Section 4.3, a non-linear least squares optimization problem is formulated to integrate depth estimates from defocus to monocular SLAM. Finally, in Section 4.4, the proposed algorithm is experimentally evaluated by processing the output of ORB-SLAM [15] to obtain an accurate metric scale map from a monocular camera. [2]

4.1 Scale Drift Elimination Strategy

In practical implementations of monocular SLAM, the scale factor in different parts of the map may vary due to scale drift [4]. This is illustrated in Fig. 4.1 where the camera travels in the direction of the arrows and returns to the origin. The blue dots show the map points generated by monocular SLAM. Ideally, Λ_1, Λ_2, and Λ_3 which are scale factors in the different local regions should be the same value. However, the accumulated scale

[2] The work presented in this chapter was reported in "Eliminating Scale Drift in Monocular SLAM Using Depth From Defocus," in IEEE Robotics and Automation Letters, vol. 3, no. 1, pp. 581-587, Jan. 2018. Also, it will be presented at 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, May. 2018.

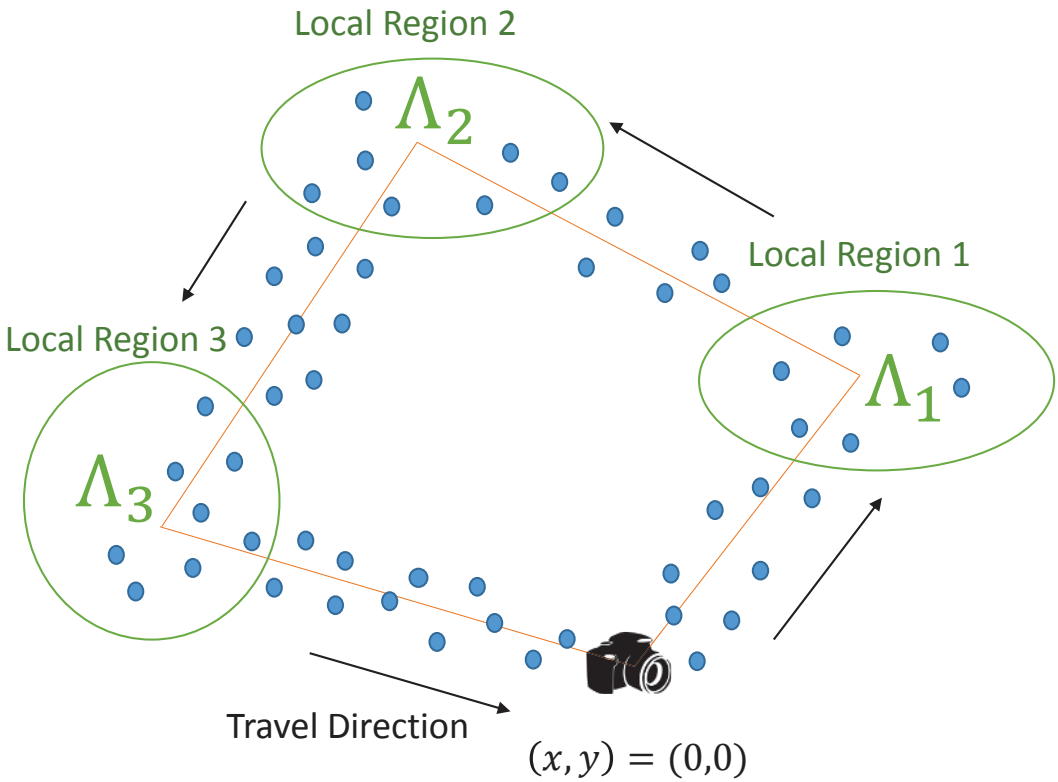


FIGURE 4.1: Illustration of the scale difference caused by the scale drift in local regions of the map. The blue dots show the map points generated by monocular SLAM algorithm.

drift, which is invariably present in practical monocular SLAM algorithms, makes them different.

Many monocular SLAM algorithms currently available, including ORB-SLAM [15], generate a map of point features present in the environment and the camera poses of a selected set of images, known as keyframes. The feature points observed among keyframes are detected, and their correspondences are determined by a feature detector and descriptor such as ORB [12]. The location of the point feature in the map is called the feature location. In this research, the feature location and keyframe pose estimates generated by monocular SLAM are post-processed to correct scale drift. The DfD algorithm is applied in arbitrary selected local regions of the map, and the metric scales of these are estimated. As a result, the scale differences among them can be corrected. In order to integrate the DfD with keyframe-based SLAM systems, a non-linear least squares optimization is proposed. The amounts of defocus blur estimated at feature locations detected by monocular SLAM algorithm are incorporated into the proposed optimization procedure. The method

proposed in this chapter makes use of the texture correction factor λ proposed in Chapter 3 and a method for correcting the impact of motion blur on depth estimates described in Section 4.2. A method for avoiding the focal plane ambiguity problem is also presented in Section 4.3. Although the experimental evaluations presented in this chapter use ORB-SLAM which is the best monocular SLAM algorithm currently available, the proposed algorithm can be used to enhance the output from any keyframe-based monocular SLAM algorithms.

4.2 Eliminating the Impact of Motion Blur

This section presents a method for eliminating motion blur effect on defocus blur estimates. When both defocus and motion contribute to image blur, Eq. (3.3) can be rewritten as

$$I^i = G(\sigma_m^i) * G(\sigma_b^i) * I_f^i, \tag{4.1}$$

where σ_b^i is the motion blur amount and σ_m^i is from Eq. (3.11). The subscript i is used to denote the i-th feature point of a scene. The composite blur amount is

$$\sigma_{mb}^i = \sqrt{{\sigma_m^i}^2 + {\sigma_b^i}^2}. \tag{4.2}$$

In [13], the projection length of motion blur onto x and y axes of the image plane is defined as the motion blur vector:

$$\mathbf{b} = [l\cos\theta \; l\sin\theta]^T, \tag{4.3}$$

where $l = 2\sigma_b^i$ is the length of motion blur along the long axis and θ is the angle between the long axis of motion blur and x axis. Also, they suggested that the motion blur vector is closely related to the optical flow. Referring to this idea, this research proposes the following equation to estimate the amount of motion blur from the optical flow:

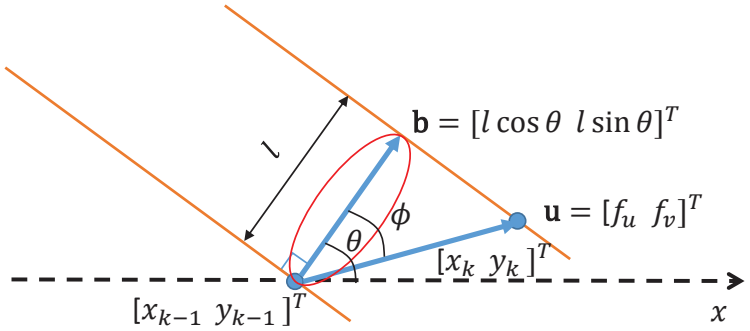


FIGURE 4.2: Illustration of Eq. (4.4). The blue dots show the corresponding feature points between the successive images, the orange lines show the edges at the feature points. The red ellipse shows the size of motion blur. ϕ is an internal angle formed by vectors $\mathbf{b}$ and $\mathbf{u}$. For simplicity, it is assumed that $T_e = T_f$ in this figure.

$$\sigma_b^i = \frac{T_e}{2T_f}|f_u \cos\theta + f_v \sin\theta|, \tag{4.4}$$

where $\mathbf{u} = [f_u \;\; f_v]^T$ is the optical flow vector. T_e and T_f are the exposure time and the frame period of the camera, respectively. This is illustrated in Fig. 4.2. The blue dots show the corresponding feature points between two successive images and the orange lines show the edges at the feature points. The red ellipse shows the size of motion blur. ϕ is an internal angle formed by vectors $\mathbf{b}$ and $\mathbf{u}$. For simplicity, it is assumed that $T_e = T_f$ in this figure. Note that θ corresponds to the angle between the vertical line of the edge and the x-axis. The length $l = |\mathbf{b}|$ can be written as

$$\begin{aligned} l &= \frac{T_e}{T_f}|\mathbf{u}|\cos\phi \\ &= \frac{T_e}{T_f}|\mathbf{u}|\frac{\mathbf{u}\cdot\mathbf{b}}{|\mathbf{u}||\mathbf{b}|} \\ &= \frac{T_e}{T_f}|f_u \cos\theta + f_v \sin\theta|. \end{aligned} \tag{4.5}$$

Using the relationship given by $l = 2\sigma_b^i$ with Eq. (4.5), Eq. (4.4) can be derived.

Solving Eq. (4.2) with Eq. (4.4) yields σ_m^i. This is illustrated in Fig. 4.3. In this experiment, the chart with a tilted edge pattern and a checkerboard shown in Fig. 4.3(a) was positioned to face the camera at a distance of two meters and moved from side to

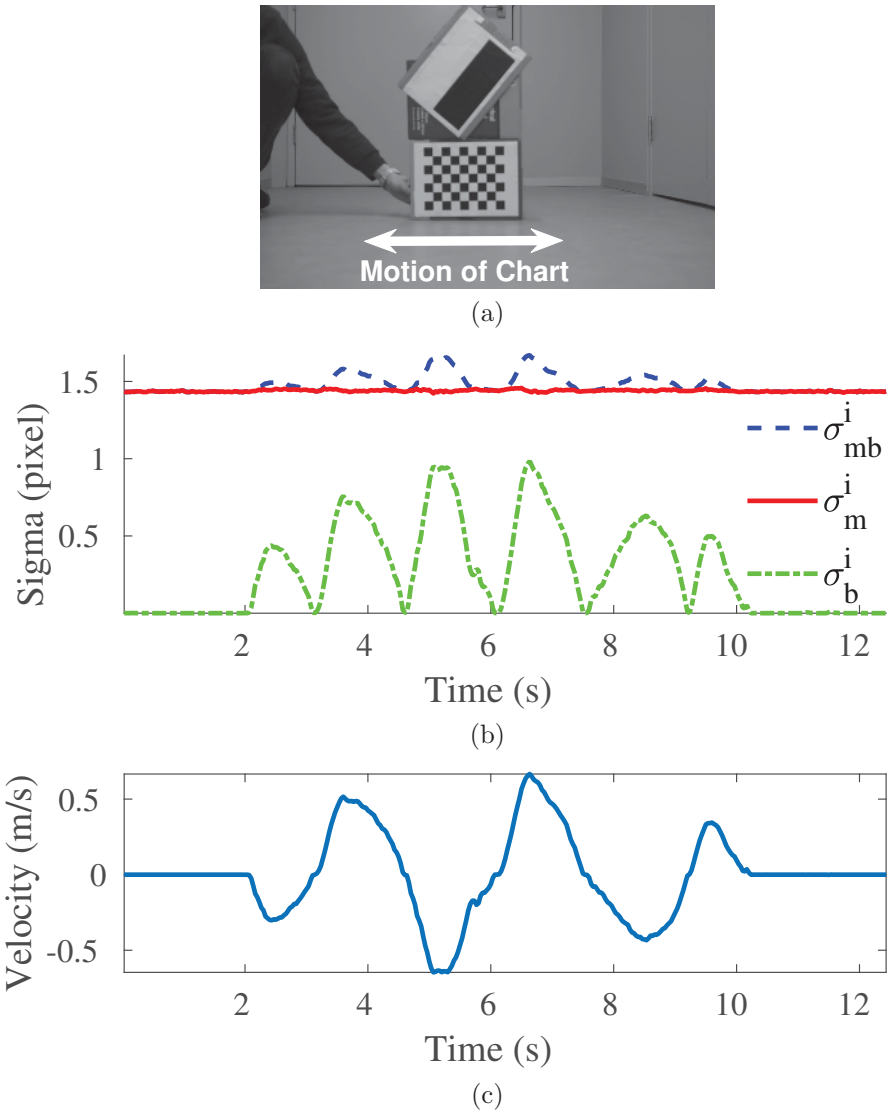


FIGURE 4.3: Demonstration of Eq. (4.4). (a) shows the chart with a tilted binary edge pattern and a checkerboard. The chart was positioned to face the camera at a distance of two meters and moved from side to side with the velocity shown in (c). In (b), the blue dash line, the red solid line, and the green dot-dash line show σ^i_{mb}, σ^i_m, and σ^i_b. As expected, σ^i_m is nearly constant. The exposure time was 8 ms and the frame period was 33 ms.

side with the velocity shown in Fig. 4.3(c). As the distance between the camera and the chart stayed constant during the experiment, the expected value of σ^i_m was constant. The composite blur amount σ^i_{mb} and the edge direction θ were measured at the edge location of the tilted chart and σ^i_b was calculated by Eq. (4.4) using the optical flow detected at the corners of the checkerboard. As shown in Fig. 4.3(b), almost constant σ^i_m was obtained as a result of the elimination of the estimated σ^i_b from σ^i_{mb}. The result shows that this

method can clearly eliminate the motion blur effect on defocus estimates. In the evaluation of the proposed algorithm with ORB-SLAM presented in Section 4.4, ORB feature points present in a keyframe and the subsequent frame are used to compute optical flow. When the corresponding feature point cannot be found in the subsequent frame, the projection of the corresponding map point onto the subsequent frame is used.

4.3 Scale Optimization

4.3.1 Optimization

Using the texture correction factor λ proposed in Chapter 3, this section formulates the non-linear least squares optimization problem to estimate metric scale. In addition, the focal plane ambiguity in DfD is avoided in the proposed formulation.

As described in Section 3.3, the scale factor Λ is defined as

$$d^{i,j} = \Lambda \cdot z^{i,j}, \tag{4.6}$$

where the subscripts i, j are used to denote the i-th map point seen from the j-th keyframe in the selected local region. $z^{i,j}$ can be obtained from the map point $\mathbf{p}_w^i = [x_w^i \ y_w^i \ z_w^i]^T$ and the keyframe pose $\mathbf{T}^j = [\mathbf{r}^j|\mathbf{t}^j]$ created by monocular SLAM as

$$\mathbf{p}^{i,j} = \mathbf{r}^{j^{-1}} \cdot (\mathbf{p}_w^i - \mathbf{t}^j), \quad \mathbf{p}^{i,j} = [x^{i,j} \ y^{i,j} \ z^{i,j}]^T, \tag{4.7}$$

where $\mathbf{r}^j$ is the rotation matrix and $\mathbf{t}^j$ is the translation vector of the keyframe pose.

To estimate the scale factor Λ and the texture correction factor λ_i, the following non-linear least squares minimization problem derived from the Depth-Defocus function with the texture correction factor shown in Eq. (3.11) and the definition of the scale factor shown in Eq. (4.6) is formulated:

$$\underset{\Lambda,\lambda^i}{\operatorname{argmin}} \sum_{i,j}(\sigma_m^{i,j} - \lambda^i \cdot D(\Lambda \cdot z^{i,j}))^2, \tag{4.8}$$

where $\sigma_m^{i,j}$ is measured at the feature points extracted by a feature detector such as ORB [12]. Note here that this formulation can avoid the focal plane ambiguity. In [31], it is pointed out that solving the inverse function of the Depth-Defocus function given by Eq. (3.10) causes multiple depth solutions across the focal plane. To avoid this, the optimization problem is formulated without inverting Eq. (3.10), enabling the minimization to yield a single scale solution. Another advantage of this approach is that it is not necessary to use all the feature points detected by SLAM algorithm for solving the optimization problem given by Eq. (4.8). A small number of feature points is adequate. Therefore, this research adopts the strategy of selecting a set of good feature points in order to estimate the scale.

First of all, a good initial guess is required to solve the optimization given in Eq. (4.8), to avoid the possibility of converging to local minima different from the truth. The procedure to obtain the initial guess is presented in the following.

4.3.2 Initial Guess

First, a set of feature points with sharp edges is selected to minimize the impact of blur texture ambiguity and therefore the optimization problem of Eq. (4.8) is simplified to

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{i,j}(\sigma_m^{i,j} - D(\Lambda \cdot z^{i,j}))^2. \tag{4.9}$$

To evaluate the edge strength, this research proposes an index $smg^{i,j}$ which is the multiplication of the estimated blur $\sigma_m^{i,j}$ and the gradient magnitude around the feature point $mg^{i,j} = || \bigtriangledown I^{i,j}||$ where $\bigtriangledown$ means the gradient. Typically, the gradient magnitude is used to evaluate the edge strength in the image [60]. However, the gradient magnitude is depth-variant when defocus blur exists. This is illustrated in Fig. 4.4. Fig. 4.4(a) and (b) are low-contrast and binary edge patterns, respectively. In Fig. 4.4(c), the blue $\times$ and

the red + show $\sigma_m^{i,j}$ measured on (a) and (b), respectively. In Fig. 4.4(d), the cyan × and the magenta + show the edge strength evaluated by the index $mg^{i,j}$ measured on (a) and (b), respectively. As can be seen, the index $mg^{i,j}$ depends on the amount of $\sigma_m^{i,j}$. On the other hand, in Fig. 4.4(d), the blue × and the red + show the edge strength evaluated by the proposed index $smg^{i,j}$ measured on (a) and (b), respectively. As can be seen, the values of $smg^{i,j}$ stay almost constant despite the changes in defocus blur. The gradient magnitude $mg^{i,j}$ is inversely proportional to $\sigma_m^{i,j}$ as shown in Fig. 4.4. Multiplying $\sigma_m^{i,j}$ and $mg^{i,j}$ can efficiently cancel the blur effect, thus $smg^{i,j}$ is a good index to evaluate the edge strength. The same Gaussian filter is used for both the blur estimates and the gradient magnitude estimates in this experiment. The results demonstrate that the proposed index can express the edge strength, independent of the amount of defocus blur.

Using the proposed index, the feature selection criterion is defined as

$$\sigma_m^{i,j} = \begin{cases} inlier & if\ e_{thl} < smg^{i,j} < e_{thh} \\ outlier & else, \end{cases} \tag{4.10}$$

where e_{thl} and e_{thh} are threshold values and decided from the value of $smg^{i,j}$ on the binary edge pattern measured in advance.

The solution of Eq. (4.9) gives an accurate scale estimate, provided a sufficient number of feature points with sharp edges exist in the scene. In situations where this is not the case, it was found that the scale estimate obtained serves as a good initial guess to the more general optimization problem given by Eq. (4.8).

4.3.3 Feature Point Selection for Optimization

Using the initial guess derived from Eq. (4.9), Eq. (4.8) is solved. In Section 3.5, it was demonstrated that the amount of defocus blur cannot be estimated correctly on complex texture such as letters, and therefore constraints of Eq. (3.11) no longer hold in these situations. Furthermore, due to the non-linearity of Eq. (3.10), depth estimation from defocus blur is only effective at short range [31]. Therefore, criteria to select feature points

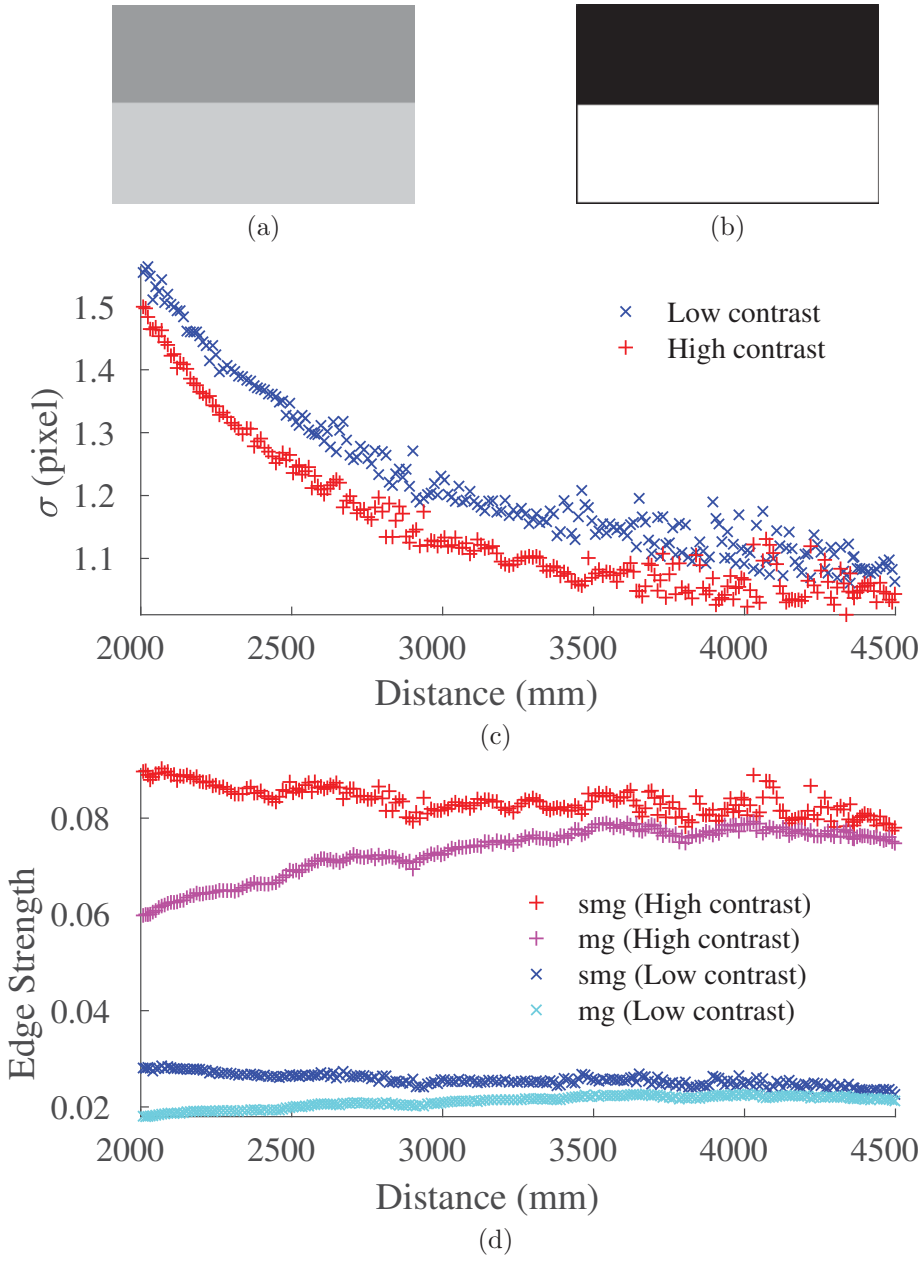


FIGURE 4.4: Demonstration of Eq. (4.10). (a) and (b) show the charts with a low-contrast edge and a binary edge, respectively. In (c), the blue × and the red + show $\sigma_m^{i,j}$ measured on (a) and (b), respectively. In (d), the cyan × and the magenta + show the edge strength evaluated by the index $mg^{i,j}$ measured on (a) and (b), and the blue × and the red + show the edge strength evaluated by the proposed index $smg^{i,j}$ measured on (a) and (b), respectively.

to be incorporated into computing the objective function defined by Eq. (4.8) are proposed as follows.

Constancy of λ^i

The feature points that satisfy the constraint given by Eq. (3.11) are selected. The ratio of $\lambda^{i,j}$ and $\lambda^{i,j-1}$ expected from the initial guess is used as an index to evaluate its constancy:

$$r^{i,j} = \frac{\lambda_{ini}^{i,j}}{\lambda_{ini}^{i,j-1}}, \tag{4.11}$$

where $\lambda_{ini}^{i,j}$ is derived from Eqs. (3.11) and (4.6) with the initial scale Λ_{ini} obtained from Eq. (4.9):

$$\lambda_{ini}^{i,j} = \frac{\sigma_m^{i,j}}{D(\Lambda_{ini} \cdot z^{i,j})}. \tag{4.12}$$

The criterion to select feature points based on Eq. (4.11) is defined as

$$\sigma_m^{i,j} = \begin{cases} inlier & if\ r_{thl} < r^{i,j} < r_{thh} \\ outlier & else, \end{cases} \tag{4.13}$$

where r_{thl} and r_{thh} are threshold values and empirically decided from the accuracy of the initial guess in advance. When satisfied with this condition, λ^i is regarded as a constant value between two keyframes.

Effective Range

The feature points within a range threshold are used. The range threshold in the metric scale can be decided from the Depth-Defocus calibration results described in Section 3.1 as

$$d_{th} = \alpha \cdot d_f, \tag{4.14}$$

where α is a lens specific value and also depends on the focal plane distance d_f. Using the initial scale Λ_{ini} obtained from Eq. (4.9), the non-scaled range threshold is given as

$$z_{th} = \alpha \cdot \frac{d_f}{\Lambda_{ini}}. \tag{4.15}$$

The criterion to select feature points based on Eq. (4.15) is defined as

$$\sigma_m^{i,j} = \begin{cases} inlier & if\ z^{i,j} < z_{th} \\ outlier & if\ z^{i,j} \geq z_{th}. \end{cases} \tag{4.16}$$

4.4 Experimental Evaluation

To evaluate the proposed algorithm, two experiments were conducted with two different cameras. The first experiment is to evaluate the proposed method in a corridor environment over a long distance causing the monocular SLAM algorithm to accumulate scale drift. The camera set used in the first experiment is a FLIR® BFLY-U3-23S6M-C camera and a Fujinon® CF16HA-1 lens shown in Fig. 4.5. The second experiment is to demonstrate that the proposed method is effective even on a small phone camera. The rear camera on iPhone SE shown in Fig. 4.6 was used. In both experiments, the focal length and f-number were fixed during capturing the image sequences. The calibration parameters for DfD were defined using the calibration process shown in Section 3.1 before the experiments.

FIGURE 4.5: The camera and lens used in Experiment 1. The field of view is about 37-degree width.

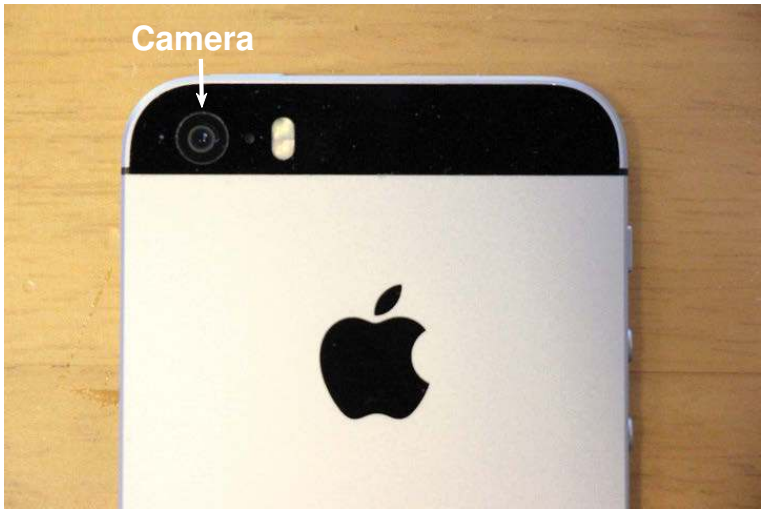


FIGURE 4.6: The rear camera on iPhone SE used in Experiment 2.

4.4.1 Experiment 1: Eliminating scale drift in a corridor environment

4.4.1.1 Dataset

A number of image sequences with 752 × 480 pixels resolution at 30 fps were captured with the camera shown in Fig. 4.5, walking in a corridor environment at the University of Technology Sydney. The maps and camera trajectories generated by ORB-SLAM from the datasets are shown in Fig. 4.7. In Fig. 4.7(a), the camera moved in the direction of the arrows and returned to the origin to make a closed loop. In Fig. 4.7(b), the camera traveled along a figure of eight trajectory as indicated by the arrows in the same environment to

Table 4.1: Scale and Scale Drift in Experiment 1

Trajectory 1			Trajectory 2		
	Scale(mm/unit)	Drift(%)		Scale(mm/unit)	Drift(%)
CA	9134	-	C1	4740	-
CB	9241	1.2	C2	4480	-5.5
			C3	5135	8.3
			C4	5051	6.6

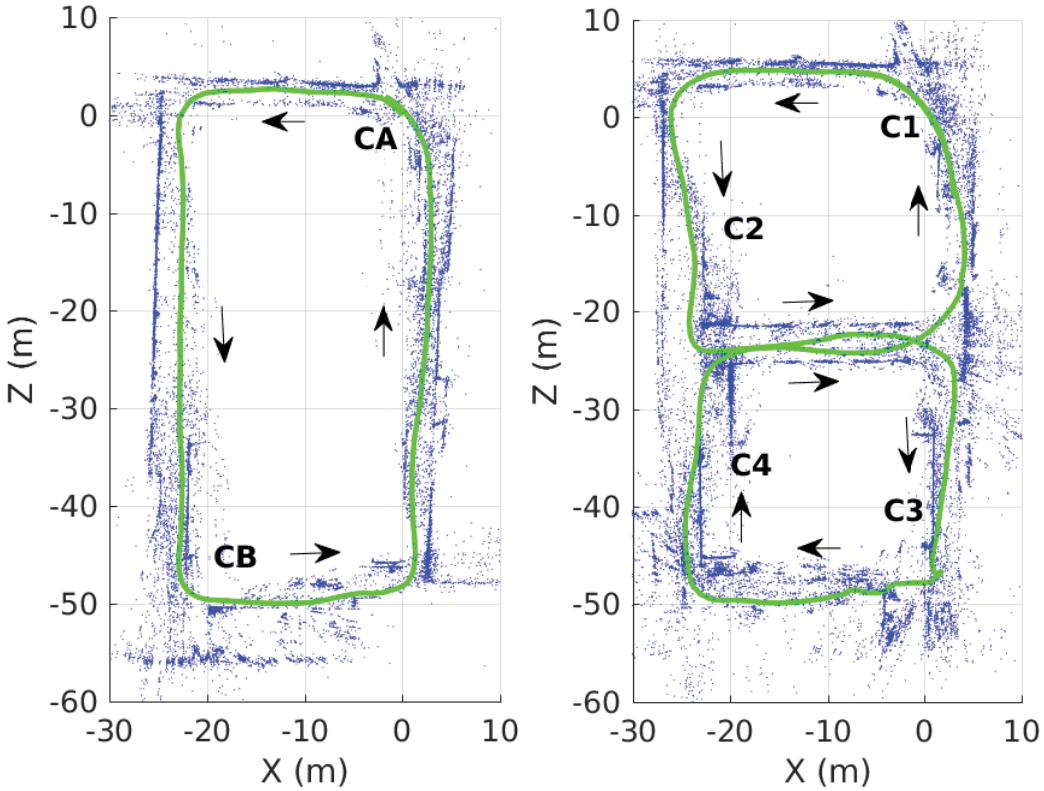
make two closed loops. Checkerboard patterns were placed at the locations CA, CB, C1, C2, C3, and C4 so that the true scale in each local region could be computed. Fig. 4.7(c) shows examples of keyframes including the checkerboards indicated in Fig. 4.7(b).

The scale estimated using the checkerboard patterns and the scale drift in each local region are shown in Table 4.1. The mean ratio of the true distances between the camera and the checkerboard patterns to the distances obtained from ORB-SLAM were used to estimate scale. The true distance was measured by the checkerboard detection algorithm [65] implemented in MATLAB®. The range over which the checkerboard was detected reliably was about seven meters. The scale drift was computed relative to the initial scale estimated at CA for trajectory 1 and C1 for trajectory 2. The ORB-SLAM result was found to be quite accurate in the case of the trajectory 1 in Fig. 4.7(a), where the scale drift at CB was only 1.2%. On the other hand, in the trajectory 2 in Fig. 4.7(b), the maximum scale drift was 8.3% perhaps due to the presence of multiple sharp turns.

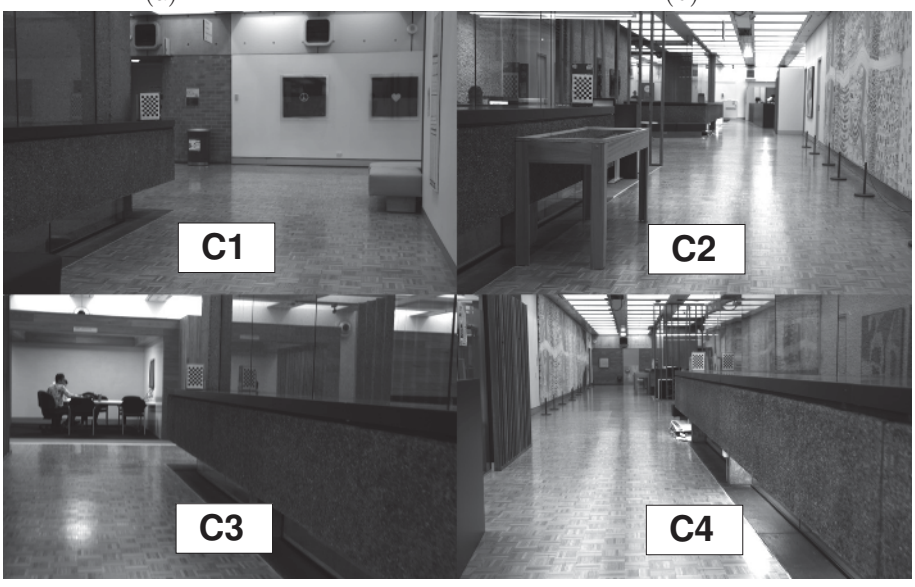
4.4.1.2 Scale estimation result

Ten keyframes around each checkerboard pattern were used in the optimization process. The parameters used for DfD are shown in Table 4.2. The threshold values for optimization are shown in Table 4.3. The trust-region reflective method [66] implemented in MATLAB® was used to solve for the initial guess using Eq. (4.9) and for the full solution using Eq. (4.8).

Table 4.4 shows the results of optimization. The error in the scale estimate is calculated as $e = (\Lambda_e - \Lambda_t)/\Lambda_t$ where subscripts e and t are used to denote the estimation and the



(a) (b)



(c)

FIGURE 4.7: In (a) and (b), the green lines show the camera trajectory, and the blue dots show the point cloud of feature points generated by ORB-SLAM. The scale was reconstructed using the mean value of the scales computed using checkerboard patterns and shown in Table 4.1. Some turns of the trajectory used to capture (b) were sharper than the trajectory shown in (a).

TABLE 4.2: Parameters used in DfD for Experiment 1

ϕ_1	ϕ_2	ϕ_3	b_f[mm]	f[mm]	d_f [mm]	N_c
-0.317	0.0825	4.20	16.9	16.8	8000	1.4

TABLE 4.3: Threshold values used in the optimization for Experiment 1

e_{thl}	e_{thh}	r_{thl}	r_{thh}	z_{th}
0.03	0.15	0.8	1.2	0.37 d_f/Λ_{ini}

TABLE 4.4: Error in Scale Estimate in Each Area (%) in Experiment 1

Trajectory 1			Trajectory 2		
	Initial Guess	Optimization		Initial Guess	Optimization
CA	-6.77	-0.03	C1	3.19	-0.20
CB	-2.11	0.17	C2	8.14	-0.01
			C3	-10.40	0.14
			C4	0.83	-0.01

truth, respectively. As can be seen, solving for the simplified optimization problem given by Eq. (4.9) results in a metric scale with an error of 10% or less. Although this result is not adequate to correct for the scale drift, it is a good initial guess for the optimization problem defined by Eq. (4.8). The final errors in the scale estimates are less than 0.20%. This is illustrated in Fig. 4.9. Fig. 4.9(a) and (c) show $z^{i,j}$ vs $\sigma_m^{i,j}$ in all keyframes in the local regions around C3 and C4, respectively. In (c), a set of feature points distributed around the true approximation curve $\sigma^{i,j} = D(z^{i,j})$ were detected for obtaining the initial guess. Fig. 4.9(d) is an example of the keyframes in the local region including C4. This image demonstrates that the feature points used to obtain the initial guess were selected at the edge positions. On the other hand, in (a), the number of feature points selected for the initial guess was smaller than in (c) and resulted in an initial guess to the scale estimation error of 10.4%. Fig. 4.9(b) is an example of the keyframes in the local region including C3. This image was a little darker than (d) and it was difficult to find the feature points with sharp edges. However, in the optimization step defined by Eq. (4.8), the proposed feature selection algorithm was able to select the feature points which satisfied the Eq. (3.11) to reduce the scale estimation error to 0.14%. Fig. 4.10 shows $z^{i,j}$ vs $\sigma_m^{i,j}$ and examples of

TABLE 4.5: RMSE of keyframe positions (mm) in Experiment 1

	ORB-SLAM ONLY	ORB-SLAM with DfD
CB	34	18
C2	198	19
C3	361	36
C4	263	9

the keyframes in the other local regions.

Fig. 4.11 shows the camera poses corrected by the estimated scales. The results of ORB-SLAM only were scaled up with the true scale at C1 for the purpose of comparison. The RMSE of keyframe positions is shown in Table 4.5. All keyframes which can see the checkerboard within seven meters are used for calculating RMSE as

$$RMSE = \left(\frac{1}{m}\sum_{j=1}^{m} ||\mathbf{t}^j - \mathbf{t}_t^j||^2\right)^{\frac{1}{2}}, \tag{4.17}$$

where $\mathbf{t}^j$ is the translational components of the keyframe pose, $\mathbf{t}_t^j$ is its truth obtained from the checkerboard detection algorithm [65], and m is the number of keyframes. In the trajectory 2, RMSE of ORB-SLAM was around 300 mm. The proposed method was able to reduce the RMSE to below 40 mm. The results from this experiment demonstrated that the proposed method could correct the scale drift accurately from a monocular camera only. Also note that the proposed method could recover the scale of the environment correctly without any prior knowledge about the scene. In the trajectory 1, RMSE of ORB-SLAM was 34 mm and the result of the proposed method was 18 mm. These results show that the proposed method has a positive impact on SLAM results even when the scale drift is already relatively small.

Fig. 4.8 shows the box plot of the absolute errors between the estimated keyframe positions and the ground truth. As can be seen from Fig. 4.8, the proposed method could reduce the maximum, minimum and median errors in all local regions.

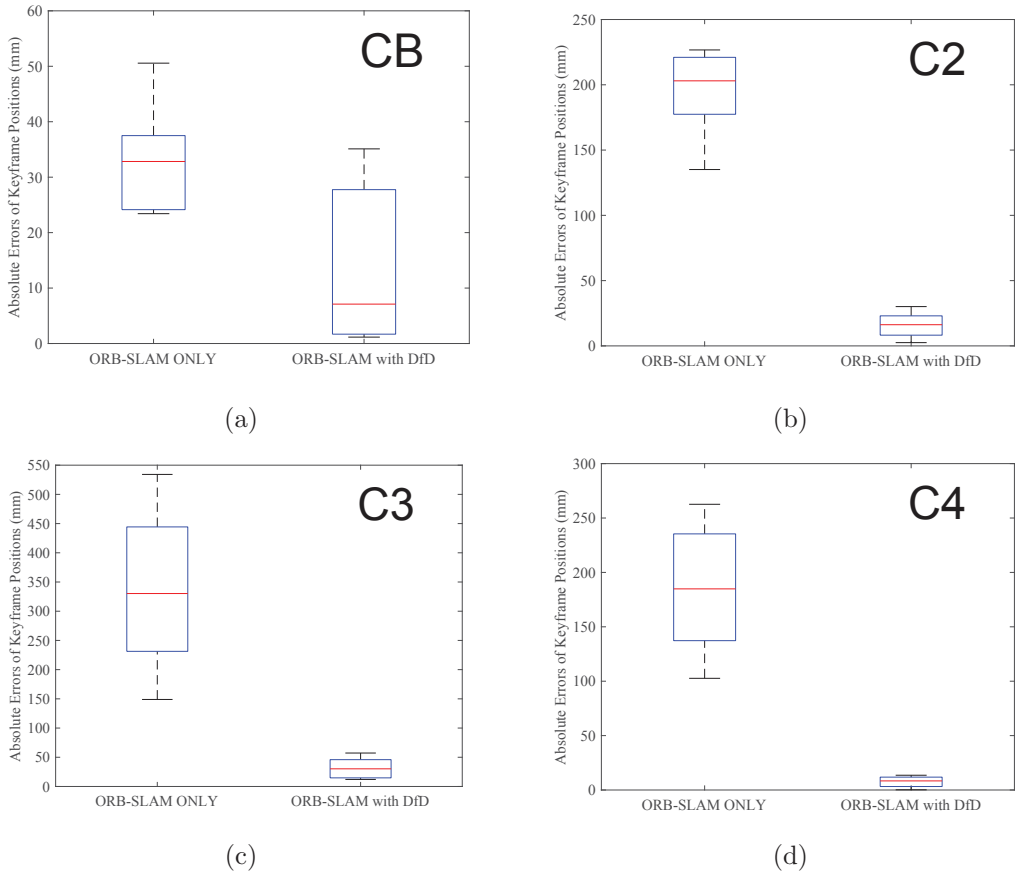


FIGURE 4.8: The box plot showing the absolute errors between the estimated keyframe positions and the ground truth in the local regions CB (a), C2 (b), C3(c), and C4(d). The box lengths indicate the interquartile range (first to third quartiles). The line in the center of the boxes indicates the median value. The whiskers down to the minimum and up to the maximum.

4.4.2 Experiment 2: Demonstration using a small camera

4.4.2.1 Dataset

A sequence of images with 640 × 480 pixels resolution at 30 fps was captured with the rear camera on an iPhone SE shown in Fig. 4.5, walking in an office environment shown in Fig. 4.12(a). The map and camera trajectories generated by ORB-SLAM from the dataset are shown in Fig. 4.12(b). In Fig. 4.12(b), the camera moved in the direction of the arrows and made a closed loop. Checkerboard patterns were placed at the locations

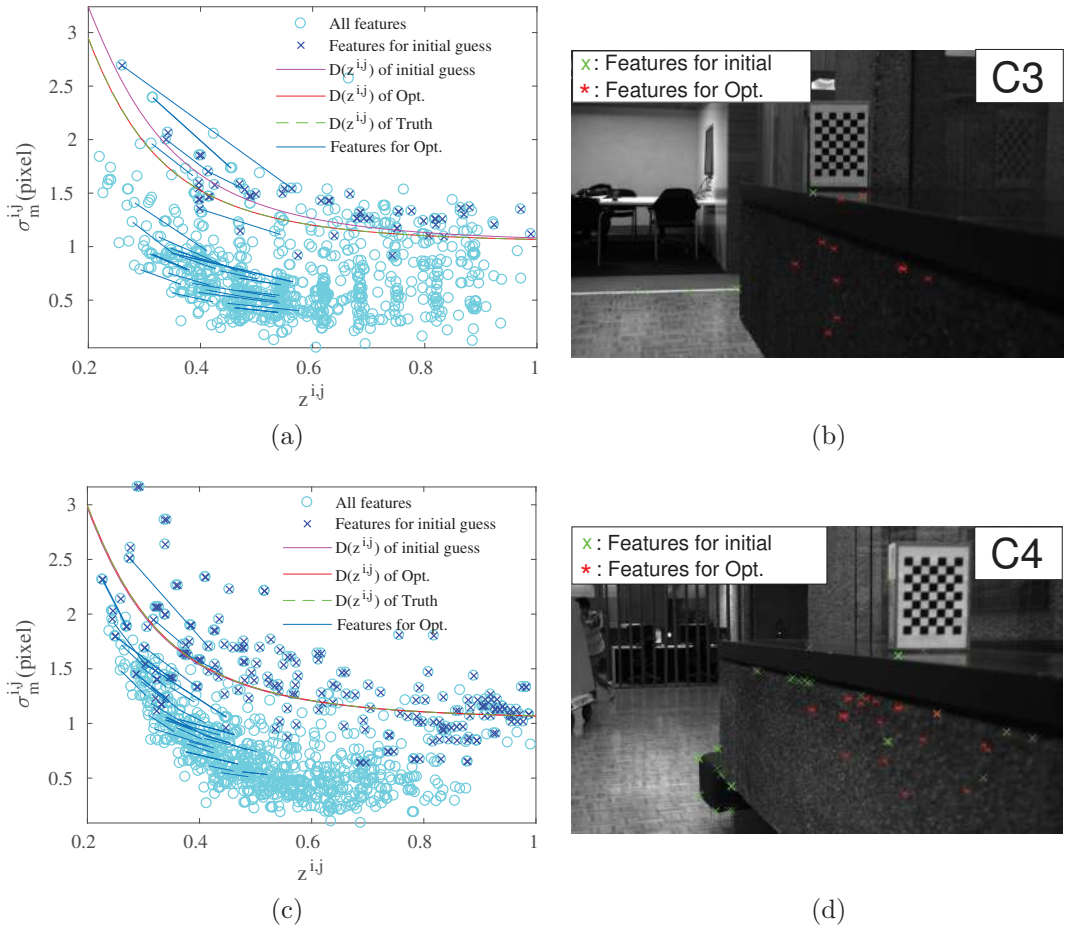


FIGURE 4.9: $z^{i,j}$ vs $\sigma_m^{i,j}$ in local regions C3 (a) and C4 (c), and the examples of keyframes in C3 (b) and C4 (d). In (a) and (c), the cyan o's show all feature points, the blue x's show the feature points selected for the initial guess. Each blue line connects the same feature for different keyframes, which is selected for the optimization. The magenta, orange, and green lines show the approximations by $\sigma^{i,j} = D(z^{i,j})$ as results of the initial guess, the optimization, and the truth. In (b) and (d), the green x's show the feature points selected for the initial guess, and the red $*$'s show the feature points selected for the optimization.

To be fair, feature points on the checkerboards were excluded for the optimizations.

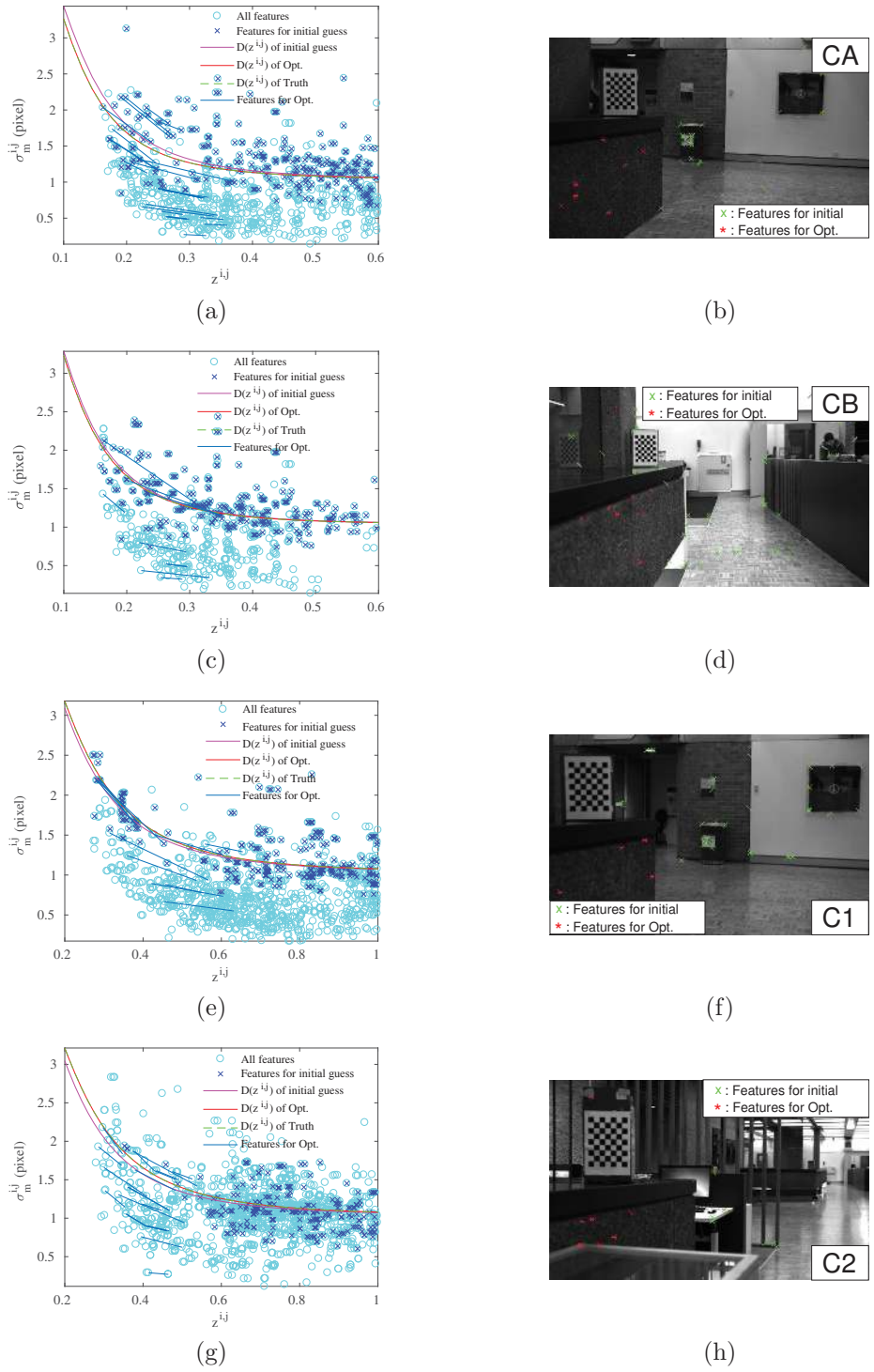


FIGURE 4.10: $z^{i,j}$ vs $\sigma_m^{i,j}$ in local regions (CA(a), CB(c), C1(e), C2(g)) and the examples of keyframes (CA(b), CB(d), C1(f), C2(h)). In (a), (c), (e), and (g), the cyan o's show all feature points, the blue x's show the feature points selected for the initial guess. Each blue line connects the same feature for different keyframes, which is selected for the optimization. The magenta, orange, and green lines show the approximations by $\sigma^{i,j} = D(z^{i,j})$ as results of the initial guess, the optimization, and the truth. In (b), (d), (f), and (h), the green x's show the feature points selected for the initial guess, and the red $*$'s show the feature points selected for the optimization. To be fair, feature points on the checkerboards were excluded for the optimizations.

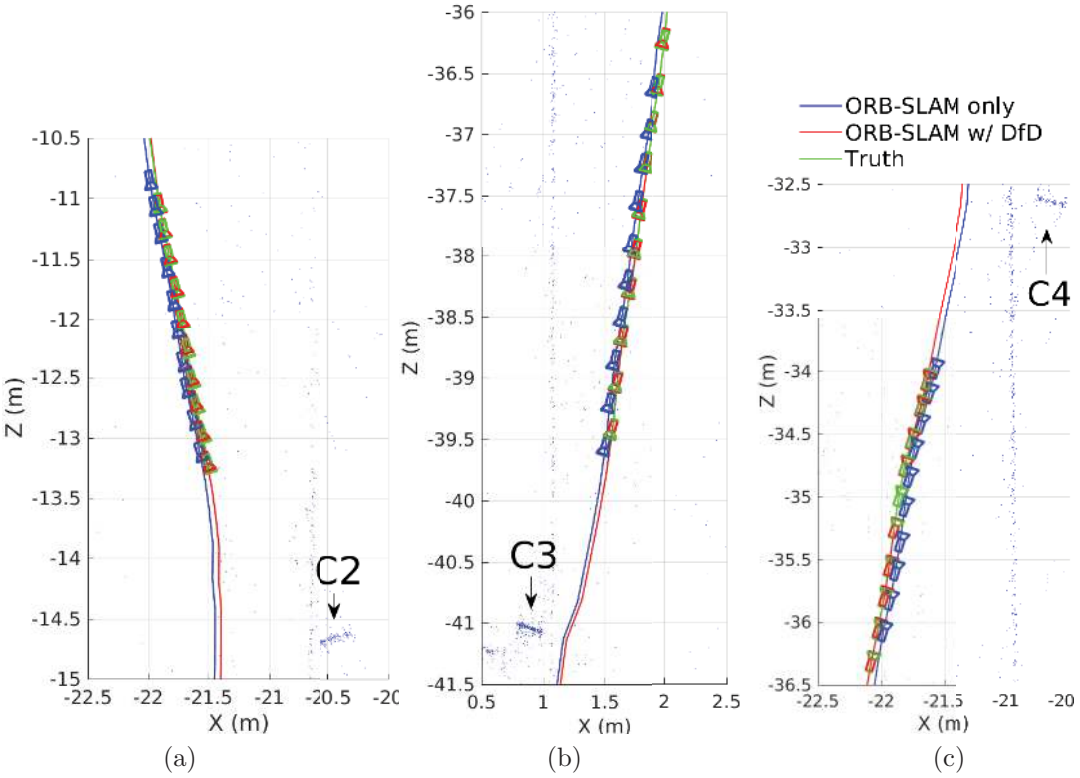
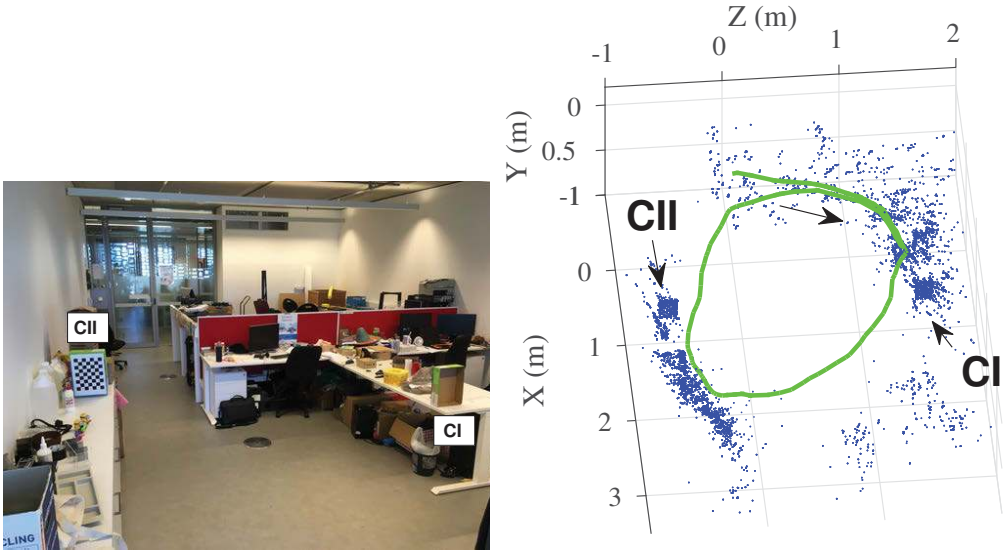


FIGURE 4.11: The map and camera poses reconstructed by the estimated scale in C2 (a), C3 (b), and C4 (c). The blue lines show the trajectory generated by ORB-SLAM. The red lines show the trajectory corrected by the estimated scales. The green lines show ground truth obtained from the checkerboard detection algorithm. The point clouds indicated by arrows are the map points on the corresponding checkerboards.

CI and CII so that the true scale in each local region could be computed. The scale drift in the local region around CII was about 1.1%.

4.4.2.2 Scale estimation result

Table 4.6 shows the results of optimization. The final estimation errors in the scale estimates are less than 0.08%. Fig. 4.13(a) and (c) show $z^{i,j}$ vs $\sigma_m^{i,j}$ in ten keyframes in each local region around CI and CII, respectively. As can be seen, adequate amounts of defocus blur were observed and the proposed algorithm could select the good feature points for optimization properly in the environment. The results of RMSE are shown in Table 4.7. The absolute errors are shown in Fig. 4.14. The effective measuring range for this small camera was only about 300 mm. This result demonstrated that the proposed algorithm was effective even on this small phone camera when at least some of the feature points



(a) (b)

FIGURE 4.12: The map and camera poses generated by using iPhone SE. (a) shows the office environment. (b) is the map and the camera trajectory reconstructed by iPhone SE. In (b), the green line is the trajectory and blue dots are the map points generated by ORB-SLAM.

TABLE 4.6: Error in Scale Estimate in Each Area (%) in Experiment 2

	Initial Guess	Optimization
CI	1.00	0.03
CII	7.18	0.08

TABLE 4.7: RMSE of keyframe positions by iPhone SE (mm) in Experiment 2

	ORB-SLAM ONLY	ORB-SLAM with DfD
CII	31	24

were observed within the effective measuring range. The parameters used for DfD are shown in Table 4.8. The threshold values for optimization are shown in Table 4.9.

4.5 Discussion

A problem of non-linear least squares optimization is that the estimates may converge to local minima different from the true solution when the objective function has multiple

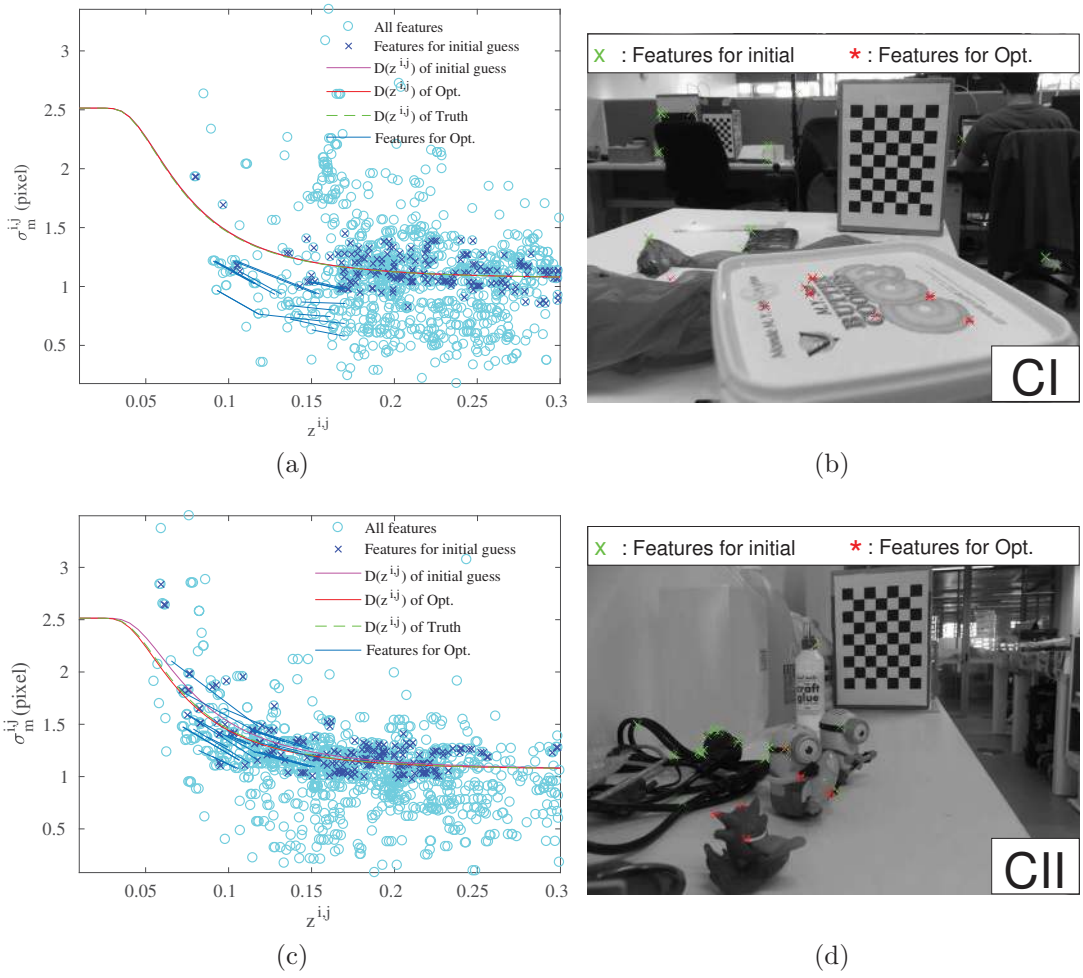


FIGURE 4.13: $z^{i,j}$ vs $\sigma_m^{i,j}$ in local regions CI (a) and CII (c), and the example of keyframes in CI (b) and CII(d). In (a) and (c), the cyan o's show all feature points, the blue x's show the feature points selected for the initial guess. Each blue line connects the same feature for different keyframes, which is selected for the optimization. The magenta, orange, and green lines show the approximations by $\sigma^{i,j} = D(z^{i,j})$ as results of the initial guess, the optimization, and the truth. In (b) and (d), the green x's show the feature points selected for the initial guess, and the red $*$'s show the feature points selected for the optimization. To be fair, feature points on the checkerboards were excluded for the optimizations.

local minima. The results are sensitive to the initial values. To obtain a good initial guess, a staged strategy with a feature selection algorithm was proposed. First, a set of feature points with sharp edges was selected to minimize the impact of blur texture ambiguity and simplify the optimization problem given by Eq. (4.8) to Eq. (4.9). Also, the estimated initial guess was used to remove the outliers and select a set of good feature points to solve the optimization problem defined by Eq. (4.8). The several experimental results in Section 4.4 demonstrated that the proposed algorithm was effective to estimate suitable

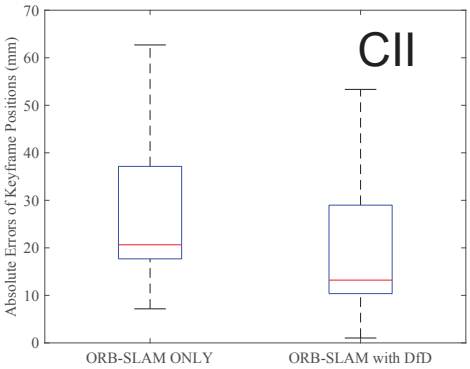


FIGURE 4.14: The box plot showing the absolute errors between the estimated keyframe positions and the ground truth in the local region CII.

TABLE 4.8: Parameters used in DfD for Experiment 2

ϕ_1	ϕ_2	ϕ_3	b_f[mm]	f[mm]	d_f [mm]	N_c
-0.694	0.0400	2.51	5.2	5.1	800	1.4

TABLE 4.9: Threshold values used in the optimization for Experiment 2

e_{thl}	e_{thh}	r_{thl}	r_{thh}	z_{th}
0.07	0.09	0.8	1.2	0.33 d_f/Λ_{ini}

initial guess and resulted in the accurate metric scale estimate.

Some criteria to select feature points to be incorporated into computing the objective functions defined by Eq. (4.8) and (4.9) were introduced. The parameters of e_{thl} and e_{thh} depend on the camera and lens, and thus can be determined by using the binary edge chart shown in Fig. 4.4 in advance. The value of z_{th} is systematically decided with the initial guess obtained from Eq. (4.9). On the other hand, the parameters of r_{thl} and r_{thh} do not have any choice but to be empirically determined. This issue relates to how change in texture happens through the image sequence, which was discussed in Section 3.5. A further investigation how to systematically determine the parameters of r_{thl} and r_{thh} will be conducted as a future work.

The effective measuring range of DfD depends on the lens, especially its focal length and aperture size. For the camera used in Section 4.4, this is approximately three meters.

The scale estimation is not feasible if all the visible feature points fall outside the effective measuring range of the DfD technique. This limitation makes the proposed method difficult to be used in a large scale environment. For example, defocus blur is not significant in the KITTI dataset [67] which is a well-known public dataset for automotive scenes. Due to the outdoor scenes where feature locations in the environment are at a relatively large distance from the camera, sufficient number of feature points cannot be observed within the effective measuring range. The same is true in the TUM dataset [68], which is a well-known public dataset for indoor scenes. The short focal length of the camera used in the dataset limits the effective measuring range to very short. Therefore, the lens and camera properties need to be selected to suit a given application scenario. The strength of this method combining DfD and SLAM is that it is not necessary to estimate depth to all feature points in a scene. A small number of feature points nearby the camera is enough to estimate an accurate metric scale of the scene. As shown in Subsection 4.4.2, the metric scale can be estimated even on a small phone camera when at least some of the feature points were observed within the effective measuring range. Therefore, it still has a potential to be used for many applications.

Although it did not appear in the experiment shown in Section 4.4, a possible failure scenario relates to the ability to obtain a suitable initial guess to the optimization problem defined by Eq. (4.8). If sufficiently sharp edges within the measuring range are not available, the initial guess may be too poor, and the method may converge to a local minimum.

The proposed algorithm only relies on the local accuracy of the underlying SLAM algorithm. Although the sale drift lowers the global accuracy of the map generated by monocular SLAM, the effect is small in the local map. It can be assumed that the local accuracy of the SLAM algorithm is good enough to estimate the metric scale precisely by solving the optimization problem defined by Eq. (4.8) even without the loop-closure. Therefore the proposed algorithm could be used before or after loop-closure.

Obtaining the sparse defocus map of an image with 752 $\times$ 480 pixels resolution used in the experiment shown in Subsection 4.4.1 takes about 0.4 seconds in MATLAB® with Intel® Core™ i5-6300U CPU at 2.40GHz $\times$ 4. The optimization including the initial

guess estimation needs about 0.3 seconds in MATLAB®. It is expected that with an efficient implementation in C, these times could be substantially reduced. However, it is important to note that the proposed technique is a post-processing step and therefore does not influence the real-time operation of the underlying SLAM algorithm.

4.6 Conclusion

This chapter described a method for correcting scale drift in monocular SLAM with the aid of DfD and illustrated it using the ORB-SLAM algorithm. Using the amount of defocus blur estimated on ORB feature points together with the map points and keyframe poses obtained from ORB-SLAM, the metric scales in selected local regions were estimated. The impact of motion blur in DfD was expressed using the optical flow, the edge direction at the feature, and the ratio of the exposure time to the frame period. The focal plane ambiguity problem in DfD was avoided using non-linear least squares optimization without inverting the Depth-Defocus function which causes multiple depth solutions across the focal plane. Eliminating the motion blur effect and avoiding the focal plane ambiguity by using the output of monocular SLAM makes the proposed method suitable for robotic applications with dynamic motion on a large field. Two experiments were conducted with two different cameras to evaluate the proposed algorithm. The first experiment demonstrated that the proposed algorithm was effective to recover the scale factor for SLAM and eliminate the effect of scale drift in a corridor environment. In the second experiment, it was shown that even on a small phone camera, the proposed methodology was effective to estimate accurate metric scale in an office environment, when at least some of the feature points observed were within the measuring range.

In this work, the output from ORB-SLAM is post-processed through a non-linear optimization algorithm. Therefore, while the local maps are accurate, the global locations of these regions are not corrected for scale drift. Given that the scale drift is relatively small, it could be argued that accurate local maps are adequate for many practical applications.

Chapter 5

Conclusion

This thesis presented a metric scale monocular simultaneous localization and mapping framework based on depth from defocus. The findings are targeted towards replacing the large and costly conventional 3D vision-based systems, such as stereo-based or RGB-D based, with small, inexpensive, and versatile monocular-based systems. In spite of the advantage that DfD can estimate the depth of a scene without the need for additional devices or prior knowledge about the scene, this concept still has not found a practical use for scale estimation on monocular camera systems since the topic developed thirty years ago, due to the presence of several ambiguities. The proposed methodologies that use information generated by camera motion enabled the resolution of all the ambiguities in DfD. The first methodology for metric scale estimation with EKF was experimentally verified in a cluttered desk environment. The results demonstrated that a texture correction factor could express the extent of blur due to texture in DfD and the EKF framework could estimate the metric scale of the scene. The second methodology for monocular SLAM with optimization was evaluated on ORB-SLAM framework in datasets captured with an industrial camera walking in a corridor environment. The results demonstrated that all the ambiguities in DfD could be resolved by using information from SLAM and the estimated metric scale could correct the scale drift in arbitrary selected local regions. Also, it was shown that even on a small phone camera, the proposed methodology was effective to estimate accurate metric scale in an office environment, when at least some of the feature points observed were within the effective measuring range. The proposed monocular-based

SLAM system which can create an accurate metric scale map will contribute to developing small and inexpensive autonomous mobile robots.

5.1 Summary of Contributions

A correction factor for scale error caused by blur due to texture in DfD

Several experimental results revealed that the scale error caused by blur due to texture is mainly caused by the difference of the contrasts at edge locations on the texture. It turned out that the low-contrast edge makes the estimated blur amount larger than truth and this behavior is depth-invariant. In the experiment with edge charts and the more complex scene with a human face, a constant gain factor could express the extent of blur due to texture well. This factor was termed the texture correction factor. The proposed formulation enabled DfD to be used in practical environments surrounded by texture with weak edges.

An EKF framework with DfD for producing an accurate metric scale map of a scene

An EKF framework to estimate the scale factor of a scene and the texture correction factor for each feature was proposed. Information on non-scaled distances to feature locations and image velocities incorporated in EKF enabled the estimation of the texture correction factors and the accurate metric scale of a scene. At first, the effectiveness of the EKF framework was evaluated with the simple edge chart. Then, the algorithm was evaluated in a more challenging cluttered desk environment and validated.

A method for eliminating the impact of motion blur in DfD

It was found that the amount of motion blur could be expressed by using the optical flow, the edge direction at the feature, and the ratio of the exposure time to the frame period. The experiment using the tilted edge chart and checkerboard demonstrated that

the amount of motion blur was estimated by using the proposed formulation and the impact on the defocus estimation could be clearly eliminated. The proposed motion blur elimination method enabled DfD to be used in a dynamic scene observed from a moving camera.

A technique to avoid the focal plane ambiguity in DfD

The inverse of the Depth-Defocus function yields multiple depth solutions across the focal plane. To avoid this, non-linear optimization was formulated without inverting the Depth-Defocus function. The experimental results demonstrated that the optimization problem could be solved without the typical assumption that the focus point was put on the nearest or farthest point of a scene. The avoidance of focal plane ambiguity made the proposed algorithm suitable for robotic applications needing a wide-range observation.

A non-linear optimization with DfD to post-process output of SLAM for eliminating scale drift

A non-linear least squares optimization problem to integrate DfD with monocular SLAM was formulated. Incorporating the defocus blur estimates together with the output keyframe and feature location estimates generated by a monocular SLAM algorithm into the optimization enabled the estimation of the accurate metric scale in an environment. A staged strategy with feature selection algorithm to avoid the possibilities that the solutions converge to local minima different from the true solution was proposed. The experimental results demonstrated that a good initial guess for the optimization could be derived using the selected feature points with sharp edges. Also, it was seen that the selected feature points within the effective measuring range made it possible to solve the optimization problem and estimate an accurate metric scale. As a result of scale drift elimination, the accuracy of the map generated by ORB-SLAM was improved in the experiments.

5.2 Discussion of Limitations and Future Work

There are several limitations of the framework based on DfD discussed in this thesis. In Chapter 3, the texture correction factor to express the extent of blur due to texture in DfD was proposed. However, the assumption that the texture correction factor is depth-invariant does not hold when there is a significant change in texture or illumination during the image sequence. For example, it was seen in Subsection 3.4.2 that although the texture at the spine of a book appeared as a single edge when the camera was at a distance, the rich texture due to letters of the book appeared when the camera was nearby. In that case, the depth-invariant correction factor does not work well. To address this issue, noise to the constraint function was added in EKF. The experiment result demonstrated that the additive noise could release the constraint that the texture correction factor was constant. An alternative strategy is to select feature points that are expected to have constant texture correction factors, which was presented in Section 4.3. As a future work, the usage of a texture discrimination algorithm based on a machine learning could be considered. Learning and identifying texture in a scene will be effective for selecting feature points to estimate accurate metric scale. Also, analyzing the impact of the different type of texture in DfD may give a hint on how to correct the extent of blur due to a complex texture to which the texture correction factor cannot be applied.

The effective measuring range of DfD depends on the lens, especially its focal length and aperture size. The scale estimation is not feasible if all the visible feature points fall outside the effective measuring range. This limitation makes the proposed method difficult to be used in a large scale environment. For an outdoor scene such as an automotive scene, the maximum range of over five meters will be at least required to estimate the scale of the scene. The lens to achieve this should have longer than 28 millimeters of focal length and smaller than 1.4 of f-number. However, such a long focal length makes the field of view narrow, making the lens unsuitable for its intended purpose of observing the environment. In addition, the staged strategy presented in Chapter 4 requires feature points with sharp edges within the range. If a sufficient number of feature points which satisfy the selection criteria is not observed, the obtained initial guess may be too poor, and the optimization result may fall into a local minimum different from the true solution.

One possible solution is to combine with another scale estimation method, such as using geometrical constraints in a scene or using information from IMU introduced in Chapter 2. Although it compromises the advantage of the proposed method without the need for any scene limitations or additional sensors, the combination with different scale estimation algorithm has a potential not only to extend the measuring range but also to improve the accuracy of the scale estimation. Investigating the effectiveness of the combination of the proposed algorithm and another scale estimation method is an avenue for future work.

The optical flow based motion blur elimination algorithm was presented in Chapter 4. The experiment using the tilted edge chart and checkerboard confirmed that the impact of the motion blur on the defocus estimation could be clearly eliminated. However, the motion of the chart in this experiment was only horizontal. A further investigation will be required to clarify to what extent this model is applicable to the general rotational and translational camera motions. The use of IMU to investigate the relationship between camera motion and motion blur is a fruitful avenue for future work.

The algorithm proposed in Chapter 4 uses a separate post-processing optimization to the monocular SLAM optimization. This approach results in a loosely-coupled system. Although the proposed approach enables the estimation of the scale of arbitrary selected local regions in a map generated by a monocular SLAM algorithm, it cannot generate the entire global map with a metric scale. Another possibility is to develop a tightly-coupled system where DfD is jointly incorporated within the monocular SLAM optimization process. Adding the constraints of DfD to the monocular SLAM bundle adjustment optimization has a potential to improve the accuracy of camera pose and feature location estimates in monocular SLAM and to result in an accurate global metric map. Future work will focus on exploring the effectiveness of the tightly coupled strategy.

Bibliography

[1] M. Hagele. Robots conquer the world [turning point]. *IEEE Robotics & Automation Magazine*, 23(1):116–118, 2016.

[2] G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (6):1052–1067, 2007.

[4] H. Strasdat, J. Montiel, and A. J. Davison. Scale drift-aware large scale monocular SLAM. *Robotics: Science and Systems VI*, 2, 2010.

[5] M. Subbarao and G. Surya. Depth from defocus: a spatial domain approach. *International Journal of Computer Vision*, 13(3):271–294, 1994.

[6] S. Zhuo and T. Sim. Defocus map estimation from a single image. *Pattern Recognition*, 44(9):1852–1858, 2011.

[7] S. Liu, F. Zhou, and Q. Liao. Defocus map estimation from a single image based on two-parameter defocus model. *IEEE Transactions on Image Processing*, 25(12): 5943–5956, 2016.

[8] A. Sellent and P. Favaro. Which side of the focal plane are you on? In *2014 IEEE International Conference on Computational Photography (ICCP)*, pages 1–8, 2014.

[9] J. Lin, X. Ji, W. Xu, and Q. Dai. Absolute depth estimation from a single defocused image. *IEEE Transactions on Image Processing*, 22(11):4545–4550, 2013.

[10] V. Srikakulapu, H. Kumar, S. Gupta, and K. S. Venkatesh. Depth estimation from single image using defocus and texture cues. In *2015 5th National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG)*, pages 1–4, 2015.

[11] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5): 1255–1262, 2017.

[12] 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)*, pages 2564–2571. IEEE, 2011.

[13] S. Dai and Y. Wu. Motion from blur. In *2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[14] E. Alexander, Q. Guo, S. Koppal, S. Gortler, and T. Zickler. Focal flow: Measuring distance and velocity with defocus and differential motion. In *European Conference on Computer Vision*, pages 667–682, 2016.

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[16] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, 2004.

[17] E. S. Jones and S. Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, 30(4):407–430, 2011.

[18] C. Paramanand and A. N. Rajagopalan. Inferring image transformation and structure from motion-blurred images. In *British Machine Vision Conference (BMVC)*, pages 73.1–73.12, 2010.

[19] C. Paramanand and A. N. Rajagopalan. Shape from sharp and motion-blurred image pair. *International journal of computer vision*, 107(3):272–292, 2014.

[20] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard. 3-D mapping with an RGB-D camera. *IEEE Transactions on Robotics*, 30(1):177–187, 2014.

[21] J. Gräter, T. Schwarze, and M. Lauer. Robust scale estimation for monocular visual odometry using structure from motion and vanishing points. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 475–480, 2015.

[22] Z. Dingfu, Y. Dai, and L. Hongdong. Reliable scale estimation and correction for monocular visual odometry. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 490–495, 2016.

[23] S. Song, M. Chandraker, and C. C. Guest. High accuracy monocular SFM and scale correction for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):730–743, 2016.

[24] N. Fanani, A. Stürck, M. Barnada, and R. Mester. Multimodal scale estimation for monocular visual odometry. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1714–1721, 2017.

[25] T. Botterill, S. Mills, and R. Green. Correcting scale drift by object recognition in single-camera SLAM. *IEEE Transactions on Cybernetics*, 43(6):1767–1780, 2013.

[26] D. P. Frost, O. Kähler, and D. W. Murray. Object-aware bundle adjustment for correcting monocular scale drift. In *Proceedings of 2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4770–4776, 2016.

[27] R. Zhang, P. Tsai, J. E. Cryer, and M. Shah. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 21(8):690–706, 1999.

[28] R. Basri, D. Jacobs, and I. Kemelmacher. Photometric stereo with general, unknown lighting. *International Journal of computer vision*, 72(3):239–257, 2007.

[29] C. H. Esteban, G. Vogiatzis, and R. Cipolla. Multiview photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):548–554, 2008.

[30] S. K. Nayar and Y. Nakagawa. Shape from focus. *IEEE Transactions on Pattern analysis and machine intelligence*, 16(8):824–831, 1994.

[31] C. Wöhler, P. d'Angelo, L. Krüger, A. Kuhl, and H.-M. GroB. Monocular 3D scene reconstruction at absolute scale. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64(6):529–540, 2009.

[32] A. N. Rajagopalan and S. Chaudhuri. Optimal selection of camera parameters for recovery of depth from defocused images. In *Proceedings of 1997 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 219–224, 1997.

[33] P. Favaro and S. Soatto. A geometric approach to shape from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):406–417, 2005.

[34] Y. Y. Schechner and N. Kiryati. Depth from defocus vs. stereo: How different really are they? *International Journal of Computer Vision*, 39(2):141–162, 2000.

[35] A. P. Pentland. A new sense for depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(4):523–531, 1987.

[36] J. H. Elder and S. W. Zucker. Local scale control for edge detection and blur estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7): 699–716, 1998.

[37] D. Chen, H. Chen, and L. Chang. Fast defocus map estimation. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3962–3966. IEEE, 2016.

[38] M. Watanabe and S. K. Nayar. Minimal operator set for passive depth from defocus. In *1996 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 431–438, 1996.

[39] M. Watanabe and S. K. Nayar. Rational filters for passive depth from defocus. *International Journal of Computer Vision*, 27(3):203–225, 1998.

[40] M. Watanabe, S. K. Nayar, and M. N. Noguchi. Real-time computation of depth from defocus. In *SPIE the International Society for Optical Engineering*, pages 14–25, 1996.

[41] C. Zhou, S. Lin, and S. K. Nayar. Coded aperture pairs for depth from defocus and defocus deblurring. *International journal of computer vision*, 93(1):53–72, 2011.

[42] B. Bascle, A. Blake, and A. Zisserman. Motion deblurring and super-resolution from an image sequence. *Computer Vision-ECCV'96*, pages 571–582, 1996.

[43] C. Paramanand and A. N. Rajagopalan. Depth from motion and optical blur with an unscented kalman filter. *IEEE Transactions on Image Processing*, 21(5):2798–2811, 2012.

[44] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17 (1-3):185–203, 1981.

[45] H. Kumar, S. Gupta, and K. S. Venkatesh. Resolving focal plane ambiguity in depth map creation from defocus blur using chromatic aberration. In *Proceedings of 2015 10th International Conference on Information, Communications and Signal Processing (ICICS)*, pages 1–5, 2015.

[46] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *2003 9th IEEE International Conference on Computer Vision (ICCV)*, page 1403, 2003.

[47] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.

[48] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[49] JM M. Montiel, J. Civera, and A. J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Robotics: Science and Systems*, 2006.

[50] J. Civera, A. J. Davison, and JM M. Montiel. Inverse depth parametrization for monocular SLAM. *IEEE transactions on robotics*, 24(5):932–945, 2008.

[51] L. A. Clemente, A. J. Davison, I. D. Reid, J. Neira, and J. D. Tardós. Mapping large loops with a single hand-held camera. In *Robotics: Science and Systems*, volume 2, 2007.

[52] H. Strasdat, J. Montiel, and A. J. Davison. Real-time monocular SLAM: Why filter? In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2657–2664, 2010.

[53] F. Dellaert and M. Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006.

[54] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 225–234, 2007.

[55] M. I. Lourakis and A. A. Argyros. SBA: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)*, 36(1):2, 2009.

[56] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. *Computer Vision-ECCV 2006*, pages 430–443, 2006.

[57] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. *Computer Vision-ECCV 2010*, pages 778–792, 2010.

[58] R. Mur-Artal and J. D. Tardós. Visual-inertial monocular SLAM with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[59] C. Wöhler. *3D computer vision: efficient methods and applications*. Springer Science & Business Media, 2012.

[60] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[61] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of 7th international joint conference on Artificial intelligence*, pages 674–679, 1981.

[62] J. Shi and C. Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.

[63] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An invitation to 3-d vision: from images to geometric models*. Springer, 2003.

[64] S. J. Julier and J. J. LaViola. On kalman filtering with nonlinear equality constraints. *IEEE Transactions on Signal Processing*, 55(6):2774–2784, 2007.

[65] A. Geiger, F. Moosmann, Car Ö, and B. Schuster. Automatic camera and range sensor calibration using a single shot. In *Proceedings of 2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3936–3943, 2012.

[66] T. F. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445, 1996.

[67] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.

[68] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580, 2012.