Eliminating Scale Drift in Monocular SLAM using Depth from Defocus

Tomoyuki Shiozaki and Gamini Dissanayake

***Abstract*—This paper presents a novel approach to correct errors caused by accumulated scale drift in monocular SLAM. It is shown that the metric scale can be estimated using information gathered through monocular SLAM and image blur due to defocus. A nonlinear least squares optimization problem is formulated to integrate depth estimates from defocus to monocular SLAM. An algorithm to process the output keyframe and feature location estimates generated by a monocular SLAM algorithm to correct for scale drift at selected local regions of the environment is presented. The proposed algorithm is experimentally evaluated by processing the output of ORB-SLAM [1] to obtain accurate metric scale maps from a monocular camera without any prior knowledge about the scene.**

***Index Terms*—Range Sensing, SLAM, Visual-Based Navigation.**

I. INTRODUCTION

MONOCULAR simultaneous localization and mapping (SLAM) [2] enables a mobile robot to map its environment and estimate its egomotion up to a scale. Ideally the scale, which defines the relationship between the estimated geometry and the metric map, while unknown, should stay constant. It has been recognized that many monocular SLAM algorithms are prone the scale drift, where the scale is different in different parts of the map [3]. State-of-the-art monocular SLAM systems, for example, ORB-SLAM [1] have ability to reuse the map, detect loops to close, and perform global optimization to minimize the error caused by the accumulated scale drift. However, the scale drift can still occur, especially when the camera turns quickly [4]. Although stereo cameras [5] or RGB-D cameras [6] can resolve the scale ambiguity problem, the small size and the versatility of monocular cameras are still attractive, particularly in robotic applications.

Depth from defocus (DfD) [7] is one of the approaches that can be used to estimate scale from information gathered from a monocular camera. It relies on the amount of defocus blur which is related to the distance to the scene [8]. In our previous work [9], we demonstrated that combination of DfD and image velocity in an extended Kalman filter (EKF) framework was able to estimate the metric scale of an environment accurately. In this paper, we leverage this work to integrate DfD with monocular SLAM systems.

Manuscript received: September, 9, 2017; Accepted October, 24, 2017.

This paper was recommended for publication by Editor C. Stachniss upon evaluation of the Associate Editor and Reviewers' comments.

The authors are with the Centre for Autonomous Systems, Faculty of Engineering and IT, University of Technology Sydney (UTS), NSW 2007, Australia. Email: shiozaki.tomoyuki@student.uts.edu.au; gamini.dissanayake@uts.edu.au.

Digital Object Identifier (DOI): see top of this page.

We begin with ORB-SLAM [1], which is one of the best monocular SLAM systems currently available. This system uses ORB features to represent the environment and select a set of frames (keyframes) for representing the camera poses. Our method estimates the amount of defocus blur at ORB features and uses this information to extract the metric scale from the map and keyframe poses generated by ORB-SLAM. The algorithm can be selectively applied to local regions of the map to correct the ORB-SLAM output to minimize the impact of scale drift. We note here that while the experimental evaluations presented make use of the output from ORB-SLAM, the proposed algorithm can be used to enhance the output from any keyframe or feature based monocular SLAM algorithm.

The main contributions of this paper are as follows:

- Use of DfD to reliably estimate the metric scale to feature locations observed from a given keyframe
- An algorithm based on nonlinear optimization to post-process keyframe pose and feature location estimates generated by a monocular SLAM algorithm to obtain the metric scale
- Experimental demonstration of the proposed method with ORB-SLAM

In section II, a review of related work on scale drift in monocular SLAM and DfD is described. Section III introduces the DfD method. Section IV presents the optimization algorithm for computing the metric scale. In section V, experimental results are presented. Section VI discusses and concludes the paper.

II. RELATED WORK

A. Scale Drift on Monocular SLAM

One popular approach for mitigating scale ambiguity in monocular SLAM is to impose geometrical constraints. For example, a fixed height of the camera above the ground plane is useful to estimate scale and therefore avoid scale drift [10], [11], [12], [13]. Alternatively, the size of known objects in the environment can be used as a depth cue [14], [15]. The main drawback of these methods is that they are effective in only limited scenes: on roads or environments populated with known objects.

When such additional information is not available, monocular SLAM is known to suffer from scale drift. Clemente et al. [16] proposed a filtering-based method to detect loops automatically to correct scale drift in large environments. Strasdat et al. [3] presented a scale drift-aware loop closing strategy using a keyframe-based pose-graph optimization to

build a consistent, global map. Mur-Artal et al. [1] proposed ORB-SLAM that achieves real-time tracking, mapping, and loop closing in large environments based on the pose-graph optimization algorithm. Potential for scale drift in ORB-SLAM has been recognized, and it has been shown that using a stereo or an RGBD-camera, ORB-SLAM2 [4] provides a good solution to this problem. In [17], a Visual-Inertial ORB-SLAM that uses information from an inertial measurement unit (IMU) to recover metric scale was presented. The focus of this paper is an alternative strategy based on DfD to eliminate scale drift. Our approach using DfD does not need any geometrical constraints or known structure in the environment and thus has a potential to enhance the performance of monocular SLAM in a broad range of applications.

B. Depth from Defocus

Conventional DfD methods required multiple images with different defocus levels [18], [19], thus were not especially attractive in many applications. Pentland [20] pointed out that defocus information can be extracted even from a single image provided that there are sharp edges. Elder and Zucker [21] applied the derivatives of the input image to find edges and estimate the blur amounts. Zhuo and Sim [22] used the Gaussian gradient ratio of input and reblured images that is robust to noise.

However, single image DfD methods suffer from ambiguities due to focal plane, motion blur, and texture. First, objects placed in front and behind the focal plane may be viewed with exactly the same amount of defocus blur [23]. Kumar et al. [24] used the chromatic aberration as a clue to solve the focal plane ambiguity. Second, in dynamic scenes both depth and motion contribute to the blur. Punnappurath et al. [25] developed a deep convolutional neural network for decoupling of motion and defocus blur. Third, given a single image, it cannot be differentiated whether a blur is caused by defocus or texture [22]. In our previous work [9], we demonstrated that the blur due to texture could be represented using a constant correction factor and an EKF framework was able to produce accurate metric reconstruction.

The method proposed in this paper makes use of the constant correction factor proposed in our previous work. In order to integrate with keyframe-based SLAM systems, nonlinear least squares optimization is used to obtain the metric reconstruction. Furthermore, through the use of the information gathered by monocular SLAM, ambiguities due to focal plane and motion blur can also be avoided, making the proposed algorithm suitable for robots applications.

III. Depth from Defocus

We assume that the image formation obeys the thin lens model [20] illustrated in Fig. 1. When the object located at out-of-focus distance d, a point on it is viewed with a blurred circle c at the image plane. This circle is so-called the circle of confusion (CoC), and the diameter can be written as

$$c = \frac{|d - d_f|}{d} \frac{f^2}{N(d_f - f)}, \tag{1}$$

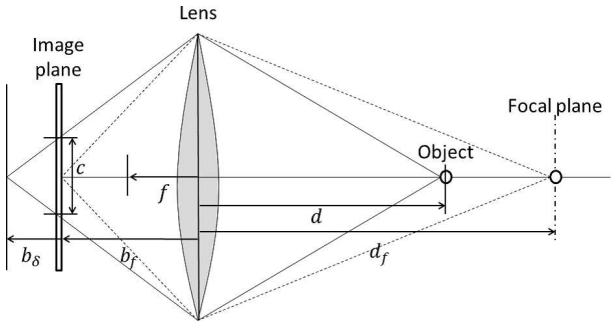


Fig. 1. Thin lens model. Reprinted from [9]. The size of c depends on the image plane distance b_f and the focal plane distance d_f. When the image plane is located at $b_f + b_\delta$, the object placed at d is best focused.

where d_f is in-focus distance, f and N are the focal length and the f-number of the camera, respectively [22]. Large $|d - d_f|$ makes CoC large. Since f-number is given as $N = f/A$ with the aperture diameter of the lens A, the blur effect is most obvious with a large lens.

The radius of σ of the Gaussian-shaped point spread function (PSF) $G(\sigma)$ can approximate the size of c as

$$c^i = \gamma \sigma^i, \tag{2}$$

$$I^i = G(\sigma^i) * I^i_f, \tag{3}$$

where subscript i is used to denote the i-th feature in an image, I^i is a small region of interest (ROI) around the feature, I^i_f is the ROI around the same feature when this feature is best focused, and $*$ indicates the convolution operation [8]. γ is a camera-specific value [26]. For estimating the value of σ^i, we use the method proposed by Zhuo and Sim [22].

The Depth-Defocus function [8] which expresses the relationship between d^i and σ^i is

$$\begin{aligned} \sigma^i = D(d^i) &= \frac{1}{\phi_1} \exp\left(-\frac{1}{\phi_2}(b_\delta(d^i))^2\right) + \phi_3, \\ b_\delta(d^i) &= \frac{d^i f}{d^i - f} - b_f, \end{aligned} \tag{4}$$

where ϕ_1, ϕ_2, and ϕ_3 are the calibration parameters. b_f is the image plane distance. The calibration process performed by measuring σ^i at the corners on the checkerboard while changing its position can generate these parameters along with f and b_f. Solving Eq. (4) yields d^i from σ^i.

However, measuring σ^i does not work well as expected on features other than those with sharp edges due to the blur texture ambiguity [8]. In our previous work [9], we found that one of the main causes was the difference of the contrast between the ROIs and demonstrated that this error could be expressed by the following equation:

$$\sigma^i_m = \lambda^i \sigma^i_t = \lambda^i D(d^i), \tag{5}$$

where σ^i_m is the measured σ^i and σ^i_t is the true σ^i measured at a sharp edge. λ^i describes the constant correction factor for the extent of blur due to texture.

Furthermore, motion blur influences the blur estimate. We propose a method to eliminate the effect of motion blur from estimated defocus blur by using optical flow. Eq. (3) is rewritten as

$$I^i = G(\sigma_m^i) * G(\sigma_b^i) * I_f^i, \tag{6}$$

where σ_b^i is the motion blur kernel and σ_m^i is from Eq. (5). The composite blur is

$$\sigma_{mb}^i = \sqrt{{\sigma_m^i}^2 + {\sigma_b^i}^2}. \tag{7}$$

In [27], the motion blur vector is expressed as $\mathbf{b} = (l\cos\theta, l\sin\theta)^T$, where $l = 2\sigma_b^i$ and θ is the edge direction. Here, the motion blur vector can be expressed by the optical flow vector $\mathbf{u} = (u,v)^T$ as $\mathbf{b} = \frac{T_e}{T_f}\mathbf{u}$, where T_e and T_f are the exposure time and the frame period of the camera, respectively. Therefore, σ_b^i can be obtained from the optical flow as

$$\sigma_b^i = \frac{T_e}{2T_f}|u\cos\theta + v\sin\theta|. \tag{8}$$

Solving Eq. (7) with Eq. (8) yields σ_m^i. This is illustrated in Fig. 2. In this experiment, the chart with a tilted edge pattern and a checkerboard shown in Fig. 2(a) was positioned to face the camera and moved from side to side with the velocity shown in Fig. 2(c). σ_{mb}^i was measured at the edge location and σ_b^i was calculated by Eq. (8) using the optical flow detected at the corners of the checkerboard. As shown in Fig. 2(b), almost constant σ_m^i was obtained as a result of the elimination of σ_b^i from σ_{mb}^i. The result shows that this method can clearly eliminate the motion blur. In the implementation of the proposed algorithm with ORB-SLAM presented in section IV, we use the ORB features present in a keyframe and the subsequent frame to get optical flow. When the corresponding feature cannot be found in the subsequent frame, we use the projection of the corresponding map point onto the subsequent frame.

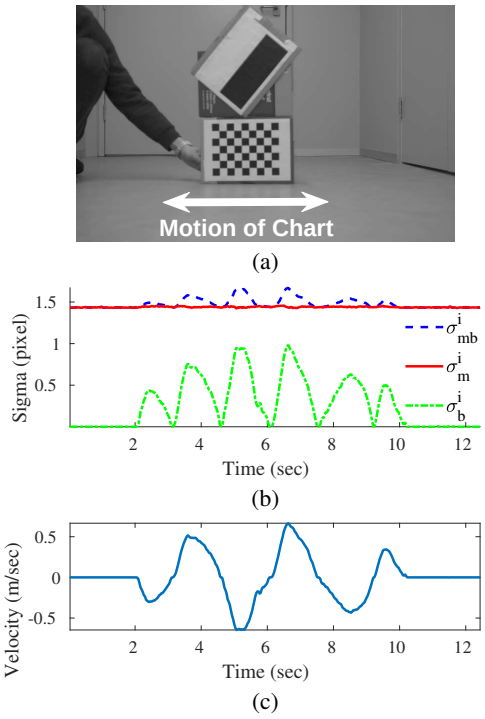


Fig. 2. Demonstration of Eq. (8). (a) shows the chart with a tilted binary edge pattern and a checkerboard. The chart was positioned to face the camera at a distance of 2m and moved from side to side with the velocity shown in (c). In (b), blue $\times$, red $+$, and green $*$ show σ_{mb}^i, σ_m^i, and σ_b^i, indicating that σ_m^i is nearly constant as expected. The exposure time was 8msec and the frame period was 33msec.

IV. Scale Optimization

A. Optimization

We begin by defining the scale factor Λ in metric scale using

$$d^{i,j} = \Lambda z^{i,j}, \tag{9}$$

where $d^{i,j}$ is the metric distance from the camera to a map point on each keyframe, $z^{i,j}$ is its up to a scale counterpart. Subscript 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 ORB-SLAM as

$$\mathbf{p}^{i,j} = {\mathbf{R}^j}^{-1}(\mathbf{p}_w^i - \mathbf{t}^j), \quad \mathbf{p}^{i,j} = [x^{i,j} \ y^{i,j} \ z^{i,j}]^T. \tag{10}$$

The scale factor Λ and texture blur correction factor λ_i are optimized by the following nonlinear least squares minimization derived from Eqs. (5) and (9):

$$\underset{\Lambda,\lambda^i}{\operatorname{argmin}} \sum_{i,j}(\sigma_m^{i,j} - \lambda^i D(\Lambda z^{i,j}))^2, \tag{11}$$

where $\sigma_m^{i,j}$ is measured at the corner extracted by the ORB features. We note here that it is not necessary to deal with the focal plane ambiguity since we do not use the inverse function of Eq. (4) that causes multiple solutions across the focal plane [8].

In our previous work [9], we demonstrated that the amount of defocus blur cannot be estimated correctly on complex texture such as letters, and therefore constraints of Eq. (5) no longer hold in these situations. Furthermore, due to the nonlinearity of Eq. (4), depth estimation from defocus blur is only effective at short range [8]. Therefore, a staged strategy is required to solve the optimization given in Eq. (11), to avoid the possibility of converging to local minima.

B. Initial Guess

We first select a set of features with sharp edges to minimize the impact of blur texture ambiguity and therefore simplify the optimization problem of Eq. (11) to

$$\underset{\Lambda}{\operatorname{argmin}} \sum_{i,j}(\sigma_m^{i,j} - D(\Lambda z^{i,j}))^2. \tag{12}$$

The edge selection criterion is given by

$$\sigma_m^{i,j} = \begin{cases} inlier & if \ e_{thl} < smg^{i,j} < e_{thh} \\ outlier & else. \end{cases} \tag{13}$$

Here $smg^{i,j}$ is the multiplication of estimated blur $\sigma_m^{i,j}$ and gradient magnitude around the feature $mg^{i,j} = ||\bigtriangledown I^{i,j}||$ where

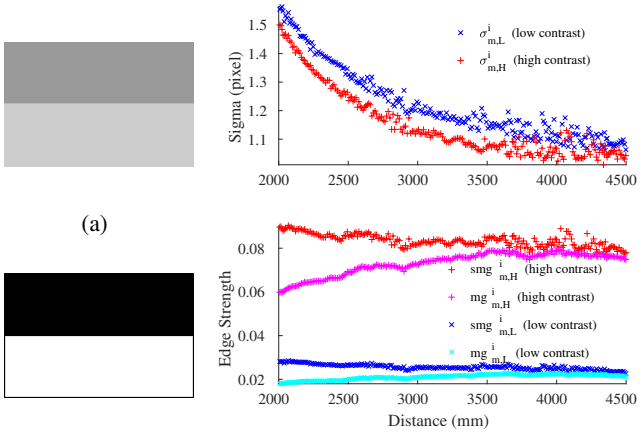


Fig. 3. Demonstration of Eq. (13). (a) and (b) show the charts with a low-contrast edge and a binary edge, respectively. In (c), smg_H^i is the multiplication of estimated blur $\sigma_{m,H}^i$ and gradient magnitude mg_H^i measured at the edge on (b). Also, smg_L^i is the multiplication of estimated blur $\sigma_{m,L}^i$ and gradient magnitude mg_L^i measured at the edge on (a).

$\bigtriangledown$ means the gradient. e_{thl} and e_{thh} are threshold values and decided from $smg^{i,j}$ of the binary edge pattern measured in advance. We can make the assumption that $mg^{i,j}$ is inversely proportional to $\sigma_m^{i,j}$. Therefore, multiplying $\sigma_m^{i,j}$ and $mg^{i,j}$ can effectively cancel the blur effect, thus is a good index to evaluate the edge strength. This is illustrated in Fig. 3. Fig. 3(a) and (b) are low-contrast and binary edge patterns, respectively. In Fig. 3(c), smg_H^i is the multiplication of $\sigma_{m,H}^i$ and mg_H^i measured from the binary edge, and so smg_L^i is from the low-contrast edge. As expected, smg_H^i and smg_L^i stay almost constant despite the change of defocus blur. The same Gaussian kernel is used for both the blur estimates and the gradient magnitude estimates. The results demonstrate that the proposed index can express the edge strength, independent of the amount of defocus blur.

The solution of Eq. (12) gives an accurate scale estimate, provided a sufficient number of features with sharp edges exist in the scene. In situations where this is not the case, we found that the scale estimate obtained serves as a good initial guess to the more general optimization problem given by Eq. (11). Features to be incorporated into computing the objective function defined by Eq. (11) can be selected as follows.

C. Feature Selection

Change of λ^i: The features that satisfy the constraint of Eq. (5) are selected as

$$\sigma_m^{i,j} = \begin{cases} inlier & if\ r_{thl} < r^{i,j} < r_{thh} \\ outlier & else, \end{cases} \tag{14}$$

$$r^{i,j} = \frac{\lambda_{ini}^{i,j}}{\lambda_{ini}^{i,j-1}}, \tag{15}$$

where r_{thl} and r_{thh} are threshold values and empirically decided from the accuracy of the initial guess in advance. $\lambda_{ini}^{i,j}$ is calculated from Eq. (5) with the initial sale Λ_{ini} estimated from Eq. (12) as

$$\lambda_{ini}^{i,j} = \frac{\sigma_m^{i,j}}{D(\Lambda_{ini} z^{i,j})}. \tag{16}$$

When satisfied with this condition, λ^i is regarded as a constant value between two keyframes.

Effective Range: The features which have $z^{i,j}$ within a range threshold are used:

$$\sigma_m^{i,j} = \begin{cases} inlier & if\ z^{i,j} < z_{th} \\ outlier & if\ z^{i,j} \geq z_{th}, \end{cases} \tag{17}$$

where z_{th} is the range threshold. The value of z_{th} is decided from the focal plane position.

Fig. 4. The camera and lens used in this experiment. The field of view is about 37-degree width. The focal length and f-number were fixed during the experiment.

V. EXPERIMENTAL EVALUATIONS

A. Dataset

To evaluate the proposed algorithm, a number of image sequences with 752 $\times$ 480 pixels resolution at 30fps were captured with a FLIR® BFLY-U3-23S6M-C camera and a Fujinon® CF16HA-1 lens shown in Fig. 4 walking in a corridor environment at University of Technology Sydney. The maps and camera trajectories generated by ORB-SLAM from the datasets are shown in Fig. 5. In Fig. 5(a), the camera moved in the direction of the arrows and returned to the origin to make a closed loop. In Fig. 5(b), the camera traveled along a figure of eight trajectory as indicated by the arrows in the same environment to make two closed loops. Checkerboard patterns were placed at locations CA, CB, C1, C2, C3, and C4 so that the true scale in each local region could be computed. Fig. 5(c) shows examples of keyframes including the checkerboards indicated on Fig. 5(b).

The scale estimated using the checkerboard patterns and the scale error in each of the local region are shown in Table I. 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 [28] implemented in MATLAB®. The range over where the checkerboard was detected reliably was about 7m. The scale error was computed relative to the initial scale estimated at CA for trajectory 1 and C1 for trajectory 2. ORB-SLAM result was found to be quite accurate in case of the trajectory 1 in Fig.

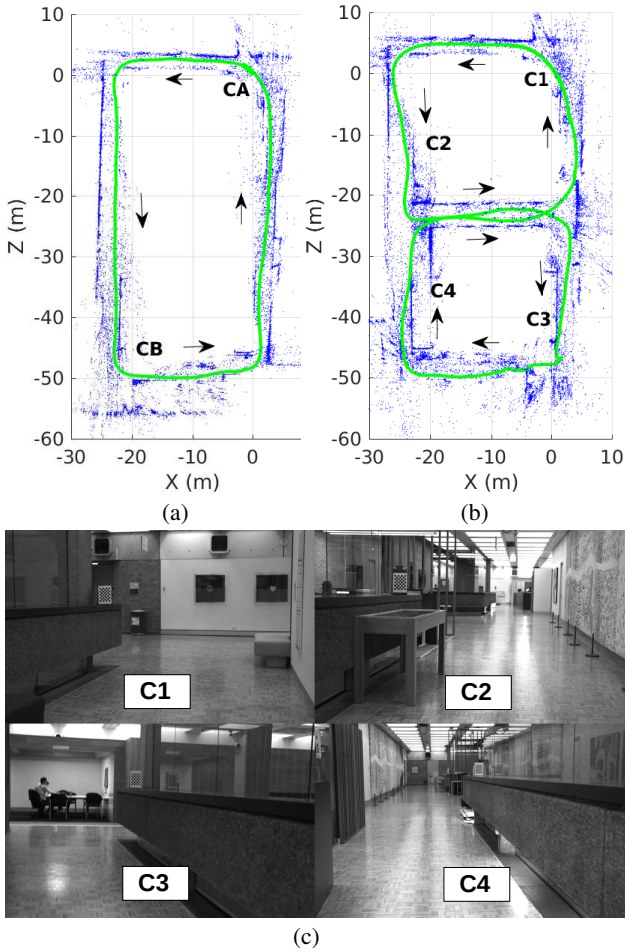


Fig. 5. In (a) and (b), green lines show the camera trajectory and blue dots show the point cloud of features generated by ORB-SLAM. The scale was reconstructed using the mean value of the scales computed using checkerboard patterns and shown in Table I. Some turns of the trajectory used to capture (b) were sharper than the trajectory shown in (a).

TABLE I
SCALE AND SCALE ERROR

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

5(a), where the scale error at CB was only 1.2%. On the other hand, in the trajectory 2 in Fig. 5(b), the maximum scale error was 8.3% perhaps due to the presence of multiple sharp turns. The root mean square errors (RMSE) of the keyframe positions are shown in Table III. The dataset used in the experiments can be made available on request.

B. Scale estimation using DfD

Ten keyframes around each checkerboard pattern were used in the optimization process. The parameters used for DfD are shown in Table V. The threshold values for optimization are shown in Table VI. The trust-region reflective method [29] implemented in MATLAB® was used to solve for the initial guess using Eq. (12) and for the full solution using Eq. (11).

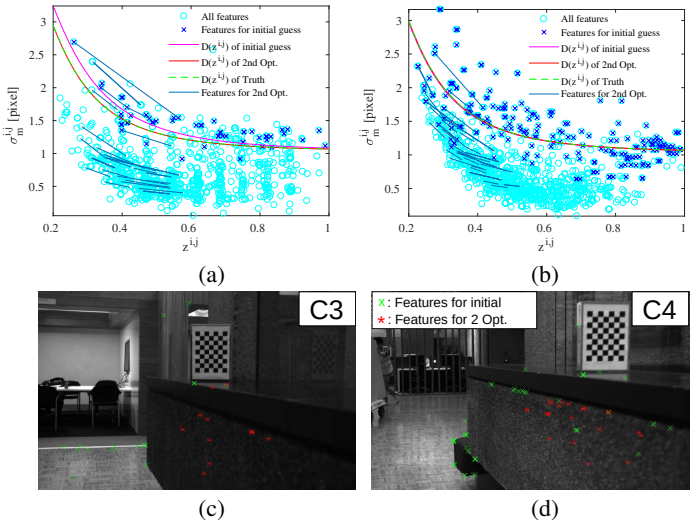


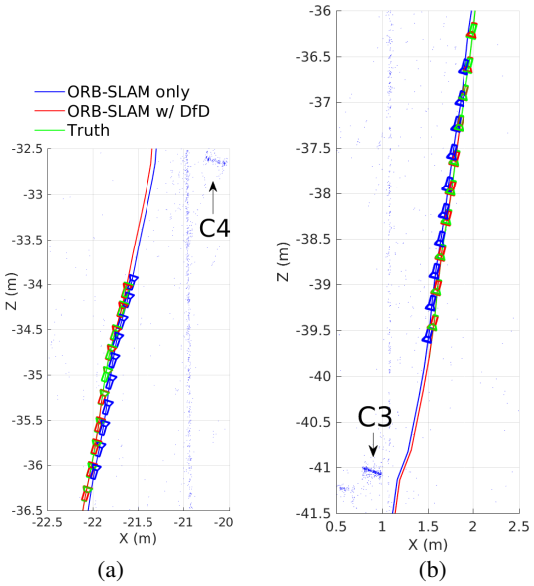
Fig. 6. In (a) and (b), Cyan 'o' show all features, blue 'x' show the features selected for the initial guess. Each blue line connects the same feature for different keyframes, which is selected for the second optimization. Magenta, orange, and green lines show the approximations $D(z^{i,j})$ as results of the initial guess, the second optimization, and the truth. (c) and (d) are the examples of keyframes in the local regions C3 and C4, respectively. Green 'x' show the features selected for the initial guess, and red '$*$' show the features selected for the second optimization. To be fair, features on the checkerboards were excluded for the optimizations.

TABLE II
ERROR IN SCALE ESTIMATE IN EACH AREA (%)

Trajectory 1			Trajectory 2		
	Initial Guess	2nd Opt.		Initial Guess	2nd Opt.
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

The error in the scale estimate is calculated as $e = (\Lambda_e - \Lambda_t)/\Lambda_t$ where subscripts e and t are the estimation and the truth, respectively.

Table II shows the results of optimization. As can be seen, solving for the simplified optimization problem given by Eq. (12) 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 Eq. (11). The final errors in the scale estimates are less than 0.20%. This is illustrated in Fig. 6. Fig. 6(a) and (b) show $z^{i,j}$ vs $\sigma_m^{i,j}$ in all keyframes in the local regions around C3 and C4, respectively. In (b), a set of features distributed around the true approximation curve $D(z^{i,j})$ were detected for obtaining the initial guess. Fig. 6(d) is an example of the keyframes in the local region including C4. This image demonstrates that the features used to obtain the initial guess were selected at the edge positions. On the other hand, in (a), the number of features selected for the initial guess was smaller than in (b) and resulted in a scale estimation error of 10.4%. Fig. 6(c) 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 features with sharp edges. However, in the second optimization step, the proposed feature selection algorithm was able to select the features which satisfied the Eq. (5) to reduce the scale estimation error to 0.14%.



(a) (b)

Fig. 7. (a) and (b) show the local maps including C4 and C3, respectively. Blue lines show the trajectory generated by ORB-SLAM. Red lines show the trajectory corrected by the estimated scales. Green lines show ground truth obtained from the checkerboard detection algorithm. Point clouds indicated by arrows are the map points on the corresponding checkerboards.

TABLE III
RMSE OF KEYFRAME POSITIONS (mm)

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

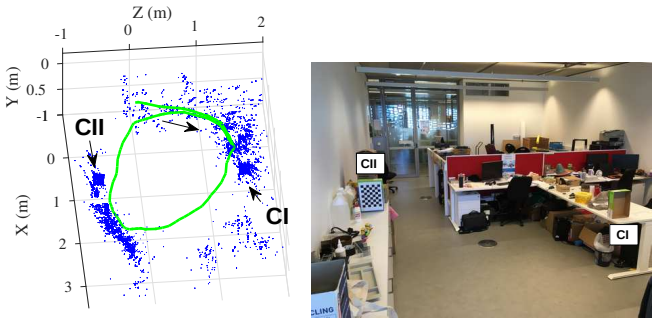
Fig. 7 shows the camera trajectories corrected by the estimated scales. The RMSE of keyframe positions are shown in Table III. All keyframes which could see the checkerboard within 7m were used for calculating RMSE as

$$RMSE = \Big(\frac{1}{m}\sum_{j=1}^{m} ||\mathbf{t}^j - \mathbf{t}_t^j||^2\Big)^{\frac{1}{2}}, \tag{18}$$

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, and m is the number of keyframes. In the trajectory 2, RMSE of ORB-SLAM was around 300mm. Our method was able to reduce RMSE to below 40mm. Results from this experiment demonstrate that the proposed method can correct the scale error accurately from only a monocular camera without any prior knowledge about the scene.

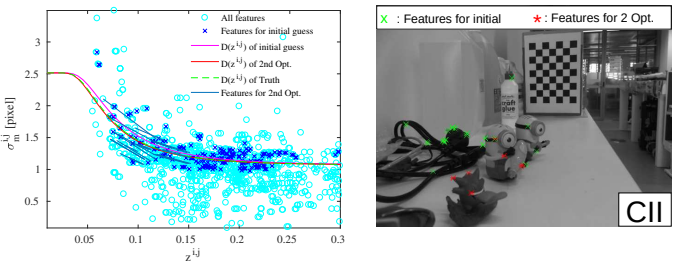
VI. DISCUSSION AND CONCLUSION

The effective measuring range of DfD depends on the lens, especially its focal length and aperture size. For the camera used in section V, this is approximately 3m. The scale estimation is not feasible if all the visible features fall outside the effective measuring range of the DfD technique. For example, defocus blur is not significant in the KITTI dataset [30] due to the outdoor scenes where features in the environment are at a relatively large distance from the camera.



(a) (b)

Fig. 8. (a) is the map and the camera trajectory reconstructed by iPhone SE. (b) shows the office environment. In (a), green line is the trajectory and blue dots are point cloud generated by ORB-SLAM.



(a) (b)

Fig. 9. In (a), Cyan 'o' show all features, blue 'x' show the features selected for the initial guess. Each blue line connects the same feature for different keyframes, which is selected for the second optimization. Magenta, orange, and green lines show the approximations $D(z^{i,j})$ as results of the initial guess, the second optimization, and the truth. (b) is the example of keyframes in the local region CII. Green 'x' show the features selected for the initial guess, and red '$*$' show the features selected for the second optimization. To be fair, features on the checkerboards were excluded for the optimizations.

The same is true in the TUM dataset [31], where the short focal length of the camera limits the effective measuring range to less than 400mm. Therefore, the lens and camera properties need to be selected to suit a given application scenario. Fig. 8(a) shows the map and the camera trajectory generated by using the rear camera on iPhone SE in an office environment shown in Fig. 8(b). The effective measuring range for this small camera is only about 300mm. Fig. 9(a) shows $z^{i,j}$ vs $\sigma_m^{i,j}$ around CII and (b) is an example of the keyframes. As can be seen, the adequate amount of defocus blur was observed and our algorithm could select the good features for optimization properly in the environment where some of the features were observed within the range. The results of RMSE are shown in Table IV.

Although it did not appear in the experiment shown in section V, a possible failure scenario relates to the ability to obtain a suitable initial guess to the optimization problem defined by Eq. (11). 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.

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

TABLE IV
RMSE OF KEYFRAME POSITIONS BY IPHONE SE (*mm*)

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

TABLE V
PARAMETERS USED IN DFD

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

TABLE VI
THRESHOLD VALUES USED IN THE OPTIMIZATION

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

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.

In this paper, we have described a method for correcting scale drift in monocular SLAM with the aid of depth from defocus and illustrate it using the ORB-SLAM algorithm. Using the amount of defocus blur estimated on ORB features together with the map points and keyframe poses obtained from ORB-SLAM, the metric scales in selected local regions are estimated. The proposed algorithm only relies on the local accuracy of the underlying SLAM algorithm and therefore could be used before or after loop closure. The experimental evaluation demonstrated the effectiveness of the proposed algorithm. In this work, the output from ORB-SLAM is post-processed through a nonlinear optimization algorithm to estimate the metric scales in local regions. 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.

Future work will focus on exploring the effectiveness of a tightly coupled strategy where defocus constraints are incorporated within the ORB-SLAM bundle adjustment process. Integration of information from inertial sensors into this process is also planned.

REFERENCES

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," IEEE Trans. Robot., vol. 31, pp. 1147-1163, 2015.

[2] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," IEEE Trans. Pattern Anal. Mach. Intell., vol. 29, pp. 1052-1067, 2007.

[3] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular SLAM," in Proc. Robot.: Sci. and Syst. VI, vol. 2, 2010.

[4] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," IEEE Trans. Robot., vol. 33, no. 5, pp. 1255-1262, 2017.

[5] T. Pire, T. Fischer, J. Civera, P. D. Cristóforis, and J. J. Berlles, "Stereo parallel tracking and mapping for robot localization," in Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst. (IROS), 2015, pp. 1373-1378.

[6] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D Mapping with an RGB-D camera," IEEE Trans. Robot., vol. 30, pp. 177-187, 2014.

[7] Y. Y. Schechner and N. Kiryati, "Depth from Defocus vs. Stereo: How Different Really Are They?," Int. J. Comput. Vision, vol. 39, pp. 141-162, 2000.

[8] C. Wöhler, P. d'Angelo, L. Krüger, A. Kuhl, and H.-M. GroB, "Monocular 3D scene reconstruction at absolute scale," ISPRS J. Photogrammetry and Remote Sens., vol. 64, pp. 529-540, 2009.

[9] T. Shiozaki, G. Dissanayake, "Monocular 3D Metric Scale Reconstruction using Depth from Defocus and Image Velocity," in Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst. (IROS), 2017, pp. 6723-6728.

[10] J. Gräter, T. Schwarze, and M. Lauer, "Robust scale estimation for monocular visual odometry using structure from motion and vanishing points," in Proc. IEEE Intell. Veh. Symp. (IV), 2015, pp. 475-480.

[11] Z. Dingfu, Y. Dai, and L. Hongdong, "Reliable scale estimation and correction for monocular Visual Odometry," in Proc. IEEE Intell. Veh. Symp. (IV), 2016, pp. 490-495.

[12] S. Sodng, M. Chandraker, and C. C. Guest, "High Accuracy Monocular SFM and Scale Correction for Autonomous Driving," IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, pp. 730-743, 2016.

[13] N. Fanani, A. Stürck, M. Barnada, and R. Mester, "Multimodal scale estimation for monocular visual odometry," in Proc. IEEE Intell. Veh. Symp. (IV), 2017, pp. 1714-1721.

[14] T. Botterill, S. Mills, and R. Green, "Correcting Scale Drift by Object Recognition in Single-Camera SLAM," IEEE Trans. Cybern., vol. 43, pp. 1767-1780, 2013.

[15] D. P. Frost, O. Kähler, and D. W. Murray, "Object-aware bundle adjustment for correcting monocular scale drift," in Proc. IEEE Int. Conf. Robot. and Autom. (ICRA), 2016, pp. 4770-4776.

[16] 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 Proc. Robot.: Sci. and Syst., vol. 2, 2007.

[17] R. Mur-Artal and J. D. Tardós, "Visual-Inertial Monocular SLAM With Map Reuse," IEEE Robot. Autom. Lett., vol. 2, pp. 796-803, 2017.

[18] A. N. Rajagopalan and S. Chaudhuri, "Optimal selection of camera parameters for recovery of depth from defocused images," in Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR), 1997, pp. 219-224.

[19] P. Favaro and S. Soatto, "A geometric approach to shape from defocus," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, pp. 406-417, 2005.

[20] A. P. Pentland, "A New Sense for Depth of Field," IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-9, pp. 523-531, 1987.

[21] J. H. Elder and S. W. Zucker, "Local scale control for edge detection and blur estimation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, pp. 699-716, 1998.

[22] S. Zhuo and T. Sim, "Defocus map estimation from a single image," Pattern Recognition, vol. 44, pp. 1852-1858, 2011.

[23] A. Sellent and P. Favaro, "Which side of the focal plane are you on?," in Proc. IEEE Int. Conf. Comput. Photography (ICCP), 2014, pp. 1-8.

[24] H. Kumar, S. Gupta, and K. S. Venkatesh, "Resolving focal plane ambiguity in depth map creation from defocus blur using chromatic aberration," in Proc. 10th Int. Conf. Inf., Commun. and Signal Process. (ICICS), 2015, pp. 1-5.

[25] A. Punnappurath, Y. Balaji, M. Mohan, and A. N. Rajagopalan, "Deep Decoupling of Defocus and Motion Blur for Dynamic Segmentation," in Proc. Eur. Conf. Comput. Vision, pp. 750-765, 2016.

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

[27] S. Dai and Y. Wu, "Motion from blur," in Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR), 2008, pp. 1-8.

[28] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster, "Automatic Camera and Range Sensor Calibration using a Single Shot," in Proc. IEEE Int. Conf. Robot. and Autom. (ICRA), 2012, pp.3936-3943.

[29] T. F. Coleman and Y. Li, "An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds." SIAM J. Optimization, Vol. 6, no.2, pp. 418445, 1996.

[30] A. Geiger, P. Lenz and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR), 2012, pp. 3354-3361.

[31] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst. (IROS), 2012, pp. 573-580.