2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

TextSLAM: Visual SLAM with Planar Text Features

Boying Li, Danping Zou*, Daniele Sartori, Ling Pei, and Wenxian Yu

***Abstract*— We propose to integrate text objects in man-made scenes tightly into the visual SLAM pipeline. The key idea of our novel text-based visual SLAM is to treat each detected text as a planar feature which is rich of textures and semantic meanings. The text feature is compactly represented by three parameters and integrated into visual SLAM by adopting the illumination-invariant photometric error. We also describe important details involved in implementing a full pipeline of text-based visual SLAM. To our best knowledge, this is the first visual SLAM method tightly coupled with the text features. We tested our method in both indoor and outdoor environments. The results show that with text features, the visual SLAM system becomes more robust and produces much more accurate 3D text maps that could be useful for navigation and scene understanding in robotic or augmented reality applications.**

I. INTRODUCTION

Visual SLAM is an important technique of ego-motion estimation and scene perception, which has been widely used in navigation for drones [1], ground vehicles or self-driving cars [2], and augmented reality applications [3]. A typical visual SLAM algorithm extracts point features [4], [5] from images for pose estimation and mapping. Recent methods [6] [7] even directly operate on pixels. However, it is well known that incorporating high-level features like lines [8], or even surfaces [9] in the visual SLAM system will lead to better performance with fewer parameters.

One type of object surrounding us that can be used as high-level features is text. Text labels in daily scenes offer rich information for navigation. They can help us recognize landmarks, navigate in complex environments, and guide us to the destination. Text extraction and recognition have been developing fast in these days [10] [11] because of the boom of the deep neural networks and the emergence of huge text datasets such as COCO-Text[12], DOST[13], and ICDAR[14]. One question raises whether texts can be integrated into a visual SLAM system to not only yield better performance but also generate high-quality 3D text maps that could be useful for navigation and scene understanding.

There are several attempts towards text-aided navigation and location in recent years. A navigation system [15], [16] for blind people, assisted with text entities, is built upon the visual-inertial SLAM system shipped on the Google Tango tablet. Similarly, Wang et al. proposed a method to extract text features [17], which are then used for fusion with Tango's SLAM outputs to facilitate closing loops. The aforementioned works have shown the great potential of integrating text features with existing SLAM systems, though in a loosely coupled manner. It is worth further investigation into a tightly coupled approach by putting texts into the SLAM pipeline instead of treating the existing SLAM system as a black box.

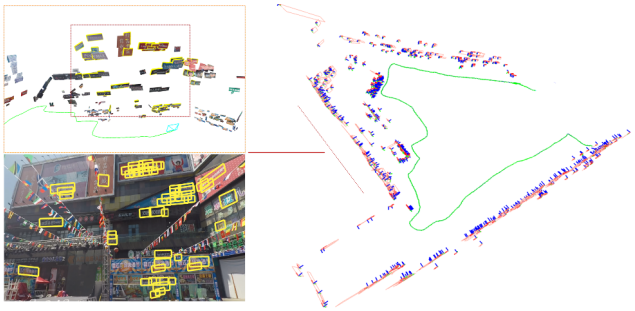

Fig. 1. TextSLAM in a shopping mall. **Left**: Detected texts in the images (in yellow rectangles) and the zoomed-in view of 3D text map. **Right**: 3D text maps and camera trajectory in top-down view. The text objects are illustrated in pink boxes and their normal directions are shown in blue.

We propose a novel visual SLAM method tightly coupled with text features. Our basic motivation is that texts are usually planar and texture-rich patch features: Texts we spotted in our daily life are mostly planar-like regions, at least for a single word or character if not the whole sentence. The rich pattern of a text entity makes the text object a naturally good feature for tracking and localization. It is demonstrated in our work that through fully exploring those characteristics of text features, we can improve the overall performance of the SLAM system, including the quality of both localization and semantic map generation. The main technical contributions of this paper includes:

1) A novel three-variable parameterization for text features is proposed. The parameterization is compact and allows instantaneous initialization of text features with small motion parallaxes.

2) The text feature is integrated in the visual SLAM system by adopting the photometric error measured by normalized sum of squared differences. Such photometric error is robust to illumination changes and blurry images caused by quick camera motions. Tracking and mapping of text features are done by minimizing the photometric errors without extra data association processes.

3) We present details for implementing such a text-based SLAM system, including initialization and update of text features, text-based camera pose tracking and back-end optimization. To our best knowledge, this is the first visual SLAM method with text features tightly integrated.

This work was supported by the Grant 61405180104 from Chinese government. All the authors are with Shanghai Key Laboratory of Navigation and Location-based Service, Shanghai Jiao Tong University.
*Corresponding author: Danping Zou (dpzou@sjtu.edu.cn)

We conduct experiments in both indoor and outdoor environments. The results show our novel text-based SLAM method achieves better accuracy and robustness, and produces much better 3D text maps than does the baseline approach. Such semantically meaningful maps will benefit navigation in man made environments.

II. RELATED WORK

a) Planar features: Planar features have been studied in visual SLAM community since the early stage. In early works [3][18][19], the planes in the scene were detected by RANSAC [20] among estimated 3D points and employed as novel features to replace points in the state. Much fewer parameters are required to represent the world using planes instead of points, hence reducing the computational cost significantly. Those works show that planar features improve both accuracy and robustness of a visual SLAM system. However, existing methods require 3D information to discover the planar features, usually using a RGB-D camera [21][22]. This becomes difficult using only image sensors. An interesting idea [23] is to assume each image patch surrounded the detected feature point as one observation of a locally planar surface. The assumption however seldom holds in realistic scenes, as feature points might be extracted from anywhere in the scene. Nevertheless, texts in realistic scenes are mostly located on planar surfaces, though sometime only locally. Therefore texts are naturally good planar features that can be easily detected by text detectors.

b) Object features: Integrating object-level features into visual SLAM systems has been receiving increasing interest in recent years [24][25][26]. Existing methods however require pre-scanned 3D models to precisely fit the observation on the image. Though recent work [27] attempted to reconstruct the 3D model of objects online with a depth camera, it is still difficult to be generalized to unseen objects with only image sensors. Another solution is adopting 3D bounding boxes [28][29] or quadrics [30] to approximate generic objects. This however suffers from loss of accuracy. Unlike generic objects, the geometry of text objects is simple. As aforementioned, text objects can be treated as locally planar features and also contain a rich amount of semantic information about the environment. It is hence worth investigating how to design a SLAM system based on the text objects.

c) Text-aided navigation: Text is a naturally good visual fiducial marker or optical positioning to assistant navigation [31][32] yet the technique is still under development [31]. Several loosely coupled text-aided navigation methods have shown a great potential for perception, navigation, and human-computer interaction. The early work [33] used the room number as a guidance for robot to autonomously move in the office-like scenes. The authors [34] annotated the text labels on the existing map generated by a laser SLAM system to help robots understand each named location. With the prior knowledge of a comprehensive map and the compass information, the authors [35] extracted 2D text information from the observations to assist localization. In the work

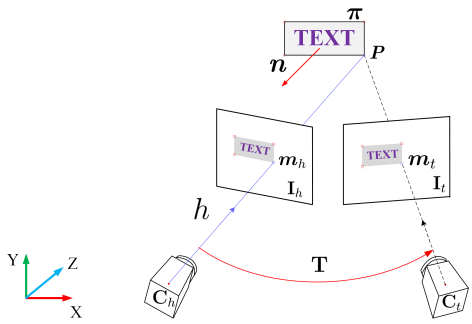

Fig. 2. A text object is compactly parameterized by $\boldsymbol{\theta}$. The inverse depth ρ of a text point $\boldsymbol{p}$ can be computed by $\rho = 1/h = \boldsymbol{\theta}^{\mathrm{T}}\tilde{\boldsymbol{m}}$ and its projection onto the target view C_t is a homography transform with respect to the relative pose $\mathbf{T}$ between the two views.

[17], the spatial-level feature named 'junction' was extracted from text objects, and then combined with the location and mapping output of Google Tango's SLAM system at the stage of loop closing. The authors present a text spotting method [15] for the assistant navigation system relying the Tango's SLAM system. Similarly, with the SLAM system of Tango, a mobile solution [16] of assistant navigation system combines various sources, such as text recognition results and speech-audio interaction, for blind and visually impaired people to travel indoor independently.

Existing text-aided methods regard SLAM systems (either vision-based or laser-based) as a black box and take less attention on the accuracy of 3D text maps. By contrast, the proposed method integrates the text objects tightly into the SLAM system to facilitate both camera tracking and mapping, and focuses on generating high accurate text map that can be used for future pose estimation and loop detection.

III. TEXT FEATURES

A. Parameterization

As discussed previously, most text objects can be regarded as planar and bounded patches. Each text patch (usually enclosed by a bounding box) is anchored to the camera frame, named as the *host frame*, when it is firstly detected on the image as shown in Fig. 2. Expressed in the coordinate system of the host frame, the plane where the text patch lies is described by the equation $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{p} + d = 0$, where $\boldsymbol{n} = (n_1, n_2, n_3)^{\mathrm{T}} \in \mathbb{R}^3$ is the normal of the plane and $d \in \mathbb{R}$ is related to the distance from the plane to the origin of the host frame; $\boldsymbol{p} \in \mathbb{R}^3$ represents the 3D point on the plane.

A straightforward parameterization of a text plane could be directly using the four parameters (n_1, n_2, n_3, d) of the plane equation. But this is a over parameterization that leads to rank deficient in the nonlinear least-squares optimization. We propose to use a compact parameterization that contains only three parameters.

$$\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)^{\mathrm{T}} = -\boldsymbol{n}/d. \tag{1}$$

We'll show that this parameterization is closely related to the inverse depth of the 3D point on the text plane.

Within the host frame, each 3D point $\boldsymbol{p} \in \mathbb{R}^3$ observed on the image is able to be represented by its normalized image coordinates $\boldsymbol{m} = (u, v)^{\mathrm{T}}$ and its inverse depth $\rho = 1/h$, where the depth h is the distance between this 3D point and the camera center. The 3D coordinates of this 3D point are computed as $\boldsymbol{p} = (uh, vh, h)^{\mathrm{T}} = h\tilde{\boldsymbol{m}}$, where $\tilde{\boldsymbol{m}}$ denotes the homogeneous coordinates of $\boldsymbol{m}$. If the 3D point locates on the text plane, we have $h \cdot \boldsymbol{n}^{\mathrm{T}}\tilde{\boldsymbol{m}} + d = 0$. The inverse depth ρ of this 3D point is then computed as

$$\rho = 1/h = -\boldsymbol{n}^{\mathrm{T}}/d \ \tilde{\boldsymbol{m}} = \boldsymbol{\theta}^{\mathrm{T}}\tilde{\boldsymbol{m}}. \tag{2}$$

That is, we can use a simple dot product to quickly infer the inverse depth of a text point from its 2D coordinates, given the text parameters $\boldsymbol{\theta}$.

On the other hand, if we have at least three points on the text patch (for example, three corners of the bounding box), with their inverse depths, we can immediately obtain the text parameters by solving

$$\begin{bmatrix} \tilde{\boldsymbol{m}}_1^{\mathrm{T}} \\ \vdots \\ \tilde{\boldsymbol{m}}_n^{\mathrm{T}} \end{bmatrix} \boldsymbol{\theta} = \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_n \end{bmatrix}, n \geq 3. \tag{3}$$

This allows us to quickly initialize the text parameters from the depth value of three corners of the text bounding box.

To fully describe a text object, properties such as the boundary of the text object and pixel values are also kept in our system. Those information can be acquired from a text detector as we'll described later.

B. Projection of the 3D text object onto a target image

Here we describe how to project a 3D text object anchored at a host frame onto the image related to the *target frame*. Let $\mathbf{T}_h, \mathbf{T}_t \in SE(3)$ represent the transformations from the host frame and the target frame to the world frame respectively. The transformation from the host frame to the target frame is $\mathbf{T} = \mathbf{T}_t^{-1}\mathbf{T}_h$. We let $\mathbf{R}, \boldsymbol{t}$ be the rotation and translation of $\mathbf{T}$. Given the text parameters $\boldsymbol{\theta}$ and the observed image point $\boldsymbol{m}$ (with the homogeneous coordinates $\tilde{\boldsymbol{m}}$) in the host frame, the 3D coordinates of point $\boldsymbol{p}$ are :

$$\boldsymbol{p} = \tilde{\boldsymbol{m}}/\rho = \tilde{\boldsymbol{m}}/(\boldsymbol{\theta}^{\mathrm{T}}\tilde{\boldsymbol{m}}). \tag{4}$$

The point is then transformed into the target frame. Let the transformed point be $\boldsymbol{p}'$. We have $\boldsymbol{p}' \sim \mathbf{R}\tilde{\boldsymbol{m}} + \boldsymbol{t}\,\tilde{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\theta}$. Here notation $\sim$ means equivalence up to a scale. Using the pinhole camera model, the coordinates of image point $\boldsymbol{m}' = (u', v')^{\mathrm{T}}$ of $\boldsymbol{p}'$ in the target frame are computed as

$$\begin{array}{ll} u' & = (\boldsymbol{r}_1^{\mathrm{T}}\tilde{\boldsymbol{m}} + t_1\tilde{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\theta})/(\boldsymbol{r}_3^{\mathrm{T}}\tilde{\boldsymbol{m}} + t_3\tilde{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\theta}) \\ v' & = (\boldsymbol{r}_2^{\mathrm{T}}\tilde{\boldsymbol{m}} + t_2\tilde{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\theta})/(\boldsymbol{r}_3^{\mathrm{T}}\tilde{\boldsymbol{m}} + t_3\tilde{\boldsymbol{m}}^{\mathrm{T}}\boldsymbol{\theta}) \end{array}, \tag{5}$$

where $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3$ are the row vectors of $\mathbf{R}$ and $\boldsymbol{t} = (t_1, t_2, t_3)^{\mathrm{T}}$. Note that (5) is in fact the homography transformation from $\boldsymbol{m}$ to $\boldsymbol{m}'$, where the homograpny matrix is defined as $\mathbf{H} \sim \mathbf{R} + \boldsymbol{t}\boldsymbol{\theta}^{\mathrm{T}}$. Hence, the whole process of projecting a 3D text object on the image plane of a target frame can be described as a homography mapping

$$\boldsymbol{m}' = \boldsymbol{h}(\boldsymbol{m}, \mathbf{T}_h, \mathbf{T}_t, \boldsymbol{\theta}). \tag{6}$$

C. Photometric error for text objects

Photometric error is used to compare the projected text object and the observed one on the image. This is done by pixel-wise comparison and similar to the direct approaches [7] which have been shown to be accurate and robust without finding corresponding feature points explicitly. The biggest issue of using photometric error is to handle the intensity changes. Existing work [7] adopts an affine model to address intensity changes, but it requires extra parameters involved in optimization and sophisticated photometric calibration to guarantee performance.

We choose to use zero mean normalized cross-correlation (ZNCC) as the matching cost to handle illumination changes. Let Ω be the set of pixels within the text region, and $\boldsymbol{m} \in \Omega$ be a text pixel. The normalized intensities for text pixels are: $\tilde{I}(\boldsymbol{m}) = (I(\boldsymbol{m}) - \bar{I}_\Omega)/(\sigma_\Omega\sqrt{N})$, where $\bar{I}_\Omega$ and σ_Ω stand for the average intensity and the standard deviation of the pixels in Ω, and N is the number of pixels. The text patch in the host frame and the predicted one in the target frame (6) are then compared by :

$$ZNCC(I_h, I_t) = \sum_{\boldsymbol{m}\in\Omega} \tilde{I}_h(\boldsymbol{m})\tilde{I}_t(\boldsymbol{m}'). \tag{7}$$

The ZNCC cost is between -1 and 1. The larger ZNCC cost indicates the two patches are more similar. However, it is difficult to directly use the ZNCC cost in visual SLAM, since it can not be formulated as a nonlinear least squares problem. We therefore adopt a variant form of ZNCC as the cost function

$$E(I_h, I_t) = \sum_{\boldsymbol{m}\in\Omega} (\tilde{I}_h(\boldsymbol{m}) - \tilde{I}_t(\boldsymbol{m}'))^2. \tag{8}$$

Though the cost function is similar to the SSD (Sum of Squared Difference) cost, it contains additional normalization process to ensure the robustness to illumination changes. If we expand this cost function as :

$$\sum_{\boldsymbol{m}\in\Omega} (\tilde{I}_h(\boldsymbol{m})^2 + \tilde{I}_t(\boldsymbol{m}')^2) - 2\sum_{\boldsymbol{m}\in\Omega} \tilde{I}_h(\boldsymbol{m})\tilde{I}_t(\boldsymbol{m}'), \tag{9}$$

we discover that minimizing this cost function is equivalent to maximizing the ZNCC cost, because $\sum \tilde{I}_h(\boldsymbol{m})^2 = 1$ and $\sum \tilde{I}_t(\boldsymbol{m}')^2 = 1$. The photometric error of a text object π with respect to the target frame t is defined as :

$$E_{photo}^{\pi,t} = \sum_{\boldsymbol{m}\in\Omega^\pi} \phi((\tilde{I}_h(\boldsymbol{m}) - \tilde{I}_t(\boldsymbol{h}(\boldsymbol{m}, \mathbf{T}_h, \mathbf{T}_t, \boldsymbol{\theta}^\pi)))^2), \tag{10}$$

where $\phi(\cdot)$ is the Huber loss function to handle possible outliers. Here, we use Ω^π to represent the text region on the image plane in the host frame. As we'll describe later, to make the computation faster, we do not use all the pixels within the text region, instead select only some of them as the reference pixels to compute the photometric error.

IV. TEXTSLAM SYSTEM

Our TextSLAM system is built upon the basic system using point features and adopts the keyframe-based framework to integrate the text features tightly. The mixture of point

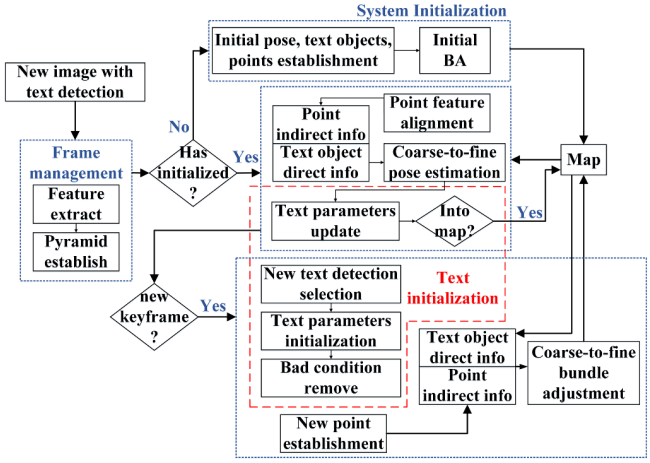

Fig. 3. An overview of TextSLAM system.

features and text features allow our system to work properly even in the scenes without text labels. Fig. 3 illustrates the flowchart of TextSLAM. we'll detail the key components in the following sections.

A. Initialization of text objects

Text objects are extracted on the image whenever a new image has been acquired. The deep learning technique has largely accelerated the text extraction development in recent years [36], [37], [38], [39]. The text detector named EAST [40] is used to extract objects in our implementation, but other text detectors may also be used, because our system does not rely on particular text detectors. Some examples of text extraction are demonstrated in Fig. 1. The outputs are arbitrary-orientation quadrilaterals enclosing the text regions. Once a text object has been extracted, we detect FAST [41] features within the text region and track them via Kanade-Lucas-Tomasi (KLT) [42] until the next key frame. Then the parameters of the text object are initialized from the tracked points. Let $\boldsymbol{m}_i \leftrightarrow \boldsymbol{m}'_i$ be the corresponding points in both views, and $\mathbf{R}, \boldsymbol{t}$ be the transformation between the two frames. From (5), we have

$$[\tilde{\boldsymbol{m}}'_i]_\times \boldsymbol{t} \tilde{\boldsymbol{m}}_i^{\mathrm{T}} \boldsymbol{\theta} = -[\tilde{\boldsymbol{m}}'_i]_\times \mathbf{R} \tilde{\boldsymbol{m}}_i, \tag{11}$$

where $\tilde{\boldsymbol{m}}_i$ and $\tilde{\boldsymbol{m}}'_i$ are the homogeneous coordinates of $\boldsymbol{m}_i$ and $\boldsymbol{m}'_i$. Note that the rank of the matrix on the left hand side is one. It requires at least three pair of corresponding points to solve $\boldsymbol{\theta}$. The text parameter is futher refined by minimizing the photometric error as defined in (10).

After initialization, we keep the text quadrilateral with the text object. The four corners can be projected onto other views (6) to predict the appearance. Note that a text object can only be initialized when its quadrilateral in the current view is not intersected with existing text objects or partially out of the image. Otherwise they are rejected as 'bad initialization'. The newly initialized text objects are kept being updated in the following frames. They are inserted into the map only if the text object has been observed in at least n_{min} (5 in our implementation) frames.

B. Camera pose estimation with text objects

Both points and text objects in the map are involved in camera pose estimation. The camera pose estimation is to minimize the following cost function

$$E(\mathbf{T}_t) = E_{point}(\mathbf{T}_t) + \lambda_w E_{text}(\mathbf{T}_t), \tag{12}$$

where $\mathbf{T}_t \in SE(3)$ represents the current camera pose. E_{point} and E_{text} come from the feature points and the text objects respectively. Note that E_{point} consists of geometric errors, or reprojection errors, but E_{text} contains only photometric errors

$$E_{text} = \sum_{\pi} E_{photo}^{\pi,t}. \tag{13}$$

The trade-off between them needs to be regulated by the weight λ_w since they are in different units (position difference vs intensity difference).

The weight λ_w is computed as $\lambda_w = \sigma_{rep}/\sigma_{photo}$. σ_{rep} represents the standard deviation of the reprojection error of a pair of corresponding points (in both x and y directions) and σ_{photo} represents the standard deviation of the photometric error of a text object as defined in (8). Those standard deviations can be acquired through a small set of training data (given corresponding points and text patches).

Optimization of the cost function (12) is a nonlinear least squares problem. As the photometric cost E_{text} is highly nonlinear, it requires a good initial guess of $\mathbf{T}_t$ to avoid being trapped in a local minimum. We firstly use a constant velocity model to predict the camera pose. Based on this prediction, we then apply a coarse-to-fine method for the optimization, and the camera pose is estimated by minimizing (12) iteratively.

C. Bundle Adjustment with text objects

We apply bundle adjustment from time to time in a local window of key frames similar to [4]. The cost function of bundle adjustment consists the point part and the text part :

$$E(\boldsymbol{x}) = E_{point}(\boldsymbol{x}) + \lambda_w E_{text}(\boldsymbol{x}). \tag{14}$$

The cost function resembles that of camera pose estimation while involves more parameters to be optimized. The variable $\boldsymbol{x}$ include the camera poses of key frames in the local window, the 3D coordinates of point features, and the text parameters. We also adopt a coarse-to-fine method to optimize (14) as in camera pose estimation.

Though we may use all the pixels within the text region to evaluate the photometric errors, a more efficient way is to use a small part of them. The representative pixels are selected with the minimum number of 15 by FAST [41], and further applied to evaluate photometric errors.

V. Experiment

A. Data collection

We collected a set of image sequences for evaluation in both indoor and outdoor scenes. Fig. 4 shows our device for the data collection. It consists of the hand-held camera and optical markers for the acquisition of ground truth

Fig. 4. The indoor test scene is shown on the left. The data collection device equipped with the GoPro camera, is presented on the right.

TABLE I

TABLE I: LOCALIZATION PERFORMANCE. RPE (0.1 M) AND APE (M)

Seq.	ORB-SLAM		Point-only		TextSLAM	
	RPE	APE	RPE	APE	RPE	APE
Indoor_01	0.342	0.161	0.192	0.223	0.188	0.196
Indoor_02	0.300	0.150	0.189	0.164	0.187	0.150
Indoor_03	0.228	0.144	0.207	0.166	0.209	0.155
Indoor_04	0.188	0.145	–	–	0.173	0.171
Indoor_05	0.277	0.120	0.172	0.160	0.174	0.161

The bar '–' indicates the algorithm fails to finish the whole trajectory.

trajectories. All image sequences were resized to 640×480 for tests.

B. Indoor scene with ground truth

The indoor environment, with text labels randomly placed, is shown in Fig. 4, which is equipped with a motion capture system to obtain the ground truth trajectories of millimeter accuracy within an area of $4m \times 4m$. Three methods were evaluated : our text-based method, our point-only method, and ORB-SLAM [4] where loop closing was disabled for fair comparison.

a) Trajectory estimation: We present both the relative pose error (RPE) and the absolute pose error (APE) in Tab. I. Note that the errors of those methods are very close to each other. The reason must be the test scene is very small and highly textured, in which using only feature points should work well. ORB-SLAM slightly outperforms both of our methods, which is not surprising because ORB-SLAM adopts a sophisticated map reuse mechanism based on the covisibility pose graph. Our methods currently are odometry systems virtually. Nevertheless, we still observe a performance gain in using text features compared with our point-only implementation.

We also evaluate the robustness of the proposed method under fast camera motion. The rapid motion causes severe image blur as shown in Fig. 5, inducing the failure of our point-only method in all cases. ORB-SLAM failed only at one test. This is also because its well implemented relocalization mechanism. By contrast, our text-based method works well in those tests and performs much more accurate as shown in Tab. II. This is largely due to our direct approach towards text objects using photometric errors.

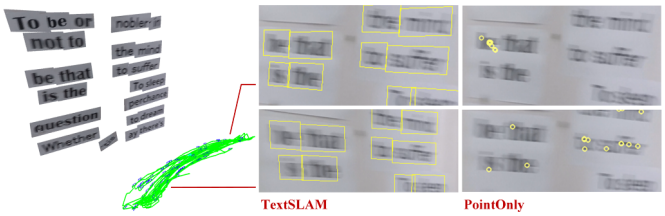

Fig. 5. TextSLAM is robust to blurry images caused by rapid camera motions. The estimated 3D text map and camera trajectory of TextSLAM are shown on the left. By contrast, point-only method failed to track feature points on severely blurry images as shown on the right.

TABLE II

TABLE II: INDOOR RAPID PERFORMANCE. RPE (0.1 M) AND APE (M)

Seq.	ORB-SLAM		Point-only		TextSLAM	
	RPE	APE	RPE	APE	RPE	APE
Rapid_01	–	–	–	–	0.271	0.029
Rapid_02	0.367	0.063	–	–	0.322	0.031
Rapid_03	0.338	0.061	–	–	0.212	0.022

The bar '–' indicates the algorithm fails to finish the whole trajectory.

b) 3D text maps: We use the angular error of each estimated text plane and visually inspection to evaluate the mapping performance. The angular error measures the difference between the estimated normal $\boldsymbol{n}_t$ of the text plane and the ground truth $\boldsymbol{n}_{gt}$, namely $\alpha = \arccos(|\boldsymbol{n}_t^{\mathrm{T}}\boldsymbol{n}_{gt}|/\|\boldsymbol{n}_t\|\|\boldsymbol{n}_{gt}\|)$. $\boldsymbol{n}_{gt}$ was acquired by a few optical markers on the text plane as shown in Fig. 4. Since no visual SLAM system generates text maps directly available for the evaluation, we implemented a loosely-coupled system based on ORB-SLAM for comparison by fitting the text plane from the 3D text points using three-point RANSAC.

The statistic of angular errors are presented in Fig. 8 and one of the mapping results is visually presented in Fig. 7. The results demonstrate that the 3D text map produced by TextSLAM is substantially better than that of the plane fitting approach based on ORB-SLAM. The reason is that the point clouds generated by ORB-SLAM are in fact noisy as shown in Fig. 7. We carefully checked what causes those noisy points and speculate that it may be caused by incorrect feature location or correspondences in the images.

C. Real-world tests

The outdoor experiment is in a commercial center as shown in the first column in Fig. 6. This daily environment is full of various challenges, including text objects with various sizes, fonts, backgrounds and languages, the complex occlusion, the reflection of glass and the dynamic pedestrians.

Since it is difficult to acquire the ground truth for either the camera trajectory or the 3D text map, we present only visual results to show the efficacy of our method. The second and three columns in Fig. 6 illustrate reconstructed 3D text labels and the estimate trajectories. A side-by-side comparison with ORB-SLAM from the top-down view is also presented on the last column.

The deep learning text detector sometimes produces false detections, the text object initialization (introduced in

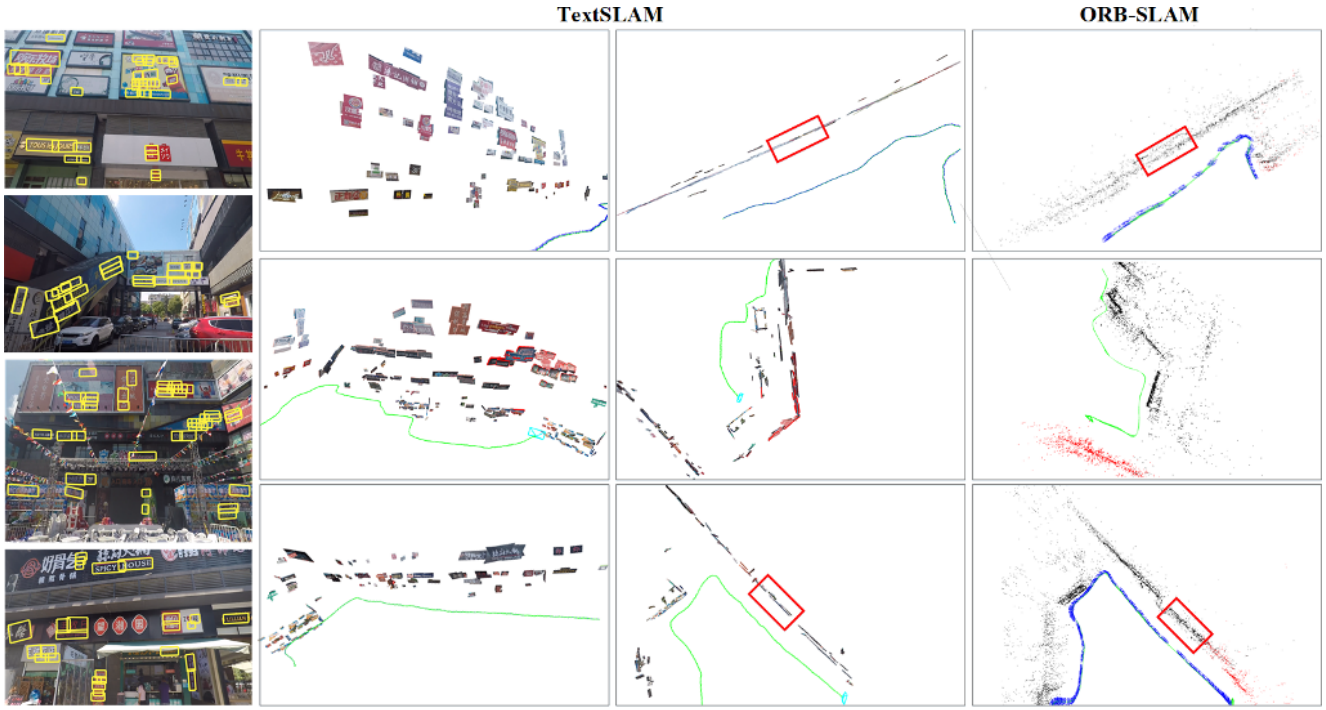

Fig. 6. Real world tests in a shopping center. The text detection results are shown in the first column. Three typical locations are shown and enlarged in each row. The second and third columns show the close-up and top-down views. For comparison, the ORB-SLAM performance of the same locations are also presented in the fourth column. We can observe the noisy point clouds visually, as enclosed in red rectangles.

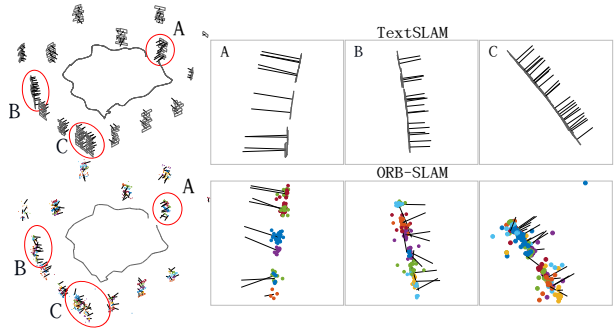

Fig. 7. Though RANSAC was adopted, plane fitting on the point clouds from ORB-SLAM still produced noisy results (as shown in the bottom row). TextSLAM avoids such problem by matching or tracking a text object as a whole by using photometric errors.

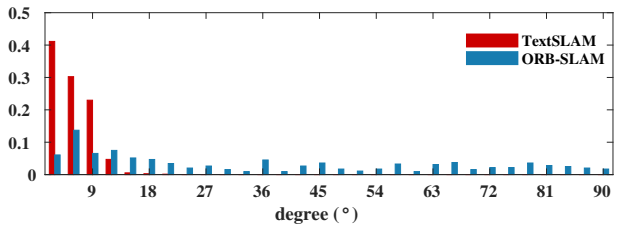

Fig. 8. The statistic distribution of the angular errors. The results of TextSLAM and ORB-SLAM are illustrated in red and blue, respectively.

SectionIV-A) filters out bad text objects automatically before insertion to the map. Our TextSLAM also works properly when no text labels found, such as the parking lot of the commercial center.

Though no ground truth is available, we can still observe the noisy point clouds in the ORB-SLAM results. As highlighted by rectangles in Fig. 6, the text maps better reveal the planar structures of the scene than the noisy point clouds from ORB-SLAM. As aforementioned, the noisy points are caused by either incorrect feature matching or detection, preventing ORB-SLAM from acquiring accurate 3D text maps.

VI. Conclusion & Future Work

We present a novel visual SLAM method tightly coupled with the planar text features. Experiments have been conducted in both artificial indoor cases with ground truth and real world scenes. The results show that our text-based SLAM method performs better than point-only SLAM method does, especially in blurry video sequences caused by rapid camera motions. However, the localization performance gain is not as large as we expected in the indoor tests, even when point-based methods generate very noisy point clouds. This could be that camera pose estimation is less sensitive to noisy points because robust approaches are usually involved. Nevertheless, the results demonstrate that our method generates much more accurate 3D text maps than the loosely-coupled method based on the state-of-the-art visual SLAM system. Future work includes incorporating text semantics into our system and acquiring real world datasets with highly accurate ground truth for quantitative evaluation.

References

[1] L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, “Autonomous visual mapping and exploration

with a micro aerial vehicle," *Journal of Field Robotics*, vol. 31, no. 4, pp. 654–675, 2014.

[2] H. Lategahn, A. Geiger, and B. Kitt, "Visual slam for autonomous ground vehicles," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1732–1737.

[3] D. Chekhlov, A. P. Gee, A. Calway, and W. Mayol-Cuevas, "Ninja on a plane: Automatic discovery of physical planes for augmented reality using visual slam," in *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 2007, pp. 1–4.

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *null*. IEEE, 2003, p. 1403.

[6] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 15–22.

[7] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[8] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "Structslam: Visual slam with building structure lines," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1364–1375, 2015.

[9] A. J. Trevor, J. G. Rogers, and H. I. Christensen, "Planar surface slam with 3d and 2d sensors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3041–3048.

[10] Y. Zhu, C. Yao, and X. Bai, "Scene text detection and recognition: Recent advances and future trends," *Frontiers of Computer Science*, vol. 10, no. 1, pp. 19–36, 2016.

[11] X.-C. Yin, Z.-Y. Zuo, S. Tian, and C.-L. Liu, "Text detection, tracking and recognition in video: a comprehensive survey," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2752–2773, 2016.

[12] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "Coco-text: Dataset and benchmark for text detection and recognition in natural images," *arXiv preprint arXiv:1601.07140*, 2016.

[13] M. Iwamura, T. Matsuda, N. Morimoto, H. Sato, Y. Ikeda, and K. Kise, "Downtown osaka scene text dataset," in *European Conference on Computer Vision*. Springer, 2016, pp. 440–455.

[14] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, *et al.*, "Icdar 2015 competition on robust reading," in *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*. IEEE, 2015, pp. 1156–1160.

[15] X. Rong, B. Li, J. P. Muñoz, J. Xiao, A. Arditi, and Y. Tian, "Guided text spotting for assistive blind navigation in unfamiliar indoor environments," in *International Symposium on Visual Computing*. Springer, 2016, pp. 11–22.

[16] B. Li, J. P. Munoz, X. Rong, Q. Chen, J. Xiao, Y. Tian, A. Arditi, and M. Yousuf, "Vision-based mobile indoor assistive navigation aid for blind people," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 702–714, 2019.

[17] H.-C. Wang, C. Finn, L. Paull, M. Kaess, R. Rosenholtz, S. Teller, and J. Leonard, "Bridging text spotting and slam with junction features," 2015.

[18] A. P. Gee, D. Chekhlov, W. W. Mayol-Cuevas, and A. Calway, "Discovering planes and collapsing the state space in visual slam." in *BMVC*, 2007, pp. 1–10.

[19] A. P. Gee, D. Chekhlov, A. Calway, and W. Mayol-Cuevas, "Discovering higher level structure in visual slam," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 980–990, 2008.

[20] M. Y. Yang and W. Förstner, "Plane detection in point cloud data," in *Proceedings of the 2nd int conf on machine control guidance, Bonn*, vol. 1, 2010, pp. 95–104.

[21] 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*. IEEE, 2012, pp. 573–580.

[22] P. Kim, B. Coltin, and H. Jin Kim, "Linear rgb-d slam for planar environments," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 333–348.

[23] N. Molton, A. J. Davison, and I. D. Reid, "Locally planar patch features for real-time structure from motion." in *Bmvc*. Citeseer, 2004, pp. 1–10.

[24] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[25] D. Gálvez-López, M. Salas, J. D. Tardós, and J. Montiel, "Real-time monocular object slam," *Robotics and Autonomous Systems*, vol. 75, pp. 435–449, 2016.

[26] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. Montiel, "Towards semantic slam using a monocular camera," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1277–1284.

[27] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, "Fusion++: Volumetric object-level slam," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 32–41.

[28] S. Yang and S. Scherer, "Cubeslam: Monocular 3-d object slam," *IEEE Transactions on Robotics*, 2019.

[29] ——, "Monocular object and plane slam in structured environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3145–3152, 2019.

[30] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[31] S. Houben, D. Droeschel, and S. Behnke, "Joint 3d laser and visual fiducial marker based slam for a micro aerial vehicle," in *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2016 IEEE International Conference on*. IEEE, 2016, pp. 609–614.

[32] R. Tapu, B. Mocanu, and T. Zaharia, "A computer vision-based perception system for visually impaired," *Multimedia Tools and Applications*, vol. 76, no. 9, pp. 11 771–11 807, 2017.

[33] M. Tomono and S. Yuta, "Mobile robot navigation in indoor environments using object and character recognition," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 1. IEEE, 2000, pp. 313–320.

[34] C. Case, B. Suresh, A. Coates, and A. Y. Ng, "Autonomous sign reading for semantic mapping," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3297–3303.

[35] N. Radwan, G. D. Tipaldi, L. Spinello, and W. Burgard, "Do you see the bakery? leveraging geo-referenced texts for global localization in public maps," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4837–4842.

[36] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "Fots: Fast oriented text spotting with a unified network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5676–5685.

[37] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li, "Single shot text detector with regional attention," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3047–3055.

[38] M. Busta, L. Neumann, and J. Matas, "Deep textspotter: An end-to-end trainable scene text localization and recognition framework," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2204–2212.

[39] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 745–753.

[40] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "East: an efficient and accurate scene text detector," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 2642–2651.

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

[42] J. Shi and C. Tomasi, "Good features to track," Cornell University, Tech. Rep., 1993.