

Visual SLAM for Small Indoor Vehicles

A project of the 2017 Robotics Course

Peihong Yu Jia Zheng

School of Information Science and Technology, ShanghaiTech University

{yuph, zhengjia}@shanghaitech.edu.cn

Abstract

Traditional approaches to stereo visual SLAM rely on point correspondences to estimate camera trajectory and build a map of the environment, consequently, their performance deteriorates in low-textured scenes where it is difficult to find a large or well-distributed set of point features.

In this paper, we propose a stereo visual SLAM systems that use line segments to work robustly in a wider variety of scenarios, and evaluate our approach in common benchmark.

1. Introduction

Simultaneous localization and mapping (SLAM) have become popular topics in recent years due to their wide application in robot navigation, 3D reconstruction, and virtual reality.

The traditional approaches consist of the detection and matching of point features between frames, and then, the estimation of the camera motion through least-squares minimization of the reprojection errors between the observed and projected points. It is noticeable that the performance of such approaches usually deteriorates in low textured environments, where it is difficult to find a large set of keypoint features. In contrast, line segments are usually abundant in human-made scenarios, which are characterized by regular structures rich in edges and linear shapes, for instance in city and indoor scenes, or in the so-called Manhattan worlds.

Point and line features are complementary in a camera localization system: point features provide good discrimination, but are view-dependent, while line features are robust to viewing changes, but are more fragile.

In this paper, we propose a solution that can simultaneously leverage points and lines information. We demonstrate this through various experiments.

2. Related Work

2.1. SLAM

2.1.1 Stereo SLAM

Monocular SLAM suffers from scale drift and may fail if performing pure rotations in exploration, while RGBD-SLAM can hardly deal with large-scale outdoor scene and glasses reflectance. By using stereo camera, all these issues are solved and can led to reliable visual SLAM solutions.

A remarkable early stereo SLAM system was the work in [11]. It was the first stereo SLAM exploiting both close and far points, using an inverse depth parameterization for reliable estimation for the latter. Also, based on conditionally independent divide and conquer extended-Kalman-filter SLAM, it was able to operate in larger environments than other approaches at that time.

Some works using direct methods emerge in recent years. These aim at computing geometry and motion directly from the images thereby skipping the intermediate keypoint selection step. One of the most recent ones is stereo-DSO [16]. It jointly optimizes for all the model parameters within the active window, including the intrinsic and extrinsic camera parameters of all keyframes and the depth values of all selected pixels. It also combines temporal multi-view stereo and static stereo and optimize their integration with marginalization using the Schur complement, which lead to more precise reconstruction results than other direct approaches and higher reconstruction density than feature-based methods.

To achieve higher computational speed and ensure guarantees for optimality and consistency, semi-direct VO(SVO) [5] is proposed. It uses direct methods to track and triangulate pixels that are characterized by high image gradients, but relies on proven feature-based methods for joint optimization of structure and motion. Together with a robust probabilistic depth estimation algorithm, it can provide efficiently tracking of pixels lying on weak corners and edges in environments with little or high-frequency texture.

The above papers are more related to visual odometry. Based on monocular ORB-SLAM [9], ORB-SLAM2 [10]

was proposed. It was the first open-source SLAM system for monocular, stereo, and RGBD cameras, including loop closing, relocalization and map reuse. It has three main parallel threads: tracking, local mapping, and loop closing, then creates a fourth thread to perform full BA after a loop closure. The tracking thread localizes the camera with every frame by finding feature matches to the local map and minimizing the reprojection error applying motion-only BA; the local mapping thread manage the local map and optimize it, perform local BA; the loop closing thread detects large loops and corrects the accumulated drift by performing a pose-graph optimization. The full BA thread was then performed to achieve the optimal structure and motion solution.

2.1.2 Line based SLAM

One of the remarkable approaches that employs line features is the one in [14], where the authors propose an algorithm to integrate them into a monocular Extended Kalman Filter SLAM system (EKF-SLAM). Other works employ edge landmarks as features in monocular SLAM, as the one reported in [4], which does not include the information of the local planar patch as in the case of keypoints, but also considers local edge segments, hence introducing new valuable information as the orientation of the so-called edgelets. In that work they derive suitable models for those kind of features and use them within a particle-filter SLAM system, achieving nearly real-time performance. More recently, authors in [5] also introduced edgelets in combination with intensity corners in order to improve robustness in environments with little or high-frequency texture.

More recently, based on ORB-SLAM [9] framework, PL-SLAM [12] proposes a solution that can simultaneously leverage points and lines information while keep computational cost. In order to preserve the real-time characteristics of ORB-SLAM [9], they have carefully chosen, used and implemented fast methods for operating with lines in all stages of the pipeline: detection, triangulation, matching, culling, re-localization and optimization. Line segments in an input frame are detected by mean of LSD [15], an $O(n)$ line segment detector, where n is the number of pixels in the image. Then, lines are pairwise matched with lines already present in the map using a relational graph strategy [17]. This approach relies on lines' local appearance (Line Band Descriptors) and geometric constraints and is shown to be quite robust against image artifacts while preserving the computational efficiency.

Another stereo version of PL-SLAM [7] build upon their previous Visual Odometry approach presented in [6]. The camera motion is recovered through non-linear minimization of the projection errors of both point and line segment features, with an ad-hoc implementation of the combined

algorithm to solve the bundle adjustment problem for this particular case. An extension of the bag-of-words approach takes into account the description of both points and lines segments to improve the loop-closure process.

2.2. Line Segment Detection

Three key features define a pixel riding on a line segment: 1) relatively strong gradient; 2) connectivity with other pixels having strong gradient; 3) linear shape and reasonable length of the connectivity.

According to the means of finding these pixels, the currently mainstream line segment detectors generally can be categorized into perceptual grouping and Hough ones. We mainly introduce the perceptual grouping methods here.

LSD [15] proposes line support region, as line segment candidates, by applying region growing algorithm on the pixel-wise gradient, then filters and aligns the candidates through multiple tools including rectangular approximation, a contrario model and Helmholtz principle, *etc.*

As the region growing is susceptible to noises in the image, LSD frequently fragment the line segments or failed to detect them. EDLines [3] connects local gradient peaks through ridges between them, and executes least squaring line fitting trough the paths to break them into line segments, which are also refined through Helmholtz principle. The evaluation illustrated that EDLines is more proof to noise.

The line segment candidates generated by EDLines, however, have poor smoothness. Liu *et al.* [8] adopt an edge chaining strategy to generate more smooth line-support regions, while improving noise-proof ability. They also in introduce smaller eigenvalue analysis to accelerate the straightness criterion, and finally detect the line segments in a top-down way.

Besides noise, gradient is also vulnerable to image resolution, LSD and EDLines, therefore, work anomaly in high resolution images. To relieve this problem, Salaun *et al.* [13] proposed multi-scale LSD(MLSD). It starts from coarser scale, applying a classical LSD; detects line segments at finer scales with the aid of detected line segments at previous scales; and finally merge them together. MLSD shows a robustness to image scale and well performs over high-resolution images.

2.3. ROS Packages

Camera Calibration [1] allows easy calibration of monocular or stereo cameras using a checkerboard calibration target. This package uses OpenCV camera calibration, and work with any camera driver node satisfying the standard ROS camera interface.

Navigation [2] is a 2D navigation stack that takes in information from odometry, sensor streams, and a goal pose

and outputs safe velocity commands that are sent to a mobile base.

3. System Description

We build a stereo visual SLAM system that use line segments to work robustly. We'll derive the relative pose by trifocal tensor between two consecutive frames in the following part.

3.1. Trifocal Tensor

Trifocal Tensor describes all the projective geometric relations between three views that are independent of scene structure.

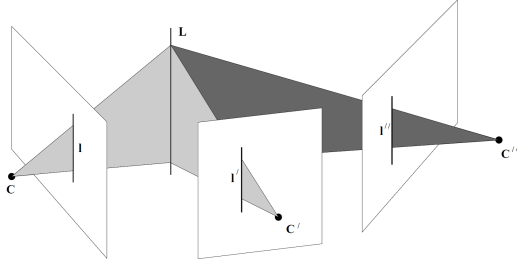


Figure 1. Trifocal Tensor.

Suppose a line L in 3D space is imaged as the corresponding triplet $l \longleftrightarrow l' \longleftrightarrow l''$ in three views, which are indicated by their centers, C, C', C'' , and image planes, as shown in Figure 1.

Let the camera matrices for the three views be

$$\begin{cases} \mathbf{P} = [\mathbf{I} \mid \mathbf{0}] \\ \mathbf{P}' = [\mathbf{A} \mid \mathbf{a}_4] \\ \mathbf{P}'' = [\mathbf{B} \mid \mathbf{b}_4] \end{cases}$$

where \mathbf{A} and \mathbf{B} are 3×3 matrices, and the vectors \mathbf{a}_i and \mathbf{b}_i are the i -th column of the respective camera matrices.

Each image line back-projects to a plane, which can be parameterized as

$$\begin{cases} \boldsymbol{\pi} = \mathbf{P}^T \mathbf{l} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ \boldsymbol{\pi}' = \mathbf{P}'^T \mathbf{l}' = \begin{pmatrix} \mathbf{A}^T \mathbf{l}' \\ \mathbf{a}_4^T \mathbf{l}' \end{pmatrix} \\ \boldsymbol{\pi}'' = \mathbf{P}''^T \mathbf{l}'' = \begin{pmatrix} \mathbf{B}^T \mathbf{l}'' \\ \mathbf{b}_4^T \mathbf{l}'' \end{pmatrix} \end{cases}$$

Let $\mathbf{M} = [\boldsymbol{\pi}, \boldsymbol{\pi}', \boldsymbol{\pi}'']$. Points on the 3D line L can be represented as $\mathbf{X} = \alpha \mathbf{X}_1 + \beta \mathbf{X}_2$, with \mathbf{X}_1 and \mathbf{X}_2 linearly independent. Such points lie on three planes and so $\mathbf{M}^T \mathbf{X} = \mathbf{0}$. Consequently \mathbf{M} has a 2-dimensional null-space since $\mathbf{M}^T \mathbf{X}_1 = \mathbf{0}$ and $\mathbf{M}^T \mathbf{X}_2 = \mathbf{0}$, so \mathbf{M} has rank 2. Denoting

$$\mathbf{M} = [\boldsymbol{\pi}, \boldsymbol{\pi}', \boldsymbol{\pi}''] = \begin{bmatrix} 1 & \mathbf{A}^T \mathbf{l}' & \mathbf{B}^T \mathbf{l}'' \\ 0 & \mathbf{a}_4^T \mathbf{l}' & \mathbf{b}_4^T \mathbf{l}'' \end{bmatrix}$$

since the rank of \mathbf{M} is 2, there is a linear dependence between its columns \mathbf{m}_i , which may be written as $\mathbf{m}_1 = \alpha \mathbf{m}_2 + \beta \mathbf{m}_3$.

Just looking at the bottom row of \mathbf{M} , we can get $\alpha \mathbf{a}_4^T \mathbf{l}' + \beta \mathbf{b}_4^T \mathbf{l}'' = 0$, which follows

$$\begin{cases} \alpha = k(\mathbf{b}_4^T \mathbf{l}'') \\ \beta = -k(\mathbf{a}_4^T \mathbf{l}') \end{cases} \text{ for some scalar } k$$

Applying this to the top 3-vectors of each column shows that (up to a homogeneous scale factor), we get

$$\begin{aligned} \mathbf{l} &= (\mathbf{b}_4^T \mathbf{l}'') \mathbf{A}^T \mathbf{l}' - (\mathbf{a}_4^T \mathbf{l}') \mathbf{B}^T \mathbf{l}'' \\ &= (\mathbf{l}''^T \mathbf{b}_4) \mathbf{A}^T \mathbf{l}' - (\mathbf{l}'^T \mathbf{a}_4) \mathbf{B}^T \mathbf{l}'' \end{aligned}$$

The i -th coordinate l_i of \mathbf{l} may therefore be written as

$$\begin{aligned} l_i &= \mathbf{l}''^T (\mathbf{b}_4 \mathbf{a}_i^T) \mathbf{l}' - \mathbf{l}'^T (\mathbf{a}_4 \mathbf{b}_i^T) \mathbf{l}'' \\ &= \mathbf{l}'^T (\mathbf{a}_i \mathbf{b}_4^T) \mathbf{l}'' - \mathbf{l}''^T (\mathbf{a}_4 \mathbf{b}_i^T) \mathbf{l}' \end{aligned}$$

and introducing the notation

$$\mathbf{T}_i = \mathbf{a}_i \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_i^T$$

the incidence relation can be written as

$$l_i = \mathbf{l}'^T \mathbf{T}_i \mathbf{l}''$$

The set of three matrices $\{\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3\}$ constitute the trifocal tensor in matrix notation. The full relation can be written as $\mathbf{l}^T = \mathbf{l}'^T [\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3] \mathbf{l}''$.

3.2. Trifocal Tensor in Stereo

At stereo situation, we use trifocal tensor to calculate the relative pose between two adjacent frames. We choose 2 views in one stereo frame, and one of other view in its adjacent frame.

Let the origin of stereo camera set is \mathbf{O} , the camera centers in this coordinate frame is \mathbf{c} and \mathbf{c}' , as shown in Figure 3.2. Then we can get

$$\begin{aligned} \mathbf{A} &= \mathbf{I} \\ \mathbf{a}_4 &= \mathbf{c} - \mathbf{c}' \\ \mathbf{B} &= \mathbf{R} = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3] \\ \mathbf{b}_4 &= \mathbf{t} - \mathbf{c}'' + \mathbf{R}\mathbf{c} \end{aligned}$$

Substitute this to the equations of trifocal tensor, we have

$$\begin{aligned} \mathbf{T}_1 &= \mathbf{a}_1 \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_1^T \\ &= \begin{bmatrix} \mathbf{b}_4^T \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} - (\mathbf{c} - \mathbf{c}') \mathbf{r}_1^T \end{aligned}$$

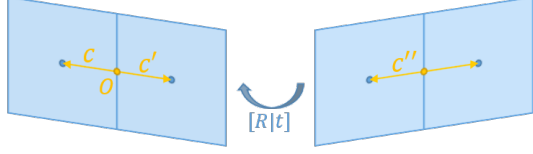


Figure 2. Trifocal tensor in stereo.

$$\begin{aligned} \mathbf{T}_2 &= \mathbf{a}_2 \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_2^T \\ &= \begin{bmatrix} \mathbf{0} \\ \mathbf{b}_4^T \\ \mathbf{0} \end{bmatrix} - (\mathbf{c} - \mathbf{c}') \mathbf{r}_2^T \\ \mathbf{T}_3 &= \mathbf{a}_3 \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_3^T \\ &= \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{b}_4^T \end{bmatrix} - (\mathbf{c} - \mathbf{c}') \mathbf{r}_3^T \end{aligned}$$

Each \mathbf{T}_i is a function of \mathbf{r}_i and \mathbf{t} , from $\mathbf{l}_{proj}^T = \mathbf{l}^T [\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3] \mathbf{l}''$. Ignore the details of derivation, we can derive and rearrange them to the following equation

$$\mathbf{l}_{proj} = \begin{bmatrix} a_{11} \mathbf{t} + a_{12} \vec{\mathbf{F}} + a_{13} \\ a_{21} \mathbf{t} + a_{22} \vec{\mathbf{F}} + a_{23} \\ a_{31} \mathbf{t} + a_{32} \vec{\mathbf{F}} + a_{33} \end{bmatrix}$$

where

$$\begin{aligned} a_{11} &= \mathbf{l}'_x \mathbf{l}''^T \\ a_{12} &= \mathbf{l}'_x [\mathbf{c}_x \mathbf{l}''^T \quad \mathbf{c}_y \mathbf{l}''^T \quad \mathbf{c}_z \mathbf{l}''^T] + [\mathbf{l}'^T (\mathbf{c} - \mathbf{c}') \mathbf{l}''^T \quad 0 \quad 0] \\ a_{13} &= -\mathbf{l}'_x \mathbf{l}''^T \mathbf{c}'' \\ a_{21} &= \mathbf{l}'_y \mathbf{l}''^T \\ a_{22} &= \mathbf{l}'_y [\mathbf{c}_x \mathbf{l}''^T \quad \mathbf{c}_y \mathbf{l}''^T \quad \mathbf{c}_z \mathbf{l}''^T] + [0 \quad \mathbf{l}'^T (\mathbf{c} - \mathbf{c}') \mathbf{l}''^T \quad 0] \\ a_{23} &= -\mathbf{l}'_y \mathbf{l}''^T \mathbf{c}'' \\ a_{31} &= \mathbf{l}'_z \mathbf{l}''^T \\ a_{32} &= \mathbf{l}'_z [\mathbf{c}_x \mathbf{l}''^T \quad \mathbf{c}_y \mathbf{l}''^T \quad \mathbf{c}_z \mathbf{l}''^T] + [0 \quad 0 \quad \mathbf{l}'^T (\mathbf{c} - \mathbf{c}') \mathbf{l}''^T] \\ a_{33} &= -\mathbf{l}'_z \mathbf{l}''^T \mathbf{c}'' \\ \vec{\mathbf{r}} &= [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3]^T \end{aligned}$$

The \mathbf{l}_{proj} is solved with its two correspondences and the trifocal tensor constrain, we also have its real expression, which will leads to the following constrains.

$$\begin{aligned} \mathbf{l} \times \mathbf{l}_{proj} &= \begin{bmatrix} \mathbf{l}_y a_{31} - \mathbf{l}_z a_{21} \\ \mathbf{l}_z a_{11} - \mathbf{l}_x a_{31} \\ \mathbf{l}_x a_{11} - \mathbf{l}_y a_{11} \end{bmatrix} \mathbf{t} + \begin{bmatrix} \mathbf{l}_y a_{32} - \mathbf{l}_z a_{22} \\ \mathbf{l}_z a_{12} - \mathbf{l}_x a_{32} \\ \mathbf{l}_x a_{22} - \mathbf{l}_y a_{12} \end{bmatrix} \vec{\mathbf{r}} \\ &+ \begin{bmatrix} \mathbf{l}_y a_{33} - \mathbf{l}_z a_{23} \\ \mathbf{l}_z a_{13} - \mathbf{l}_x a_{33} \\ \mathbf{l}_x a_{23} - \mathbf{l}_y a_{13} \end{bmatrix} = 0 \end{aligned}$$

In the minimal case, the solver just needs two lines, but it will get more robust solution when take more lines as input.

The solver leads to many solutions, from which we can use energy function to get the optimal solutions.

4. System Evaluation

We evaluate our algorithms in different scenarios.

4.1. Simulation test of Solver

We compare our solver with another one which takes 3D lines as input. In the simulation, we first generate 2 random lines in 3D with known rotation and translation as our ground truth. For each line, we project them to the each views, and add noise to their end points. The triangulated 3D lines by projected 2D lines will be given to the 3D-3D solver as input, and the 2D lines will be given to our solver as input.

We test our solver under different noise level from 0.1 pixel to 1 pixel. For each noise level, we run 1,000 experiments independently.

The subscript angular, absolute, relative in Table 4.1 denote the angular, absolute, and relative error respectively. The percent means the acceptable solution rate, whose error angular of rotation is under 1° .

4.2. Visualization

The visual result will be shown in demo part.

5. Conclusion

In this paper, we propose a stereo visual SLAM systems based on combination of both keypoints and line segments features. We evaluate our solver in simulation test and real world scenario.

References

- [1] ROS camera calibration. http://wiki.ros.org/camera_calibration. Accessed: 2017-11-01.
- [2] ROS navigation. <http://wiki.ros.org/navigation>. Accessed: 2017-11-01.
- [3] C. Akinlar and C. Topal. Edlines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, 2011.
- [4] E. Eade and T. Drummond. Edge landmarks in monocular slam. *Image and Vision Computing*, 27(5):588–596, 2009.
- [5] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2017.
- [6] R. Gomez-Ojeda and J. Gonzalez-Jimenez. Robust stereo visual odometry through a probabilistic combination of points and line segments. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 2521–2526. IEEE, 2016.

Table 1. Median error of the Solver under different noise level.

	method	0.1 pixel	0.2 pixel	0.3 pixel	0.4 pixel	0.5 pixel
R_angular	2D	0.041561	0.086806	0.115743	0.160887	0.207837
	3D	0.024125	0.048615	0.066685	0.088656	0.108230
R_absolute	2D	0.001026	0.002143	0.002857	0.003972	0.005130
	3D	0.000595	0.001200	0.001646	0.002188	0.002671
R_relative	2D	0.000592	0.001237	0.001649	0.002293	0.002962
	3D	0.000344	0.000693	0.000950	0.001263	0.001542
t_angular	2D	0.045247	0.072911	0.812981	0.645973	0.539083
	3D	0.125775	0.046887	0.050562	0.504173	0.310335
t_absolute	2D	0.001631	0.001899	0.013702	0.003430	0.015165
	3D	0.002718	0.001042	0.002997	0.009640	0.004665
t_relative	2D	0.001540	0.001386	0.019255	0.019213	0.012373
	3D	0.002253	0.000819	0.002723	0.009137	0.005904
percent	2D	0.863520	0.849421	0.837419	0.831117	0.820181
	3D	0.798000	0.809000	0.811000	0.816000	0.829000

	method	0.6 pixel	0.7 pixel	0.8 pixel	0.9 pixel	1.0 pixel
R_angular	2D	0.219625	0.250154	0.266318	0.319854	0.374803
	3D	0.137028	0.155243	0.194368	0.186282	0.226535
R_absolute	2D	0.005421	0.006174	0.006573	0.007895	0.009251
	3D	0.003382	0.003832	0.004798	0.004598	0.005592
R_relative	2D	0.003130	0.003565	0.003795	0.004558	0.005341
	3D	0.001953	0.002212	0.002770	0.002655	0.003228
t_angular	2D	0.282563	0.937200	0.255073	0.82506	2.98127
	3D	0.151625	0.264036	0.14124	0.243137	0.559882
t_absolute	2D	0.010145	0.015607	0.010470	0.032519	0.052408
	3D	0.003078	0.003241	0.003198	0.004776	0.012710
t_relative	2D	0.007373	0.017282	0.006985	0.030461	0.055965
	3D	0.003054	0.006438	0.003045	0.004884	0.012843
percent	2D	0.831646	0.7979	0.845946	0.828084	0.783465
	3D	0.825000	0.795000	0.778000	0.786000	0.802000

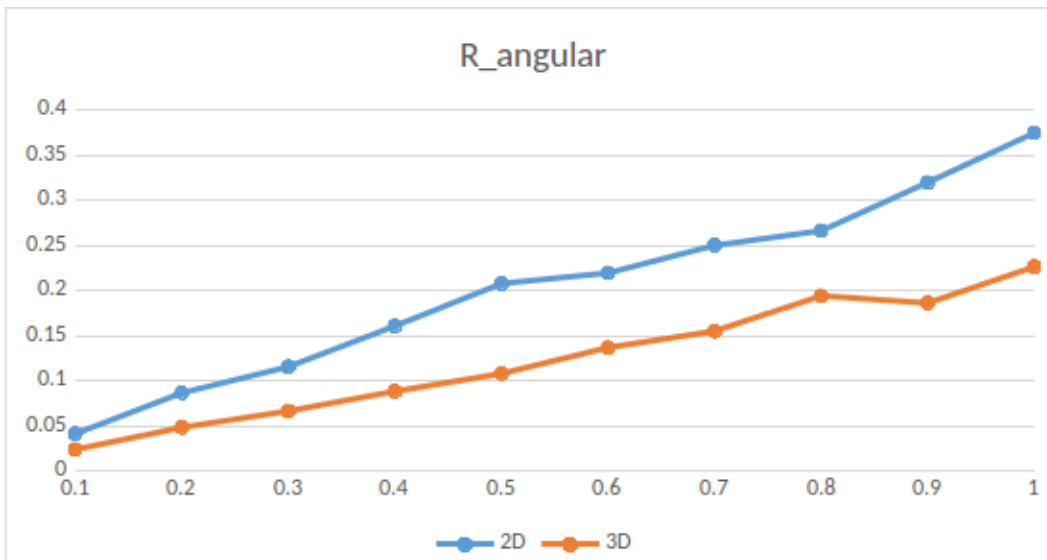


Figure 3. The angular error of rotation versus noise level.

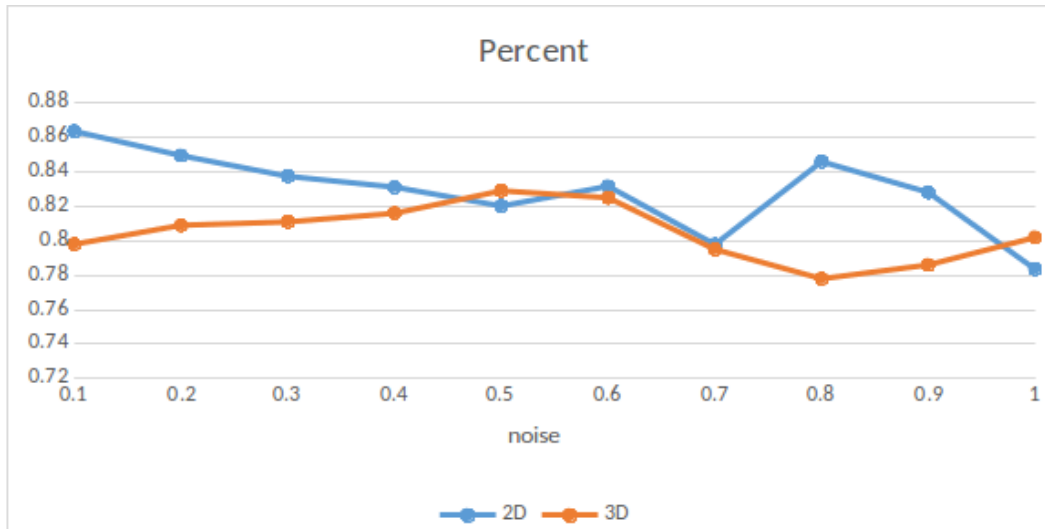


Figure 4. The acceptable solution percent versus noise level.

- [7] R. Gomez-Ojeda, F.-A. Moreno, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: a stereo slam system through the combination of points and line segments. *arXiv preprint arXiv:1705.09479*, 2017.
- [8] D. Liu, Y. Wang, Z. Tang, and X. Lu. A robust and fast line segment detector based on top-down smaller eigenvalue analysis. In *Fifth International Conference on Graphic and Image Processing*, pages 906916–906916. International Society for Optics and Photonics, 2014.
- [9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.
- [10] R. Mur-Artal and J. D. Tardos. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, PP(99):1–8, 2016.
- [11] L. M. Paz, P. Pinies, J. D. Tardos, and J. Neira. Large-scale 6-dof slam with stereo-in-hand. *IEEE Transactions on Robotics*, 24(5):946–957, 2008.
- [12] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. PL-SLAM: Real-time monocular visual slam with points and lines. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4503–4508. IEEE, 2017.
- [13] Y. Salaün, R. Marlet, and P. Monasse. Multiscale line segment detector for robust and accurate sfm. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 2000–2005. IEEE, 2016.
- [14] P. Smith, I. D. Reid, and A. J. Davison. Real-time monocular slam with straight lines. 2006.
- [15] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–732, 2010.
- [16] R. Wang, M. Schwörer, and D. Cremers. Stereo DSO: large-scale direct sparse visual odometry with stereo cameras. *CoRR*, abs/1708.07878, 2017.
- [17] L. Zhang and R. Koch. An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency. *Journal of Visual Communication and Image Representation*, 24(7):794–805, 2013.