RGB-D visual odometry with point and line features in dynamic environment

Shuai Wang\textsuperscript{1} and Baoling Han
Beijing Institute of Technology, Beijing, China

\textsuperscript{1}E-mail: 1543990541@qq.com

Abstract. Vision-based simultaneous localization and mapping (SLAM) technology is the key to realize autonomous navigation of mobile robots. When the robot is in an unfamiliar environment, it usually uses the point features of the surrounding environment to estimate its pose. However, if the feature information in the environment is not rich and there are many dynamic objects, the camera trajectory cannot be accurately estimated. To this end, this paper proposed an RGB-D visual odometry that combines point features and line features simultaneously. The dynamic line features are eliminated by calculating the static weight of the line features, and the camera pose is estimated based on the point features and the remaining line features. Compared with other feature-based SLAM systems, the performance and accuracy of systematic pose estimation can be improved in the absence of feature points or dynamic environments.

1. Introduction
Simultaneous Localization and Mapping (SLAM) is a research hotspot in the field of robotics and computer vision. For mobile robots working in unfamiliar environments, the primary problem is that the robot can locate and map at the same time. SLAM is a comprehensive system that helps robots find their location and reconstruct the map of surroundings. The SLAM process consists of two parts: the front-end and the back-end. The front end is also known as the visual odometry VO [1-5], in which information collected from sensors (monocular cameras, stereoscopic cameras, RGB-D cameras or lidar) is analyzed and calculated to determine the robot's position and direction. This paper focuses on the visual SLAM based on RGB-D camera. RGB-D camera can be used to obtain RGB information and corresponding depth information at the same time, so as to solve the problem of depth localization caused by the single acquisition of two-dimensional scene information by the visual sensor. But RGB-D camera cannot perform well in environments with strong illumination changes and more dynamic objects. On the basis of this, the visual odometry is built by using point feature and line feature fusion to further overcome the defect that the point feature is difficult to track and match due to scene or illumination changes in the real environment. Secondly, we calculated the probability of the line features in the static environment, and eliminated the lines belonging to the dynamic objects using the static weight method, and the camera pose is calculated by the remaining static lines. Compared with the existing methods based on the individual point features, the method can effectively reduce the tracking error and improve the Accuracy and robustness, especially in environments with strong illumination changes and more dynamic objects.
2. Related work

2.1. Line feature and its application in SLAM

Line feature is in a higher level compared to point feature. Line feature is more robust to illumination and viewing angle changes, and it can express the geometric information of the scene more intuitively.

The line features in the image are extracted by the LSD [6] algorithm, and the feature descriptors are calculated by the LBD algorithm. LSD is a line segment feature extraction algorithm proposed by Von et al. in 2012. Its core idea is to combine pixel points with similar gray and gradient directions to generate line segment support fields, and then construct a rectangle to determine whether the line segment support domain can be as a line feature. The algorithm can obtain sub-pixel precision detection results in linear time, also can control the number of false detections by itself, which has the advantages of high extraction speed and strong robustness.

LBD [7] is a line segment description algorithm proposed by Zhang et al. in 2013. Its main idea is to establish a line segment support domain in a local area of a line segment, then use several strips to describe the linear support domain. Compared with the MSLD algorithm, the algorithm introduces weight coefficients, which has the advantages of high calculation speed and strong robustness.

Many methods combine the point features and line features to realize the visual odometry. Yaan Lu [8] realized the visual odometer by combining the point features and line features, detected the line segment on the image of the RGBD camera, performed the least square on the spatial line segment through the depth map, and proved the combination of point features and line features can reduce the uncertainty of the pose estimation. Ruben [9] proposed a point-and-line visual odometer based on binocular camera. The reprojection error of point and line features was calculated separately during optimization, and the weights of the point and line features are adjusted according to the error statistics. But their works cannot perform well in dynamic environments. The proposed method eliminates the dynamic line features by calculating the static weight, and successfully improved the performance and accuracy of systematic pose estimation in a dynamic environment.

2.2. Dynamic SLAM and VO

SLAM methods have developed over 30 years, there are many excellent works about dynamic SLAM or VO. To simplify positioning and mapping problems, most of the current visual odometry methods assume that the environment is static, but dynamic objects are inevitably present in real-world environments. Dynamic objects cause large intensity/depth differences [10]. The minimum energy function [11] cannot effectively reflect the correct camera pose. Therefore, it is necessary to detect, distinguish and remove dynamic objects. Paper [12] uses pixel intensity difference between images to detect the boundaries of dynamic objects, realizes the clustering segmentation [13] of dynamic object points by depth map, the performance is stable under dynamic scene, but the real-time performance is not ideal. Paper [14] first calculates the dense optical flow through color images, then uses point clustering method to detect dynamic objects, this method effectively improves the robustness of visual odometers in dynamic environments, but there are still problems with optical flow estimation and clustering time-consuming. Paper [15] uses the depth difference of past frames to calculate the static background model, the method is stable in dynamic environment, but there is an aperture problem. When the dynamic object is parallel to the camera plane, only the boundary of the dynamic object can be detected, and the influence of the dynamic objects cannot be completely eliminated. In this paper, the dynamic line features are eliminated by calculating the static weight of the line features, and the camera pose is estimated based on the point features and the remaining line features.

3. Research method

3.1. System overview

Figure 1 is a block diagram of a visual odometer system [16] based on line features. It is mainly divided into two parts: detection and tracking. For each newly obtained image, extract the ORB point...
features in the color image, perform point feature matching and estimate the initial transformation matrix. In the color image, the line features are extracted by LSD (line segment detector) algorithm, which are described and matched. The static weight of the line feature is calculated according to the initial transformation matrix, and the dynamic line features are eliminated. Finally, the matching static line features are used to estimate the camera pose.

Figure 1. Overview of the visual odometry system.

3.2. Initial pose calculation by point features
The ORB algorithm [17] is a feature point detection and description algorithm based on visual information, which is widely used in visual SLAM. The basic process of the ORB-based camera pose estimation method is as follows: First, ORB feature points are extracted from the color image, and the descriptors are calculated. Then, the K nearest neighbor (KNN) algorithm [18] is used to match the feature points according to the binary descriptor of the feature points. Finally, the 3D coordinates of the matched ORB feature points are calculated and the initial transformation is solved. To improve the system robustness, the initial transformation matrix is estimated by the ICP algorithm [19] under the RANSAC framework.

3.3. Line features
3.3.1. Line feature extraction and matching. Suppose that given 2 frames of image source frame S and target frame M, \( l_m \) is a line segment on frame M and \( l_n \) is a line segment on frame N. The angles of the two line segments are \( \theta_m \) and \( \theta_n \) respectively, and the lengths of the two line segments are \( |l_m| \) and \( |l_n| \) respectively, and the distance from the midpoint in \( l_m \) to the other line segment \( l_n \) is \( d \). If \( l_m \) and \( l_n \) satisfy the following conditions, they are considered to be matched.

(a) The distance of the LBD description operators between \( l_m \) and \( l_n \) is less than a given threshold \( \gamma \).

(b) The directional difference between \( l_m \) and \( l_n \) is less than the given threshold \( \varphi_1 \), namely:

\[ |\theta_m - \theta_n| < \varphi_1 \]  \hspace{1cm} (1)

(c) The lengths of \( l_m \) and \( l_n \) are approximate, namely:

\[ \frac{\min(|l_m|, |l_n|)}{\max(|l_m|, |l_n|)} > \varphi_2 \]  \hspace{1cm} (2)

(d) The distance from the midpoint of \( l_m \) to \( l_n \) is less than a given threshold, namely:
3.3.2. Geometric representation of a spatial line. For RGB-D image, each pixel has color gray value \( l_{RGB} \) and depth value \( d_k \in \mathbb{R}^4 \) in frame \( k \). Assume that the coordinates of a point \( p_k \) in the space corresponding to pixel \( x \) are \((x, y, z, 1)^T \), which matches \( p_{k-1} \). \( p_k \) and \( p_{k-1} \) achieve rigid body migration by homogeneous transformation matrix \( T_{k-1}^k \in \text{SE}(3) \). Homogeneous transformation matrix \( T_{k-1}^k \) is a Euclidean transformation consisting of rotation and translation.

\[
T_{k-1}^k = \begin{bmatrix} R_{k-1}^k & t_{k-1}^k \\ 0_{1 \times 3} & 1 \end{bmatrix} p_{k-1}
\]

Equation (4)

Among the formula \( R_{k-1}^k \in \text{SO}(3) \) is a \( 3 \times 3 \) rotation matrix and \( t_{k-1}^k \in \mathbb{R}^3 \) is a \( 3 \times 1 \) translation vector. Similarly, the transformation matrix \( T_{w}^c \in \text{SE}(3) \) is used to represent the transformation from the world coordinate system to the camera coordinate system. It contains a rotation matrix \( R_w^c \in \text{SO}(3) \) and a translation vector \( t_w^c \in \mathbb{R}^3 \), as shown in equation (5):

\[
T_{w}^c = \begin{bmatrix} R_w^c & t_w^c \\ 0_{1 \times 3} & 1 \end{bmatrix}
\]

Equation (5)

Given a straight line \( L \), it can be represented by two points. Assuming that the homogeneous coordinates of the two points are \( X_1 = (x_1, y_1, z_1, 1)^T \) and \( X_2 = (x_2, y_2, z_2, 1)^T \), respectively, and the non-homogeneous coordinates are represented by \( \tilde{X}_1 \) and \( \tilde{X}_2 \), then the Plücker coordinates of the spatial straight line \( L \) are:

\[
\begin{bmatrix} \tilde{X}_1 \\ \tilde{X}_2 \\ r_{\tilde{X}_2} \tilde{X}_1 - r_{\tilde{X}_1} \tilde{X}_2 \end{bmatrix} = \begin{bmatrix} n \\ \nu \end{bmatrix} \in \mathbb{P}^5
\]

Equation (6)

\( L \) is a \( 6 \times 1 \) vector containing \( n \) and \( \nu \), where \( \nu \) is the direction vector of the line, \( n \) is the moment vector of the line, and \( n \) is perpendicular to the plane determined by the origin and the line. Here \( \mathbb{P}^5 \) refers to the 5-dimensional projective space.

Convert the line \( L_w \) from the world coordinate system to the camera coordinate system, denoted \( L_c \), according to reference [20] we denote \( L_c \) in Plücker coordinates:

\[
L_c = \begin{bmatrix} n_c \\ \nu_c \end{bmatrix} = H_w^c L_w = \begin{bmatrix} R_w^c & [t_w^c] \times & R_w^c \\ 0_{3 \times 3} & R_w^c \end{bmatrix} L_w
\]

Equation (7)

Where \([\cdot] \times\) denotes the antisymmetric matrix of the vector and \( H_w^c \) denotes the straight line transformation matrix. According to the known camera internal reference, the line \( L_c \) is projected onto the image plane, which is denoted \( l_c \).

\[
l_c = Kn_c = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ -f_y c_x & f_x c_y & f_x f_y \end{bmatrix} n_c = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix}
\]

Equation (8)

Among the formula, \((f_x, f_y, c_x, c_y)\) is the camera internal reference, \( f_x, f_y \) are the focal lengths of the camera on the x-axis and y-axis, \( c_x, c_y \) are the aperture centers of the camera, and \( K \) is the projection matrix of the line.

3.3.3. Line feature static weight estimation. Given the matched line segments \( l_i \) and \( l_i' \) in the two frames \( S \) and \( M \), the line segment \( l_i \) belongs to the source frame \( S \), the line segment \( l_i' \) belongs to the target frame \( M \), and the extreme point of \( l_i \) is the point \( X'_i, X'_i' \in \mathbb{R}^3 \), as shown in the Figure 2. \( T_{\text{initial}} \) The static line segment \( l_i \) after the initial transformation matrix \( T_{\text{initial}} \) is denoted \( l_i^{\text{initial}} \), the extreme point of which is point \( X_s, X_e \in \mathbb{R}^3 \). The static weight of the line segment \( l_i \) is represented by \( w_i^{SM} \), which can be confirmed with the spatial distance between \( t_i^{\text{initial}} \) and the line segment \( l_i' \). The two line segments have a common vertical vector \( N \) and their spatial distance is \( d_0 \), then:

\[
N = \frac{X_s X_e'}{X_s X_e'} \times \frac{X'_s X'_e}{X'_s X'_e}
\]

Equation (9)
According to Reference [21], this paper uses $t$ distribution to estimate static weights.\[ w_i^{S,M} = \frac{\nu_0 + 1}{\nu_0 + [(d_D - \mu_D)/\sigma_D]^2} \] (11)

$\nu_0$ is the degree of freedom of the $t$ distribution, $\mu_D$ is the mean, and $\sigma_D$ is the variance. In the experiment, $\nu_0$ is set to 10 according to the empirical value, $\mu_D$ is set to 0, and $\sigma_D$ is estimated by the distance absolute median function Median. That $d_D$ is 0 indicates the line feature has a high static probability, a small $d_D$ indicates a small noise interference, the line feature is from a static environment, a large $d_D$ indicates a large interference, and a straight line feature is from a dynamic object.

The algorithm flow is shown in Algorithm 1.

**Algorithm 1.** Line feature static weight.

Input: Source frame S and target frame M linear matches between the two frames and. Initial transformation matrix $T_{initial}$;  
Output: the line feature static weight $w_i^{S,M}$ of each match i 

For each match i, do  
Calculate the 3D endpoint coordinates of the matching line segment;  
Calculate the distance $d_D$ between line segment $l_i^{T_{initial}}$ after the $T_{initial}$ translation and the matching spatial straight line of $l_i$;

end for

Calculating the variance $\sigma_D$ formula (12);  
for each match i, do  
Estimate static weight $w_i^{S,M}$ formula (11);

end for

For the case where the static object moves or the dynamic object moves slowly, the static weight of the line segment obtained by comparing the source frame S with the adjacent previous frame F is also included in the static weight of the current line. For reference [22], equation (13) expresses the calculation of this static weight.

$w_s(i) = \alpha w_t^{F,S} + (1 - \alpha) w_t^{S,M}$ (13)
Where \( w_{S}^{PS} \) is the static weight of the source frame and the previous frame segment, which is the initial term; \( w_{S}^{SM} \) is the static weight of the source frame and the target frame segment, which is the updated item.

Combining the static weights of multiple frames is based on two considerations: 1) Using the initial term, if the static object starts moving, the static line segment can become a dynamic line segment, and the new dynamic line segment can only be detected in the target frame; 2) Using the updated item, in the time interval of one frame, the moving distance of the moving object is small, and the distance may fall in the static feature caused by noise when the line segment is matched, which is not enough to distinguish the dynamic object. This paper gives certain initial and updated items, weight coefficient \( \alpha \) can be adjusted. The value of \( \alpha \) is set by empirical value: \( \alpha = 0 \), when the lines in the source frame \( S \) has no match in the previous frame \( F \); \( \alpha = 0.25 \), in other cases.

When there is no matched line segment in the source frame \( S \) and the previous frame \( F \), the \( \alpha \) value is 0, only the update item is considered.

In this paper, the dynamic and static linear features are distinguished according to the static weight of the calculated line features. When it is less than the given threshold, it is considered to be a dynamic linear feature, and when it is greater than the given threshold, it is considered to be a static linear feature, so the dynamic line features can be deleted after they are detected. Figure 3 is an example of static weight estimation in some sequences of the TUM datasets [23].

![Figure 3. Line feature extraction and static weighting examples.](image)

The 1st column: the original RGB images. The 2nd column: LSD line features (red lines). The 3rd column: results of lines after static weighting (green for static lines). The 4th column: results of static lines and feature points.

3.3.4. Matches based on static line features and points features. In order to reduce the influence of dynamic objects on pose estimation, this paper removed the dynamic line features, the static linear features and point features are used to solve the camera pose.

4. Experiment and result

4.1. Experiments
In order to test the empirical accuracy and robustness of the method, we performed a comprehensive evaluation of the TUM data set. The data set contains 39 sequences that are captured by Kinect or Xtion in two different indoor environments. It consists of a composite captured color and depth image with a real trajectory from the motion capture system. In the fr3 series (including the sitting xyz, sitting halsphere, sitting rpy, sitting static, walking xyz, walking halsspace, walking rpy, walking static, etc.), camera always faces the table, the trajectories of the camera movement and the ways people moving are different in different sequences, which is a typical dynamic scene sequence set. Among them, "sitting" belongs to low dynamic environment, "walking" belongs to high dynamic
environment. At the same time, in order to verify the performance of this algorithm in static environment, we also performed experiments in static environments of fr2/desk and fr3/long office.

The relative pose error (RPE) and absolute trajectory error (ATE) are employed as the performance metrics, which are usually used to evaluate the odometry method.

All the experiments were obtained on a laptop with Intel i7 CPU (2.8GHz) and 8 GB RAM. It should be noted that for the computations only the CPU was utilized in the proposed approach.

In order to verify the accuracy of the visual odometer, this paper uses absolute pose error (ATE) as the evaluation index. We compared the method in this paper with the DVO, ORB point feature method and RGBD-SLAM method, and the comparison results are shown in Table 1. The first column is a sequence of TUM data sets, which are divided into static environment, low dynamic environment and high dynamic environment.

4.2. Results analysis
It can be seen from Table 1 that in the low dynamic environment, the performance of this method is improved compared with the DVO method; In the high dynamic environment, the performance of this method is significantly better than the ORB point feature method, which significantly reduces the tracking error.

This method can maintain more accuracy in camera pose estimation in dynamic environments. There are two main reasons: 1) Using linear features on color images, it can effectively find feature matching, which is more conducive to more accurate estimation of pose; 2) Using the static weight method of this paper to eliminate the line features of dynamic objects, effectively reduced the influence of dynamic objects on the algorithm.

Table 1. Comparison of absolute trajectory error (the unit is meter) between proposed method and three state-of-the-art methods.

| Sequence          | Proposed method | DVO  | ORB  | RGBD SLAM |
|-------------------|-----------------|------|------|-----------|
| Static            |                 |      |      |           |
| fr1/xyz           | 0.013           | 0.014| 0.031| 0.017     |
| fr1/desk2         | 0.025           | 0.020| 0.096| 0.026     |
| fr3/office        | **0.012**       | 0.035| 0.086| 0.031     |
| Low dynamic       |                 |      |      |           |
| fr3/sitting xyz   | 0.014           | 0.019| 0.056| **0.011** |
| fr3/sitting halfspace | **0.021**     | 0.038| 0.073| **0.021** |
| High dynamic      |                 |      |      |           |
| fr3/walking rpy   | **0.032**       | 0.112| **0.102** | 0.035     |
| fr3/walking xyz   | **0.014**       | 0.043| 0.126| 0.038     |
| fr3/walking halfspace | **0.012**    | 0.035| 0.086| 0.031     |

5. Conclusions
In this paper, a method based on line features for dynamic environment RGB-D visual odometer is proposed. This method uses the point features to estimate the initial pose, calculates the static weight of the line features based on the initial pose estimation, which effectively distinguishes and eliminates the dynamic line features from the matching line features, and finally estimates the more accurate camera pose through the static line features. The experimental results on the public data set show that, compared with the existing ORB point feature method, this method reduces the trajectory errors in challenging dynamic environments. Therefore, the proposed method effectively reduces the influence of dynamic objects on the pose estimation of the camera, also makes the visual odometer more accurate and robust in a dynamic environment.

References
[1] Kerl C, Sturm J, et al. 2013 Robust odometry estimation for RGB-D cameras IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE 3748-3754
[2] Whelan T, Johannsson H, et al. 2013 Robust real-time visual odometry for dense RGB-D
mapping IEEE International Conference on Robotics and Automation Piscataway, USA: IEEE 5724-5731

[3] Kerl C, Sturm J, et al. 2013 Dense visual SLAM for RGB-D cameras IEEE/RSJ International Conference on Intelligent Robots and Systems Piscataway, USA: IEEE 2100-2106

[4] Gutierrez-Gomez D, Mayol-Cuevas W, et al. 2016 Dense RGB-D visual odometry using inverse depth Robotics and Autonomous Systems 75: 571-583

[5] Jaimez M, Kerl C, et al. 2017 Fast odometry and scene flow from RGB-D cameras based on geometric clustering IEEE International Conference on Robotics and Automation Piscataway, USA: IEEE 3992-3999

[6] Von Gioi R G, Jakubowicz J, et al. 2010 LSD: A fast line segment detector with a false detection control IEEE transactions on pattern analysis and machine intelligence 32(4): 722-732

[7] Zhang L, Koch R 2013 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

[8] Lu Y, Song D 2015 Robust RGB-D Odometry Using Point and Line Features 2015 IEEE International Conference on Computer Vision (ICCV) IEEE Computer Society

[9] Gomezojeda R, Jiménez J G 2016 Robust stereo visual odometry through a probabilistic combination of points and line segments IEEE International Conference on Robotics & Automation

[10] Sun Y, Ming L, et al. 2017 Improving RGB-D SLAM in dynamic environments: A motion removal approach Robotics & Autonomous Systems 89(Complete):110–122

[11] Urgaonkar R, Neely M J 2011 Network Capacity Region and Minimum Energy Function for a Delay-Tolerant Mobile Ad Hoc Network IEEE/ACM Transactions on Networking 19(4):1137-1150

[12] Sun Y X, Liu M, et al. 2017 Improving RGB-D SLAM in dynamic environments: A motion removal approach Robotics and Autonomous Systems 89: 110-122

[13] Mouysset S, Zbib H, et al. 2013 Physics in Medicine & Biology 58(19):6931

[14] Wang Y, Huang S 2015 Towards dense moving object segmentation based robust dense RGB-D SLAM in dynamic scenarios International Conference on Control Automation Robotics & Vision

[15] Kim D H, Kim J H 2017 Effective Background Model-Based RGB-D Dense Visual Odometry in a Dynamic Environment IEEE Transactions on Robotics 32(6):1565-1573

[16] Majdik A L, Tamas L, et al. 2010 Visual odometry system to build feature based maps for mobile robot navigation Control & Automation

[17] She J, Xu R, Ning C 2015 Image stitching technology based on ORB and improved RANSAC algorithm Journal of Jiangsu University of Science & Technology

[18] Saini I, Singh D, et al. 2013 Delineation of ECG Wave Components Using K-Nearest Neighbor (KNN) Algorithm: ECG Wave Delineation Using KNN Tenth International Conference on Information Technology: New Generations

[19] Rusinkiewicz S, Levoy M 2002 Efficient Variants of the ICP Algorithm International Conference on 3-d Digital Imaging & Modeling

[20] Bartoli A, Sturm P 2004 The 3D line motion matrix and alignment of line reconstructions International Journal of Computer Vision 57(3): 159-178

[21] Kerl C, Sturm J, et al. 2013 Dense visual SLAM for RGB-D cameras IEEE/RSJ. International Conference on Intelligent Robots and Systems Piscataway, USA: IEEE 2100-2106

[22] Li S, Lee D 2017 RGB-D SLAM in dynamic environments using static point weighting IEEE. Robotics and Automation Letters 2(4): 2263-2270

[23] Sturm J, Engelhard N, et al. 2012 A benchmark for the evaluation of RGB-D SLAM. Systems IEEE/RSJ International Conference on Intelligent Robots and Systems Piscataway, USA: IEEE 573-580