1 Introduction

For embedded systems it is useful to use a traditional method for the Visual Odometry or for the Visual SLAM. These traditional approaches are subdivided into 2 categories:

1. **feature-based (or indirect) methods**.
2. **direct methods**. The paper is on these type, because it is more accurate and robust than the other also in a low-textured scenario.

2 Paper aims

There are two contribution that the paper want to give:

1. Extend the photometric error to lines,
2. Reduce the computational complexity using two-step appraoch composed by two optimization problems.

3 Related works

(Io non citerei gli altri metodi di line detection, ma solo il direct che è quello che analizziamo) Direct methods minimize the photometeric error to estimate camera poses and point depths. In some works, in order to correct the drift of the robot they added a close-loop. Since, the photometeric error is to measure the difference between the intensity values of points in the reference and target image, it is not defined for lines but only for points. In general, the collinear constrain is used to regulate the depth estimation. But, using this constraint plus the photometric error, there is

- no guarantee to reach the optimal solution,
- huge computational cost.

Many researchers tried to solve these two problems. Therefore, they reduce the complexity of the problem, fixing some parameters or initializing the collinear points. In both case, there are one problem. In the first case we can reach a suboptimal solution instead of an optimal one. In the second case, the results depend on the inizialization of the collinear points.

4 Preliminaries

4.1 Photometric Error

To understand the photometric error, we need two images I_i and I_j called reference and target image. Suppose that $\boldsymbol{x}$ is observed by I_i (so $\boldsymbol{x} \in \Omega_i$, where Ω_i is the domain of image I_i), and that it is also observed in image I_j at $\boldsymbol{x'} \in \Omega_j$. The relation between $\boldsymbol{x}$ and $\boldsymbol{x'}$ is:

$$\mathbf{x'} = \prod_c (\mathbf{R}_{ij} \prod_c^{-1} (\mathbf{x}, d) + \mathbf{t}_{ij}) \tag{1}$$

where: $\prod_c$ and $\prod_c^{-1}$ are the projection and back-projection function with the camera intrinsic parameter c, $\mathbf{R}_{ij}$ and $\mathbf{t}_{ij}$ are the rotation and translation from i to j, and d is inverse depth of the point.

With this relation we can understand the photometric error for a point:

$$E_{xj} = \sum_{x \in N_x} w_x ||(I_j[\boldsymbol{x'}] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i[\boldsymbol{x}] - b_i)||_y \tag{2}$$

where t_i and t_j are the exposure time of I_i and I_j, a_i, a_j, b_i, b_j are the affine brightness transform parameters, w_x is a gradient dependent weighting factor, N_x is a set of neighbothood of $\boldsymbol{x}$, and $|| \cdot ||_y$ is the Huber norm.

4.2 Plücker Coordinates

A 3D line can be represented by a Plücker coordinates. Given two points p_1 and p_2 on a 3D line, the 3D coordinates of the line are:

$$\boldsymbol{L} = [\boldsymbol{m}; \boldsymbol{d}], \quad \boldsymbol{m} = \boldsymbol{p_1} \times \boldsymbol{d}, \quad \boldsymbol{d} = \frac{\boldsymbol{p_2} - \boldsymbol{p_1}}{||\boldsymbol{p_2} - \boldsymbol{p_1}||_2} \tag{3}$$

These coordinates are homogeneous, we normalize by d. We can can write the distance from a point $\boldsymbol{p}$ to $\boldsymbol{L}$ as:

$$e(\boldsymbol{L}, \boldsymbol{p}) = \boldsymbol{m} - \boldsymbol{p} \times \boldsymbol{d} \tag{4}$$

5 Model

Given 3 points in image I_i that stay in the same line l, let call d_1 and d_2 the inverse depth of the two endpoints of the line l. We can calculate the p_1 and p_2 using the depths and the back-projection function. Now we can calculate the $\boldsymbol{m}$ and $\boldsymbol{d}$ and so we can define the line $\boldsymbol{L}$ (that is a function of d_1 and d_2). Given a point $\boldsymbol{x}$ in $\boldsymbol{L}$ we can calculate the 3D projection $\boldsymbol{p_x}$ in a closed-form solution using the line, d_1, d_2 and the camera intrinsic matrix $\mathbf{K}$.

We can define the photometric Error for Line. We can calculate it by calculate the photometric error for the two endpoints and for a set of internal points sampled.

Firstly, we need to calculate the error for some internal points. Therefore, we define $\boldsymbol{l}$ the line see by the image I_i (a projection of the 3D line $\boldsymbol{L}$). At this point, we can take a internal point $\boldsymbol{x}$, then we can calculate $\boldsymbol{p_x}$, after we can project it into I_j, in this way we have $\boldsymbol{x}$ and $\boldsymbol{x'}$ and so we can calculate the photometric error for that point. Let define it as E^l_{xj}.

For the endpoints $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ of $\boldsymbol{l}$ we can calculate the pthomoetric error for both points. Moreover, assume N internal points of $\boldsymbol{l}$, which forms a set X. Hence, we can formulate the error for the 2D line $\boldsymbol{l}$ as:

$$E_{lj} = E_{x_1 j} + E_{x_2 j} + \sum_{x \in X} E^l_{xj} \tag{5}$$

Note that E_{lj} depends only on the inverse depth of x_1 and x_2 not on the number of points sampled from $\boldsymbol{l}$. This because for the calculation of $\boldsymbol{p_x}$ we need only d_1 and d_2.

Now we can define the collinear constraint for line association. We need to add a collinear constraint in order to regulate the depths of the endpoints. Suppose that $\boldsymbol{L}$ is seen by $M_{\boldsymbol{L}}$ poses whose indices are in set $O_{\boldsymbol{L}}$. Let define, $i \in O_{\boldsymbol{L}}$ and the 2D line observation of $\boldsymbol{L}$ as $\boldsymbol{l}_i$. Let define $\boldsymbol{p}_{i,1}$ and $\boldsymbol{p}_{\boldsymbol{i},\boldsymbol{2}}$ to represent the two 3D endpoints for $\boldsymbol{l}_i$, these two points are in the local camera coordinate system. We can transport them into the global coordinate system using $\boldsymbol{T}_i^{-1}$. At this point, let call $\boldsymbol{q}_{i,1}$ and $\boldsymbol{q}_{i,2}$ the global coordinates. This work in order to find lines use a LSD (Line Segment Detection). Each line is associated with a support region. Let ρ_i represent the width of the support region of $\boldsymbol{l}_i$. Therefore, ρ_i represent also the uncertainty of the two endpoints for the line $\boldsymbol{l}_i$, this. also, impact the uncertainty of the 3D endpoints $\boldsymbol{q}_{i,1}$ and $\boldsymbol{q}_{i,2}$. Therefore, they use $\frac{1}{\rho_i}$ to weigh the collinear constraint. Furthermore, we can define the collinear constriant as:

$$E_{\boldsymbol{L}} = \sum_{i \in O_{\boldsymbol{L}}} \frac{1}{\rho_i} (||e(\boldsymbol{L}, \boldsymbol{q}_{i,1})||_2^2 + ||e(\boldsymbol{L}, \boldsymbol{q}_{i,2})||_2^2) \tag{6}$$

Where $e(\cdot)$ is the distance between $\boldsymbol{L}$ and $\boldsymbol{q}_{i,j}$ for $j = 1, 2$.

We can compute the model of our approach. That is the combination of the photometric error for points, line and the collinear constraint for line association. Therefore, it is:

$$E = \sum_{i \in F} \sum_{\boldsymbol{p} \in P_i} \sum_{j \in obs(\boldsymbol{p})} E_{\boldsymbol{p}j} + \sum_{i \in F} \sum_{\boldsymbol{l} \in L_i} \sum_{j \in obs(\boldsymbol{l})} E_{\boldsymbol{l}j} + \sum_{\boldsymbol{L} \in L} E_{\boldsymbol{L}} \tag{7}$$

Where, the first term is for all points in a photo, the second is for all line in a photo and the third term is the collinear constraint. Moreover, i runs over all keyframes F; P_i and L_i are the sets of 2D points and 2D line in keyframe i; $obs(\boldsymbol{p})$ and $obs(\boldsymbol{l})$ are the sets of keyframes where $\boldsymbol{p}$ and $\boldsymbol{l}$ are visible; L is the set of 3D lines. Lastly, for $obs(\boldsymbol{l})$, we consider that $\boldsymbol{l}$ is visible at keyframe j, if any sampled points on $\boldsymbol{l}$ are visible at keyframe j.

Now we need to do the windowed optimization. In this way we can balance accuracy and efficiency. They adopt a method similar to the incremental bundle adjustment with Schur matrix, in this way they can remove wrong associations and large photometric errors. Moreover, the error that we want to minimize is the total error defined up.

The last aspect is the two-step minimization. We need to consider 7. We need to notice that the third term correlates points, lines and poses. Therefore, if camera poses and the inverse depths of the 2D endpoints are given, this term is equivalent to fitting 3D lines to sets of points. On the other hand, if the parameters of the line are fixed, the resulting cost function 7 for pose and inverse depths can be minimized only considering the photometric error. Moreover, this algorithm always converges. This happen because first we minimize the third term and the we perform LM in order to find a better values for poses and inverse depths.

6 Front-end

Going more into details, we need to understand the3D line initialization and the 2D line combination.

3D line Initialization Given a new 2D line $\boldsymbol{l}$ we track the sampled 2D points in a subsequent image minimizing the photometric error. This set build up a set of correspondences $\{\boldsymbol{x} \leftrightarrow \boldsymbol{x'}\}$, but given to the ambiguity along line $\boldsymbol{l}$ and the pose error, these correspondences are not well accurate. However, these can generate an accurate line-line correspondences. We fit a 2D line $\boldsymbol{l'}$ for the tracked points $\boldsymbol{x'}$. At this point, we can use line triangularization in order to calculate the 3D line. After we can calculate or update the inverse depths of the endpoints of $\boldsymbol{l}$. Moreover, when a new line association is available we update the 3D endpoints by averaging. Furthermore, lines with large photometric errors can be removed.

2D line Combination They merge parallel lines that are too close each other. They sum up the magnitude of the gradient along the line and keep the one with the largest magnitude.

7 Results and conclusion

As shown in the images their algorithm has better results than the stat-of-art algorithms. Moreover, they compare different errors, like alignment error, rotation drift and scale drift. For all the approaches they use the same dataset and they extend DPLVO in order to have the same front-end of their approach (DPLVO++). Moreover, they also consider two variations of their algorithm:

- EDPLVO-LA: the collinear term in 7 is removed
- EDPLVO_Joint: they use 4 DoF representation to parametrize the 3D line.

Since they want to reduce the computational complexity, they also consider the runtime of the tried method. Their approach is the fastest among the one in the state-of-the-art with 96ms for the EDPLVO_Joint case that is the heaviest one.
Their algorithm significantly decreases the number of variables in the optimization and makes the collinearity exactly met. Furthermore, they introduce a two-step optimization method to speed up the optimization and prove its convergence.

8 Consideration regarding the poster

I think that we can place everything into 3 columns:

1. Column 1:
   (a) Goal of the paper
   (b) preliminaries, in this way we can show something
   (c) Fig 2
2. Column 2:

(a) Fig 3, in order to understand the initial idea
(b) the formulation of the photometric error for lines
(c) collinear constraint and the total error (we can plot only the total error and we can say something about the collinear constraint + the optimization with windows)
(d) the two-steps used to solve the optimization problem.

3. Column 3:
    (a) Some graph with the results
    (b) a small conclusion.