Scale Drift-Aware Large Scale Monocular SLAM

Huangyao

2015-09-15

内容

1. 论文主要工作

2. 基础知识

3. SLAM 系统

4. 闭环检测

5. 参考资料

一、 论文主要工作

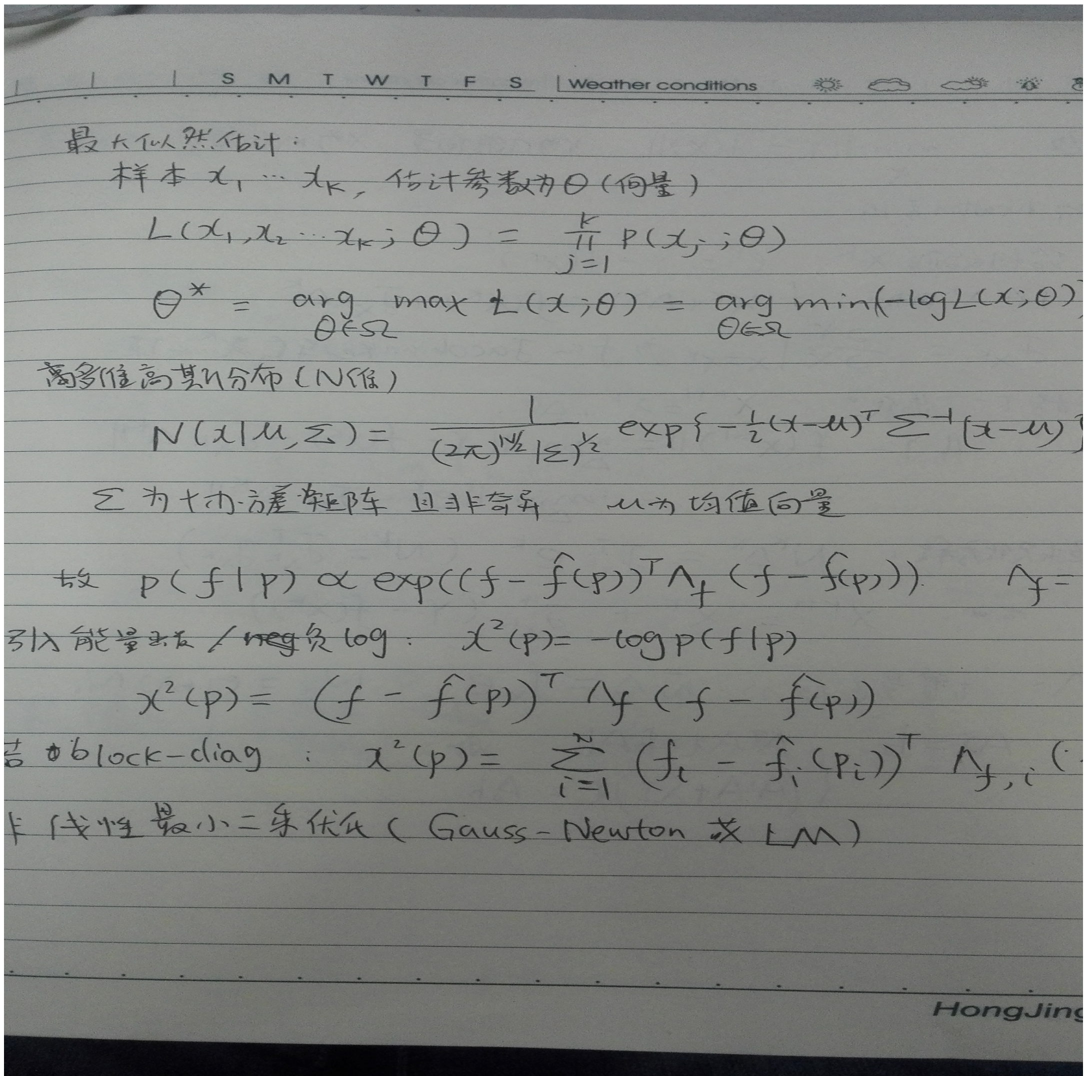
1、不同于双目以及立体SLAM，monoSLAM存在尺度问题

2、基于姿态图优化的技术来纠正rotation,translation, scale偏移

3、scale偏移在loop closures解决

二、 基础知识

最大似然估计



二、 基础知识

最小二乘与牛顿迭代法

S M T W T F S Weather conditions

问题： $\min_{X} \|Y - f(X)\|$ Y为n维向量，X为m维向量。

一阶 Newton 方法：

给初始解 X^k， $e^k = Y - f(X^k)$

f在X^k附近 $f(X^k + \Delta^k) = f(X^k) + J_{X^k}\Delta^k$

$J_{X^k} = \frac{\partial Y}{\partial X}\Big|_{X=X^k}$ 为f的Jacobian矩阵在X^k的值.

寻找下一个迭代点： $X^{k+1} = X^k + \Delta^k$.

$$\|Y - f(X^{k+1})\| = \min_{\Delta^k} \|Y - f(X^k) - J_{X^k}\Delta^k\|$$
$$= \min_{\Delta^k} \|e^k - J_{X^k}\Delta^k\|$$

正规化方程： $N^k\Delta^k = J_{X^k}^T e^k$ $(N^k = J_{X^k}^T J_{X^k})$

故 $X^{k+1} = X^k + J_{X^k}^{+}(Y - f(X^k))$.

二 LM： 增量正规方程 $N\Delta = J_X e$. $N_{ii} = (1+\lambda)N_{ii}$

$$Ax = b \quad \begin{cases} A^TAx = Ab \\ (A^TA + \lambda I)x = Ab \end{cases}$$

HongJing paper

二、 基础知识

旋转矩阵与李代数（李群）

$$\mathtt{T} = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathtt{0} & 1 \end{bmatrix} \quad \text{with} \quad \mathtt{R} \in \mathbf{SO}(n)\ , \quad \mathbf{t} \in \mathbb{R}^n$$

李代数se(n)可表示旋转矩阵；在R3空间中，可用6-vectors表示（w,v)^T
W=(w1,w2,w3)表示绕三轴旋转，v为平移变换t的旋转表示方式；
se(3)李代数可以通过指数映射为SE(3)群（属于李群）；

$$\exp_{\mathbf{SE}(3)}(\omega, v) := \begin{bmatrix} \exp_{\mathbf{SO}(3)}(\omega) & \mathtt{V}v \\ \mathtt{0} & 1 \end{bmatrix} = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathtt{0} & 1 \end{bmatrix}. \tag{10}$$

Here, $\exp_{\mathbf{SO}(3)}(\omega) = \mathtt{I} + \frac{\sin(\theta)}{\theta}(\omega)_\times + \frac{1-\cos(\theta)}{\theta^2}(\omega)_\times^2$ is the well-known *Rodrigues' formula*, $\mathtt{V} = \mathtt{I} + \frac{1-\cos(\theta)}{\theta^2}(\omega)_\times + \frac{\theta-\sin(\theta)}{\theta^3}(\omega)_\times^2$, $\theta = ||\omega||_2$, and $(\cdot)_\times$ is an operator which maps a 3-vector to its skew-symmetric matrix. Since $\exp_{\mathbf{SE}(3)}$ is surjective, there exists also an inverse relation $\log_{\mathbf{SE}(3)}$.

二、 基础知识

Rodrigues' Rotation Formula

Rodrigues' rotation formula gives an efficient method for computing the rotation matrix $\mathrm{R} \in SO(3)$ corresponding to a rotation by an angle θ about a fixed axis specified by the unit vector $\hat{\omega} = (\omega_x, \omega_y, \omega_z) \in \mathbb{R}^3$. Then $\mathrm{R}_{\hat{\omega}}(\theta)$ is given by

$$\mathrm{R}_{\hat{\omega}}(\theta) = e^{\tilde{\omega}\theta} \tag{1}$$

$$= \mathrm{I} + \tilde{\omega}\sin\theta + \tilde{\omega}^2(1-\cos\theta) \tag{2}$$

$$= \begin{bmatrix} \cos\theta + \omega_x^2(1-\cos\theta) & \omega_x\omega_y(1-\cos\theta) - \omega_z\sin\theta & \omega_y\sin\theta + \omega_x\omega_z(1-\cos\theta) \\ \omega_z\sin\theta + \omega_x\omega_y(1-\cos\theta) & \cos\theta + \omega_y^2(1-\cos\theta) & -\omega_x\sin\theta + \omega_y\omega_z(1-\cos\theta) \\ -\omega_y\sin\theta + \omega_x\omega_z(1-\cos\theta) & \omega_x\sin\theta + \omega_y\omega_z(1-\cos\theta) & \cos\theta + \omega_z^2(1-\cos\theta) \end{bmatrix} \tag{3}$$

,

where I is the 3×3 identity matrix

and $\tilde{\omega}$ denotes the antisymmetric matrix with entries

$$\tilde{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{4}$$

Note that the entries in this matrix are defined analogously to the differential matrix representation of the curl operator.

Note that

$$\tilde{\omega}\,\omega = \mathbf{0}, \tag{5}$$

http://mathworld.wolfram.com/RodriguesRotationFormula.html

二、 基础知识

- Keen way: Rodrigues's formula!

$$\mathbf{x}' = \mathbf{x} + (\sin\theta)\, \hat{\mathbf{n}} \times \mathbf{x} + (1 - \cos\theta)\, \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \mathbf{x})$$

- Define "cross product matrix" N:

$$N = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$$

so that

$$N\mathbf{x} = \hat{\mathbf{n}} \times \mathbf{x}$$

$$\mathrm{T}_{(i+1)} = \exp_{\mathrm{SE}(3)}(\delta) \cdot \mathrm{T}_{(i)} \ .$$

http://www.cs.cmu.edu/afs/cs/academic/class/16741-s07/www/Lecture7.pdf

三、 SLAM系统构成

1、Front-End: 跟踪图片中的features，并且获取3D points对应点后估计姿态;

2、Back-End: 在一系列关键帧上执行Bundle Adjustment优化

三、 SLAM系统构成

相机投射函数

$$\hat{\mathbf{z}}(\mathtt{T}_i, \mathbf{x}_j) = \mathbf{proj}(\mathtt{K} \cdot \mathtt{T}_i \cdot \mathbf{x}_j) \ .$$

BA Optimisation

Feature Tracking and Pose Optimisation

主要任务是通过tracking估计当前相机姿态pose Ti，同时处理特征和关键帧初始化问题；

三、 SLAM系统构成

Input：a set of keyframe poses Tkey:i and 3D points xj ∈ X which are associated with 2D measurements zkey:i,j .

Step1：估计当前帧和先前帧之间的相对的姿态Delta_Ti,借助 dense variational optical flow on GPU；

Step2：根据delta_Ti 来估计当前帧的姿态

$$\mathbf{T}_i^{[0]} = \Delta\mathbf{T}_i \cdot \mathbf{T}_{i-1}^{-1}.$$

Step3： 根据估计的姿态，评估地图中所有landmarks的可能看到的map point构成一个集合；

Step4：Active search。搜索的中心根据投射函数确定，搜索的区域大小根据Ti和xj的不确定性；

Step5: template matching using normalised cross-correlation 且只在FAST特征点上匹配

Output: a set of 2D-3D matches between image locations zk and feature points xk.

最后：后端使用BA（LM)优化pose Ti, 代价函数为pseudo-Huber cost function(robust kernel)

$$\chi^2(\mathrm{T}_i) = \sum_k \left(\mathbf{z}_k - \hat{\mathbf{z}}_k(\mathrm{T}_i, \mathbf{x}_j)\right)^2$$

三、 SLAM系统构成

特征初始化

our method employs a set of information filters. Each filter estimates the position of the a single landmark given the current pose estimate.

关键帧dropping

四、闭环检测

Step1 根据特征描述子，创建当前帧和loop帧SURF特征匹配对

Step2 根据当前帧可以在loop帧中看到的三维特征点创建dense surface model

Step3 根据dense surface模型可以计算surf特征的深度信息

Step4 根据2D-3D surf对应关系，采用P4P+RANSAC算法计算出6 DOF的 loop constraint T(loop）

四、闭环检测

Loop Closure Correction

1、Loop closing问题可以看做一个large BA问题，消耗时间多；而且非凸，可能陷入局部最小；
2、One solution would be to optimise over relative constraints between poses using pose-graph optimisation
3、6DOF约束可以有效地纠正旋转和平移的漂移问题，但不处理尺度漂移；因此须使用S（7自由度）来进行优化；
4. 必须将6DOF的Ti转换为7DOF的Si; 只有闭环约束时s(loop)!=1

$$\mathbf{r}_{i,j} = \log_{\mathrm{Sim}(3)}(\Delta \mathrm{S}_{i,j} \cdot \mathrm{S}_i \cdot \mathrm{S}_j^{-1}) \ . \tag{23}$$

The graph of similarity constraints is optimised by minimising:

CSparse library.

$$\chi^2(\mathrm{S}_2, ..., \mathrm{S}_m) = \sum_{i,j} \mathbf{r}_{i,j}^\top \Lambda_{\Delta \mathrm{S}i,j} \mathbf{r}_{i,j} \tag{24}$$

5、将map中得point位置进行优化

$$\mathbf{x}_j^{\mathrm{cor}} = (\mathrm{S}_i^{\mathrm{cor}})^{-1}(\mathrm{T}_i \mathbf{x}_j)$$

五、参考资料

Strasdat H, Montiel J M M, Davison A J. Scale Drift-Aware Large Scale Monocular SLAM[C]//Robotics: Science and Systems. 2010, 2(3): 5.