LSD-SLAM中运动参数的求取过程

(2016/4/24 第一次创建, 2016/4/26 更新目标函数为lsd-slam当中所采用的目标函数)

LSD_SLAM采用了Compositional Approach的方法来求取关键帧相对当前帧的运动参数p，即旋转和平移。

由于关键帧已经采用了关键帧到上一帧的运动参数作为关键帧到当前帧的运动参数的估计值$\tilde{p}$，将点云投影到了当前帧，所以下一步应该求的是一个运动参数Δp。
于是我们有优化的目标函数

$$E(\Delta p) = \sum_i [\tilde{I_1}(\tilde{x}(x_i;\Delta p)) - I_0(x_i)]^2 \tag{1}$$

其中I_0为关键帧, $\tilde{I_1}$为关键帧的点warp到当前帧上的位置在当前帧上的值. 要在(1)式中求解参数Δp，属于非线性最小二乘问题。我们对$\tilde{I_1}(\tilde{x}(x_i;\Delta p))$做一阶泰勒展开，得到

$$\tilde{I_1}(\tilde{x}(x_i;\Delta p)) = \tilde{I_1}(\tilde{x}(x_i;0)) + \frac{\partial \tilde{I_1}}{\partial \tilde{x}} \cdot \frac{\partial \tilde{x}}{\partial p} \cdot \Delta p \tag{2}$$

令$\frac{\partial \tilde{I_1}}{\partial \tilde{x}} \cdot \frac{\partial \tilde{x}}{\partial p} = \tilde{J_1}(x_i)$，我们将$(2)$代入$(1)$，可得

$$E(\Delta p) \approx \sum_i [\tilde{J_1}(x_i)\Delta p + e_i]^2 \tag{3}$$

(1)中的非线性最小二乘问题被转化为(3)中的线性最小二乘问题, 其中$e_i = \tilde{I_1}(\tilde{x}(x_i;0)) - I_0(x_i)$. 我们在$(3)$中对$\Delta p$求偏导，并令偏导的结果为零，可得

$$\sum_i \tilde{J_1}^T(x_i)\tilde{J_1}(x_i)\Delta p = -\sum_i \tilde{J_1}^T(x_i)e_i \tag{4}$$

令(4)中的$\sum_i \tilde{J_1}^T(x_i)\tilde{J_1}(x_i) = A$，$-\sum_i \tilde{J_1}^T(x_i)e_i = b$，有了$A$和$b$，我们就可以得到$\Delta p$。

至于$\tilde{J_i}(x_i)$的求解, 在LSD_SLAM中, 采用了$\vec{\omega} = \theta \cdot \hat{n} = (\omega_x, \omega_y, \omega_z)$来表示绕着轴$\hat{n}$旋转$\theta$角度. 根据罗德里格斯公式, 有

$$R(\hat{n},\theta) = I + sin\theta[\hat{n}]_\times + (I - cos\theta)[\hat{n}]_\times \tag{5}$$

其中$[\hat{n}]_\times$为$\hat{n}$的内积矩阵,

$$[\hat{n}]_\times = \begin{bmatrix} 0 & -\hat{n}_z & \hat{n}_y \\ \hat{n}_z & 0 & -\hat{n}_x \\ -\hat{n}_y & \hat{n}_x & 0 \end{bmatrix} \tag{6}$$

在旋转角度θ非常小的情况下, 我们可以得到

$$R(\vec{\omega}) = I + \theta[\hat{n}]_\times = I + [\vec{\omega}]_\times \tag{7}$$

于是我们有

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & -\hat{\omega}_z & \hat{\omega}_y \\ \hat{\omega}_z & 1 & -\hat{\omega}_x \\ -\hat{\omega}_y & \hat{\omega}_x & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right) \tag{8}$$

我们对$\tilde{x}$和$\tilde{y}$, 分别求其在$p = (0,0,0,0,0,0)$处对$p = (t_x, t_y, t_z, \hat{\omega}_x, \hat{\omega}_y, \hat{\omega}_z)$的偏导, 可得

$$\frac{\partial \tilde{x}}{\partial t_x} = \frac{f_x}{z} \tag{9}$$

$$\frac{\partial \tilde{y}}{\partial t_x} = 0 \tag{10}$$

$$\frac{\partial \tilde{x}}{\partial t_y} = 0 \tag{11}$$

$$\frac{\partial \tilde{y}}{\partial t_y} = \frac{f_y}{z} \tag{12}$$

$$\frac{\partial \tilde{x}}{\partial t_z} = -\frac{xf_x}{z^2} \tag{13}$$

$$\frac{\partial \tilde{y}}{\partial t_z} = -\frac{yf_y}{z^2} \tag{14}$$

$$\frac{\partial \tilde{x}}{\partial \vec{\omega}_x} = -\frac{xyf_x}{z^2} \tag{15}$$

$$\frac{\partial \tilde{y}}{\partial \vec{\omega}_x} = -f_y(1+\frac{y^2}{z^2}) \tag{16}$$

$$\frac{\partial \tilde{x}}{\partial \vec{\omega}_y} = f_x(1+\frac{x^2}{z^2}) \tag{17}$$

$$\frac{\partial \tilde{y}}{\partial \vec{\omega}_y} = \frac{xyf_y}{z^2} \tag{18}$$

$$\frac{\partial \tilde{x}}{\partial \vec{\omega}_z} = \frac{-yf_x}{z} \tag{19}$$

$$\frac{\partial \tilde{y}}{\partial \vec{\omega}_z} = \frac{xf_y}{z} \tag{20}$$

即

$$\tilde{J}_1(x) = \begin{bmatrix} \frac{\partial \tilde{I}_1}{\partial x} & \frac{\partial \tilde{I}_1}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{f_x}{z} & 0 & -\frac{xf_x}{z^2} & -\frac{xyf_x}{z^2} & f_x(1+\frac{x^2}{z^2}) & \frac{-yf_x}{z} \\ 0 & \frac{f_y}{z} & -\frac{yf_y}{z^2} & -f_y(1+\frac{y^2}{z^2}) & \frac{xyf_y}{z^2} & \frac{xf_y}{z} \end{bmatrix} \tag{21}$$

其中$(\frac{\partial \tilde{I}_1}{\partial x}, \frac{\partial \tilde{I}_1}{\partial y})$为$\tilde{I}_1$的梯度.

$\tilde{J}_1(x)$的代码实现见`SE3Tracker.cpp`中的函数`calculateWarpUpdate()`.