LSD-SLAM

Created	@October 4, 2023 6:27 PM
Tags	

В основе **Large Scale Direct SLAM** лежит метод прямой визуальной одометрии (direct visual odometry), который оптимизирует геометрию напрямую на image intensities, что позволяет получить из изображения максимум информации, а не полагаться на ключевые точки или фичи.

Основные нововведения

Semi-dense depth filtering formulation - позволяет строить визуальную одометрию в режиме реального времени на CPU и смартфонах, но за счёт этого метода как такового мы только фиксируем движение камеры. **Статья, на которую ссылается автор**

Pose-graph optimization - мир представляется в виде кадров в вершинах графа, соединенных возможными изменениями позы (*мне кажется, это подразумевается под “pose-pose constraints”*). Граф оптимизируется при помощи библиотеки g2o

Авторы предлагают комбинировать:

1. **direct image alignment** coupled with filtering-based estimation of **semi-dense depth maps**
2. Граф с ключевыми кадрами в вершинах и рёбрами, обозначающими трехмерные изменения положения в пространстве.

А дальше идёт матан…

Условные обозначения:

1. R - латинские заглавные буквы - матрицы
2. ξ - малые буквы (*греческие?*) - векторы
3. $\Omega \subset \mathbb{R}^2$ - множество нормализованных пикселей изображения, то есть учитывающих сопутствующую калибровку камеры (*латеховская буква для вещественных чисел тут не отрисовывается, так что будет с бэкслэшэм, чтобы с матрицами не путать*)

4. $I : \Omega \to \mathbb{R}$ - изображение (*надеюсь, дальше пояснят, в чём суть этого отображения*)*Upd: Пояснения не было*, так что просто предположу, что мы отображаем каждый пиксель в какое-то конкретное вещественное число по его значениям каналов цветов. Самое простое – перемножить b*g*r, но, наверное, нам будет достаточно использовать ЧБ изображение, что уменьшит наш разброс значений с 2^32 до 2^8
5. $D : \Omega \to \mathbb{R}^+$ - per-pixel inverse depth map (попиксельное отображение обратной глубины)
6. $V : \Omega \to \mathbb{R}^+$ - inverse depth variance map (отображение дисперсии обратной глубины)
7. $d = z^{-1}$ - инверсия глубины точки z

Трансформация твёрдого тела (3D Rigid body transformation)

Описывается матрицей $G \in SE(3)$, которая включает вращение (R) и переход (t):Для оптимизации записи матрица отображается в вектор \R6:= SE(3)(G). Переход от кадра i к кадру j описывается обозначается ij, далее вводится операция конкатенации векторов: ij = ikkj = SE(3)(GikGkj) = SE(3)((ik)(kj)).Далее мы определяем функцию перехода для пикселя p за счёт вектора (*честно, я не понимаю, что за магия тут творится (особенно справа, где мы, согласно нотации выше, умножаем матрицу 2 на 2 на вектор длины 4 и получаем вектор длины 4, но в результате мы добавляем… глубину в изображение?*):

$$\mathbf{G} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \quad \text{with} \quad \mathbf{R} \in \mathrm{SO}(3) \text{ and } \mathbf{t} \in \mathbb{R}^3.$$

$$\omega(\mathbf{p}, d, \boldsymbol{\xi}) := \begin{pmatrix} x'/z' \\ y'/z' \\ 1/z' \end{pmatrix} \quad \text{with} \quad \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} := \exp_{\mathfrak{se}(3)}(\boldsymbol{\xi}) \begin{pmatrix} \mathbf{p}_x/d \\ \mathbf{p}_y/d \\ 1/d \\ 1 \end{pmatrix}.$$

Трехмерное преобразование подобия (3D Similarity Transformations)

Здесь определяется матрица S, которая описывает вращение, масштабирование и смещение:Операции для вектора здесь описываются

аналогичным образом, что и для трансформации твёрдого тела, с той лишь разницей, что \R7 (пространство SIM(3))(*откуда добавляется степень свободы несколько загадочно… Наверное (даже очень вероятно), логарифмирование матриц - операция совсем иного рода, нежели простое логарифмирование*)Статья по этой теме

$$\mathbf{S} = \begin{pmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \quad \text{with} \quad \mathbf{R} \in \mathrm{SO}(3),\ \mathbf{t} \in \mathbb{R}^3 \ \text{ and } \ s \in \mathbb{R}^+.$$

Weighted Gauss-Newton Optimization on Lie-Manifolds

(*пожалуй, тут без перевода*). Два изображения выравниваются при помощи метода Гаусса-Ньютона (что такое r, вообще не ебу, МАМА, МНЕ СТРАШНО): (*Ладно, спокойно… Тут ведь чётко отмечено, что вся формула за суммой - это и есть r2i().И вообще, осталось чуть матана… Вроде бы…*) Далее подбирается такое , которое минимизирует ошибку. Сначала считается (0) по описанным выше формулам.Затем считается left-multiplayed increment (n) путём нахождения минимума для второго приближения Гаусса-Ньютона для E:Где, J - производная сложенного остаточного вектора (stacked residual vector) r= (r1, ..., rn)T с учётом левоумноженного инкремента (left-multiplayed increment) и J^T J Gauss-Newton approximation of the Hessian of E. (метод Гаусса-Ньютона на Википедии)Далее мы получаем новую оценку через умножение с вычисленным инкрементом: (n+1)=(n)(n)Для устойчивости к отображений и других искажений было решено итеративно пересчитывать матрицу весов W = W((n)) , которая понижает вес самых больших остатков. Получили следующие формулы:В итоге мы получаем важную информацию о корреляциях между шумом в разных измерениях. Использоваться это будет в оптимизации графа и, *насколько я понял, в g2o этот метод УЖЕ, МАТЬ ВАШУ, РЕАЛИЗОВАН!!! Зачем я прожил эти 2 часа жизни?.. Ладно, будем надеяться, что хотя бы первые пункты из математической аннотации не были записаны впустую…*

$$E(\boldsymbol{\xi}) = \sum_i \underbrace{\left(I_{\mathrm{ref}}(\mathbf{p}_i) - I\left(\omega(\mathbf{p}_i, D_{\mathrm{ref}}(\mathbf{p}_i), \boldsymbol{\xi})\right)\right)^2}_{=: r_i^2(\boldsymbol{\xi})},$$

$$\delta\boldsymbol{\xi}^{(n)} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{r}(\boldsymbol{\xi}^{(n)}) \quad \text{with} \quad \mathbf{J} = \left.\frac{\partial \mathbf{r}(\boldsymbol{\epsilon} \circ \boldsymbol{\xi}^{(n)})}{\partial \boldsymbol{\epsilon}}\right|_{\boldsymbol{\epsilon}=0},$$

$$E(\boldsymbol{\xi}) = \sum_i w_i(\boldsymbol{\xi}) r_i^2(\boldsymbol{\xi}),$$

and the update is computed as

$$\delta\boldsymbol{\xi}^{(n)} = -(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}\mathbf{J}^T\mathbf{W}r(\boldsymbol{\xi}^{(n)}).$$

Распространение неопределённости (Propagation of Uncertainty)

Позволяет рассчитать неопределённость на выходе на основании степени неопределённости данных на входе:

Propagation of uncertainty is a statistical tool to derive the uncertainty of the output of a function $f(\mathbf{X})$, caused by uncertainty on its input $\mathbf{X}$. Assuming $\mathbf{X}$ to be Gaussian distributed with covariance $\boldsymbol{\Sigma}_\mathbf{X}$, the covariance of $f(\mathbf{X})$ can be approximated (using the Jacobian $\mathbf{J}_f$ of f) by

$$\boldsymbol{\Sigma}_f \approx \mathbf{J}_f \boldsymbol{\Sigma}_\mathbf{X} \mathbf{J}_f^T. \tag{11}$$

Общее описание LSD-SLAM, но уже более подробно:

1. Получается новое изображение и рассчитывается его трансформация твёрдого тела на основании прошлого кадра
2. Через фильтр глубины (пункт b) и либо ключевой кадр обновляется, либо заменяется (если камера перешла слишком далеко)
3. Если ключевой кадр был заменён, делается вывод, что карта глубины нового кадра не изменится, а значит его можно внедрить в глобальную карту с использованием коррекции при помощи расчёта трёхмерного преобразования подобия. Расчёт проводится с ближайшими ключевыми кадрами, сохранёнными ранее + непосредственным предшественником.

Для инициализации алгоритма достаточно задать случайную карту глубины и большую дисперсию

Карта представлена графом позиций, где каждая вершина i содержит изображение Ii:i \R, его обратную карту глубины Di:Di \R+и дисперсию обратной глубины Vi:Di \R+, причём глубина и её дисперсия определены только для части пикселей: Di . Рёбра содержат трехмерное преобразование подобия (3D Similarity Transformations) ij, а также матрицу ковариации ij (*тут ещё и ковариации вылезли… Что они значат вообще?..*)

1. Пункт (f.i) - получение нового изображенияНа основании данных прошлого кадра i и нового изображения j рассчитывается самая вероятная трансформация твёрдого тела jiпри помощи минимизации variance-normalized photometric errorДисперсия остатков 2rp(p,ji)считается при помощи распространения неопределенности (e.5)(*каким оно тут боком оказалось, если для этой сигмы итак вроде бы уже формула есть?..*). Далее мы *предполагаем (!?)* Gause Image Intensity noise 2IДля минимизации используется итеративно перевзвешиваемый метод оптимизации Гаусса-Ньютона (e.iv)

$$E_p(\boldsymbol{\xi}_{ji}) = \sum_{\mathbf{p}\in\Omega_{D_i}} \left\| \frac{r_p^2(\mathbf{p},\boldsymbol{\xi}_{ji})}{\sigma^2_{r_p(\mathbf{p},\boldsymbol{\xi}_{ji})}} \right\|_\delta \tag{12}$$

$$\text{with} \quad r_p(\mathbf{p},\boldsymbol{\xi}_{ji}) := I_i(\mathbf{p}) - I_j(\omega(\mathbf{p}, D_i(\mathbf{p}), \boldsymbol{\xi}_{ji})) \tag{13}$$

$$\sigma^2_{r_p(\mathbf{p},\boldsymbol{\xi}_{ji})} := 2\sigma_I^2 + \left(\frac{\partial r_p(\mathbf{p},\boldsymbol{\xi}_{ji})}{\partial D_i(\mathbf{p})}\right)^2 V_i(\mathbf{p}) \tag{14}$$

$$\left\|r^2\right\|_\delta := \begin{cases} \dfrac{r^2}{2\delta} & \text{if } |r| \le \delta \\ |r| - \dfrac{\delta}{2} & \text{otherwise.} \end{cases}$$

2. (f.ii) Расчёт карты глубины и работа с ключевыми кадрами.Смещение камеры расчитывается по формуле dist(ji) := TjiWji, где W - диагональная матрица весов (*мы их высчитывали в (e.iv) или это вообще новые веса, подбираемые случайно?..*)Если было принято решение заменить ключевой

кадр, по формулам из статьи пункта (b) вычисляем на основании карты глубину прошлого кадра карту нового.Если ключевой кадр не меняется, в текущем производится пересчёт карты глубины по формулам из статьи пункта (b)

3. Ограничения и прямое выравнивание изображений с учётом скейлинга.
   a. Если был выбран новый ключевой кадр, то зафиксированный сравнивается со своим предшественником и проводится минимизация ошибки для расчета трёхмерного преобразования подобия (в нём кроме фотометрической погрешности rp считается также погрешность глубины rd):Поиск минимального ji проводится также методом Гаусса-Ньютона

$$E(\boldsymbol{\xi}_{ji}) := \sum_{\mathbf{p}\in\Omega_{D_i}} \left\| \frac{r_p^2(\mathbf{p},\boldsymbol{\xi}_{ji})}{\sigma^2_{r_p(\mathbf{p},\boldsymbol{\xi}_{ji})}} + \frac{r_d^2(\mathbf{p},\boldsymbol{\xi}_{ji})}{\sigma^2_{r_d(\mathbf{p},\boldsymbol{\xi}_{ji})}} \right\|_\delta , \tag{17}$$

where the photometric residual r_p^2 and $\sigma^2_{r_p}$ is defined as in (13) - (14). The depth residual and its variance is computed as

$$r_d(\mathbf{p},\boldsymbol{\xi}_{ji}) := [\mathbf{p}']_3 - D_j([\mathbf{p}']_{1,2}) \tag{18}$$

$$\sigma^2_{r_d(\mathbf{p},\boldsymbol{\xi}_{ji})} := V_j([\mathbf{p}']_{1,2}) \left(\frac{\partial r_d(\mathbf{p},\boldsymbol{\xi}_{ji})}{\partial D_j([\mathbf{p}']_{1,2})} \right)^2 + V_i(\mathbf{p}) \left(\frac{\partial r_d(\mathbf{p},\boldsymbol{\xi}_{ji})}{\partial D_i(\mathbf{p})} \right)^2 , \tag{19}$$

where $\mathbf{p}' := \omega_s(\mathbf{p}, D_i(\mathbf{p}), \boldsymbol{\xi}_{ji})$ denotes the transformed point. Note that the Hu-

   b. Далее, если был добавлен новый ключевой кадр Ki в карту, находим ближайшие к нему 10 ключевых кадров Kj, ... , kjn, и при помщи алгоритма из этой статьи находим замыкания графа. Чтобы избежать ложных обанаружений, мы дополнительно проверяем разницу векторов между кадрами Kjk и Ki:Если разница достаточно мала, мы заключаем, что кадры Kjk и Ki образуют цикл и корректируем карту.

$$e(\boldsymbol{\xi}_{j_k i}, \boldsymbol{\xi}_{ij_k}) := (\boldsymbol{\xi}_{j_k i} \circ \boldsymbol{\xi}_{ij_k})^T \left(\boldsymbol{\Sigma}_{j_k i} + \mathrm{Adj}_{j_k i} \boldsymbol{\Sigma}_{ij_k} \mathrm{Adj}_{j_k i}^T \right)^{-1} (\boldsymbol{\xi}_{j_k i} \circ \boldsymbol{\xi}_{ij_k})$$

   c. Тут речь идёт об увеличении радиуса сближения для пространства SIM(3), но, насколько я понял, эти увеличения совсем незначительны

4. Оптимизация карты проводится в фоновом режиме на всех кадрах (*как я понял, W - каждый кадр, добавленный в итоговую карту*) по классическому методу из (e.iv):

convention from Sec. 2.2 – defined by (W defining the world frame)

$$E(\boldsymbol{\xi}_{W1} \dots \boldsymbol{\xi}_{Wn}) := \sum_{(\boldsymbol{\xi}_{ji}, \boldsymbol{\Sigma}_{ji}) \in \mathcal{E}} (\boldsymbol{\xi}_{ji} \circ \boldsymbol{\xi}_{Wi}^{-1} \circ \boldsymbol{\xi}_{Wj})^T \boldsymbol{\Sigma}_{ji}^{-1} (\boldsymbol{\xi}_{ji} \circ \boldsymbol{\xi}_{Wi}^{-1} \circ \boldsymbol{\xi}_{Wj}). \quad (21)$$

5. Итог… (*Мама, роди меня обратно!*) Общая концепция алгоритма ясна, а вот детали заставляют чувствовать неимоверную боль. очень надеюсь, что описаны функции в статье были именно потому, что это статья, а в реализации мы сможем пользоваться библиотечными функциями. Опишу буквально в паре слов алгоритм ещё раз и прикрепляю схему: получаем кадр и считаем для него и текущего ключевого смещение твёрдого тела в SE(3) пространстве, используя метод Гаусса-Ньютона; решаем на основании данных о смещении, нужно ли нам обновить данные текущего ключевого кадра или создать новый. Если создаём новый ключевой кадр, то прошлый добавляем в карту, посчитав для него смещение со скейлингом в SIM(3). По полученным смещениям и параметрам самих изображений (карты глубины) пытаемся обнаружить замыкания. После всего этого производим оптимизацию карты на основании подсчитанных смещений