BundleFusion 解析

2016-10-25 15:32:32 Xingyin-Fu 阅读数 8649 更多

#Method overview

BundleFusion 和之前所有的帧匹配（keyframe，每个 chunk 的第一帧），匹配采用 sift 描述子，这种方式和 SLAM 中一般通过邻近的帧计算 pose 不一样。

为了减少优化时优化的变量个数，采用 local-to-global 的位子优化策略。local 范围内，将全部帧按照划分成等大小的 chunk，先在 chunk 内做 pose 优化。然后在 global 范围内，所有的 chunk 的 pose 放在一起优化。这种 local-to-global 的方式，可以减少优化的变量个数，加速优化过程。

在建图部分，采用 voxel hashing 算法，不同的是，BundleFusion 中每个 chunk 的 pose 一直在做改变，pose 的改变需要映射到全局的 TSDF 模型中。方法是，挑选出 pose 改变量最大的 chunks（10 个），按照将帧融合到 TSDF 时的 pose，做 de-integration，将之前融合进 TSDF 模型中的数据，从 TSDF 模型中减掉。然后，按照优化之后的 pose，将 chunk 的数据 re-integrate 到 TSDF 模型中。

算法流程图：

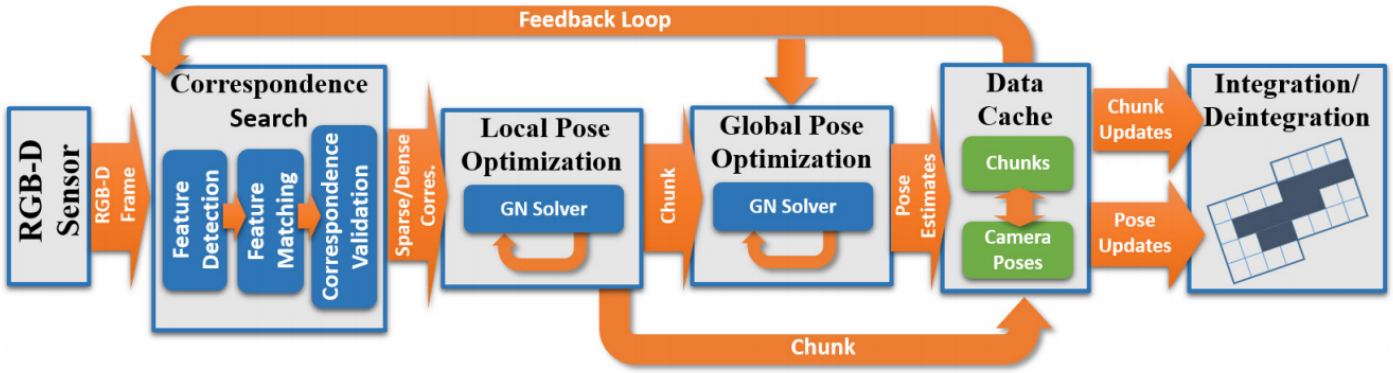

Fig. 2. Our global pose optimization takes as input the RGB-D stream of a commodity sensor, detects pairwise correspondences between the input frames, and performs a combination of local and global alignment steps using sparse and dense correspondences to compute per-frame pose estimates.

#Global Pose Alignment

对于新获取的帧，和之前的帧累计到一个 chunk 时（每 11 帧组成一个 chunk），先在 chunk 内匹配，做 local 优化，优化后，用 chunk 内的第一帧表示 chunk，chunk 内的所有帧的特征组合在一起，表示该 chunk 内特征，然后新的 chunk 和之前所有的 chunk 匹配，做 pose 优化，SIFT 特征提取和匹配都在 GPU 上做，计算一帧 SIFT 关键点和提取关键点描述子占用 4-5ms，匹配一次耗时大概 0.05 ms，特征匹配不可避免会有错误的匹配，作者设计了严格的筛选机制。

##Correspondence Filter

###Key point correspondence filter

对于帧 f_i 和 f_j，f_i 中 3D 点集 P，f_j 中 3D 点集 Q，并且的 pose 满足重投影误差小，并且用来计算 pose 的 3D 点，位置分布也好。匹配的特征是 greedily aggregated，对于每个新加入的匹配，采用 Kabsch 算法极小化匹配特征的 RMSD 误差计算 pose，然后，检测用来计算 pose 的两组特征 P_{cur} 和 Q_{cur} 分布是否满足要求（匹配的特征点都处在一条直线上，或者是 rotational symmetry），方法是计算特征分布的条件数，和 P_{cur}、Q_{cur} 间的协方差，如果条件数或者协方差很大，则系统认为不稳定，放弃新加入的匹配，如果对于计算的 pose T_{ij}，P_{cur} 和 Q_{cur} 间的冲投影误差太大，也放弃加入的匹配。对于新加入的匹配，依次这样做。如果最后得到的 pose 不成立，两帧之间所有的匹配都会被移除。

###Surface area filter

对于帧 f_i 的特征集合 P_{cur}和 f_j 的特征集合 Q_{cur}，将 3D 特征集合投到由两帧的主轴方向组成的平面上，如果投影区域的 bounding box 的面积面积不够大，两帧之间的匹配也被放弃。

###Dense verification

对于帧 f_i 和 f_j 抽样后大小 80×60 的图像，根据计算的 pose T_{ji} 采用投影算法找匹配，然后计算匹配点之间深度、法向量和颜色的差。

如果上述验证，帧 f_i 和 f_j 之间的匹配都可以满足，则认为两帧之间的匹配有效，建立的匹配用来做后续的 pose 优化。

##分层优化

BundleFusion 和 SFM 思路很像，为了应对系统实时性能的要求，优化分两级进行。按照邻接关系，将整个序列图像划分成等大小的

chunk（新帧加入时，数量累计到一个 chunk 才开始做 local 优化），每个 chunk 内 local 优化后，用 chunk 内的第一帧图像代表该 chunk，chunk 内所有帧提取的特征做融合，chunk 的特征用融合后的特征表示。chunks 间也建立匹配关系，做 global 优化。

###Local intra-chunk pose optimization
序列图像中每 11 帧组成一个 chunk，每个 chunk 间有一帧的交叠，通过交叠，可以将 chunks 的 pose 变换到同一个坐标系。在 chunk 内的每两帧之间，都做特征匹配，intra-chunk 的 pose 优化项，包括稀疏特征匹配误差，和 dense 匹配的误差，优化时，每帧的 pose 都设为单位矩阵。优化后，对于每两帧之间都做 dense verification，如果任意两帧之间的误差太大，则在 global 优化时，整个 chunk 的信息都会被放弃。

###Per-chunk keyframes
chunk local pose 优化做完后，计算表示该 chunk 的特征。对于空间一个点，chunk 内的帧可能有几个表示，需要做融合。方法是，根据 local pose 优化计算的 pose，将特征变换到同一个坐标系，把空间位置和描述子相似的特征合并为一个特征，特征的位置用所有特征的位置最小二乘拟合得到。特征合并后，intra-chunk 内的特征，描述子和匹配关系可以舍弃，chunk 内每帧的信息，在下采样后是被保留的，mapping 用的也是下采样后的图像，在 pose 优化后，pose 变化大帧从 TSDF 模型中做 de-integration 和 re-integration 都是采用的下采样后的图像，所以，chunk 内下采样后的图像不能被舍弃。

###Global inter-chunk pose optimization
global 优化同 chunk 内 local 的优化，global 优化时，chunk 的 pose 可以通过 chunk 间交叠的帧计算得到，global pose 优化后，所有帧的 pose 通过每个 chunk 的 pose 也可以计算得到。

###Pose alignment as energy optimization
优化项包括稀疏特征优化项和稠密优化项，稀疏特征优化项的目标函数是特征 3D 点的欧氏距离（不是通常用的 3D 到 2D 的重投影误差），目标函数如下：

$$E_{\text{sparse}}(\mathcal{X}) = \sum_{i=1}^{|\mathbf{S}|} \sum_{j=1}^{|\mathbf{S}|} \sum_{(k,l)\in\mathbf{C}(i,j)} \left\| \mathcal{T}_i \mathbf{p}_{i,k} - \mathcal{T}_j \mathbf{p}_{j,l} \right\|_2^2 .$$

###Dense alignment
稠密的优化项目标函数是常用的匹配点的点到平面距离几何误差，和匹配点的像素值误差。

有个问题是，优化的时候，特征点的位置是固定不变的（并不是 BA），inter-chunk 间的帧建立帧和帧之间的特征匹配，chunk 间建立 chunk 和 chunk 的特征匹配，上式中特征在帧空间，或者 chunk 空间的 3D 坐标是不变的，所以，在做 global 优化时，dense 匹配优化，会被稀疏特征优化重置（理解是dense 优化是迭代，包括匹配点的正确建立，随着迭代进行才可以正确建立，而稀疏特征的正确匹配不需要迭代，一次成功），只在最后扫描结束后在 global 优化时，才包含 dense 的优化项。

##Fast and Robust Optimization Strategy
优化时，优化变量包括所有帧的 pose，变量个数很大。作者采用 Gauss-Newton 法极小化非线性目标函数，采用 Preconditioned Conjugate Gradient 法求解方程组。

假设优化的非线性目标函数：

$$\mathcal{X}^* = \underset{\mathcal{X}}{\operatorname{argmin}} E_{align}(\mathcal{X}) .$$

$$E_{align}(\mathcal{X}) = \sum_{i=1}^{R} r_i(\mathcal{X})^2 .$$

residuals 写成向量形式：

$$\mathbf{F}(\mathcal{X}) = [\ \ldots,\ r_i(\mathcal{X}),\ \ldots\ ]^T$$

$$E_{refine}(\mathcal{X}) = ||\mathbf{F}(\mathcal{X})||_2^2$$

一阶 Taylor 展开：

$$\mathbf{F}(\mathcal{X}^k) = \mathbf{F}(\mathcal{X}^{k-1}) + \mathbf{J}_\mathbf{F}(\mathcal{X}^{k-1}) \cdot \Delta\mathcal{X}, \Delta\mathcal{X} = \mathcal{X}^k - \mathcal{X}^{k-1}$$

$$\Delta\mathcal{X}^* = \underset{\Delta\mathcal{X}}{\operatorname{argmin}} \underbrace{||\mathbf{F}(\mathcal{X}^{k-1}) + \mathbf{J}_\mathbf{F}(\mathcal{X}^{k-1}) \cdot \Delta\mathcal{X}||_2^2}_{E_{lin}(\Delta\mathcal{X})}$$

高斯牛顿求解线性方程组：

$$\mathbf{J}_\mathbf{F}(\mathcal{X}^{k-1})^T \mathbf{J}_\mathbf{F}(\mathcal{X}^{k-1}) \cdot \Delta\mathcal{X}^* = -\mathbf{J}_\mathbf{F}(\mathcal{X}^{k-1})^T \mathbf{F}(\mathcal{X}^{k-1})$$

对于上述等式求解，采用 Preconditioned Conjugate Gradient 法，计算时变量个数和 residual 项都很多，矩阵和向量的维度会很大，这里不会显式地计算矩阵的值，只计算其中的非零项，计算都在 GPU 中进行。

###Correspondence and Frame Filtering
为了移除错误的匹配，对于 frame 间的匹配计算误差的最大值，如果最大值太大，则两帧之间所有的匹配都会被移除。

#Dynamic Reconstruction
系统中，global pose 优化会改变所有帧的 pose，作者使用 de-integration 和 re-integration 根据 pose 的变化，更新 TSDF 值。TSDF 模型表示采用 voxel hashing 的形式，RGB-D 数据的 integration 和 de-integration 是对称的，integrate 的数据，可以按照 integrate 时的 pose，从模型中做 de-integration 减掉。

integration 操作：

$$\mathbf{D}'(\mathbf{v}) = \frac{\mathbf{D}(\mathbf{v})\mathbf{W}(\mathbf{v}) + w_i(\mathbf{v})d_i(\mathbf{v})}{\mathbf{W}(\mathbf{v}) + w_i(\mathbf{v})}, \ \mathbf{W}'(\mathbf{v}) = \mathbf{W}(\mathbf{v}) + w_i(\mathbf{v}).$$

de-integration 操作：

$$\mathbf{D}'(\mathbf{v}) = \frac{\mathbf{D}(\mathbf{v})\mathbf{W}(\mathbf{v}) - w_i(\mathbf{v})d_i(\mathbf{v})}{\mathbf{W}(\mathbf{v}) - w_i(\mathbf{v})}, \ \mathbf{W}'(\mathbf{v}) = \mathbf{W}(\mathbf{v}) - w_i(\mathbf{v}).$$

作者对于优化的位姿按照位姿的变化的大小排序，将 pose 变化最大的 10 帧数据先按照优化前的 pose 做de-integration，从模型中减掉，然后再按照优化后的 pose 做 re-integration。

#Limitations
特征坐标在优化时是保持不变的，如果 SIFT 特征偏移几个像素，或者深度图像中特征位置处噪声比较大，这种误差会造成最小二乘求解的 SIFT 特征点的位置不对，从而优化的 pose 有误差。理想情况下，系统应该做 BA 优化，把特征的坐标也放在优化中，但是，这样计算量会很大。

实测 BundleFusion 在重复结构纹理的时候，效果不是特别好；建立的三维结构可能因为上述原因，特征的位置不对，使得重建的三维结构也可能会有偏差。

文章参考：
“BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration”