# pcl\_kinfu\_app 目标函数修改设计报告

**Changelog**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 变更人 | 变更说明 | 变更时间 |
| V1.0 | 张琛 | 初稿 | 2015/01/18 |
| V1.1 | 张琛 | 增加使用轮廓点集优化配准的实现方法描述 | 2015/01/23 |

目录

[pcl\_kinfu\_app 目标函数修改设计报告 1](#_Toc410002849)

[**Changelog** 1](#_Toc410002850)

[**1.** **引言** 2](#_Toc410002851)

[**1.1.** **编写目的** 2](#_Toc410002852)

[**1.2.** **问题描述** 2](#_Toc410002853)

[**1.3.** **任务目标** 2](#_Toc410002854)

[1.3.1. 增加RGB颜色信息约束 2](#_Toc410002855)

[1.3.2. 仅使用depth点云，但增加轮廓点的权重（杨学连之前的思路） 3](#_Toc410002856)

[**1.4.** **实现环境** 3](#_Toc410002857)

[**1.5.** **参考文献** 3](#_Toc410002858)

[**2.** **实现步骤** 3](#_Toc410002859)

[2.1. 增加RGB颜色信息约束的实现 4](#_Toc410002860)

[2.2. 仅使用depth点云，但增加轮廓点的权重的实现 4](#_Toc410002861)

[**3.** **测试方法** 5](#_Toc410002862)

1. **引言**
   1. **编写目的**

在理解 pcl\_kinfu\_app 程序的算法细节基础上，对现有的代码实现进行修改，在目标函数（cost function）上增加惩罚项，与原目标函数优化结果进行对比。

* 1. **问题描述**

Kinect Fusion （pcl）开源实现中，用到icp算法对两个点集进行配准，求解点集之间的刚性变换 (R, t)。在3D场景的几何形状较为平凡时（e.g., 大的平面， 对称的圆柱体、花瓶，etc.），icp配准可能会出现累积偏移误差（drift）现象。可能需要修改原有的目标函数，增加新的惩罚项，约束优化空间进行求解，以消除偏移误差。

* 1. **任务目标**

Pcl 中的 Kinect fusion 实现，采用了POINT-TO-PLANE ICP算法[1]，对物体（点云）的刚性变换进行求解。其目标函数为：

(1)

其中，是源数据点，是目标点集中的对应点；是 点处的单位法向量，M、 均为 4\*4 3D 刚性变换矩阵。

我们尝试对式 (1) 进行修改：

### 1.3.1. 增加RGB颜色信息约束

目前有以下两种思路：

a) 若在 **之外**增加一项，使目标函数变为：

(2)

其中， 为第 i 对对应点之间的RGB空间的欧氏距离；w为权重系数，用于平衡“+”前后两项间的权重比。

b) 若在 **之内**进行修改，意味着仍使用POINT-TO-PLANE ICP 相同的优化过程，所以约束信息必然是逐对应点的。这里仍使用RGB空间的欧氏距离，即将原来点集信息由 (x, y, z) 3D空间扩展到 (x, y, z, r, g, b) 6D 空间进行ICP迭代。

~~此时，公式(1) 中， 均为6D向量，~~

（没想通， 未解决）

输入：

1. OpenNI驱动捕捉的 RGB-D 数据；

2. 修改过约束条件的目标函数；

输出：

1. 新的 (R, t) 序列；

2. 生成的3D场景模型；

### 1.3.2. 仅使用depth点云，但增加轮廓点的权重（杨学连之前的思路）

对原有点集C提取轮廓后，得到轮廓点集，以及正常点 。假设视窗中物体（或场景）的轮廓具有相对更高的精度，配准过程中，增加C1的权重w：

(3)

其中，分别为点集对应的第i帧源数据点，对应点，法向量；则为对应的第i帧源数据点，对应点，法向量.

* 1. **实现环境**

OS: windows7 x64

IDE: visual studio 2010

Lang: C/C++

Device: XBOX 360 Kinect (Model 1414)

Drivers: OpenNI

3rd party LIBs: pcl, VTK, Boost, Eigen, FLANN, OpenNI

* 1. **参考文献**

[1] Low K L. Linear least-squares optimization for point-to-plane icp surface registration[J]. Chapel Hill, University of North Carolina, 2004.

[2] Arun K S, Huang T S, Blostein S D. Least-squares fitting of two 3-D point sets[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1987 (5): 698-700.

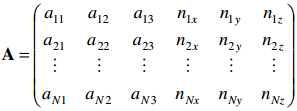
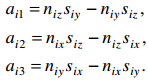
1. **实现步骤**

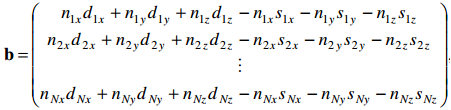
原来的代码实现对公式(1)进行变换为：

(4)

对应：

(5)

，其中  (6)

 (7)

则问题变为线性最小二乘问题。采用SVD分解方法得到：

(8)

进而得到 (R, t)

### 2.1. 增加RGB颜色信息约束的实现

对于 1.3.1-a) 提到的方法，由于增加了惩罚项 ，无法再转化为公式(4)~(8)的过程。可能的实现方案可以如下：

1. 设定初值R=identity()；

2. 每次迭代，的计算方法为：记前一帧彩色图像为，当前帧为，上一帧得到的旋转为 ，则

(9)

3. 采用自适应阈值方法，迭代得到较好的权重值 w。这里每次迭代需要评估结果，可能的方法有：

a) 使用3D扫描仪的obj点云作为ground-truth进行评估；

b) 以原来代码实现得到的 (R, t)序列作为 groundtruth进行评估；

计划尝试使用优化算法库 levmar进行实现。

### 2.2. 仅使用depth点云，但增加轮廓点的权重的实现

假设所有点权重相同，公式(1)可近似为公式(4)，采用线性最小二乘求解。而公式(3)中对轮廓点集赋予不同权重w，等价于在公式(4)中，对n\*6**矩阵A中的对应轮廓点的行**乘以w，其余计算过程完全不变。因此，代码修改的主要工作集中在如何**标记出点集C中的轮廓点，对cuda中计算矩阵A, b的部分，判断某一行是否为轮廓点**，如果是，乘以权重系数w。

具体修改步骤如下：

1. 轮廓点提取：

采用pcl已有api（关键词outlier\_removal）：

model\_outlier\_removal

radius\_outlier\_removal

statistical\_outlier\_removal

进行轮廓点提取，需要对可用的API效果对比测试（1day）

输入： raw-depth-data C

输出： a) C1, C2

b) **index\_C2**, 即C2在C中各点对应序号。

编码困难：

① pcl-oni-grabber得到的是organized datasets，能否直接得到array of indices？

② 对于unorganized datasets，能否直接得到array of indices? 不得不遍历吗？未解决

2. cpp接口修改：

从cpp到cu文件的接口为estimateCombined函数：

|  |
| --- |
| void estimateCombined (const Mat33& Rcurr, const float3& tcurr, const MapArr& vmap\_curr, const MapArr& nmap\_curr, const Mat33& Rprev\_inv, const float3& tprev, const Intr& intr, const MapArr& vmap\_g\_prev, const MapArr& nmap\_g\_prev, float distThres, float angleThres, DeviceArray2D<float>& gbuf, DeviceArray<float>& mbuf, float\* matrixA\_host, float\* vectorB\_host); |

其中， 传入数据主要为：vmap\_curr，nmap\_curr，vmap\_g\_prev，nmap\_g\_prev。需要将当前帧（curr）的vmap改为vmap\_inl, vmap\_outl， nmap改为nmap\_inl, nmap\_outl；前一帧（prev）不用改。 轮廓点、正常点的计算已经由第一步求出。

3. GPU求解矩阵A, b 代码修改：

输入： C, **index\_C2**（第一步求得）

操作： 向量 “float row[7];” 逐元素赋权值w

输出： A, b两矩阵（分别 6\*6, 6\*1）

1. **测试方法**

分别使用原始 pcl\_kinfu\_app 以及改进后的 fusion 结果（obj mesh 文件）， 与 3D 扫描仪导出的 obj 点云进行 ICP 配准， 分别计算RMS误差。