



SLAM & Mesh

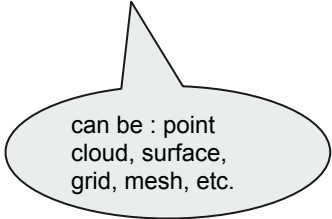
L.Y.



Simultaneous localization and mapping

Objective目的:

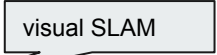
- localization定位
- mapping (make a map)制图



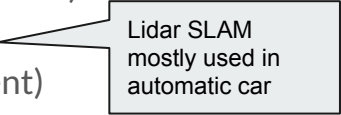
can be : point
cloud, surface,
grid, mesh, etc.

What we have as resource资源:

- camera (mostly is our only resource)
- lidar (help for better distance)
- imu (help for estimate movement)



visual SLAM



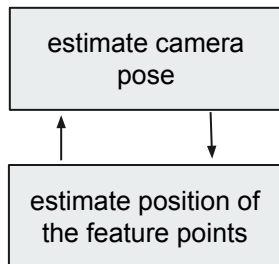
Lidar SLAM
mostly used in
automatic car

-> the algorithm is based on our
resources to achieve our objective.
算法是:使用现有的资源达到我们的
目的是手段。

Overview of the algorithm



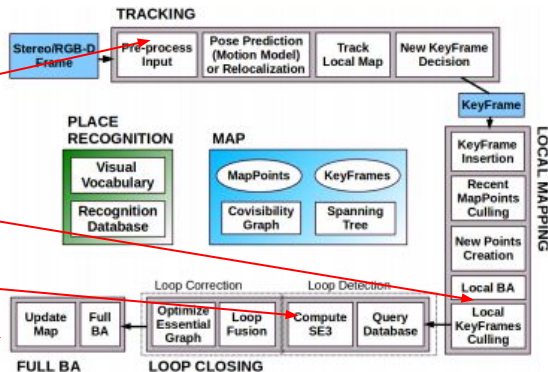
sequence of
RGB images
depth(optional)



1. 提取不同角度拍摄到的照片的**特征点**, 进行匹配。
2. 综合得到的**匹配**, 结合相机的物理模型, 还有运动模型, 得到**相机位姿**的估计。
3. 利用相机位姿的估计, 优化得到更好的**点云位置**。
4. 得到相机的**位姿和特征点点云地图**

ORB SLAM system

- 追踪相机的位姿
- BA(非线性优化算法)-> 优化三维点的位置 -> 得到特征点的点云
- 回环检测(检测整体的漂移, 一旦检测到会进行全局优化Full BA, 非线性优化所有点云的位置和之前所有相机位姿)



Overview of the algorithm

I think it is better to think it as a list of equations to solve.

- position of a point in one point of view * the transformation from camera view to the world view = position of a point in the world coordinate system.

We have the upper equations for each of the feature points we detected.

from this list of equations we can solve the position of the feature points and the position of the camera.

我们有上面的一系列方程 -> 求解他们(使用之前提到的非线性优化方法)可以得到三维特征点的位置和相机的位置。(黑点:特征点位置, 蓝色:相机的位姿)

这一类视觉SLAM(ORB_SLAM为例)只能得到特征点的点云+相机位姿的估计。

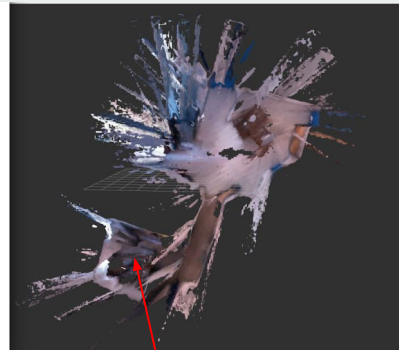
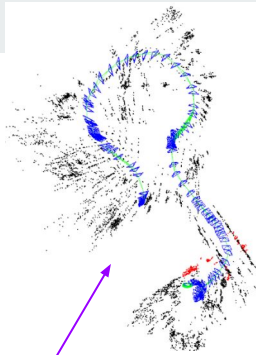
如果我需要mesh, 需要更好看的建图怎么办?

我能想到的方法:

- 使用SLAM的稀疏的特征点点云>生成稠密的mesh。
- 使用另一系列的SLAM算法: 基于深度相机的SLAM(例如bundle fusion, kinect fusion等)。
- 使用这一类SLAM的相机位姿+深度相机的稠密点云建图。

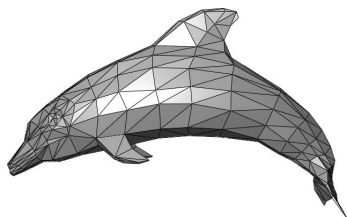
这一类算法的主要目的是制图->非常不适合定位的应用。
运算量非常大。

我们目前使用的算法, ORB_SLAM定位+voxblox制图



Mesh建图

Mesh从何而来 -> mesh其实是一系列的三角形->三角形的基础是三个顶点
mesh其实是对点云的加工, 按照特定的方式, 将点云 连接成三角形就得到了 mesh。



6D ai

SDK State: Running
Load State: None
Error: None
Save State: None
Error: None
Location ID:

因为可以注意到
mesh的三角形都有
相同的尺寸。

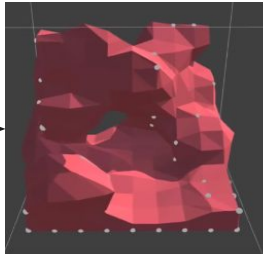
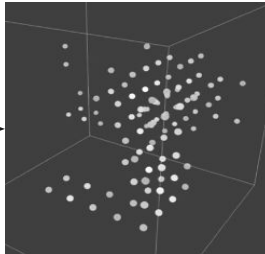
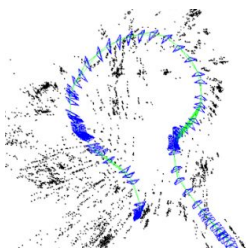
Ok, 那么6d ai公司的单目实时mesh建图是怎么做的。

- 首先, 他们的制图不稠密, 同样还是栅格化的三维空间, 他们的地面不平整, 细节不完整。
- 其次, 他们的demo只是给了简单场景的demo。(我也找到了类似的简化版https://github.com/WeiHuang1994/ORBSLAM_with_Texture)

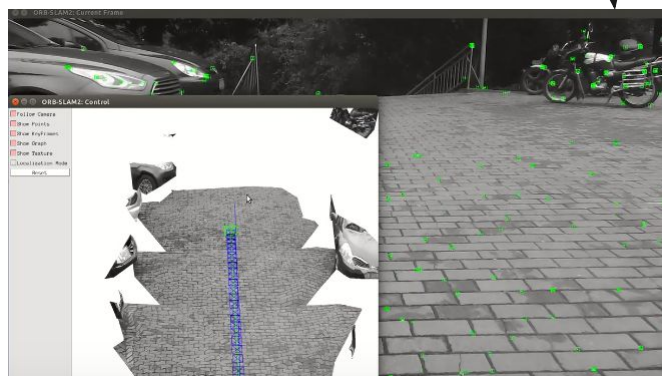
我们要复制他们的demo怎么做?

我想出的解决方案:

1. 稠密化稀疏的点云 -> 栅格化 -> [marching cube](https://www.youtube.com/watch?v=B_xk71YopsA) 建立mesh (应该能得到最像6d ai的结果)



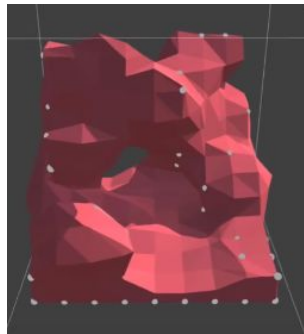
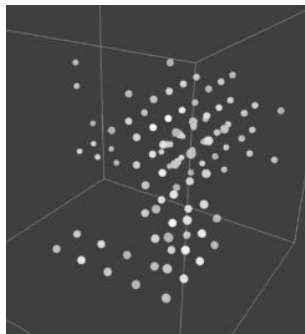
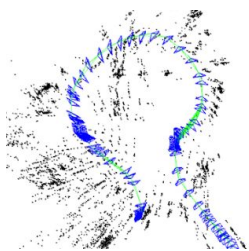
2. 运用现有的稀疏特征点点云 -> 寻找合适的方案三角化, 连接成mesh



Mesh建图

我们的做法:

目前的方案最多的是 栅格化点云 -> 再根据栅格化的点云得到 mesh (例marching cube算法)
由于我们有深度相机, 可以得到更加稠密的点云 (可以得到精度更高的栅格), 同时有更加精确的深度。



存在问题:

所有空间位置的三角形都有相同的尺寸 -> 平面会有过多的细节, 但是角落的细节可能会不足。

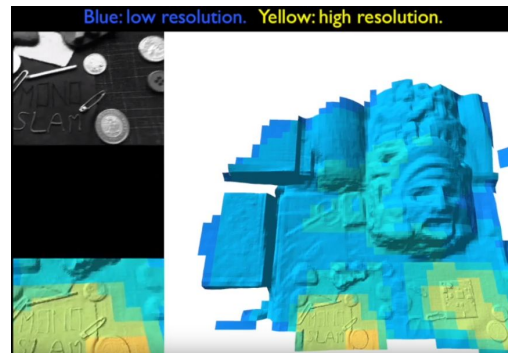
解决方案:

下一页

Dynamic Level of Detail

动态分布栅格的尺寸(Dynamic Level of Detail)。

例: (<https://www.youtube.com/watch?v=gCGg9MrMgjk&t=57s>)



Using Machine learning

没有找到类似的开源算法。不过可以借助深度学习, 提供语义信息, 基于语义信息来建立mesh。

lots of work

我不是这方面的专家, 可能有局限性。

Post processing mesh



可能的应用

- 遮挡和物理互动: 我们可以用深度相机预先得到的更加精确的mesh(同时还能节省手机端有限的运算能力)。
- 动态物体的mesh建立: 需要另外一套专门针对动态物体的SLAM算法才能做到, 这一套算法和基于静态物体的定位算法(我们的算法)应该是冲突的, 只能分开运算。(另外, 可以发现6d ai的demo中没有动态物体出现, 我以狭隘的眼光推测他们不能处理动态物体)



Thank you