

科研笔记，记录人生，见证成长。

谢东海 首都师范大学

7.20 ~ 22

周五开车到学校，然后乘坐地铁去 T1，和吴俣乘坐海航飞机到长沙出差。陈前和家宝带上台式机坐高铁已经提前参加了展会，我们住在文化气息挺浓，配套也很有特色的亚朵酒店，还吃了长沙小吃和麻辣小龙虾。508 联系的岳阳的项目，关键是要做出变化检测的效果。但是目前已有的方案只有基于 yolo v3 的 coco 库训练模型，可以用来检测汽车，人体等。后面要考虑基于 coco 库来增加训练样本，或者是只针对几种目标来训练。能否在目标模型的基础上来继续训练，还是要从头开始训练呢？YOLO v3 的训练过程包括几个步骤：首先是用 labelimg 来标注数据，接着将数据组织为 pascal voc 的目录结构 (work/programs/deeplearning/generate_voc_dir.py)，然后调用程序将 xml 的标注文件转换为 txt 文件(voc_label.py)，最后手工构造网络文件(cfg)、标签文件(names)、以及数据文件(data)就可以进行训练了。Python 由于其跨平台和易用性，可以用来编写中间的很多批处理的程序。

当前帧后全景底图的匹配我使用了 HoG 特征，并且进行了一些改进，将角度归一化到了 180 以内。这是原来做点云深度图匹配采用互信息算法时顺便编写的代码，还是有一些效果。

但是真正要做变化检测还是挺难的，遥感中基于像素对齐的方法肯定不适用，只能考虑采用语义分割的方法来做。深度学习的文献我也下载了两篇，后续要继续跟进。

7.23

今天上午跟家宝一起继续调试 modis sara 算法，采用 modis brdf 数据产品来计算地表反射率，发现和 mod09 的存在明显差异。同时也验证了地表反射率不

稳定的问题，对反演的结果影响比较大，特别是气溶胶光学厚度较大时透过率的计算不准确，这个还没想好有什么办法来提高。

下午去亦庄开会，听取了激光 slam 的产品研究进展。目前公司采用的是 google 的开源代码，跑通后用来处理自己组装仪器测量的数据。所以核心要把算法看懂并进行改造，适应测绘的需要。

晚上研究了 map2dfusion，发现这篇文章做的正好是我的思路，将 slam 的成果用到地图数据生产中来。后面要把代码调通，看看是否能达到文章中的效果。

Map2DFusion: Real-time Incremental UAV Image Mosaicing based on Monocular SLAM

Shuhui Bu¹, Yong Zhao¹, Gang Wan², and Zhenbao Liu¹

Abstract—In this paper we present a real-time approach to stitch large-scale aerial images incrementally. A monocular SLAM system is used to estimate camera position and attitude, and meanwhile 3D point cloud map is generated. When GPS information is available, the estimated trajectory is transformed to WGS84 coordinates after time synchronized automatically. Therefore, the output orthoimage retains global coordinates without ground control points. The final image is fused and visualized instantaneously with a proposed adaptive weighted multiband algorithm. To evaluate the effectiveness of the proposed method, we create a publicly available aerial image dataset with sequences of different environments. The experimental results demonstrate that our system is able to



这篇文章的代码开源。

7.25

K. SAKURADA, T. OKATANI: SCENE CHANGE DETECTION USING CNN FEATURES 1

Change Detection from a Street Image Pair using CNN Features and Superpixel Segmentation

Ken Sakurada
sakurada@vision.is.tohoku.ac.jp

Takayuki Okatani
okatani@vision.is.tohoku.ac.jp

Abstract

This paper proposes a method for detecting changes of a scene using a pair of its vehicular, omnidirectional images. Previous approaches to the problem require the use of a 3D scene model and/or pixel-level registration between different time images. They are also computationally costly for estimating city-scale changes. We propose a novel change detection method that uses features of convolutional neural network (CNN) in

K. SAKURADA, T. OKATANI: SCENE CHANGE DETECTION USING CNN FEATURES 2

Figure 1: Example of an image pair of a scene captured two months apart.

To cope with these issues, some of the previous studies consider the problem in the 3D domain. They assume that a 3D model of a scene is given beforehand or can be created from images, and that the input images can be registered to the model with pixel-level accuracy [10, 11, 12]. However, a 3D model is not always available for every city. Besides, it is sometimes hard to perform precise image registration, due to lack of sufficient visual features. These are particularly the case when the scene undergoes enormous amount of changes. Working in the 3D domain tends to require large computational cost, which can be another difficulty when we want to detect changes for a large city.

Thus, we tackle the change detection problem in the 2D domain. That is, we consider detecting changes based on the direct comparison of a pair of images. The major issue is then how to deal with the above unwanted visual differences (i.e., viewpoint differences etc.). To cope with this, we propose to use the features extracted by convolutional neural networks (CNNs). To be specific, we use a fully trained CNN for large-scale object recognition task [13] in a transfer learning setting. It was reported in the literature that using activation of the upper layers of a CNN trained for a specific task can be reused for other visual classification tasks. Several recent researches imply that the upper layers of CNNs represent

长沙项目要用视频来进行变化检测，上面这篇文章的思路比较接近。特别是这篇

文献不要求进行精确的配准，而是用深度学习提取特征的方式来进行变化检测。数据采用了全景方式，这个的确是很好选择，因为可以获取 360 度范围场景。利用 GPS 来获取距离最近的全景作为比较的图像对，由于拍摄时每隔几米采集一张全景，所以视点肯定存在偏移。而 CNN 的一个优点就是对平移、旋转等造成的图像变化鲁棒。

7.26

长沙视频变化检测（工程：YOLOv3-2）的一个简单思路：

- ✓ 将拼接后的全景图均匀分成多个块，对每块计算 HOG 特征；
- ✓ 对视频的每一帧计算 HOG 特征，然后根据相关系数来计算全景图中对应的那块；
- ✓ 在粗定位的全景图像块和视频帧之间进行匹配，采用 orb 特征；
- ✓ 利用 RANSAC 仿射变换将视频帧 warp 到全景图像块上面，实现几何对齐；
- ✓ 后续在对齐的基础上进行变化检测；

基本的思路还是要进行几何对齐，但是不一定要精确定位，后续考虑用深度学习来进行变化检测。

SARA 算法计算的结果很不理想，MOD09 有很多数据有问题，我们现在用 brdf 来弥补，但是反演的结果还是有很大差距。然后感觉透过率也有问题，SARA 使用的是经验公式，直接根据几何角度和光学厚度来计算出了透过率。后续可以用 6s 或者 rt3 来计算双程透过率，看看效果如何。实在不行就用通用的气溶胶模式来计算双程透过率。

Map2dfusion 的编译：安装 qt4，安装 libqglviewer，由于版本的问题，这两个库都得用源代码编译的方式来安装。还有 glew。原来可以用 apt-get install 来安装，但是现在都没法安装，都得用手工编译来安装，手工安装到也简单，首先 make，然后 sudo make install 就可以。安装好后，还需要修改 makefile 的 config 文件中

的 `qt` 动态库的目录，然后将 `usr/local/lib` 目前放到 `/etc/ld.so.conf` 中，进入 `/etc` 目录来运行 `ldconfig`，这样程序才能找到安装后的第三方库的位置。所以整个过程非常麻烦，现在的问题是我根本没有办法找到代码执行的流程。

7.27

虽然 `map2dfusion` 的编译过程无比痛苦，但是拼接的效果还是很不错的。现在需要研究其代码的调用过程。如果不能看到实时拼接的效果，我也没有信心去看那么复杂的代码。



感觉速度还是很快的，而且接缝线和匀色处理的也很好，真是很厉害啊。但我仔细看代码后发现，这个程序需要输入路径文件，但是程序里面并没有说明这个路径文件是怎么来生成的。所以我觉得这个算法只是做了一个实时匀色的效果，而没有做 `slam` 的效果！！！！！从作者的 `github` 网站上发现了：
<https://github.com/zdzhaoxiong/GSLAM> 我的理解是作者将 `slam` 单独写成了一个模块。而且还给予这个模块将 `orb-slam` 也给重写了，非常的厉害，完全把 `slam` 的原理和代码吃透了。我给作者写了封信，他回复也证实了这点。

Deeplab 的训练流程：

- 利用批处理文件 `labelme_convert.py` 将所有的.json 文件转换为对应的文件夹，

并把文件夹中的 `label.png` 拷贝到指定的目录里面；

- 利用批处理文件 `labelconvert.py` 将 16bit 的图像格式转换为 8bit；
- 构造目录树结构: `generate_voc_dir.py`
 - `ImageSets`
 - ◆ `Main` 目标检测的样本训练与测试集
 - ◆ `Segmentation` 语义分割的样本训练与测试集
 - `JPEGImages` 样本图像文件
 - `SegmentationClass` 语义分割的标记图像文件
- 生成 `train.txt` 和 `test.txt`, 放在 `ImageSets/SegmentationClass` 中
- 调用转换函数将样本文件转换为 `tfrecord` 格式, 转换的时候要记得把图像格式加上: `build_karst_data.py -image_type="jpg"`
- 开始训练: `local_test_karst.shcd`

`Labelme_convert.py` 和 `labelconvert.py` 的路径为
`media/xdh/work/work/Programs/deeplearning/deeplab`

7.28 周六

现在确定思路，就是将 `orb-slam2` 和 `map2dfusion` 结合起来，用 `orb-slam2` 来计算轨迹和姿态，然后用 `map2dfusion` 来拼接。

7.30 周一

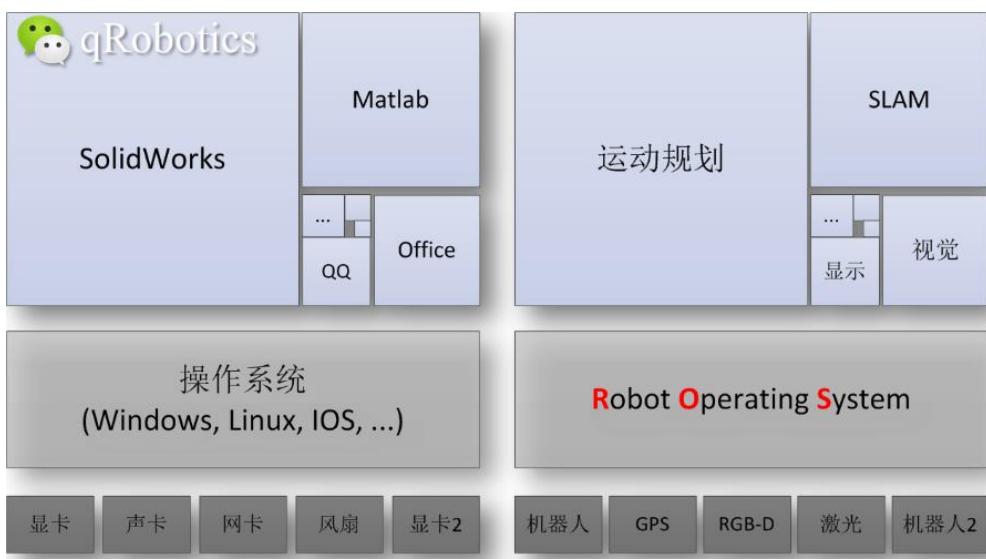
今天上午将视频数据和 `orb-slam2` 对接起来，可以直接读取视频中的数据来进行 `slam`，但是整个流程对我来说还是黑箱，数据结构和算法都没怎么看懂。算法最后将轨迹输出，包括位置信息和姿态，姿态转换为了四元数。程序需要输入相机的参数，但是拍摄时摄像头没有进行标定，因此我将图像的宽度缩放至 640，然后估计了一个内方位元素。后续最好进行标定。明天准备弄清楚轨迹数据和

map2dfusion 直接的联系。

下午去亦庄对接 cartographer 软件包，安装平台推荐为 ubuntu 14.04，只能将笔记本的系统重装。这个激光 slam 的算法也很复杂，看不懂，唉。。。和小川的交流中，他说目前的问题有：1) cartographer 无法适应快速数据采集，目前是用背包来采集，当装在车上时，由于激光采集的频率不够，会导致点云匹配失败；
2) 拼接后点云的精度也不够高，目前测试的精度大概在 20cm 左右；

7.31 周二

今天林允晖将 cartographer ros 编译成功，从网上下载的数据也可以跑通。
Ros 是什么东西我还不是很清楚，但在机器人领域 ros 被广泛的使用。



我继续研究 orb-slam2 的数据结构和算法。目前理解的数据结构是用 Frame 来保存特征点信息，KeyFrame 来保存关键帧的信息。每帧图像首先传入 Frame 中，进行特征点信息提取后，图像的存储空间是不被保存的。KeyFrame 的数量不多，因此为了方便后续的实时拼接，我在 KeyFrame 类中增加了一个变量来存储关键帧对应的图像数据。Orb-slam2 加载图像时，同时加载了时间戳。我改成了加载视频，然后根据帧率来添加时间戳。最后存储轨迹文件时，第一列为时间，后面为位置和姿态，姿态是用四元数来描述的。

```

1 0.040000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 1.0000000
2 0.880000 0.0020972 -0.0240697 -0.0011894 -0.0002384 -0.0003401 -0.0008274 0.9999996
3 0.920000 0.0016618 -0.0273298 -0.0015996 -0.0014355 -0.0001975 -0.0011803 0.9999983
4 2.720000 0.0058781 -0.0852272 -0.0052058 -0.0024305 -0.0009318 -0.0023051 0.9999940
5 3.560000 0.0061544 -0.1193247 -0.0061020 -0.0055552 -0.0005334 -0.0018977 0.9999827
6 5.000000 0.0079731 -0.1732705 -0.0104999 -0.0068374 -0.0009102 -0.0024559 0.9999732
7 5.520000 0.0086700 -0.1905496 -0.0114170 -0.0057930 -0.0009015 -0.0026501 0.9999793
8 6.840000 0.0122643 -0.2440577 -0.0087194 -0.0079378 -0.0016881 -0.0028171 0.9999631
9 8.560000 0.0146046 -0.3162570 -0.0030870 -0.0112365 -0.0015323 -0.0032636 0.9999304
10 10.280000 0.0194996 -0.3877737 0.0032466 -0.0131411 -0.0028416 -0.0028688 0.9999055
11 11.880000 0.0222940 -0.4624285 0.0094040 -0.0179357 -0.0027380 -0.0032860 0.9998300
12 12.240000 0.0222320 -0.4794367 0.0111521 -0.0189620 -0.0024093 -0.0034037 0.9998115
13 12.600000 0.0239364 -0.4931004 0.0119596 -0.0181017 -0.0026185 -0.0030796 0.9998280

```

Map2dfusion 中根据时间戳来存储对应的图像文件，希望明天我自己做的数据能够和 map2dfusion 进行对接。

8.1 周三

今天继续调试 map2dfusion，我在 windows 下面建立了一个工程来调试核心代码，可以运行，但是拼接会死掉。大致知道了 map2dfusion 的流程，启动了两个线程，一个用来加载图像，一个用来进行实时的拼接和匀色。

下午去亦庄继续研究背包的软件。

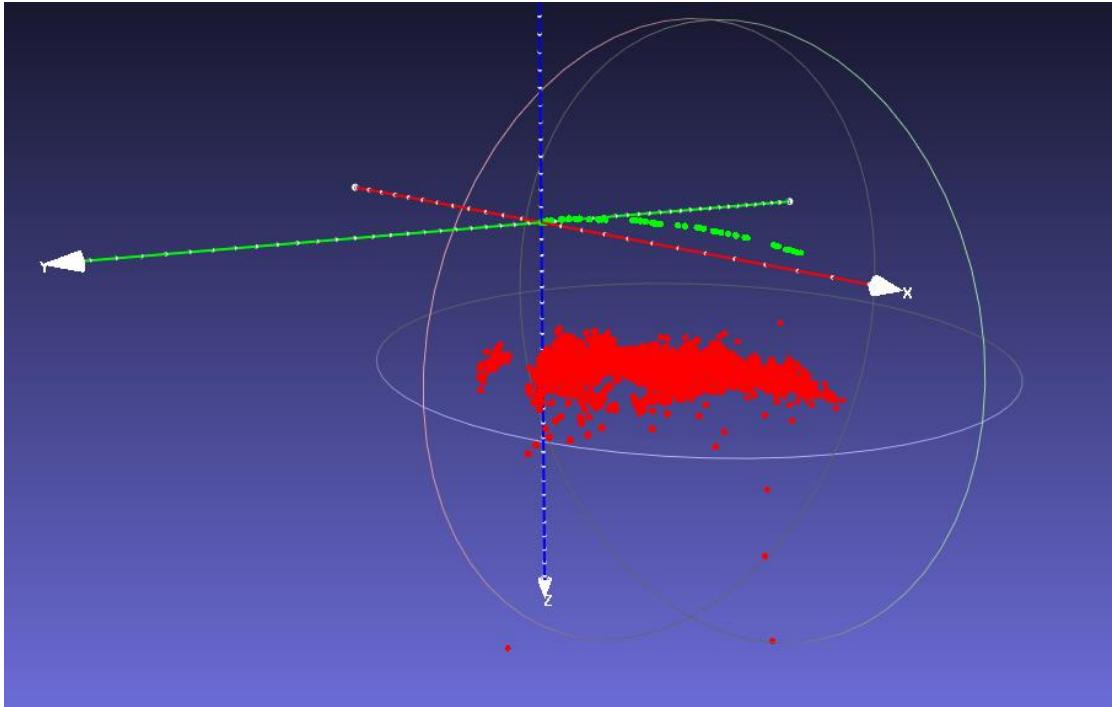
8.2 周四

在 windows 下面用 debug 来运行 map2dfusion 的融合代码时调用 renderFrame 函数总是出现死机的情况。实在没有办法我改用 release 来编译，然后可以运行出融合的结果。renderFrame 可以每加入一张图像就进行一次融合。根据代码分析，我发现融合也是基于 opencv 来完成的，核心函数包括：restoreImageFromLaplacePyr，该函数就是利用拉普拉斯金字塔图像来重构出拼接图像。



图、不同数量的图像重建后的效果

基本思想应该是利用位置和姿态数据来计算每张相片在地面投影后的范围，然后进行 warp。这跟我以前做拼接采用的是同样的思路。现在的问题是，如何根据 slam 的结果来获取拟合平面的参数？这个我看的很晕，感觉和我们常用的摄影测量坐标系不一样。



图、slam 输出的点云和相机位置

Slam 输出的点云坐标系中，Z 轴是朝下方的，这跟 bundler 不太一样。

Map2dFusion 中从相机坐标到平面坐标的转换代码为：

```
pose=plane.inverse()*pose; //plane coordinate
```

pose 为相机的位置和姿态参数。

根据我的理解，map2dfusion 首先将自由网的坐标原点移动到了拟合平面的中心点，然后进行旋转让自由网的点云拟合的平面在新坐标系下面尽可能的水平。根据拟合平面的旋转与平移量，将相机的位置和姿态都转换到新的坐标系中，然后计算每个相机在新的坐标系中的覆盖范围。自由网中的投影公式为：

$$x \approx R_f X_f + T_f$$

校正后的点云设为 X_g ，假设校正后点云坐标系和自由网坐标系的关系为：

$$R_g X_g = X_f - T_g$$

那么有：

$$\begin{aligned}x &\approx R_f(R_g X_g + T_g) + T_f \\x &\approx R_f R_g X_g + R_f T_g + T_f \\x &\approx R_f R_g [X_g + \text{inv}(R_f R_g)(R_f T_g + T_f)]\end{aligned}$$



图、没有进行平面拟合的拼接效果

8.3 周五

经过测试，可以根据点云数据来拟合平面位置和旋转矩阵。

```
//added by xiedonghai, 2018.8.3
int PlantFit()
{
    char* filename = "C:\\\\Work\\\\Data\\\\videomosaic\\\\my\\\\mapts.ply";

    int npt;
    double* px = NULL;
    double* py = NULL;
    double* pz = NULL;
    int* pr = NULL;
    int* pg = NULL;
    int* pb = NULL;

    ReadPly(filename, &npt, &px, &py, &pz, &pr, &pg, &pb);

    double R[9];
    FitPlane1(px, py, pz, npt-100, R);

    double mx = 0;
```

```

double my = 0;
double mz = 0;
for (int i = 0; i < npt - 100; i++){
    mx += px[i];
    my += py[i];
    mz += pz[i];
}
mx /= npt - 100;
my /= npt - 100;
mz /= npt - 100;

printf("translation: %lf %lf %lf \n", mx, my, mz);

printf("rotation: \n");
for (int i = 0; i < 9; i++)
{
    printf("%lf ", R[i]);
    if ((i + 1) % 3 == 0)
        printf("\n");
}

for (int i = 0; i < npt - 100; i++){
    double fpt[3];
    fpt[0] = px[i];
    fpt[1] = py[i];
    fpt[2] = pz[i];
    double res[3];
    mult(R, fpt, res, 3, 3, 1);
    px[i] = res[0];
    py[i] = res[1];
    pz[i] = res[2];
}
WritePly("c:\\temp\\rotatemap.ply", npt - 100, px, py, pz, pr, pg, pb);

invers_matrix(R, 3);

//from rotation to quartion
Eigen::Matrix<double, 3, 3> M;
M << R[0], R[1], R[2], R[3], R[4], R[5], R[6], R[7], R[8];
Eigen::Quaterniond q(M);

FILE* fp = fopen("c:\\temp\\planefit-quatern.txt", "w");
fprintf(fp, "%lf %lf %lf %lf \n", q.x(), q.y(), q.z(), q.w());
fclose(fp);

```

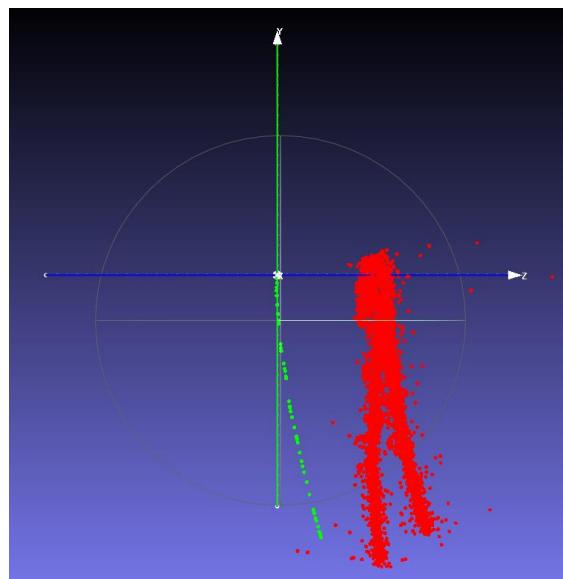
```
//  
return 0;  
}
```

然后作为输入给 map2dfusion，得到的拼接效果如下：



图、平面拟合后的拼接效果

可以看出平面拟合后的效果改善明显。接下来的工作就是如何将拼接和 slam 集成到一个平台中。为了提高拼接后的图像分辨率，我将 Default.cfg 文件中的 Map2D.Scale 设置为 1。



图、利用平面拟合来校正点云

可以看出，经过校正后的点云和 XY 平面基本上水平，这样对后续的拼接是有帮助的。

8.4 周六

上午继续面试，但是来应聘的员工大多水平不高，也没有太多的经验。后续要多找找实习生。

8.5 周日

最近天气热得厉害，不开空调简直没法干活，继续看 `slam` 的代码和原理。
Orb-slam2 的三个线程（`Tracking` 为主进程，`localmapping` 和 `loopclosing` 为进程中的两个线程）的结构还是很难看懂，特别是什么变量需要进行访问控制，而哪些不需要，这个比较麻烦。但是一定要坚持，我最开始看的时候，完全是晕的。可即使这样，我也对 `slam` 有了比较直观的印象，现在第二次认真的看代码，感觉就没有那么的陌生了。

✓ 初始化：首先找到一帧特征点数量大于 100 个的图像，将其作为初始帧 `mInitialFrame`，并且进行初始化算法类的初始化：`mplInitializer = new Initializer(mCurrentFrame, 1.0, 200);`；如果初始帧的下一帧的特征点数量小于 100，那么重新开始进行初始帧的选择与算法类的初始化。在满足条件的相邻两帧之间进行匹配：

```
ORBmatcher matcher(0.9,true);
int nmatches = matcher.SearchForInitialization(mInitialFrame,mCurrentFrame,
mvbPrevMatched,mvIniMatches,100);
```

如果匹配数量小于 100，那么重新开始进行初始化。

如果满足匹配点数据的要求，那么就进行初始化：`bool Initializer::Initialize(const Frame &CurrentFrame, const vector<int> &vMatches12, cv::Mat &R21, cv::Mat &t21, vector<cv::Point3f> &vP3D, vector<bool> &vbTriangulated)`

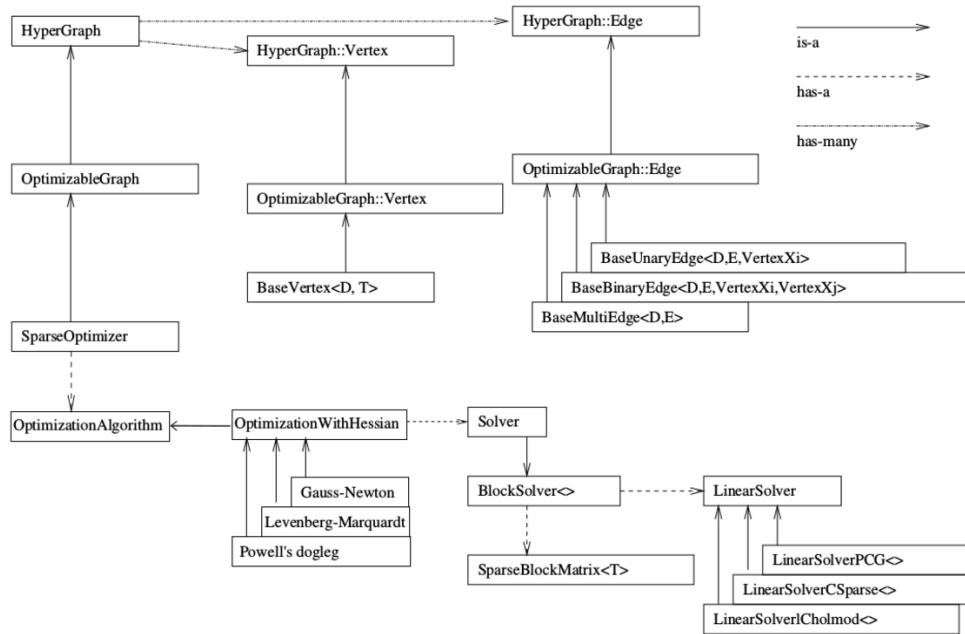
根据文章中提到的 Fundamental Matrix 和 Homography 的算法来进行初始化。
找到符合要求的相邻两帧后，将这两帧作为关键帧。

8.6 周一

除了代码，Github 上面还有很多讲理论知识的内容。比如：

<https://github.com/Ewenwan/MVision/tree/master/vSLAM>

这个网页收集了大量的关于视觉 slam 的资料。



图、g2o 类图

在 g2o 中，参数都放到 vertex 里面，不同的 vertex 之间通过 edge 来连接。

下午去顺义公司，商量了一下拼接的接口，还是要做成一个动态库来调用。将最后的拼接和匀色做到一个线程里面。

视频拼接输入参数：

- ✓ 相机内方位元素参数文件；
- ✓ BoW 词袋文件

将拼接组织为一个动态库文件，参数文件在初始化的时候传入动态库。

8.8 周三

昨天好像什么都没干。上午电话面试了几位实习生，下午去验车发现排气管坏了，马上回家后送到安慧去修理。今天上午和王润柱继续对接，用背包采集了6楼的楼道的数据，基本上都拼接上了，但是程序还需要继续熟悉。还讨论cartographer的算法细节，然后还讨论了一下目前slam的进展，感觉收获挺多的，以后要多多进行交流。特别是讨论中提到的几个github上最新的开源库：VINS，SVO等，要好好去研究一下。**激光 slam**是后续工作的一个重点，需要花更多的精力去做，特别是**精度验证**以及**算法的改进**。下午去验车比较顺利，不到一个小时就弄完。晚上继续看cartographer对应的文献，有了上午的交流感觉看起来轻松多了。

Real-Time Loop Closure in 2D LIDAR SLAM

Wolfgang Hess¹, Damon Kohler¹, Holger Rapp¹, Daniel Andor¹

Abstract—Portable laser range-finders, further referred to as LIDAR, and simultaneous localization and mapping (SLAM) are an efficient method of acquiring as-built floor plans. Generating and visualizing floor plans in real-time helps the operator assess the quality and coverage of capture data. Building a portable capture platform necessitates operating under limited computational resources. We present the approach used in our backpack mapping platform which achieves real-time mapping and loop closure at a 5 cm resolution. To achieve real-time loop closure, we use a **branch-and-bound** approach for computing scan-to-submap matches as constraints. We provide experimental results and comparisons to other well known approaches which show that, in terms of quality, our approach is competitive with established techniques.

loop closure detection. Some methods focus on improving on the computational cost by matching on extracted features from the laser scans [4]. Other approaches for loop closure detection include histogram-based matching [6], feature detection in scan data, and using machine learning [7].

Two common approaches for addressing the remaining local error accumulation are particle filter and graph-based SLAM [2], [8].

Particle filters must maintain a representation of the full system state in each particle. For grid-based SLAM, this quickly becomes resource intensive as maps become large; e.g. one of our test cases is 22,000 m² collected over a 3 km

如果能够把slam技术的细节吃透，那么后面可以将视觉、IMU等设备和激光进行集成，获得更好的精度。另外在网上看到邸老师的一篇综述文章，有时间下载来看看。

论文推荐| 邱凯昌：视觉SLAM技术的进展与应用

2018-07-18 08:57

 技术 / 机器人 / 数码

视觉SLAM技术的进展与应用

邱凯昌¹, 万文辉¹, 赵红颖², 刘召芹¹, 王润之¹, 张飞舟²

1. 中国科学院遥感与数字地球研究所遥感科学国家重点实验室, 北京 100101;
2. 北京大学遥感与地理信息系统研究所, 北京 100871

8.9 周四

今天将 map2dfusion 的代码整合为一个线程类, 集成到 orb-slam2 的框架中了。但是运行就死掉了, 多线程的调试非常的麻烦, 我也没有找到好的方法。而且 orb-slam2 工程的编译超慢。

8.10 周五

今天将实时拼接的效果调试出来。参考了 orb-slam2 中的显示进程 Viewer 处理结束的方式, 将处理结束的函数拷贝到了类 MultiBandMap2DCPU 中。在 MultiBandMap2DCPU::run() 函数中执行无限循环的一个过程。

```
void MultiBandMap2DCPU::run()
{
    mbFinished = false;
    mbStopped = false;

    cv::namedWindow("mosaic", WINDOW_NORMAL);
    cv::resizeWindow("mosaic", 1024, 640);

    while (1)
    {
        //initialize the data
        vector<KeyFrame*> vpKFs = mpMap-> GetAllKeyFrames();
```

```

int nKeyFrames = vpKFs.size();

if (nKeyFrames > mnCurrentKeyNumber){
    mnCurrentKeyNumber = nKeyFrames;
    printf("current key number: %d \n", mnCurrentKeyNumber);
}
else{
    //printf("key frames: %d \n", nKeyFrames);
    continue;
}

//printf("key frames number: %d \n", nKeyFrames);

if (nKeyFrames==20 || (nKeyFrames>0 && nKeyFrames%100==0 ) )
{
    sort(vpKFs.begin(), vpKFs.end(), KeyFrame::lId);

    // plane fitting
    const vector<MapPoint*> &vpMPs = mpMap->GetAllMapPoints();
    int nmappt = vpMPs.size();
    double* px = new double[nmappt];
    double* py = new double[nmappt];
    double* pz = new double[nmappt];
    int npt = 0;
    for (size_t i = 0, iend = vpMPs.size(); i < iend; i++)
    {
        if (vpMPs[i]->isBad())
            continue;
        cv::Mat pos = vpMPs[i]->GetWorldPos(),
        px[npt] = pos.at<float>(0);
        py[npt] = pos.at<float>(1);
        pz[npt] = pos.at<float>(2);
        npt++;
    }
    double mx, my, mz, wx, wy, wz, w;
    PlaneFit(px, py, pz, npt, mx, my, mz, wx, wy, wz, w);
    pi::SE3d plane = pi::SE3d(mx,my,mz,wx,wy,wz,w);
    delete[] px;
    delete[] py;
    delete[] pz;

    //camera parameters
}

```

```

PinHoleParameters camParas;
camParas.fx = mFx;
camParas.fy = mFy;
camParas.cx = mCx;
camParas.cy = mCy;
camParas.w = mWd;
camParas.h = mHt;

//frame pos and image data
printf("generating frames and rendering... \n");
deque<std::pair<cv::Mat, pi::SE3d> > frames;
for (int i = 0; i < nKeyFrames; i++)
{
    //printf("%d ", i);
    std::pair<cv::Mat, pi::SE3d> frame;

    KeyFrame* pKF = vpKFs[i];
    frame.first = pKF->mFrameImage;

    double x, y, z, wx, wy, wz, w;
    cv::Mat R = pKF->GetRotation().t();
    vector<float> q = Converter::toQuaternion(R);
    cv::Mat t = pKF->GetCameraCenter();

    x = t.at<float>(0);
    y = t.at<float>(1);
    z = t.at<float>(2);
    wx = q[0];
    wy = q[1];
    wz = q[2];
    w = q[3];

    printf("pos: %lf %lf %lf %lf %lf %lf %lf \n", x, y, z, wx, wy, wz, w);
    frame.second = plane.inverse()*pi::SE3d(x,y,z,wx,wy,wz,w);

    frames.push_back(frame);
}

//prepare
printf("\n prepare... \n");
prepare(plane, camParas, frames);

printf("render... \n");
for (int i = 0; i < nKeyFrames; i++)

```

```

{
    printf("%d ", i);
    renderFrame(frames[i]);
}

/*printf("mosaic...\n");
Mat result;
mosaic(result);
cv::imshow("mosaic", result);
cv::waitKey(100);*/
//printf("save mosaic... \n");
//imwrite("c:\\temp\\mosaic.jpg", result);
}

//feed new frame
if (nKeyFrames > 20)
{
    sort(vpKFs.begin(), vpKFs.end(), KeyFrame::lId);

    KeyFrame* pKF = vpKFs[nKeyFrames-1];
    Mat image = pKF->mFrameImage;

    double x, y, z, wx, wy, wz, w;
    cv::Mat R = pKF->GetRotation().t();
    vector<float> q = Converter::toQuaternion(R);
    cv::Mat t = pKF->GetCameraCenter();

    x = t.at<float>(0);
    y = t.at<float>(1);
    z = t.at<float>(2);
    wx = q[0];
    wy = q[1];
    wz = q[2];
    w = q[3];

    printf("pos: %lf %lf %lf %lf %lf %lf \n", x, y, z, wx, wy, wz, w);
    pi::SE3d pos = pi::SE3d(x, y, z, wx, wy, wz, w);

    printf("feed image ... \n");
    //feed(image, pos);
    std::pair<cv::Mat, pi::SE3d> frame(image, prepared->_plane.inverse()*pos);
    renderFrame(frame);

//mosaic

```

```

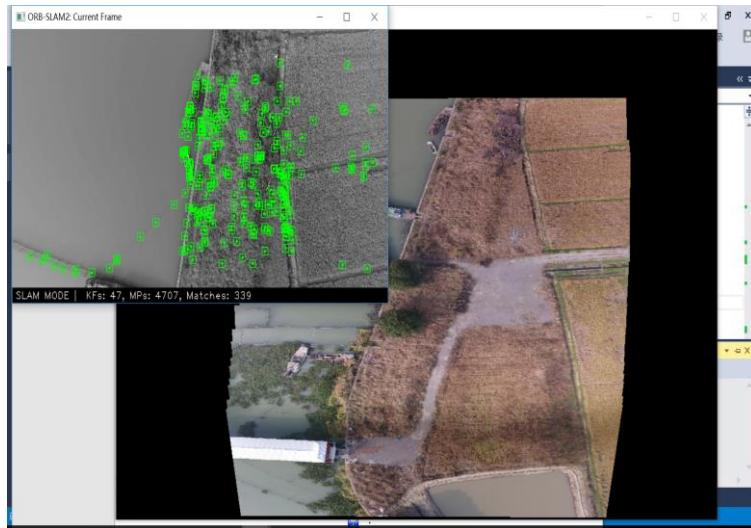
//if (nKeyFrames % 5 == 0)
{
    printf("mosaic...\n");
    Mat result;
    mosaic(result);
    cv::imshow("mosaic", result);
    cv::waitKey(100);
    //printf("save mosaic... \n");
    //imwrite("c:\\temp\\mosaic.jpg", result);
}
}

if (Stop())
{
    while (isStopped())
    {
        usleep(3000);
    }
}

if (CheckFinish())
{
    break;
}

usleep(3000);
}
}

```



图、slam 与实时拼接的效果图

算法的基本步骤是：

- ✓ 初始化：利用 20 张关键帧图像以及 slam 恢复的位置和姿态来初始化拼接的范围和分块匀色的数据结构。在 Prepare 函数中完成。并将每隔关键帧进

行 render。分块的思想是创新的地方。

- ✓ 将 slam 中最新的关键帧图像和位姿数据进行 render。
- ✓ 利用 opencv 函数来实现图像的拼接重建。核心算法是拉普拉斯金字塔，我自己写过，但是写的太复杂。Opencv 的函数我没有用过，需要花些时间来熟悉和掌握。



从图中可以发现明显的重叠现象，这个应该是关键帧的整体优化时参数会变化导致的。

8.12 周日

周日过来继续调试变化检测的算法。下午和吴宇、徐崇斌一起讨论了一下界面的设计。但目前最重要的是将变化检测的效果做出来，因为我采用仿射变换来进行几何对齐的方式效果很差。我打算采用相似变换的方式来试试。

如果采用深度学习，能不能用特征提取器来对图像进行处理，得到特征图像。然后看看特征图像能否反映出变化。

8.13 周一

上午面试了两个实习生，和孟想交流了一下，下午去北大口腔医院给雯琦挂号。看了下关于 deeplab 的文献，但是没有看懂。目前最紧迫的是将变化检测的效果做出来，我从 github 上面找到了一个网页：

Unstructured-change-detection-using-CNN

This repository contains implementation of the paper 'Mohammed El Amin, Arabi & Liu, Qingjie & Wang, Yunhong. (2016). Convolutional neural network features based change detection in satellite images. 100110W. 10.1117/12.2243798'.

Instruction to run the file

The feat.py file accepts the two file paths corresponding to the two input patches as arguments.

For example to find Unstructured change in two image patches img1.PNG and img2.PNG, both located in the ./data/ folder by running

```
$ python feat.py ./data/img1.PNG ./data/img2.PNG
```

文章我只看到了摘要

Convolutional Neural Network Features Based Change Detection in Satellite Images

Arabi Mohammed El Amin*, Qingjie Liu*, Yunhong Wang

State Key Laboratory of Virtual Reality Technology and System

School of Computer Science and Engineering, Beihang University, Beijing 100191

ABSTRACT

With the popular use of high resolution remote sensing (HRRS) satellite images, a huge research efforts have been placed on change detection (CD) problem. An effective feature selection method can significantly boost the final result. While hand-designed features have proven difficulties to design features that effectively capture high and mid-level representations, the recent developments in machine learning (Deep Learning) omit this problem by learning hierarchical representation in an unsupervised manner directly from data without human intervention. In this letter, we propose approaching the change detection problem from a feature learning perspective. A novel deep Convolutional Neural Networks (CNN) features based HR satellite images change detection method is proposed. The main guideline is to produce a change detection map directly from two images using a pretrained CNN. This method can omit the limited performance of hand-crafted features. Firstly, CNN features are extracted through different convolutional layers. Then, a concatenation step is evaluated after an normalization step, resulting in a unique higher dimensional feature map. Finally, a change map was computed using pixel-wise Euclidean distance. Our method has been validated on real bi-temporal HRRS satellite images according to qualitative and quantitative analyses. The results obtained confirm the interest of the proposed method.

Keywords: Convolutional Neural Network (CNN), Change Detection (CD), High Resolution Remote Sensing (HRRS)

1. INTRODUCTION

Change detection (CD) is the heart process of many applications utilizing remote sensing images. It leads to the identification of changes that has occurred on the Earth's surface by processing two (or more) images acquired at different times that cover the same geographical area. CD has a wide range of uses, including land use and land cover change monitoring, risk assessment, urban growth studies and environmental investigation.

Based on hand-engineered features, a variety of algorithms have been proposed to solve the CD problem, such as image differencing (ID) [1], image rationing (IR) [1], principal component analysis (PCA) [2], change vector analysis (CVA) [3], expectation maximization (EM) [4], graph cut [5], the Parcel-based method [6] and Markov random field [7]. To calculate these hand-engineered features, parameters such as sizes, scales and directions should be prudently and elaborately selected. Also, features selection and combination is another obstacle for HR imagery CD.

Our approach is inspired by the recent success of CNN model [8, 9]. This model transform sequentially a given input to the expected output through a sequence of processing steps [8, 9], producing a hierarchy of feature maps via learned filters. CNNs trained on specific tasks are capable to automatically learn complex features from images and achieve superior performance compared to hand-crafted features [8, 9].

This work follows a similar line of thought, with question in mind: "Can we re-use a pre-trained deep CNN to detect changes in Bi-Temporal HRRS satellite images?"

Several recent researches prove that the upper layers of CNNs encode highly-abstract information about the input image [10]. However, they used the CNN features for generic tasks such as classification [11], where the goal is categorizing a holistic representation of the image without considering the object location within it. Lower levels features are good for correspondence, but higher levels are needed for semantic information. To get the best of both worlds, a feature fusion strategy is employed by stacking all feature maps in a high dimension hyper feature (figure 1).

文章对应的 Python 代码如下

```
#This file contains the python implementation feature based change detector
#Author: Bhavan Vasu

import tensorflow as tf
import keras
from keras.applications.vgg19 import VGG19
from keras.preprocessing import image
from keras.applications.vgg19 import preprocess_input
import numpy as np
import matplotlib.pyplot as plt
import sys
from skimage import filters #change to 'import filter' for Python>v2.7
from skimage import exposure
from keras import backend as K

#Function to retrieve features from intermediate layers
def get_activations(model, layer_idx, X_batch):
    get_activations = K.function([model.layers[0].input, K.learning_phase()],
                                [model.layers[layer_idx].output,])
    activations = get_activations([X_batch,0])
    return activations

#Function to extract features from intermediate layers
def extra_feat(img_path):
    #Using a VGG19 as feature extractor
    base_model = VGG19(weights='imagenet',include_top=False)
    img = image.load_img(img_path, target_size=(224, 224))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x = preprocess_input(x)
    block1_pool_features=get_activations(base_model, 3, x)
    block2_pool_features=get_activations(base_model, 6, x)
    block3_pool_features=get_activations(base_model, 10, x)
    block4_pool_features=get_activations(base_model, 14, x)
    block5_pool_features=get_activations(base_model, 18, x)

    x1 = tf.image.resize_images(block1_pool_features[0],[112,112])
    x2 = tf.image.resize_images(block2_pool_features[0],[112,112])
    x3 = tf.image.resize_images(block3_pool_features[0],[112,112])
    x4 = tf.image.resize_images(block4_pool_features[0],[112,112])
    x5 = tf.image.resize_images(block5_pool_features[0],[112,112])

    F = tf.concat([x3,x2,x1,x4,x5],3) #Change to only x1, x1+x2,x1+x2+x3..so on, inorder to visualize
```

```

features from diffetrnt blocks
    return F
def main():
    if (len(sys.argv))>3:
        print "Invalid number of input arguments "
        exit(0)

#Two aerial patches with change or No change
img_path1=sys.argv[1]
img_path2=sys.argv[2]

sess = tf.InteractiveSession()

F1=extra_feat(img_path1) #Features from image patch 1
F1=tf.square(F1)
F2=extra_feat(img_path2) #Features from image patch 2
F2=tf.square(F2)
d=tf.subtract(F1,F2)
d=tf.square(d)
d=tf.reduce_sum(d,axis=3)

dis=(d.eval())    #The change map formed showing change at each pixels
dis=np.resize(dis,[112,112])

# Calculating threshold using Otsu's Segmentation method
val = filters.threshold_otsu(dis[:,::])
hist, bins_center = exposure.histogram(dis[:,::],nbins=256)

plt.title('Unstructured change')
plt.imshow(dis[:,::] < val, cmap='gray', interpolation='bilinear')
plt.axis('off')
plt.tight_layout()
plt.show()
"""

Uncomment For veiwing a graph for visualizing threshold selection
plt.subplot(144)
plt.title('Otsu Threshold selection')
plt.plot(bins_center, hist, lw=2)
plt.axvline(val, color='k', ls='--')

plt.tight_layout()
plt.show()
"""

if __name__ == "__main__":

```

main()

基本思想是利用已有的网络来提取特征，然后将特征组织起来进行变化检测。
明天接着研究，争取用 darknet 来实现特征的旋转和变化比较。

8.15 周三

今天上午继续调试 darknet 的动态库，昨天因为我的主工程是 32 位，所以折腾了半天也没有链接成功。修改后可以实现 c++ 环境下面调用 c 语言输出的动态库函数，这也为后面的调试提供了很好的环境。

下午跟钟老师去亦庄，和小川、振兴交流，后面海量点云的算法我需要尽快的熟悉，然后才能融入到公司的研发中。Mask-RCNN 的流程也要尽快的跑通，后面提取道路线会用到。

8.16 周四

今天上午去所里跟老余讨论了一下仿真书的写作，感觉很无聊。中午去眉州小吃吃饭，然后开车到顺义公司。牟总找我聊天，说到公司这边想把我纳入核心成员，并提供股份，这让我很感动。后续我自己也得继续努力，争取为公司的发展做出一点自己的贡献，特别是在算法层面，要多多的读文献，深刻理解算法。下午还跟彭齐路一起讨论了 slam 移植的问题。

8.17 周五

变化检测：吴宇的主意也不错，在拼接时保存原始的图像，然后直接在图像上进行匹配和 warp。Warp 的时候用 homography 就能比较好的模拟相机转动时的影像变化情况。计算当前帧图像的梯度方向直方图，然后和拼接时使用的图片的梯度方向直方图进行匹配。代码放在了台式机上，所以回家后就没法调

试，准备安装一个远程控制软件：teamviewer 来实现远程访问。特别是后续有关深度学习的应用会越来越多，所以需要有一台功能强大的服务器来调试代码才行。基于 darknet 的目标定位与识别打包成了一个动态库，变化检测其实单独做成一个动态库更好。传统的方法目前我能想到的就是将图像进行分块，然后计算每块的梯度方向直方图来进行比较，如果其中的某块相似性较差，那么就有可能是出现了变化。

阴影去除：包括阴影的提取和阴影去除两个部分。网上没有找到合适的代码，所以需要查查文献，看看哪种算法的效果比较好，然后实现即可。

车道线的提取与分割：准备采用 mask r-cnn 算法来实现，首先需要将网上提供的开源算法调试并跑通，看看效果。

点云算法：安装最新的 pcl 版本，然后配置通过，测试常用的 pcl 算法。

8.20 周一

周六调试了一下变化检测程序，但是效果不好。周日上午去所里开仿真写书的会议，下午带雯琦去亦庄国际会展中心看机器人的展览。今天上午带雯琦去参加社区组织的交通协管志愿者活动，并且去翠微的盒马鲜生自己动手做了玫瑰饺子。下午去理科楼参加孙老师负责的变形监测项目讨论会，主要是想基于测科院李老师的算法来对青岛车站的房顶进行变形监测。

我目前的变化检测采用方向直方图的效果不好，让我觉得很奇怪，然后特别去看了 sift flow 的文章，发现其实和方向直方图的原理是差不多的。所以想仔细来调试一下。经过对比我发现目前的算法将梯度角度归一化到了 [0,180]，主要想想解决不同拍摄模式下的匹配，考虑到我们目前的算法都是基于同一个相机，所以我将角度还原到 [0,360]，测试后发现有变化的区域的相关系数 < 0.5，比原来的算法有了提高。我将图像拆分为 64*64 的块，计算每个块的梯度分布直方图。



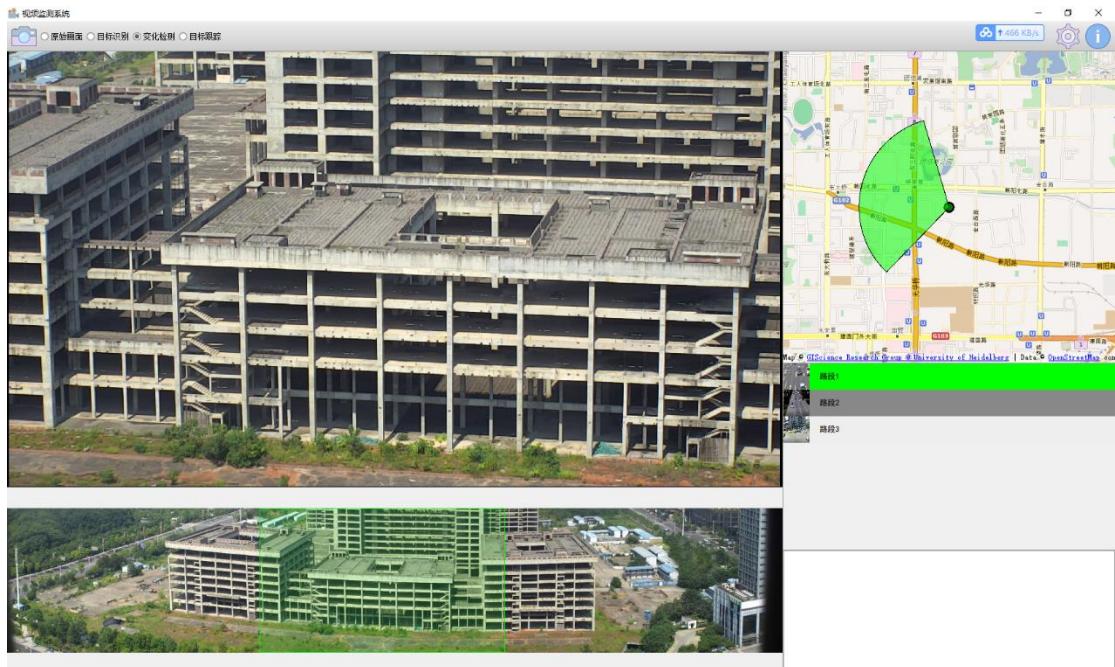


上面的图像是我利用 ps 的盖章工具人为制造的变化，利用梯度直方图计算的相关系数都在 0.5 以内。后续可以考虑将 64*64 的块再细分为 4 个 32*32 的块，将 4 个块的梯度直方图连接起来作为特征描述，这样的话效果应该会更好些。

因此目前我的变化检测思路是：利用整图的梯度直方图将视频帧和所有的底图进行匹配找到最相似的底图，利用 orb+homography+ransac 的方式来将视频帧和底图进行几何对齐，将图像拆分为 64*64 的块并计算每个块的梯度直方图特征。统计所有底图被匹配上的次数 Nr，以及每个块被检测为发生了变化的次数 Nc，如果 $Nc/Nr >$ 设定的阈值，那么就认为图像块发生了变化。根据适当的规则来将 Nr 和 Nc 置为 0。

8.21 周二

今天早上带雯琦去学钢琴，然后到学校继续调试变化检测的程序。根据昨天的思路构造了变化检测的类来管理所有相关的变量，这个方法是从 orb-slam2 的代码来学习的。在类的初始化函数中计算每张底图的梯度直方图，与当前视频帧的梯度直方图进行比较来寻找最相似的底图。为了将提取到的信息绘制到全景图上，还需要将底图和全景图来进行匹配。我采用了一个技巧，就是根据图像的顺序来大致定位出底图和全景图像分块后的哪个区域最相似，然后利用相似变换来将底图坐标变换到全景图像坐标。一直到晚上 9 点才将变化检测的接口写完并测试，然后交给小康集成到界面环境中。录完视频已经晚上 11 点多了。由于 darknet 相对独立，所以我将变化检测单独做成了一个动态库。



由于接口设计的简单，所以集成也比较顺利。后续需要提高速度，并对算法的有效性进行验证，看看这种基于块的变化检测算法到底性能如何？本来想试试深度网络的特征提取效果，但是时间有限。

8.24 周五

周三一早出发去德清，7点火车开动，遇到了同样去参加会议的杨健和郭红。我们中午一点左右到湖州市德清县，然后打车去金银岛饭店。注册后去吃自助餐，吃完饭我去办理入住。下午很困，睡了一觉，晚上去和志强，段福州会和吃饭，同桌还遇到了胡德勇老师，北师大的蒋老师和一位东南大学的老师。吃过饭后会宾馆继续看视频拼接的算法。周四上午 10.30 参加余老师组织的高分系列丛书的编委会，然后和彭斌见了面聊了一会。下午和吴俣，郭红，陈冬花一起去下渚湖湿地公园玩。5点左右我和吴俣打车到德清站，到了杭州东站乘坐 7 点高铁回北京，到达北京已经是晚上 11.30。今天图景那边跟我联系，说是视频拼接的算法会出现死掉的情况，因此还需要仔细阅读代码。

8.27 周一

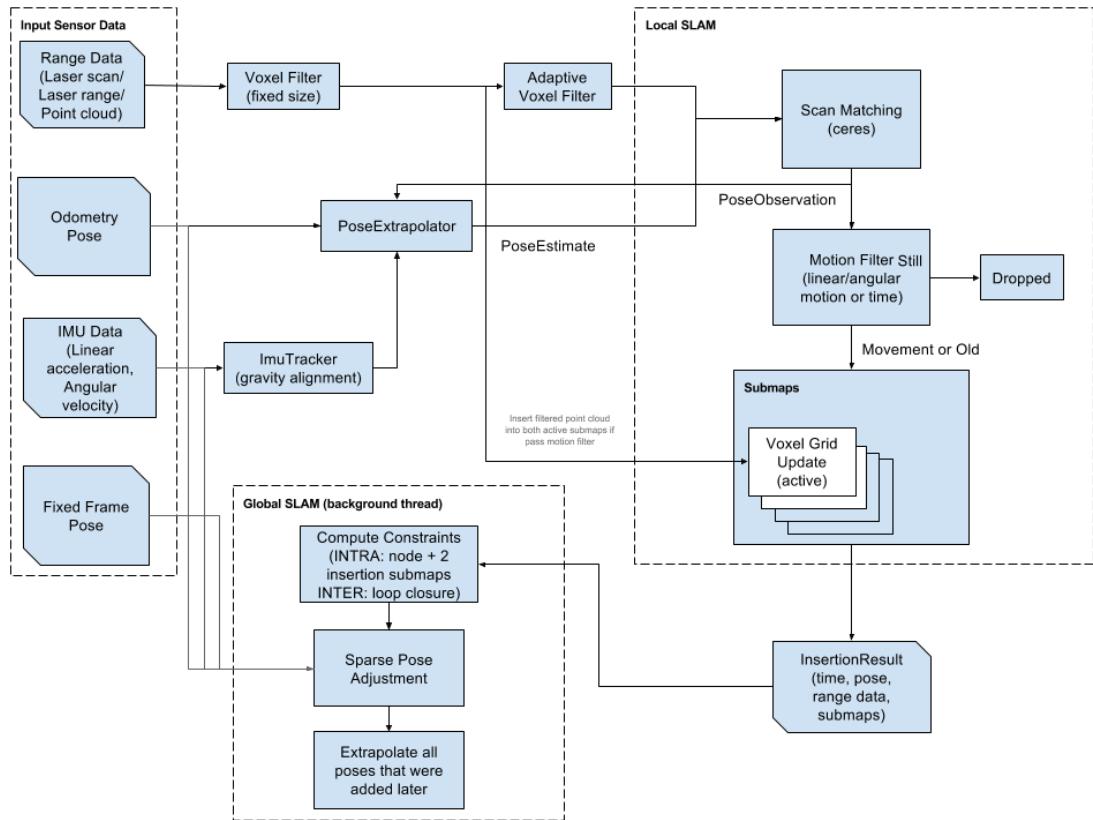
下午去亦庄进行每周的例行讨论，和一个公司进行了交流，主要想利用全景测量或者激光 slam 技术来实现地下停车场的量测以及识别统计。地下停车场测量方案：全景，激光 slam，配合 CAD 的地形图。学校这边测试激光 slam 的精度。分析轨迹的精度。全景目前最大的问题是失锁时的精度如何来保证。

8.28 周二

下午去顺义公司讨论，目前还存在的问题是：视频拼接的地理坐标计算，拼接的速度问题。同时还要关注后续的稠密点云实时构建问题。

8.29 周三

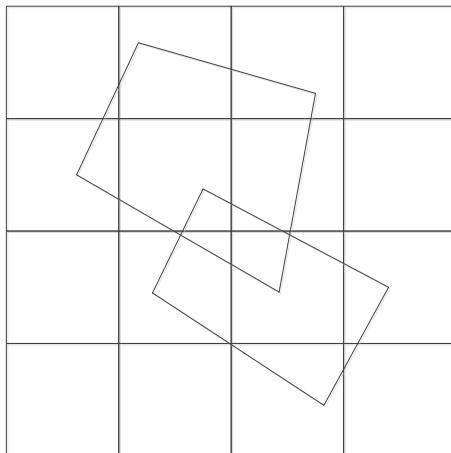
今天上午去 4s 店做保养，中午的时候去学校吃饭，然后看延禧攻略到 3 点。陈前来办公室找我讨论了一下。公司那边把推车送过来，但是螺丝没有弄好，所以激光扫描还没法用，只能明天一早去试试。因为要用背包，所以想了解一下 cartographer 的算法原理。上网查了些资料，但是看起来还是很费劲。



今天还大致看了下 cartographer 的代码，很复杂，没有什么头绪。

Map2dfusion 的加地理坐标的代码也看了一些，匀色的原理基本上看懂了。

为了提高速度和方便进行实时的匀色，这个算法采用了**分块**和**实时区域扩展**的方式。



图、分块拼接与匀色图

采用分块的优点是当新的图像进来后，可以只处理该影像覆盖的块，这样就可以提高处理的效率。在利用拉普拉斯算法进行重建时，只需要将分块的数据拷贝到一个连续的存储块中就可以了。为了解决重叠区域的重合问题，算法对每张图像构建了一个**权重图像**，以图像中心往四周逐渐降低，在进行拼接时

根据权重来填充图像数据即可。

关于变化检测：最近都没时间来弄。下一个环节是将变化区域的图像显示出来，我的想法是在算法中用变量来存储检测到的变化图像块，然后将其推送出去。

8.30 周四

今天一天都在教一楼的地下停车场用背包的激光来进行 **slam** 测量。但是对 **ros** 系统还是比较陌生，不知道是如何来调用 **carto** 进行实时的拼接和优化的。

8.31 周五

上午带雯琦去西城的北大口腔看牙，结束后打车回家，然后到 301 看肠胃，医生开好验血的单子我就去学校吃饭。下午继续编写视频拼接的代码。

9.1 周六

上午送雯琦去学校参加开学典礼，然后我去 301 抽血检验。下午带雯琦去了趟植物园。

9.2 周日

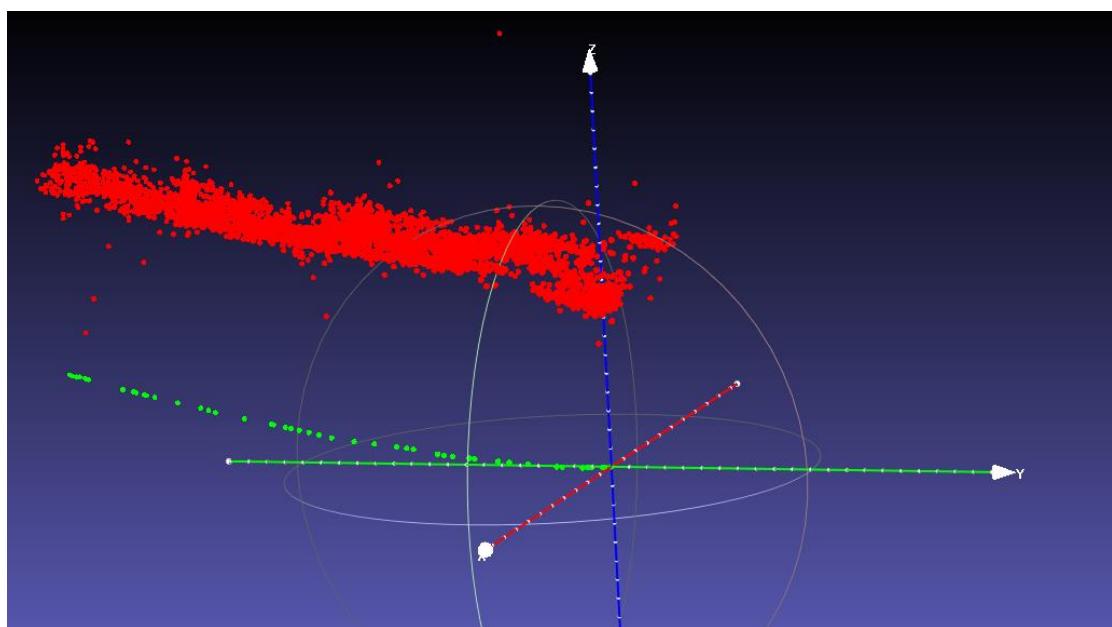
继续 508 项目联调。主要是将有变化的区域提取出来并且存到特定的路径中。我采用的思路是对检测到的矩形框进行合并（），然后将合并后的框返投影到每张基准图中，然后选择投影后最靠近图像中央的那张。为了防止重复存储，根据监测到的矩形框在全景图像上的位置来进行存储，如果框的位置没有发生变化，就不存储。

9.3 周一

目前有这么几件事情：

- 1、视频拼接加地理坐标；

关键帧是根据 `frame` 来初始化的，因此在构造 `frame` 时需要特别留意内部参数是不是被全部拷贝。在 `Frame::Frame(const Frame &frame){ }` 中就没有对 `gps` 信息进行处理。



从上图可以看出，slam 重建的三维点云 Z 值是正的，和我常用的摄影测量坐标系不太一样。我想采用三维绝对的方法来加地理坐标，但是效果很差，三维点进行旋转、缩放和平移后和大地坐标无法吻合。有可能是 `gps` 坐标本身就不太准确，也可能是局部地图恢复的相机位置和姿态本身就不准。

- 2、点云数据的车道线识别；
- 3、背包的绝对精度验证；

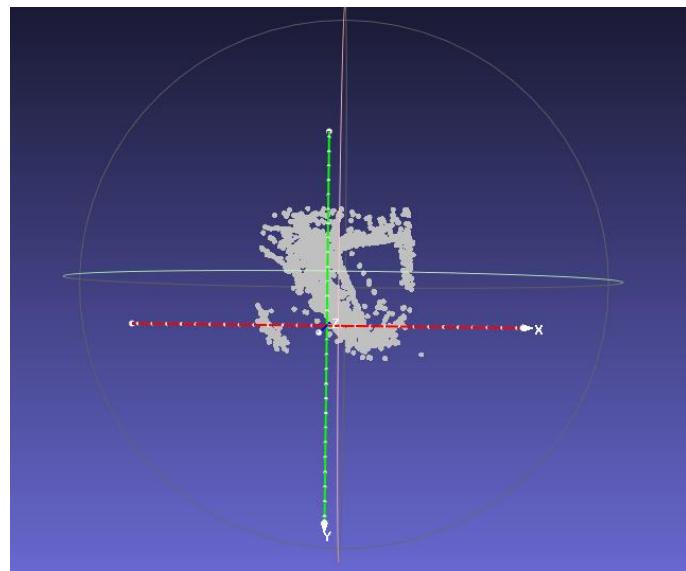
9.4 周二

上午带雯昕去打疫苗，中午跟中介去看出租的房子。继续研究视频拼接的

地理坐标问题。

9.7 周五

妈妈回四川，家里没有人照顾小孩，所以这两天都呆在家里负责做饭，很多事情都暂时没有做。在视频拼接中，采用三维坐标来进行绝对定向的效果很差，点云旋转后基本上看不出来规律。现在我准备基于二维来做：首先利用平面拟合将相机和点云转换到新的坐标系，然后利用相似变换来建立新的坐标系和大地坐标系之间关系，利用平面 warp 来实现地理坐标的建立。



图、计算机视觉的坐标系（目标在 Z 轴的正方向）

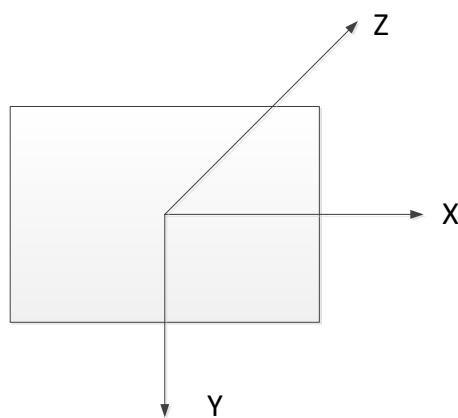
9.10 周一

难点：点云数据的自动矢量化采集时的遮挡问题；设备稳定性问题；采集时点云要覆盖整个交通标志；高温和防水功能；

9.11 周二

上午去所里参加空间信息项目的启动会，中午和吴俣、陈前去合川吃了饭。视频拼接目前已经实现了初步的地理信息添加，但是位置对不上。我目前的思路是这样的：首先根据恢复的地图三维点来进行平面拟合，根据拟合的平面将相机的位置和姿态进行旋转和平移，利用 **pos** 点将平面位置进行相似变换到大地坐标。

今天被 **orb-slam2** 的旋转矩阵搞晕了，由于 **slam** 算法一般采用计算机视觉的坐标系，和摄影测量的不太一样。



图、计算机视觉的像空间坐标系

9.13 周四

srcPts	{ size=10 }
capacity	13
allocator	
[0]	{x=0.015097676552463210 y=0.050233036205623922 z=-0.99868835487734009 ...}
[1]	{x=0.017914969295200929 y=0.017741987808890566 z=-1.0007884090525849 ...}
[2]	{x=0.017249827481925061 y=0.012577628949360112 z=-1.0017248206935148 ...}
[3]	{x=0.019474210453973226 y=0.0084588334453626415 z=-1.0014871901196052 ...}
[4]	{x=0.017721776123839548 y=-0.0036087710274310195 z=-1.0024072398947390 ...}
[5]	{x=0.019344845584209840 y=-0.014497692179240816 z=-1.0031824214018128 ...}
[6]	{x=0.018669760851825642 y=-0.027123059457008332 z=-1.0028649087549759 ...}
[7]	{x=0.018090753536762205 y=-0.035810530677711123 z=-1.0034089312229828 ...}
[8]	{x=0.019389835706401716 y=-0.043773662536592421 z=-1.0038778182954300 ...}
[9]	{x=0.018920614483692942 y=-0.049296654500706936 z=-1.0047741120214448 ...}

图、slam 获取的位置数据经过平面拟合后的效果

299083. 775623	3441868. 992323	299083. 734690	3441868. 546309
299064. 758718	3441872. 874868	299078. 074060	3441869. 991310
299061. 924096	3441874. 127083	299078. 074060	3441869. 991310
299059. 197859	3441873. 553729	299041. 557162	3441879. 103040
299052. 607662	3441876. 592886	299040. 220379	3441879. 470238
299046. 119695	3441877. 506660	299038. 215204	3441880. 021034
299039. 029168	3441880. 025283	299036. 874807	3441880. 398947
299034. 169545	3441881. 823667	299034. 864212	3441880. 965817
299029. 405911	3441882. 427934	299033. 523815	3441881. 343731
299026. 333531	3441883. 628947	299032. 183418	3441881. 721644

图、相似变换结果和经纬度转大地坐标

9.14 周五

拼接后的图片还是对不上，要奔溃了，搞了一个多星期。



图、视频拍摄的区域

299088. 429995	3441867. 640571	299083. 734690	3441868. 546309
299067. 002610	3441872. 474055	299077. 412624	3441870. 139175
299064. 466034	3441873. 702092	299077. 412624	3441870. 139175
299053. 875372	3441875. 517087	299039. 551987	3441879. 653836
299045. 097835	3441877. 969762	299037. 545006	3441880. 209991
299041. 181110	3441879. 856632	299036. 204609	3441880. 587904
299037. 670727	3441880. 084108	299034. 864212	3441880. 965817
299031. 685694	3441882. 154817	299033. 523815	3441881. 343731
299025. 433505	3441884. 366023	299031. 513220	3441881. 910601
299023. 061969	3441884. 353713	299030. 843022	3441882. 099558

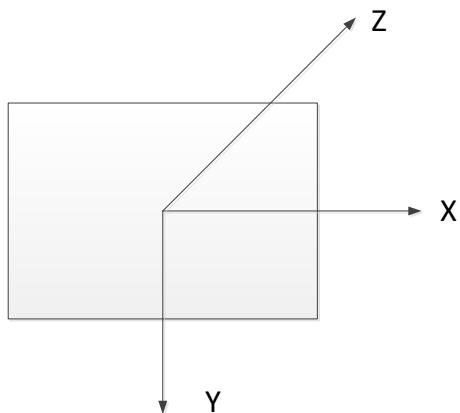
```

0 0.101841 -0.120039 -1.032964
1 0.065065 -0.111743 -1.037057
2 0.060712 -0.109635 -1.036933
3 0.042535 -0.106520 -1.038214
4 0.027470 -0.102310 -1.039089
5 0.020747 -0.099072 -1.040033
6 0.014722 -0.098681 -1.041603
7 0.004450 -0.095127 -1.041752
8 -0.006281 -0.091332 -1.043082
9 -0.010351 -0.091353 -1.042731

```

9.15 周六

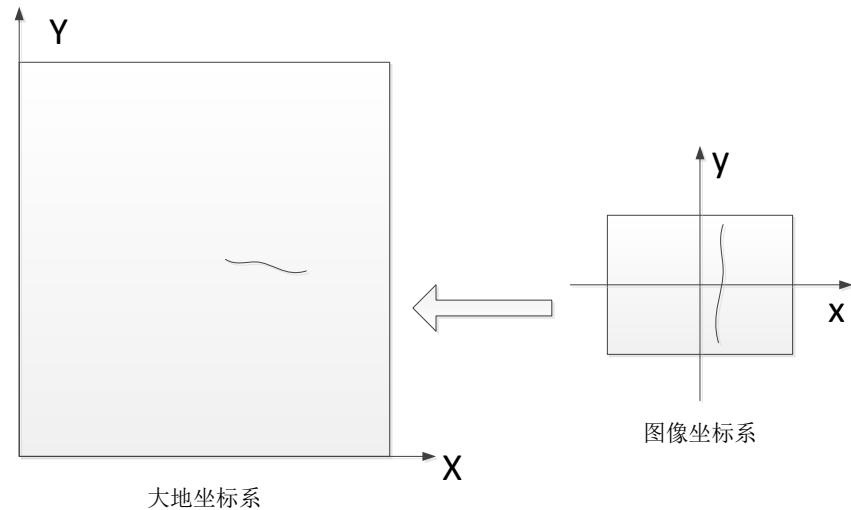
今天一早就去 301 做无痛胃肠镜，挺顺利的，感觉不到难受。继续研究 slam 和视频拼接，发现 slam 采用的坐标系采用了计算机视觉中常用方式。重建后的三维点的 z 坐标为正。而且 orb-slam2 重建后的三维点的 z 值基本上在 **1.0** 左右，不知道是什么原因。



采用这个坐标系的优点是焦距在齐次坐标里面是正值，但是跟摄影测量坐标系不同，所以从我自己的理解上会有很多不方便的地方。我一开始的思路是利用相机的平面位置和 pos 数据进行相似变换，但是怎么弄都无法和经纬的方向一致，后来发现如果采用平面位置，slam 的其实是一个左手系，而经纬度转成大地坐标后是一个右手系。所以我实在没有办法，就改变了算法的思路：首先将相机的位置转换到图像坐标，然后采用右手系来建立相似变换。直接建立拼接后好的图像和大地坐标之间的联系。

9.16 周日

上午带雯琦去跳舞，下午去五棵松绘画。今天利用图像和大地坐标之间的相似变换来加地理坐标，方向和位置基本上对了，但是比例尺还有差别。



拼接后的图像转到大地坐标系时可以使用一个简单的相似变换（或者更复杂一些的仿射、多项式变换）。由于都是右手系，因此只要一个旋转就可以实现方向的对齐。而相似变换中的旋转向量 a , b 的模其实就是分辨率，这根据相似变换的公式很容易看出来，因为旋转后需要进行一个缩放才能将图像坐标转换到大地坐标，这个缩放比例其实就是比例尺。

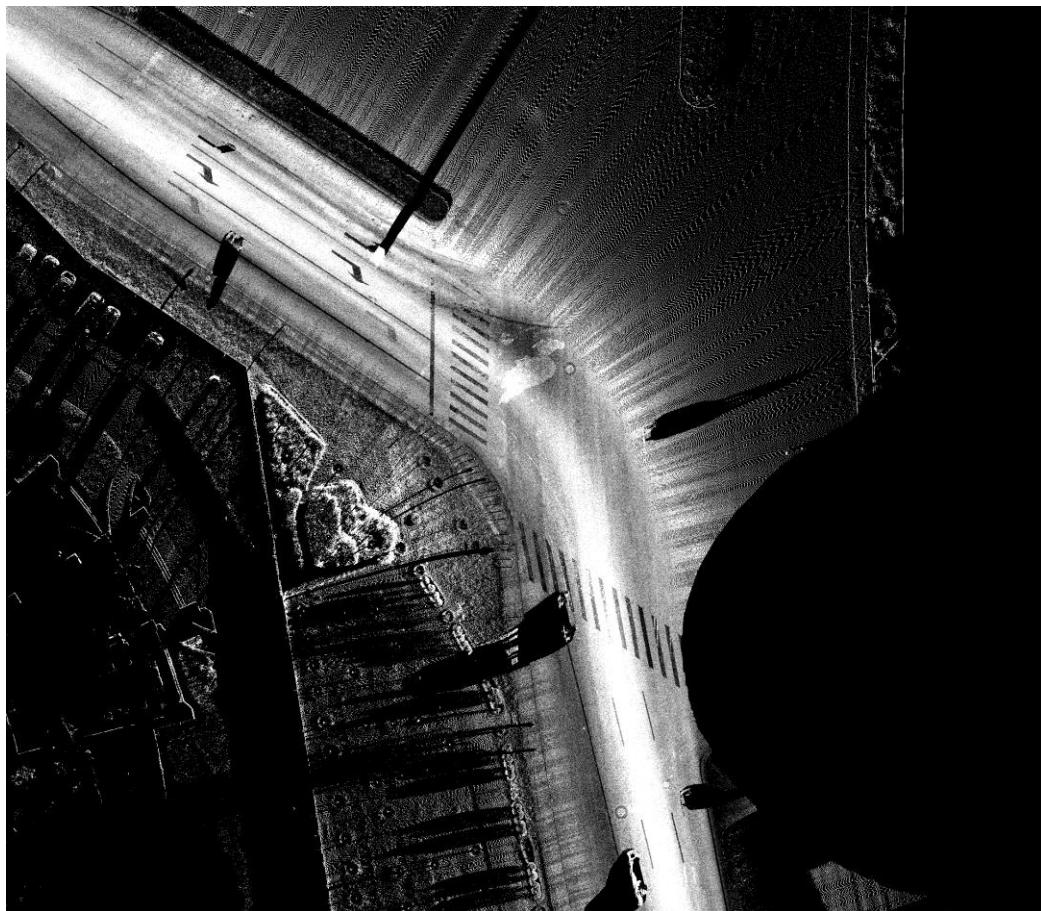


经过上述处理后的图像的方位和底图已经大致能够一致了，但是位置和比例尺还是有问题，我也不知道原因，可能是 pos，也可能是我的算法还有问题。

9.17 周一

目前工作任务列表：

- 1、视频拼接：比例尺问题，slam 失败的处理问题。还得要把 orb-slam 的核心看懂才行，这样才知道什么时候失败然后怎么来处理。
- 2、道路线提取：
- 3、写仿真书和几何的材料：
- 4、郑州答辩的 ppt：
- 5、激光 slam 程序和算法：按照 cartographer 的建议，首先在 14.04 上面安装 ros indigo。我首先安装了虚拟机，然后依据网页的命令来执行即可。



图、利用点云来生成的道路强度图

车道线和标志的提取思路：首先利用图像分割来得到前后背景，但是噪音太多了。应该是想办法把条状的区域找出来。可以利用形状特征来滤掉一部

分。然后看怎么结合 **hough** 变换来做些约束。可是噪音特别多。

9.18 周二

今天申请了理科楼的服务器访问权限，远程访问服务器来进行深度学习是一个很好的方法，不用自己搭建环境，只要有客户端软件(xshell & xftp)来连接就可以。但只能在控制台下来运行训练代码。

随着人工智能技术的不断发展，深度神经网络的学习模型开始应用到遥感领域。特别是利用深度学习来进行遥感数据的分类与识别，相比传统的分类与识别算法具有极大的优势。遥感数据具有海量的特点，也特别适合用来进行神经网络的训练。

仿真的过程就是根据已有的数据和设定的参数来得到新的数据。深度学习模型一般基于复杂的神经网络来构建，参数的数量巨大，具有极强的数据拟合能力。在仿真应用中，可以利用深度学习从离散的多光谱数据中学习出**高光谱仿真**的连续数据，其精度可以达到以假乱真的程度，特别适合仿真数据缺失情况下的图像仿真。如果我们将已有的底图看作是样本，将卫星获取的数据当作输出，那么就可以利用深度神经网络来模拟卫星的变化情况，比如模拟**不同分辨率的仿真**（类似超分）。但是几何变换没有办法用深度学习来模拟。

9.19 周三

正射出错，出来混迟早要还的。我写的递进式的处理方式速度慢而且算法也不稳定，所以准备用 **slam** 来替换，不知道能不能成功！

深度学习远程训练：

```
cd anaconda3/lib/python3.6/site-packages/tensorflow/models/research/deeplab
```

缺点是所有的操作都得在控制台下完成，不太适应。

9.21 周五

周三下午坐火车到郑州，参加周四的人才引进的答辩，还是住在锦江之星，不过没有上次住的那家舒服。农业遥感我的确不熟悉，而且评委主要对盈利模式感兴趣，感觉没啥戏。周四下午坐高铁回到北京。

周四晚上试了下 `slam` 用来处理无人机飞行用单反相机拍摄的图片，没有办法成功进行跟踪，原因在于 `slam` 处理的是视频数据，要求数据之间的重叠度要很大，满足跟踪的要求。但是单反相机拍摄的图片间隔比较大，即使能够初始化成功，也没有办法进行跟踪。`Slam` 的核心算法我还是没有弄懂，要继续加油！相机的参数需要在 `cfg.yml` 中来设定，`pos` 数据则存放在 `sub.txt` 文件中，不同的视频需要替换这两个文件。

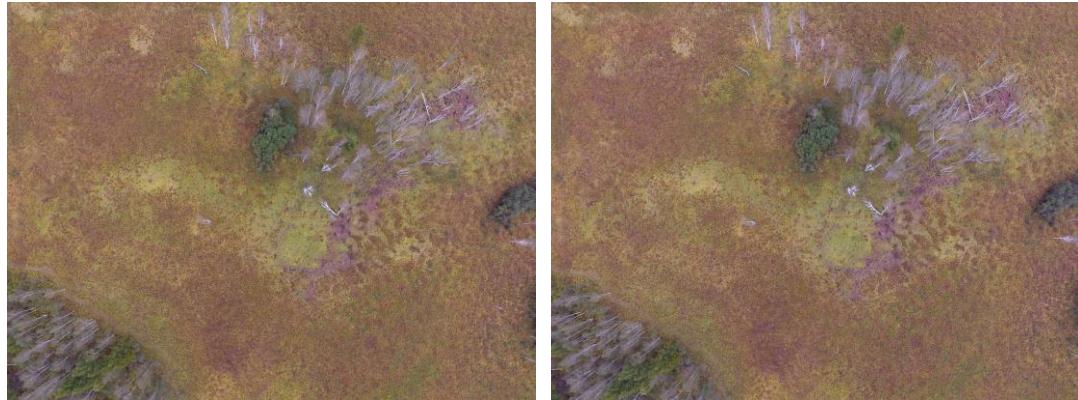
但是目前我编写的算法对比较复杂的飞行序列的效果不好，在优化的过程中参数变得越来越差，特别是焦距变得非常大，不知道是什么原因？还需要继续修改算法，看看有没有什么解决的办法。

下午还看了林允晖采集的学校的激光点云拼接数据，效果看着还不错，后续要把精度评价弄出来，就可以重点研究算法了。点云拼接和优化的代码后续要好好的看。

9.22 周六

今天带孩子们和老人去园林博物馆，风比较大，没有呆多久就回家来。白天要帮忙带雯昕，所以没有什么时间，晚上可以做些东西。

晚上调试 `sfm` 的代码，发现那段比较难拼的数据在初始化的时候选择的立体像对的基线很短，从而导致了相对定向的失败，也导致了整体拼接的失败那么如何来判断相邻两帧像片的基线合理而且匹配点也较多呢？



图、编号 67-68 的图像

Bundler 是根据匹配成功率来判断哪个像对作为初始像对，orb-slam 则是根据 homography 和 fundamental matrix 的误差来计算一个量：

$$R_H = \frac{S_H}{S_H + S_F}$$

如果 $R_H > 0.45$ ，那么就是 homography 的模式，否则就是 fundamental matrix 模式。

9.23 周日

上午带雯琦去上舞蹈课，然后去上钢琴课，下午去四季青的菜园活动了一下。



图、根据 score 挑选的初始像对 0058-0081

我没有完全用 orb-slam 文章里面的方法，而是根据 Homography 和 Fundamental Matrix 获得的正确点的数量来计算。原理是相同的，就是如果基线比较短，那

么就可以用单应性矩阵模型来进行描述。

9.24 周一

今天出发去岳阳调试代码，希望能把效果做出来。

调试 `sfm` 代码发现很奇怪的问题，焦距会越来越小，但是相机的位置分布大致还是合理的，残差也比较小，不知道是什么原因？？？焦距是最重要的的内方位元素，当焦距变小时会导致相机位置降低，覆盖的范围变大。

9.25-26 周二，周三

这两天在岳阳公司这边调试和测试代码，遇到了很多问题，对变化检测的算法又进行了一些优化：对初始化代码进行整理，解决内存泄漏问题；对图像存储进行修改，解决取图像块时的越界问题；和底图进行匹配时对相似度进行约束，如果相似度不够就不进行处理；在计算梯度直方图时，将梯度阈值设置为 **8**，这样可以将纹理比较平坦的区域也计算进行，增加有效像素的数量；将底图的索引保存到变化检测框里面，方便进行变化区域的存储。

但还是存在很多问题：特征点匹配失败导致图片对齐不准；底图和全景的匹配问题；多条带的匹配问题；相机旋转偏移时的几何对齐问题；更好的相似度计算方法等。这些问题都不好解决，需要考虑更新的算法，进行大量的测试。特别是如何将深度学习的算法引入到变化检测中。

9.27 周四

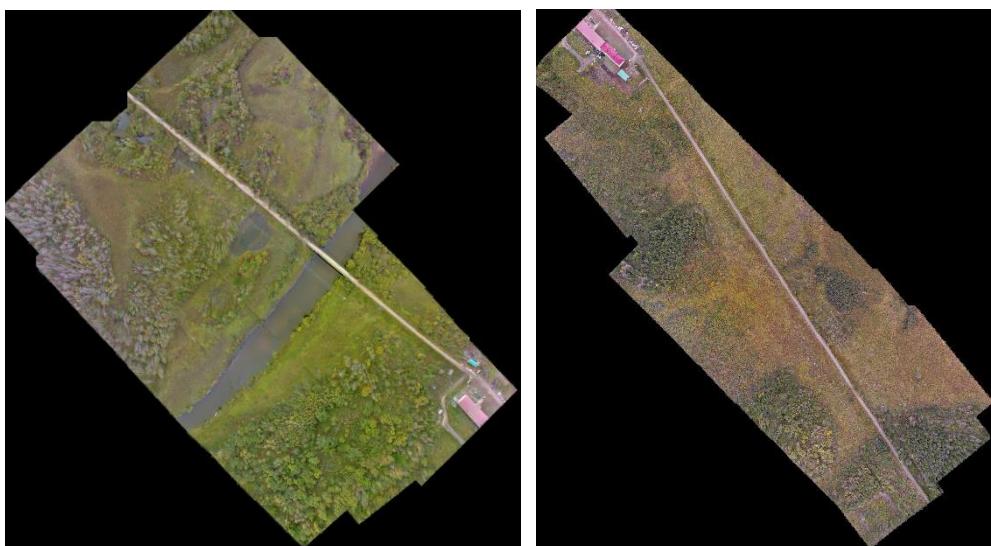
今天重点调试无人机图像的拼接，解决公司提供数据无法正常进行空三优化的问题。我打算参考 `colmap` 的一些思路。当加入新的图片时，有可能导致整体优化的结果变得更差，这个时候应该将新加入的图片删掉，恢复到没有加入

之前。

尝试在服务器上训练 deeplab，由于是纯控制台，所以操作起来不太方便。而且需要安装 labelme 和 opencv2 才行。Labelme 的 json_to_dataset.py 中还需要加入 convert() 函数才能够运行起来。但是在运行 build_lane_data.py –image_type="jpg" 的时候发现图像格式没法输入进去，我干脆把 FLAGS.image_format = "jpg"，这样就可以运行了。而且 tfrecord 目录不会自动生成，需要手工来创建。主要的麻烦就是要根据实际的路径来修改 python 代码，后续我要将所有动态的路径通过控制台传进去。

9.28 周五

今天到顺义，牟总请客吃饭，因为牟总是黑龙江人，所以吃得是东北菜。公司提供的数据拼接还是有问题，只能拼接上一部分，主要原因还是由于特征点较少，有些图片在 pnp 算法后剩余的点比较少，就没有办法加入到点云中。



图、50 张图像中两次拼接的效果

有没有什么更好的方法来提高拼接的完整性呢？？？而且在优化的过程中焦距的变化也很剧烈，最后的焦距明显是不对的。改进的方法包括：增加点的数量（把 GPU 提点的版本调试加进去看看效果）；提高匹配的成功率（修改一些参数）；固定相机的焦距；

9.30 周日

今天在学校进行工作。脑子比较乱，整理了一下，后续任务安排：

- ✓ 深度学习提取道路线：
- ✓ 正射算法的改进：提高速度和处理复杂数据的能力
- ✓ SLAM 算法的研究：研究半稠密，稠密 slam
- ✓ 点云算法的研究：研究常用的分类算法
- ✓ 改进背包：加入组合导航

今天继续尝试在服务器上训练 deeplab，但是出现了设备不能使用的问题，不知道什么原因，所以想首先测试一下最简单的手写体识别代码：

```
cd anaconda3/lib/python3.6/site-packages/tensorflow/models/tutorials/image/mnist
```

手写体代码中没有发现怎么来修改 gpu 的设备号，所以我切换到 cifar10。这里没可以用 `tf.device('/gpu:1')` 来设置 gpu 的索引号，但是诡异的事情出现了，当我运行训练代码的时候，目录下的所有文件全部消失了！

在控制台下熟悉了 vi 的一下用法，最方便的就是查找了（/待查找内容，n 遍历），所有的操作都可以利用键盘来实现，不愧为代码编辑的神器。

10.1 周一

今天国庆节，雯昕还小，所以就带老婆和孩子们去了离家不远的园林博物馆。最近北京天气非常的好，大气环境治理有了很明显的成果。北京城天高云淡，碧空如洗。园林博物馆门前布设了许多鲜花，看着让人心旷神怡。馆内有盆景展览，还将苏州和岭南的特色园林复制了过来，非常的漂亮，为中国人和中华文化自豪。

继续研究控制台下的训练，将 tensorflow 官网下载的 models 拷贝到了控制台的目录下，运行 mnist 和 cifar10 都可以成功。

```
cd /anaconda3/lib/python3.6/site-packages/tensorflow/models/tutorials/image/cifar10
```

每次进入到这个目录都非常麻烦，所以我干脆用脚本语言来调用命令实现训练，我在用户目录下面构造一个 `programs` 的目录，然后写了一个 `train.sh` 的脚本文件，其内容为：

```
#!/bin/bash
codedir="/home/xdonghai/anaconda3/lib/python3.6/site-
packages/tensorflow/models/tutorials/image/cifar10"
pwd
cd "${codedir}"
python cifar10_train.py
```

这样我就可以在用户目录下面来进行训练了。

但是在控制台下面进行数据的预处理实在太麻烦了，特别是切换目录和文件编辑，很不方便。所有代码可以考虑在 `windows` 下面写好，然后通过软件传输到服务器上去执行。

最近要重点加强点云方面的知识，为后续在公司研发打下基础。背包的组合导航知识也需要补充一些。顺义公司的无人机正射和 `slam` 也需要继续加强。

10.2 周二

为了方便进行预处理和训练，我准备把所有的操作都写成脚本，简化 `deeplab` 训练的复杂的过程。脚本本身就是编程语言，可以在其中调用 `python` 等命令来执行操作。

10.3 周三

今天我们一家去花卉大观园游玩，景色很不错。特别是温室里面的花卉种类繁多，繁花似锦，雯琦和雯昕都很开心。

准备利用一些时间来将点云的基本知识入门，准备从 `pcl` 来着手。晚上将 `pcl` 的一个程序跑通，主要是要配置 `boost`。以前的 `boost` 在 `vs2017` 中无法编译，所以下载了最新的版本。但是又缺少库文件，然后根据网上的提示来编译成功了库文件。具体操作时：先打开 `vs2017 64` 位环境的 `cmd` 环境，进入 `boost` 源目录，运行 `bootstrap.bat`，然后运行下面编译命令：`bjam stage --toolset=msvc-`

```
14.1 architecture=x86 address-model=64 --without-graph --without-graph_parallel --  
stagedir="c:\boost\boost_1_64_0\bin\vc141-x64" link=static  
runtime-link=shared runtime-link=static threading=multi debug release
```

10.4 周四

点云的实验数据可以从下面的网页下载:

<https://github.com/PointCloudLibrary/data>

pcl 的源代码也从网上下载完成，后续用 cmake 编译成功后，可以熟悉一下这些算法的原理和调用。

我让李振把我的权限提升为管理员，这样方便安装一下库，比如 zlib:

```
tar -xvf ~/Downloads/zlib-1.2.9.tar.gz  
cd zlib-1.2.9  
sudo -s  
../configure; make; make install  
cd /lib/x86_64-linux-gnu  
ln -s -f /usr/local/lib/libz.so.1.2.9/lib libz.so.1  
cd ~  
rm -rf zlib-1.2.9
```

将预处理和训练都写到一个脚本文件: train.sh 中。

10.8 周一

亦庄开会：实时出点云，实时 rtk。低成本激光。深度学习解决实际问题。点云处理的智能化处理算法。

10.11 周四

这几天杂事很多，不能集中精力来做事情。9号去机场接妈，10号给老婆

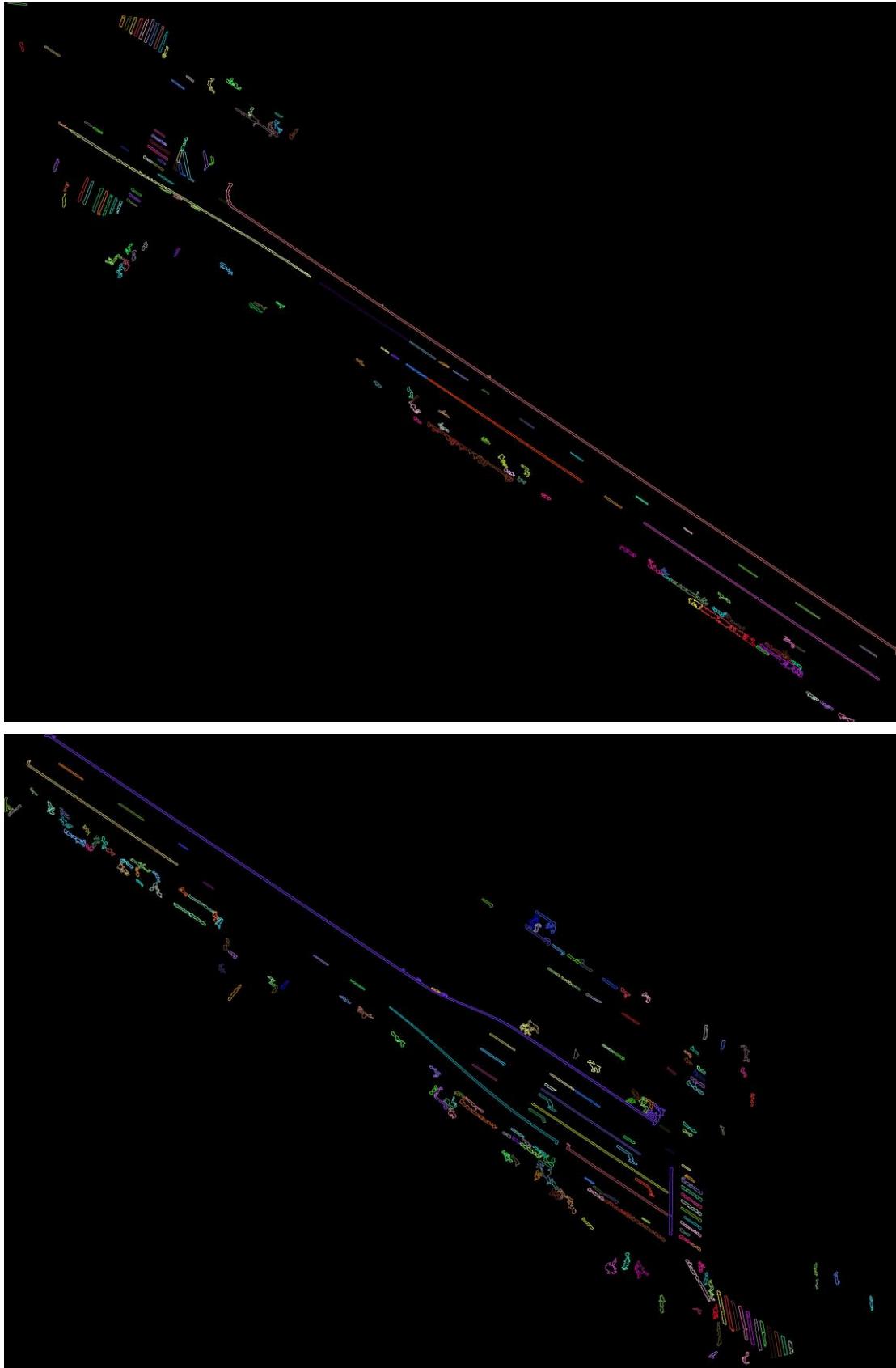
去报销保险，今天则是去建筑大学的大兴校区，时间就这样没有了。下午去东区踢足球，感觉根本就跑不动，体力太差了，被对方灌进去 5 个球，唉。。。

从点云强度图提取车道线的算法没有那么简单，但是我从 `github` 的一个网页找到了一些思路。其实就是根据车道线中间亮，两边黑的特点来设计一个简单的过滤器，就这么简单，但是我自己就是想不到。然后点云强度图有很多小黑洞，需要进行填充，否则会影响提取的效果。

10.12 周五

上午去亦庄开会，见到了张磊，了解了公司目前的一些情况。背包目前不是最紧要的设备，等到精度达到 10cm ，才会考虑产品化。

基于点云的车道线提取思路：生成强度图，填充黑洞，根据车道线特征进行分割得到二值图像（考虑水平和垂直两个方向），提取轮廓，寻找每个轮廓的主方向并将轮廓进行旋转，利用车道线是矩形的特点剔除轮廓（长宽比，轮廓的周长。。。）。效果还是不错，但是转弯的标志还需要进一步研究。



目前基于视觉的方法最麻烦的就是调参数，后续还是试试深度学习的方法。然

后看看能否结合点云的特点来降低噪音。

10.13 周六

第三专题：空地结合的建筑垃圾精准估算技术研究

本专题研究以低空无人机、飞艇、地面移动平台为载体，以倾斜摄影测量和三维激光扫描主要手段的建筑垃圾信息快速采集方法；建立空地结合的建筑垃圾监测数据精准三维可视化模型，实现建筑垃圾的边界提取，不同类型建筑垃圾的位置解析，堆体的占地面积、体积、坡度等参数的计算，为堆存区域及消纳场内建筑垃圾管控决策提供有效、可靠的测量数据支持。技术流程如下图。



图、空地结合的建筑垃圾精准估算技术流程图

在实际的研究与项目实践中，本专题可以细分为三个模块：数据采集模块，数据预处理模块，建筑垃圾参数估算模块。其中数据采集模块包括采集平台，采集设备，采集数据。预处理模块利用多角度图像 SFM(Structure From Motion)密集点云重建技术、激光点云滤波和拼接技术来获取高密度、高精度的

点云数据，利用点云数据和 POS 系统获得的控制点数据可以生成正射影像产品。基于点云数据和正射影像，并针对建筑垃圾的特点，利用图像分类和识别技术，以及点云几何处理技术可以获取建筑垃圾的体积、面积、坡度等重要信息。

1、采集模块

该模块依托目前主流的高精度数据采集平台：低空无人机、飞艇和地面移动测量平台，利用多角度拍摄图像的倾斜相机系统、单反相机和激光扫描仪来采集多角度影像和点云数据。为了能够直接获得数据的地理坐标，以及实现影像和点云数据的自动对齐，采集模块集成卫星定位数据接收模块(GPS 和北斗)和惯性导航模块来获取绝对坐标和姿态角数据。采用组合导航系统标定方法来对多个设备进行标定，获取标定数据。根据标定数据，可以实现影像和点云数据的自动对齐。倾斜相机系统一般包括 5 个镜头，可以从垂直和四个倾斜方向来拍摄影像。单反相机则利用多个航带来实现多角度影像的拍摄。激光扫描仪（LIDAR）可以直接获取高精度的点云数据。综合利用这些设备可以同时获取多角度的影像数据和点云数据。

2、数据预处理模块

数据预处理模块则利用 SFM（Structure From Motion）技术来从多角度影像中提取密集点云，SFM 是计算机视觉和摄影测量中常用的三维信息恢复技术，具体包括的技术点为：特征点提取、特征点匹配、连接点提取、相对定向、空三优化、绝对定向、密集匹配、正射生成。基于优异的特征点提取算法（如 SIFT、SURF、ORB），可以实现整个流程的全自动化处理。激光扫描获取的点云数据则需要进行拼接来得到整体的点云数据，建筑垃圾一般位于室外，因此可以直接利用卫星定位系统和惯性导航获取的站点位置和姿态数据可以实现点云的自动拼接。

3、建筑垃圾参数估算模块

建筑垃圾的参数估算模块利用图像自动识别技术来对建筑垃圾进行定位和精确的边界提取。结合传统的计算机视觉算法和深度学习的自动学习机制，可以准确的从正射影像中提取建筑垃圾的覆盖范围，计算面积。在建筑垃圾的识别基础上，利用三维点云的三角网构建算法可以对建筑垃圾进行建模，根据建模结果来计算建筑垃圾的体积和坡度等重要参数。影像数据和点云数据各有优

点。其中影像数据具有纹理信息，便于进行识别和分类，而激光点云数据可以精确的获取目标的轮廓信息。将影像数据和激光点云数据进行结合，能够利用两种数据的优点，最大限度的实现建筑垃圾的高精度、自动化参数提取。

10.14 周日

If life were caught by a clarionet, 如果生活与乐管同奏，
And a wild heart, throbbing in the reed, 狂热的心和簧片共振；
Should thrill its joy and trill its fret, 生活中的欢乐与烦恼，
And utter its heart in every deed, 全然是心灵的颤动。

Then would this breathing clarionet 当呼吸与簧管同步，
Type what the poet fain would be; 诗人将欣然命笔；
For none o' the singers ever yet 尽管没有一个歌手，
Has wholly lived his minstrelsy, 用他毕生吟唱。

Or clearly sung his true, true thought, 或纯粹唱他真实的思考，
Or utterly bodied forth his life, 或完全唱他生命之形体，
Or out of life and song has wrought 或唱生命的消逝，
The perfect one of man and wife; 或唱一个男人和妻子完美的功绩
Or lived and sung, that Life and Song 生活和歌唱，生命与歌，
Might each express the other's all, 一切都可以彼此表达，
Careless if life or art were long 要是生命或艺术被长期忽视，
Since both were one, to stand or fall: 那么好坏都一样：
So that the wonder struck the crowd, 所以群众创造奇迹，
Who shouted it about the land: 他们唤醒了土地：
His song was only living aloud, 他的歌仅仅是高亢一曲的生活，
His work, a singing with his hand! 他的劳作就是他用手唱出的歌！

10.15 周一

今天上午和下午分别听取了朱庆老师关于公司和技术方面的报告，深受启发，就是你要把一件事情坚持做下去，10年，20年就一定会有收获。我从2009年开始接触图像拼接，到现在还是在摸索，特别是密集匹配一直没有成功，而且稀疏点的重建也存在很多问题，所以还要再努力10年。也许到我50岁的时候能把这个问题搞透，同时也把相似的slam和深度学习弄清楚。

密集匹配还是要从PMVS2来看起：<https://www.di.ens.fr/pmvs/>。

10.16 周二

最近工作重点：

- 1、 编写图像融合算法，对以前的方法进行改进，采用空间距离权重的方法来进行融合。同时查找一下最新的融合文献。
- 2、 密集匹配算法的整理，这是一直想做的，但是没有坚持下来，要查找一下最新的算法。
- 3、 全景slam算法整理。审稿的文章中说精度可以达到20cm以内。所以想对算法进行改进，特别是加入g2o来进行优化
- 4、 深度学习训练车道线检测。

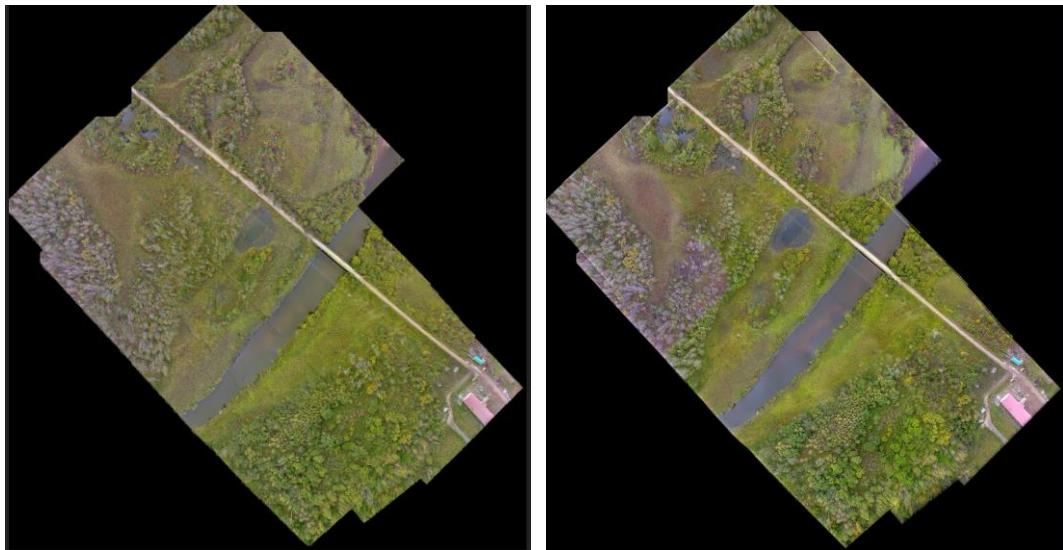
今天中午去东区踢球，热身不够，一个大脚把大腿给拉伤了，没有办法走路。这也是教训啊，剧烈运动之前一定要充分的热身。

10.17 周三

今天上午带雯昕去打疫苗，然后赶到学校差不多就吃午饭了。中午参加深度学习的培训班，三点多去接雯琦放学。

10.18 周四

受 map2d 的启发，我准备将融合的算法重新编写，大概花了两天的时间。



左：老的融合效果；右：新的融合效果



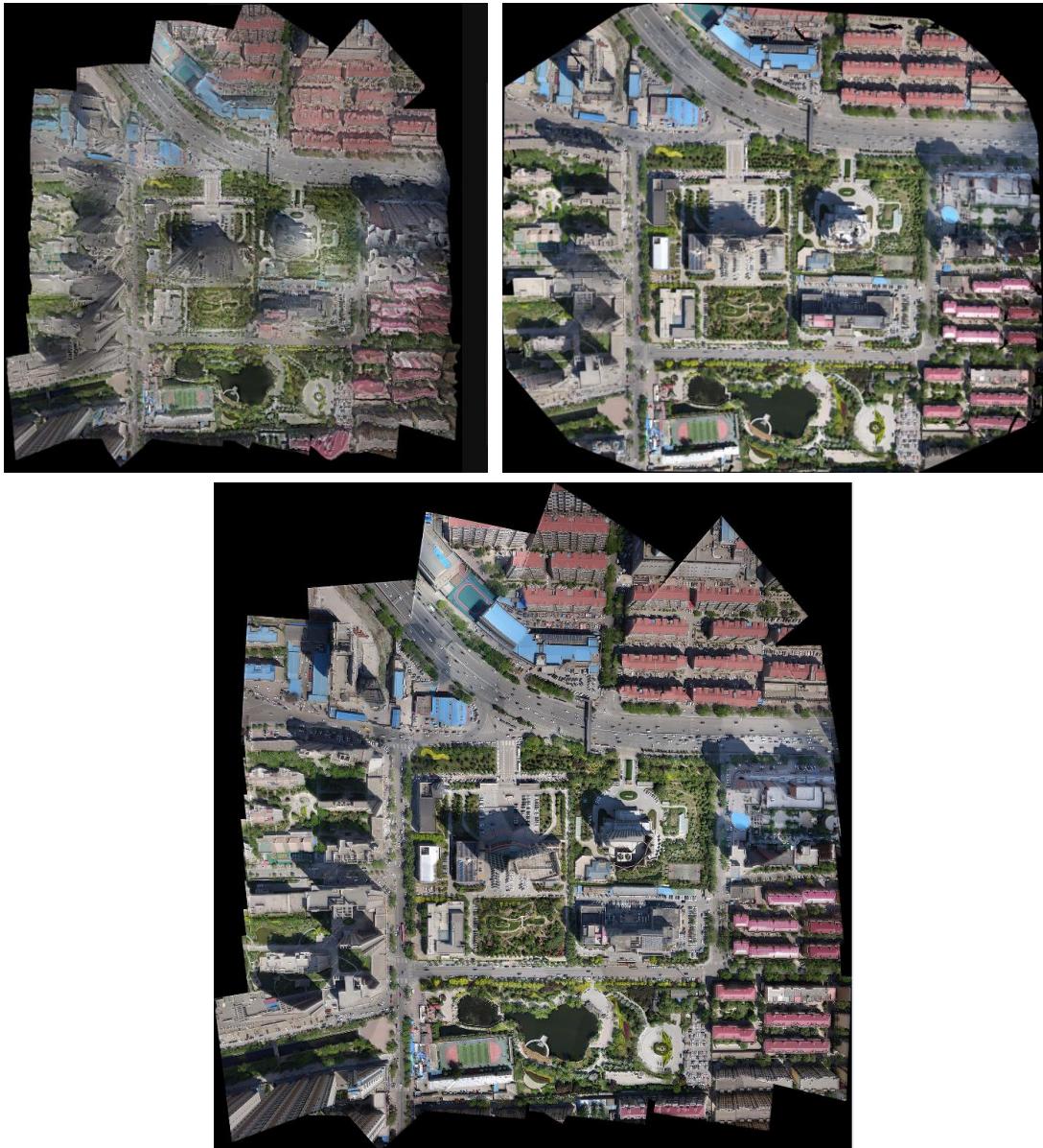
可以看出新的融合算法没有接缝的痕迹，而且色彩没有失真。整个算法是基于 opencv 提供的两个简单的函数：

`cv::detail::createLaplacePyr()` 和 `cv::detail::restoreImageFromLaplacePyr()`

10.19 周五

上午去所里和天海，吴俣商量 GF5 号结题的事情，后面要把精度验证多做些效果出来。下午去顺义公司看了视频拼接，速度还是比较慢，牟总对点云比

较关心，后续要花些时间把半稠密点云的效果做出来。



图、上面一排左边是我自己的老的拼接算法的效果，右边是 pixel4d 的效果。

下面一排是用新方法匀色的效果。

由于我目前只用了稀疏点云，而且匀色是自己写的，所以效果比起 pixel4d 来差的很多。密集匹配的算法我到目前为止还是没有搞定，所以一直没有往程序里面去加。后续还是要加进去才行。我不使用稀疏点云生成的 dsm，而是直接用地面的平均海拔来作为高程可以一定程度避免图像畸变，然后用新的匀色算法得到的结果要明显好一些。

10.22 周一