手把手教你如何用EVO评估VINS-Mono定位精度？

文档说明：

EVO安装

EVO官方链接：
https://github.com/MichaelGrupp/evo.git
安装方式：最简单的是命令行安装：

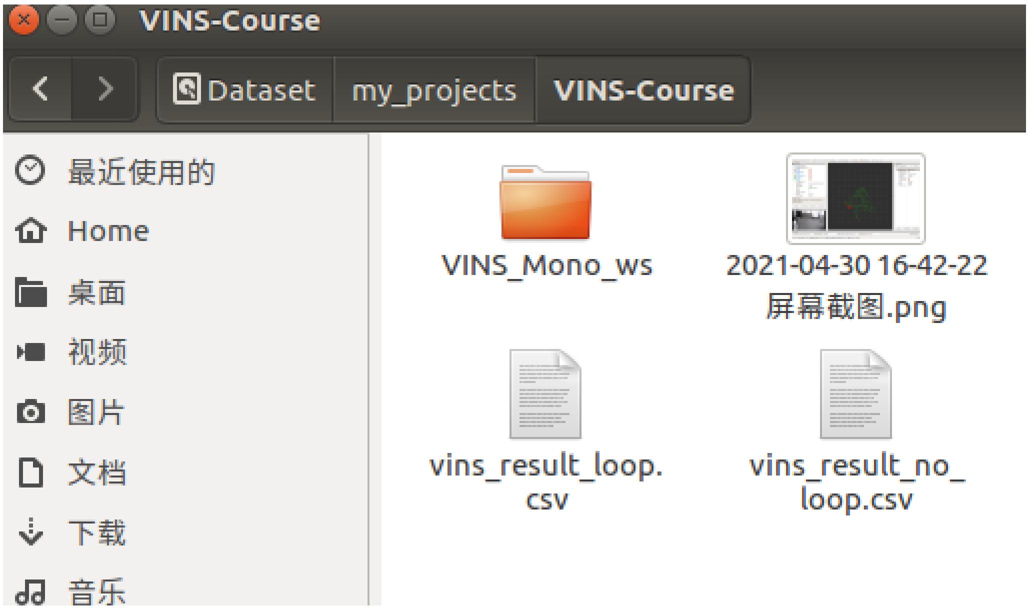
```
pip install evo --upgrade --no-binary evo
```

EVO支持三种数据集的ground truth格式：TUM，KITTI，euroc

EVO工具评估VINS-Mono的方法

以euroc数据集为例子，需要修改 VINS-Mono/config/euroc/euroc_config.yaml文件中的output_path和pose_graph_save_path，将后面的内容改为自己的路径，例如：
output_path: ~~"/home/shaozu/output/"~~
output_path: "/media/yhp/Dataset/my_projects/VINS-Course"

VINS-Mono把数据集运行结束以后生成的轨迹文件位于修改后的路径下，如图所示：



两个文件vins_result_loop.csv和vins_result_no_loop.csv分别是有回环和没有回环的轨迹。vins_result_loop.csv一共有8列，分别是时间戳，3自由度的平移XYZ，旋转的四元数qw,qx,qy,qz。vins_result_no_loop.csv多了3列分别是xyz三个方向的Vs，即速度。我跑的是V1_02_medium序列，那么他的ground_truth位于 V1_02_medium/mav0/state_groundtruth_estimate0/data.csv，把它复制到刚刚的目录下，重命名为GT_V1_02_medium.csv，方便比较.

在终端中打开包含groundtruth文件和运行结果的目录，显示ground truth轨迹

```
evo_traj euroc GT_V1_02_medium.csv -p --plot_mode xyz
```

将运行结果与ground truth对比，由于vins生成的结果与euroc ground truth的格式不一样，需要把格式统一为tum，每一列分别为：x,y,z,qz,qy,qz,qw，首先

```
evo_traj euroc GT_V1_02_medium.csv --save_as_tum
```

修改代码，首先是visualization.cpp中的pubOdometry()函数

```
// write result with tum format, for evo tools
ofstream foutC(VINS_RESULT_PATH, ios::app);
foutC.setf(ios::fixed, ios::floatfield);
foutC.precision(0);
foutC << header.stamp.toSec() << " ";
// foutC << header.stamp.toSec() * 1e9 << ", ";
foutC.precision(5);
foutC << estimator.Ps[WINDOW_SIZE].x() << " "
      << estimator.Ps[WINDOW_SIZE].y() << " "
      << estimator.Ps[WINDOW_SIZE].z() << " "
      << tmp_Q.x() << " "
      << tmp_Q.y() << " "
      << tmp_Q.z() << " "
      << tmp_Q.w() << endl;

// foutC << estimator.Ps[WINDOW_SIZE].x() << ","
//       << estimator.Ps[WINDOW_SIZE].y() << ","
//       << estimator.Ps[WINDOW_SIZE].z() << ","
//       << tmp_Q.w() << ","
//       << tmp_Q.x() << ","
//       << tmp_Q.y() << ","
```

```
//        << tmp_Q.z() << ","
//        << estimator.Vs[WINDOW_SIZE].x() << ","
//        << estimator.Vs[WINDOW_SIZE].y() << ","
//        << estimator.Vs[WINDOW_SIZE].z() << "," << endl;
foutC.close();
```

pose_graph.cpp中的updatePath()函数

```
if (SAVE_LOOP_PATH)
{
    ofstream loop_path_file(VINS_RESULT_PATH, ios::app);
    loop_path_file.setf(ios::fixed, ios::floatfield);
    loop_path_file.precision(0);
    // loop_path_file << (*it)->time_stamp * 1e9 << ",";
    loop_path_file << (*it)->time_stamp << " ";
    loop_path_file.precision(5);
    loop_path_file  << P.x() << " "
          << P.y() << " "
          << P.z() << " "
          << Q.x() << " "
          << Q.y() << " "
          << Q.z() << " "
          << Q.w() << endl;
    // loop_path_file  << P.x() << ","
    //        << P.y() << ","
    //        << P.z() << ","
    //        << Q.w() << ","
    //        << Q.x() << ","
    //        << Q.y() << ","
    //        << Q.z() << ","
    //        << endl;
    loop_path_file.close();
}
```

pose_graph.cpp中的addKeyFrame()函数

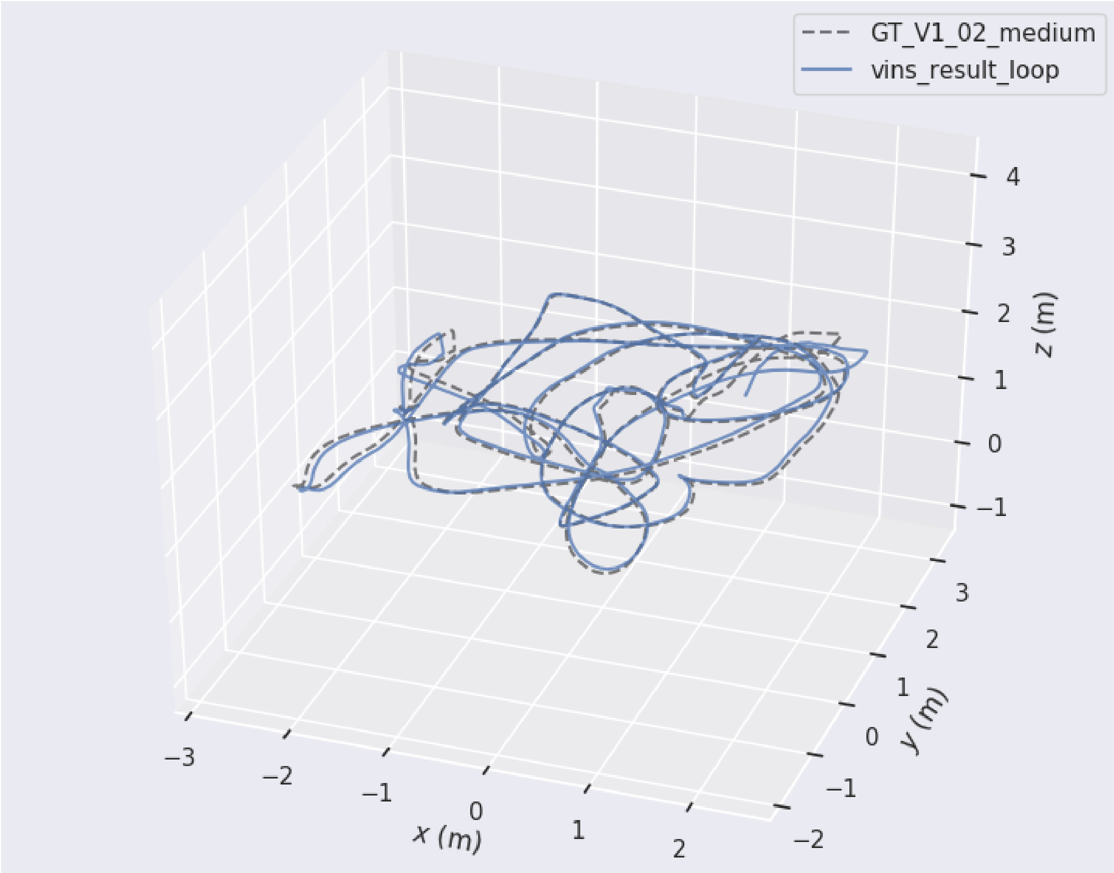
```
if (SAVE_LOOP_PATH)
{
      ofstream loop_path_file(VINS_RESULT_PATH, ios::app);
      loop_path_file.setf(ios::fixed, ios::floatfield);
      loop_path_file.precision(0);
      // loop_path_file << cur_kf->time_stamp * 1e9 << ",";
      loop_path_file << cur_kf->time_stamp << " ";
      loop_path_file.precision(5);
      loop_path_file  << P.x() << " "
            << P.y() << " "
            << P.z() << " "
            << Q.w() << " "
            << Q.x() << " "
            << Q.y() << " "
            << Q.z() << endl;
      // loop_path_file  << P.x() << ","
      //        << P.y() << ","
      //        << P.z() << ","
      //        << Q.w() << ","
      //        << Q.x() << ","
```

```
    //      << Q.y() << ","
    //      << Q.z() << ","
    //      << endl;
    loop_path_file.close();
}
```

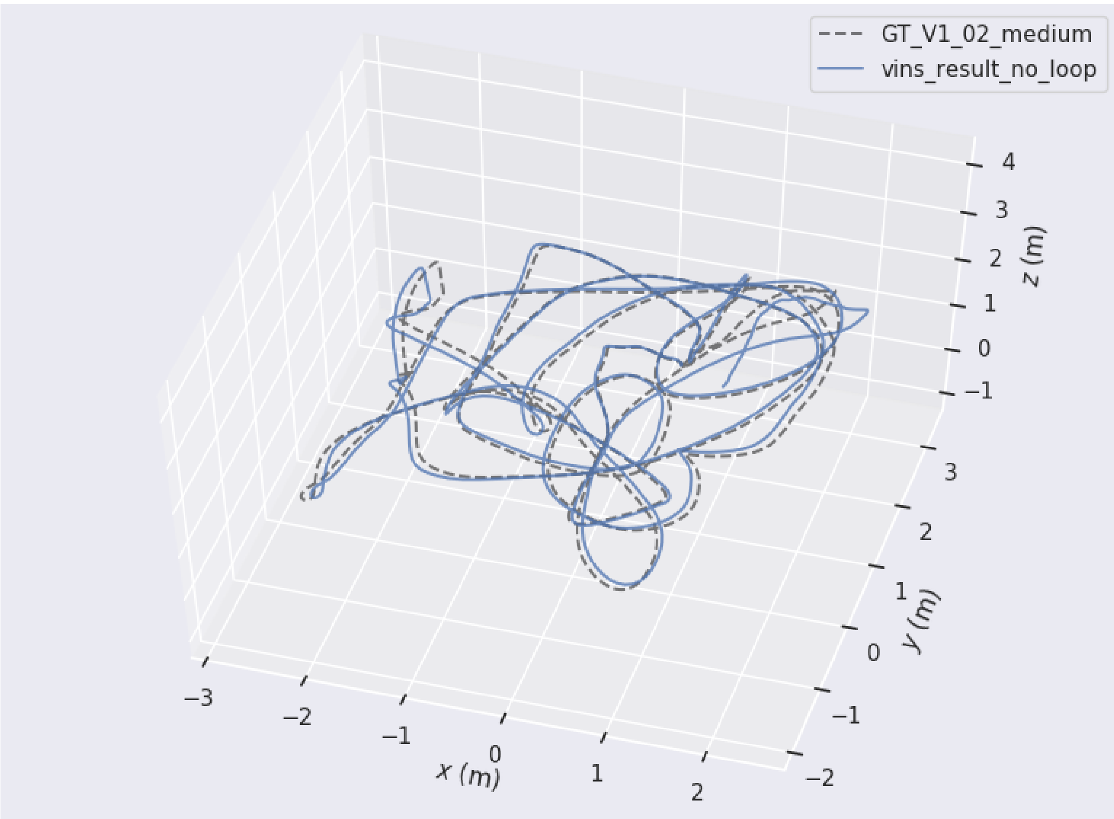
生成的文件的后缀改成tum，绘制轨迹

```
  evo_traj tum vins_result_no_loop.tum --ref=GT_V1_02_medium.tum -va --plot --
plot_mode xyz
  evo_traj tum vins_result_loop.tum --ref=GT_V1_02_medium.tum -va --plot --
plot_mode xyz
```

带回环的轨迹



不带回环的轨迹

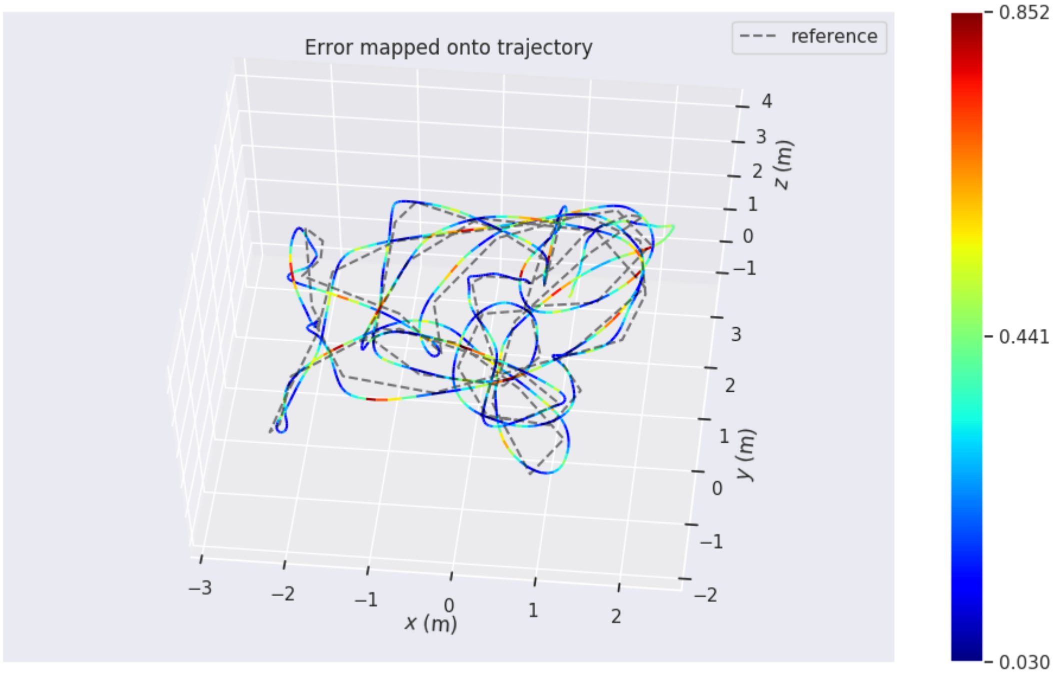


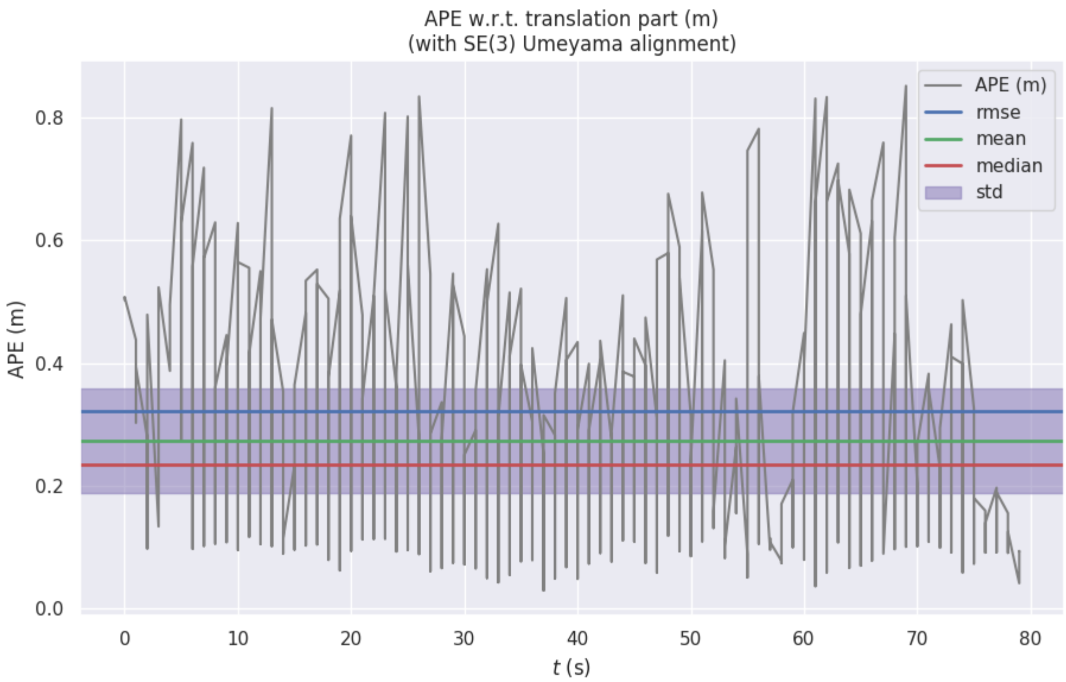
上面的例子只是轨迹的绘制，下面进行定量的评估，evo工具提供了3种误差评估方式:

```
  evo_ape -absoulte pose error 绝对位姿误差
  evo_rpe -relative pose error 相对位姿误差
  evo_rpe -for-each -sub-sequence-wise averaged pose error

  4种工具：

  evo_traj - tool for analyzing, plotting or exporting one or more trajectories
  evo_res - tool for comparing one or multiple result files from evo_ape or
evo_rpe
  evo_fig - (experimental) tool for re-opening serialized plots (saved with –
serialize_plot)
  evo_config - tool for global settings and config file manipulation
```

使用evo_ape确定轨迹误差，并保存结果

```
  evo_ape tum GT_V1_02_medium.tum vins_result_no_loop.tum -va --plot --plot_mode
xyz
```






```
yhp@yhp-Lenovo-IdeaPad-Y410P: /media/yhp/Dataset/my_projects/VINS-Course
[[ 0.7883685   0.61511038 -0.01069282]
 [-0.61519003  0.78834659 -0.00713293]
 [ 0.0040421   0.01220149  0.99991739]]
Translation of alignment:
[1.01391081 1.65981127 0.88099677]
Scale correction: 1.0
--------------------------------------------------------------------------------
Compared 792 absolute pose pairs.
Calculating APE for translation part pose relation...
--------------------------------------------------------------------------------
APE w.r.t. translation part (m)
(with SE(3) Umeyama alignment)

       max      0.851667
      mean      0.274246
    median      0.235713
       min      0.029872
      rmse      0.323078
       sse      82.668503
       std      0.170788

--------------------------------------------------------------------------------
Plotting results...
```

参考链接：

https://blog.csdn.net/weixin_38141453/article/details/105954396

https://blog.csdn.net/dcq1609931832/article/details/102465071

https://blog.csdn.net/qq_34213260/article/details/112548249