/***/
文档内容　SVO_RGBD_SLAM　代码配置及运行说明文档
作者：朱保华
修改日期：2016.9
/***/
一　配置部分

SVO 环境配置及运行方法

首先确保您的电脑已配好 ROS 环境，并且已安装好 opencv，建议安装 opencv2.4.8，
添加依赖项：

1、新建一个工作空间，即新建一个文件夹，起名为 workspace（起任何名字都可以，在本文档中起名为 workspace）。然后我们把运行 RGBD_SVO 的依赖安装在此 workspace 里。

2、安装 Sophus 库（若提示没有 cmake，先通过 apt-get cmake 指令安装 cmake）

```
cd workspace
git clone https://github.com/strasdat/Sophus.git
cd Sophus
git checkout a621ff
mkdir build
cd build
cmake ..
make
```

3、安装 Fast 库

```
cd workspace
git clone https://github.com/uzh-rpg/fast.git
cd fast
mkdir build
cd build
cmake ..
make
```

4、安装 g2o

首先安装一些 g2o 的依赖项

```
apt-get libeigen3-dev libsuitesparse-dev libqt4-dev qt4-qmake
libqglviewer-qt4-dev
```

这里可能会报错，提示我们 libqt4-dev　qt4-qmake

libqglviewer-qt4-dev 无法正确安装。

解决方法：需要上网下载安装 QT5

依赖项安装成功后，我们开始下载编译 g2o

```
cd workspace
git clone https://github.com/RainerKuemmerle/g2o.git
cd g2o
mkdir build
cd build
cmake ..
make

sudo make install
```

5、ROS 依赖

```
sudo apt-get install ros-indigo-cmake-modules
```

若显示已安装则跳过该步骤即可

6 编译工程

这时需要我们新建一个 catkin 工作空间，即一个文件夹，建议与 workspace 放在同一级目录下，起名为 catkin_ws，并在该目录下创建名为 src 的文件夹。

```
cd catkin_ws
```

把从 git 上克隆的 src 文件夹添加到工程里

```
catkin_make
```

编译该工程

二 运行部分

运行 svo

首先要注意：在接下来的步骤中，每打开一个新的终端，都要先执行指令

```
source PATH/catkin_ws/devel/setup.sh
```

其中 PATH 表示 catkin_ws 所在目录的绝对路径

或者您不想这么麻烦，每次都要添加这一句的话，也可以直接向~/.bashrc 中添加这句代码，方法为：

```
echo “source PATH/catkin_ws/devel/setup.sh” >> ~/.bashrc
source ~/.basrc
```

这样一来，就不需要每次都输入 source PATH/catkin_ws/devel/setup.sh 了。

我为大家提供的例程是运行 927 实验室 6cm 相机拍摄的图集和 EvRoc 图集。

Launch 文件已经写好。

编译配置好运行环境后，如果只是运行 EvRoc 图集或用 6cmMoveSense 相机拍的图集。只需要运行相对应的 launch 文件。（已经写好，但在不同电脑上运行时，注意修改图集路径）

注意：图集的存放是这样的：-------图集路径

---01（左图文件夹）

---disp（视差图文件夹）

roslaunch svo_ros EvRoc.launch

roslaunch svo_ros camera_6cm.launch

如果需要跑其他图集，需要修改 launch 文件。

下面简单说明一下 launch 文件需要修改的各个参数的含义：

1

<1> dataset_directory :数据集（包括左图与视差图）的路径

左图文件夹名字为 01 视差图文件件名字为 disp

<2> Depth_source :视差图的来源：我们之前将视差信息存成图像。因为存的方式不同而分为两类： Movesense 和我们自己用 SGBM 跑完存的视差图。如果跑 MoveSense 相机的图集，该参数选择“MoveSense”，如果跑我们自己用 SGBM 跑的图集，该参数填写“SGBM”。注意字母的大小写。

如果您想跑一下自己的数据集，可以在 benchmark_node.cpp 修改读图代码。

2 相机参数文件 .yaml:其存放路径在 svo_ros/param 文件夹下。

其内除了含有内参之外，还新填了参数

<1>Bf：

跑不同图集注意是否需要修改相机参数。

3 vo_accurate.yaml:

程序输入参数文件：在测试一些功能时，运行程序无需重新编译源码，只需修改下列变量。现将参数分为四部分：

<1> max_n_kfs:代码运行中允许创建的关键帧的最大数量 int

max_fts:匹配数量最大值 int

newKF_ratio：当前帧匹配点的数量小于 max_fts*newKF_ratio，则创建关键帧。float

depth_min_th:深度小于此值没有作为特征点的资格 float

depth_max_th:深度大于此值没有作为特征点的资格 float

process_pic_start: 你希望从第几帧开始跑图集。这个在调试的时候会有作用。int

process_pic_end: 你希望跑到第几帧结束。int

chang_pyr_ratio_:是否要改变金字塔比例 bool 类型

pyr_raio_：降采样比例 float 类型

<2>下面为网格法梯度点提取参数：

My_grad_size：设置网格尺寸 int

Want_piont_num：设置在每层金字塔上期待提取梯度点的最大数量 int

Constant_thred：弱纹理控制阈值，为 cell 里阈值的组成部分，最小为 0，越小代表尽可能的让弱纹理网格也提取到点。int

gridsize_down_ratio=2:每相邻金字塔网格尺寸比例。默认为 2。为 2 时意味着每层金字塔提取的梯度点数相近，越小（比如为 1.5）越意味着尺度大（原尺度最大）的层提取的点数多。float

<3> 程序调试显示参数：

show_cput_Resident_mssage: bool 是否要显示对齐残差信息

display_Residuals： 是否显示图像对齐迭代时的残差图

display_Gradpoint : bool 对齐采用梯度点时，是否显示提取的梯度点

<4> 运行模式控制参数 ：

align_only_： bool

Align_every_level_by_ItsOwns_Point_：bool

FastCorner0_2_Grad4_: bool

Featrues_grad bool 注意：此变量在 src/rpg/svo/cmakelist 中：

下附表格说明参数含义，以上四个参数的组合会产生多种运行模式：

运行模式控制参数说明：

完整 slam 模式	Featrues_grad	align_only	Align_every_level_by_ItsOwns_Point	FastCorner0_2_Grad4_
对齐：角点（对齐非分层） 匹配：角点	FALSE	false	false	false
对齐：level4 梯度点，外层角点（对齐非分层） 匹配：角点	FALSE	false	false	true

只 vo 模式	Featrues_grad	align_only	Align_every_level_by_ItsOwns_Point	FastCorner0_2_Grad4_
角点对齐	FALSE	true	false	false
梯度点对齐（非分层）	TRUE	true	false	false
梯度点对齐（分层对齐）	TRUE	true	true	false