深蓝学院
shenlanxueyuan.com

第一章作业分享

主讲人 肖映彩

纲要

深蓝学院
shenlanxueyuan.com

- 第一部分：Linux基础
- 第二部分：CMake练习

第三部分：编译运行ORB-SLAM2

第四部分：体会

- 如何定义新的环境变量

3种设置方式，以设置环境变量MY_PATH为例：

① 在终端执行命令：export MY_PATH=/my/path:$MY_PATH，只在当前终端有效。

② 在~/.bashrc文件末尾添加一行内容：MY_PATH=/my/path:$MY_PATH，然后在终端执行命令source ~/.bashrc可以使环境变量在当前终端生效，或者重启计算机使环境变量对当前用户生效。

- 如何定义新的环境变量

③ 在/etc/profile文件末尾添加一行内容：MY_PATH=/my/path:$MY_PATH，然后在终端执行命令source /etc/profile可以使环境变量在当前终端生效，或者重启计算机使环境变量对所有用户生效。

纲要

深蓝学院
shenlanxueyuan.com

- 第一部分：Linux基础
- **第二部分：CMake练习**
- 第三部分：编译运行ORB-SLAM2
- 第四部分：体会

CMake练习

深蓝学院
shenlanxueyuan.com

- 题目要求

① 将include/hello.h和src/hello.c编译成libhello.so库。

② 在useHello.c文件中调用libhello.so库中的sayHello函数，并将useHello.c编译成名为sayhello的可执行文件。

③ 默认使用Release模式编译这个工程。

④ 如果用户使用sudo make install，那么将hello.h放至/usr/local/include/下，将libhello.so放至/usr/local/lib/下。

CMake练习

深蓝学院
shenlanxueyuan.com

- 做题思路

① libhello.so是一个共享库，所以编译时应指定为SHARED模式。

② 需要通过target_link_libraries设置需要链接的库。

③ 默认使用Release模式，我理解的“默认”是用户不设置编译模式的情况，如果用户设置了就应该使用用户设置的编译模式。

④ 头文件和库文件需要采用不同的install方式。

CMake练习

深蓝学院
shenlanxueyuan.com

- **CMakeLists.txt文件内容**

```
cmake_minimum_required(VERSION 3.16)
project(hello)
IF(NOT CMAKE_BUILD_TYPE)
  set(CMAKE_BUILD_TYPE Release)
ENDIF()
```

CMake练习

深蓝学院
shenlanxueyuan.com

- **CMakeLists.txt文件内容**

```
set(SRC_PATH src/)
set(INCLUDE_PATH include/)

include_directories(${INCLUDE_PATH})
add_library(hello SHARED ${SRC_PATH}/hello.cpp)

add_executable(sayhello ${SRC_PATH}/useHello.cpp)
target_link_libraries(sayhello hello)
```

CMake练习

深蓝学院
shenlanxueyuan.com

- **CMakeLists.txt文件内容**

```
set(CMAKE_INSTALL_INCLUDEDIR /usr/local)


install(FILES ${INCLUDE_PATH}/hello.h DESTINATION
${CMAKE_INSTALL_INCLUDEDIR}/include)

install(TARGETS hello LIBRARY DESTINATION
${CMAKE_INSTALL_INCLUDEDIR}/lib)
```

纲要

深蓝学院
shenlanxueyuan.com

- 第一部分：Linux基础
- 第二部分：CMake练习
- **第三部分：编译运行ORB-SLAM2**
- 第四部分：体会

编译ORB-SLAM2

深蓝学院 shenlanxueyuan.com

- 环境

- Ubuntu20.04 + OpenCV4.2(apt-get install方式安装)

- 前提

已按照README安装好所有的依赖库

- 难点

ORB-SLAM2代码中使用的OpenCV版本与OpenCV4.2不兼容，导致直接编译会出现一堆错误

编译ORB-SLAM2

深蓝学院 shenlanxueyuan.com

- 解决方法

- 用OpenCV4.2中的数据结构和函数替代ORB-SLAM2中报错的数据结构或函数，比如：
- CvMat ---> cv::Mat
- cvSolve ---> cv::solve
- cvSVD ---> cv::SVD::compute
- .....
- 修改CMakeLists.txt文件内容:

```
# find_package(OpenCV 3.0 QUIET)
# if(NOT OpenCV_FOUND)
#    find_package(OpenCV 2.4.3 QUIET)
#    if(NOT OpenCV_FOUND)
#       message(FATAL_ERROR "OpenCV > 2.4.3 not found.")
#    endif()
# endif()

find_package(OpenCV 4.0 QUIET)
if(NOT OpenCV_FOUND)
   message(FATAL_ERROR "OpenCV 4 not found.")
endif()
```

运行ORB-SLAM2

深蓝学院 shenlanxueyuan.com

- **将myslam.cpp添加到工程中**

① 可以参照ORB-SLAM2工程中的几个例程。在Examples中创建myslam文件夹，然后将myslam.cpp文件拷贝到该文件夹中。

② 在CMakeLists.txt文件末尾添加如下内容：

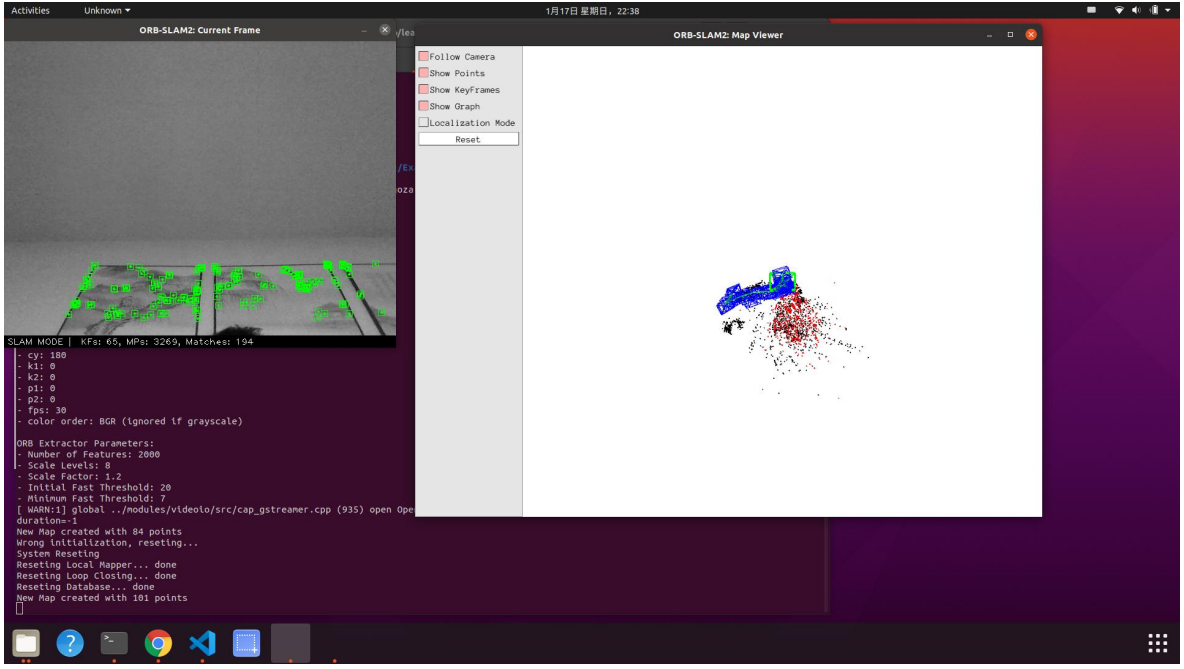
```
set(CMAKE_RUNTIME_OUTPUT_DIRECTORY
${PROJECT_SOURCE_DIR}/Examples/myslam)

add_executable(myslam Examples/myslam/myslam.cpp)

target_link_libraries(myslam ${PROJECT_NAME})
```

运行ORB-SLAM2

编译成功后，在Examples/myslam/目录下会生成可执行文件myslam，使用摄像头的运行效果如下：



纲要

深蓝学院
shenlanxueyuan.com

- 第一部分：Linux基础
- 第二部分：CMake练习

第三部分：编译运行ORB-SLAM2

第四部分：体会

体会

1. 碰到问题不要慌，冷静一下，尝试解决。
2. 考虑问题要全面。
3. 运行myslam的时候一定要确保在程序中设置的文件路径是正确的。
4. 编写的代码要对异常情况做处理，比如

```
cv::VideoCapture cap(0);
if(!cap.isOpened()){ // 判断cap是否正常
    std::cout << "can not open camera!" << std::endl;
    return -1;
}

cap >> frame;              // 读取相机数据
if (frame.empty()){     // 判断图像是否正常
    std::cout << "frame is empty!" << std::endl;
    break;
}
```

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening