HRG_MP3 使用手册

一、项目逻辑和模块原理

详见《ZJL_DynamicGesturePresentationForHrg.pptx》文件。

二、支持功能

2.1 当前版本支持 9 种静态手势，分别是 0-8 数字。

说明：

手势 3：视为“OK”命令，Mp3 播放。

手势 5：视为“Stop”命令，Mp3 停止。

手势 1：视为动态手势的“开始”触发动作。

2.2 当前版本支持 4 种动态手势，分别是“增加音量”“减小音量”“上一首”“下一首”。

说明：

“增加音量”是手势 1 全程的逆时针旋转一周。

“减小音量”是手势 1 全程的顺时针旋转一周。

“上一首”是手势 1 向左移动。

“下一首”是手势 1 向右移动。

三、操作注意事项

3.1 工作范围：

目前基于 HRG_ToF 相机实现的 demo 版本，其设定的最佳工作距离在 0.3-1.1m 之内，其中中值 0.8m 左右是最理想的工作距离。

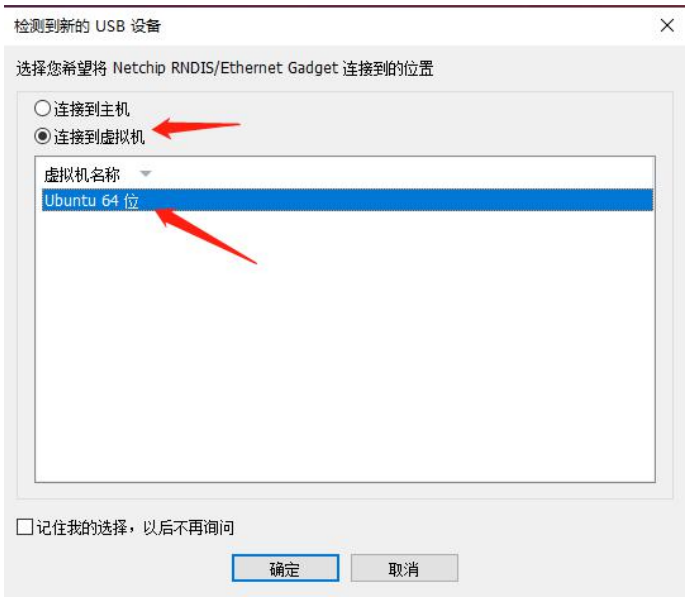
3.2 注意事项：

- 相机与手势之间最好不要有干扰物。
- 手势不要出相机视场。
- 动态手势最好**一直保持 1** 的手势进行动作。

四、操作流程

4.1 连接相机

注意：先连接电源，在将 usb 线插入电脑。出现如下界面，如下选择。

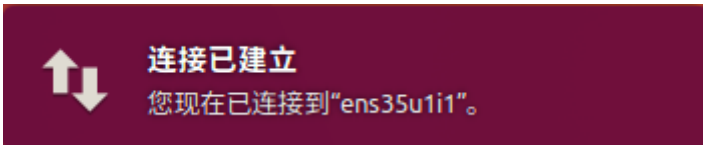


4.2 配置 ip

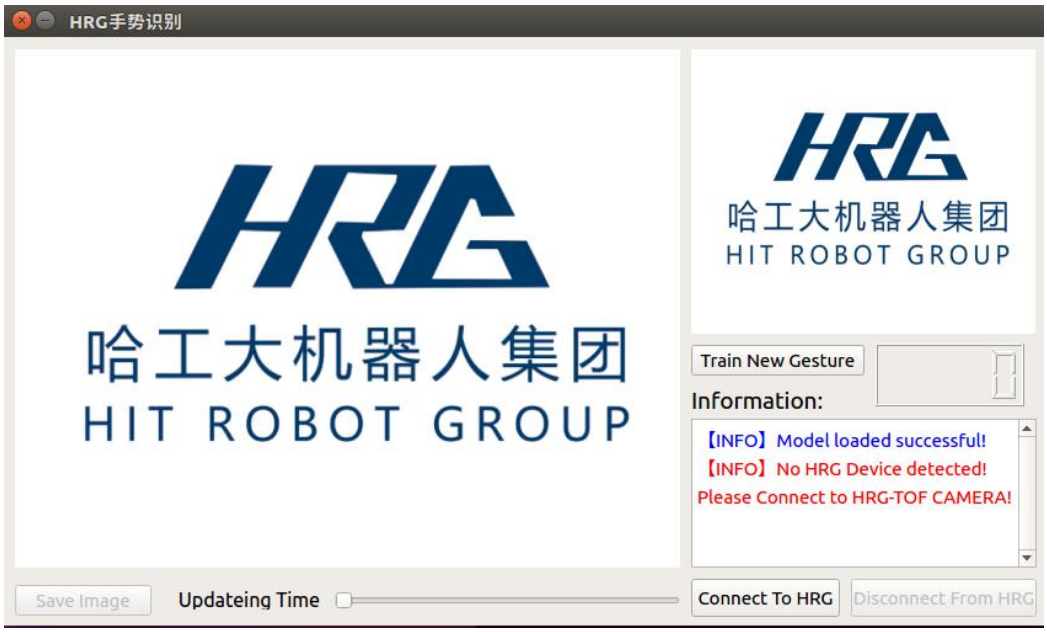
首先，将 ubuntu 下所有的网络断开连接。

然后按住 Ctrl+Alt+T，打开终端，输入 ifconfig，查看相机装置名称。如下，我这边的装置名称为：ens35u1i1

之后，在终端输入：sudo ifconfig ens35u1i1 192.168.1.123 修改 ip，出现如下所示即为连接成功。



4.3 进入 HRG_Mp3 文件，打开 bin 文件夹，选择在终端打开。输入./2HRG_MP3 即可打开软件使用。



五、开发流程

5.1 静态手势图片获取

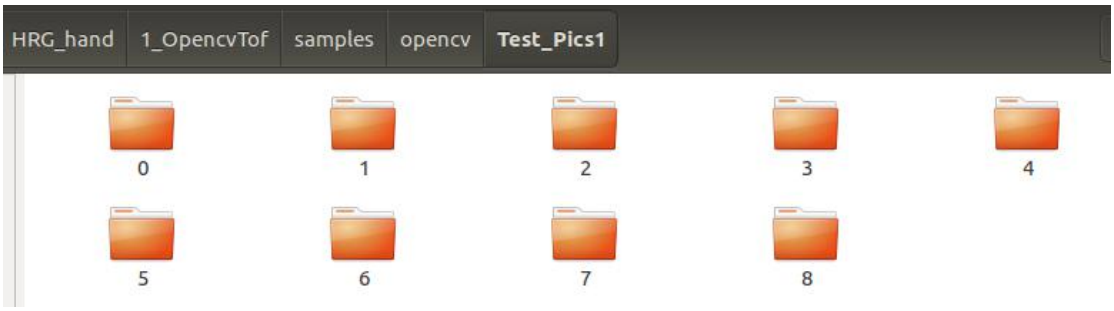
打开 1_CvSaveImg 文件夹下，进入 samples，进入 openCV_savePic，进入 build，并 cmake .. 编译，编译通过之后，进入 bin 文件夹，执行 openCV_savePic，即可进行手势图片的采集。

注意：这是用鼠标右键触发保存的；其次，保存路径需要自己设置。

5.2 静态手势图片数据集制作

进入 2_MakeTrainDataText 文件夹中。

首先，做准备工作，将上一步收集的图片数据，按照 label 分类存入对应的文件夹中，如下图所示：



例：手势 0 的所有图片就放置在“0”文件夹下，以此类推。

准备工作之后，打开 make_label_txt.py，修改图片路径问上述准备工作的文件夹的路径。

```
outer_path = '/home/zjl/HRG_hand/1_OpencvTof/samples/opencv/Test_Pics1/' # 这里是你的图片的目录
```

利用 python 执行 make_label_txt.py 文件，即可获得对应的 Train_List.txt 文件。

将得到的 txt 文件移动到自己第一步收集手势图片的文件夹下。

5.3 ResNet18 网络训练

进入 3_ResNet18 文件夹下，双击打开 work.py，将下面的图片路径修改成**第一步**自己收集的所有手势的那个文件夹的路径，**注意：不是准备工作的文件夹路径！！**

```python
# 1.数据集准备
# 根据自己定义的那个勒MyDataset来创建数据集！注意是数据集！而不是loader迭代器
#root = "F:/Rgb_Cnn/"
root1 = '/home/zjl/HRG_hand/1_OpencvTof/samples/opencv/Test_Pics/'
root2 = '/home/zjl/HRG_hand/1_OpencvTof/samples/opencv/Test_Pics/'
trans = transforms.Compose(
        [
            transforms.ToTensor(),
            transforms.Normalize(mean=(0.5, 0.5, 0.5), std=(0.5, 0.5, 0.5))
        ])
train_data = MyDataset(root=root1, datatxt='Train_list.txt', transform=trans)
test_data = MyDataset(root=root2, datatxt='Train_list.txt', transform=trans)
```

其次，如果自己的手势种类小于 10，则可以直接训练了；如果大于 10，则需要打开 cnn.py 文件，修改里面的 num_classes 为自己的种类数。

```python
ResNet(nn.Module):
ef __init__(self, ResidualBlock, num_classes=9)
   super(ResNet, self).__init__()
   self.inchannel = 64
```

接下来，执行 work.py 之后即可训练网络，训练完成之后再本目录下会得到一个 HRG_C++.pt 文件。

5.4 导入模块

在后面 c++程序中，下面的函数就是利用 libtorch 库导入刚刚训练好的 pt 模型文件到程序中。**注意：路径！**

```cpp
/*******************************************************
 * function:InitModel()
 * author:Zuo jiale
 * Date:2021-1-6
 * description:加载训练好的model，用来后续预测手势结果
 * Params:输入None
 *        输出None
 * ******************************************************/
void Widget::InitModel()
{
    /*
     * function:导入训练好的神经网络模型到程序中
     */
    try {
          module = torch::jit::load("../resource/HRG_C++.pt");
       }
       catch (const c10::Error& e) {
            ui->textInfo->append("<font color=\"#FF0000\">【INFO】Failed to load the Model!</font>");
           return;
       }
       ui->textInfo->append("<font color=\"#0000FF\">【INFO】Model loaded successful!</font>");
       return;
}
```

5.5 C++重要模块解释


bin
可执行文件
build
编译文件
include
Hrg相机所需要的头文件
lib
HRg需要的静态、动态链接库
resource
软件所需的资源文件：图片logo，手势库txt等
song
mp3需要的歌曲，可以增删
2_HRG_MP3.pro
2_HRG_MP3.pro.user
CMakeLists.txt
编译所需文件
connectthread.cpp
connectthread.h
连接HRG的多线程
dtwrecoge.cpp
dtwrecoge.h
DTW算法实现
handfinder.cpp
handfinder.h
手部提取追踪类的实现
main.cpp
mp3.cpp
mp3.h
mp3播放器的逻辑
mp3.ui
setparam.cpp
setparam.h
连接相机子窗口
setparam.ui
widget.cpp
widget.h
主软件主逻辑