哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

GPU 计算实验报告

附加实验： HandWrite Atlas 200DK 部署

学院：计算学部

姓名：钱泽凯

学号：1190202011

一、实验内容

https://kqybjxvfo3.feishu.cn/docs/doccn6sbQsQ3dpsEMRDhXiWaEJb

二、实验要求

1. 完成 HandWrite 项目部署与复现，撰写报告描述实现过程，录制演示视频
2. 使用 msame 工具推理，撰写报告描述实现过程，录制演示视频
3. 检测其他颜色的文字，撰写报告描述实现过程，录制演示视频

三、实验加分说明

1. 完成实验要求 1，可获得 3 分加分、1 本赠书；
2. 在完成实验要求 1 的基础上，完成要求 2 可获得 5 分加分、1 本赠书；
3. 在完成实验要求 1 的基础上，完成要求 3 可获得 4 分加分、1 本赠书；
4. 实验要求 1、2、3 均完成，可获得 5 分加分、2 本赠书。

四、实验过程

1. HandWrite 项目部署与复现

1.1 首先搭建环境：

分为软件环境和硬件环境，

a. 软件环境已经提供。为了简化实验，在虚拟机、存储卡上已经配置一份完整的开发和运行基础环境，从而跳过第一步环境搭建。

b. 硬件环境包括：笔记本电脑、Atlas200DK 开发板、树莓派摄像头、存储卡、USB 转 Type C 数据线。连接起来如图所示：

图 1 硬件连接实物图

1.2 下面进行部署工作：

a. 开发环境命令行中设置编译依赖的环境变量。

export DDK_PATH=$HOME/Ascend/ascend-toolkit/latest/arm64-linux

export NPU_HOST_LIB=$DDK_PATH/acllib/lib64/stub

b. **切换到 HandWrite 目录，创建目录用于存放编译文件。**

cd $HOME/samples/cplusplus/contrib/HandWrite

mkdir -p build/intermediates/host

c. **切换到 build/intermediates/host 目录，执行 cmake 生成编译文件。**

cd build/intermediates/host

make clean

cmake ../../../src -DCMAKE_CXX_COMPILER=aarch64-linux-gnu-g++ -DCMAKE_SKIP_RPATH=TRUE

d. **执行 make 命令，生成的可执行文件 main 在 HandWrite/out 目录下。**

make

e. **完成后将本地文件传输到开发板上：**

scp- r $HOME/samples/cplusplus/contrib/HandWrite HwHiAiUser@192.168.1.2:/home/HwHiAiUser

1.3 运行

a. **在虚拟机：**

```
cd $HOME/samples/cplusplus/contrib/HandWrite
bash scripts/run_presenter_server.sh
```

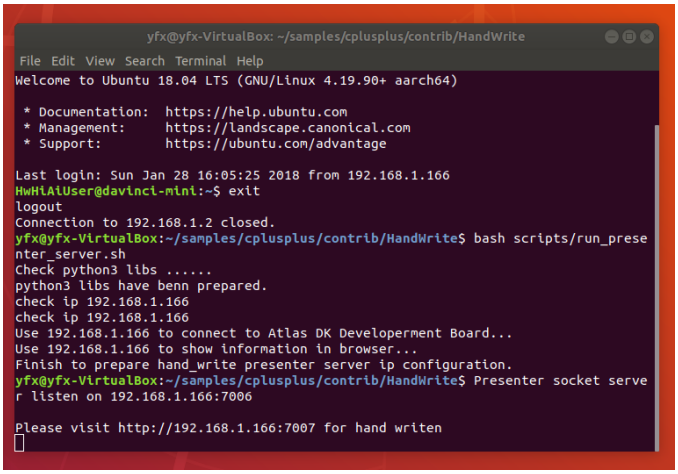


图 2 虚拟机运行

b. **在开发板：**

```
cd $HOME/HandWrite/out
./main
```

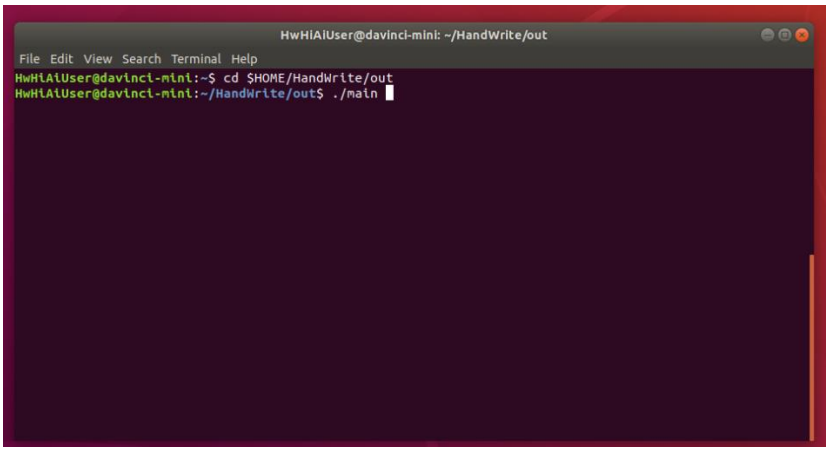


图 3 开发板运行

结果如图所示：

图 4 红色字体识别效果图

2. 使用 msame 工具推理（未完成）

3. 检测其他颜色的文字（这里改为深蓝色）

3.1 原理分析：

在图像预处理时，首先接收摄像头发送的格式为 YUV420SP 的图片，将该图片转换为 BGR 格式，该阶段输出为 BGR 格式的 Mat 图像矩阵 dst_temp，格式为 CV_8UC1，指图像文件格式使用的是无符号 8 位，最后的参数 1 表示通道数。

接下来，由于 BGR 颜色空间不连续，而本案例要设置阈值提取图像中的红色轮廓，所以将该 BGR 图像转换成 HSV 颜色空间。

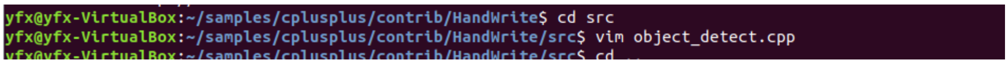
一般对颜色空间的图像进行有效处理都是在 HSV 空间进行的，然后对于基本色中对应的 HSV 分量需要给定一个严格的范围，下面是通过实验计算的模糊范围（准确的范围在网上都没有给出）。（H: 0 — 180 S: 0 — 255 V: 0 — 255）

	黑	灰	白	红		橙	黄	绿	青	蓝	紫
hmin	0	0	0	0	156	11	26	35	78	100	125
hmax	180	180	180	10	180	25	34	77	99	124	155
smin	0	0	0	43		43	43	43	43	43	43
smax	255	43	30	255		255	255	255	255	255	255
vmin	0	46	221	46		46	46	46	46	46	46
vmax	46	220	255	255		255	255	255	255	255	255

图 5 hsv 颜色对应图

3.2 代码修改：

在 src / object_detect.cpp 中是对图像预处理的部分，我们进入目录，用 vim 修改代码第 363 行，如下图所示：



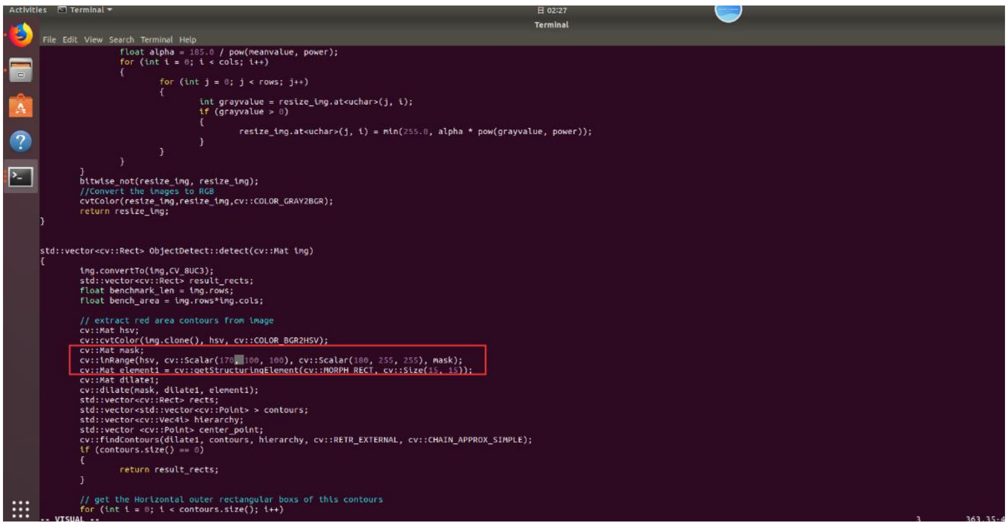


图 6 要修改的位置

将原来红色范围的 H S V 修改为深蓝色范围的 H S V,即 H:100 -124 S:43 - 255 V：46-255，如下图所示：

```
std::vector<cv::Rect> ObjectDetect::detect(cv::Mat img)
{
        img.convertTo(img,CV_8UC3);
        std::vector<cv::Rect> result_rects;
        float benchmark_len = img.rows;
        float bench_area = img.rows*img.cols;

        // extract red area contours from image
        cv::Mat hsv;
        cv::cvtColor(img.clone(), hsv, cv::COLOR_BGR2HSV);
        cv::Mat mask;
        cv::inRange(hsv, cv::Scalar(100, 43, 46), cv::Scalar(124, 255, 255), mask);
        cv::Mat element1 = cv::getStructuringElement(cv::MORPH_RECT, cv::Size(15, 15));
        cv::Mat dilate1;
        cv::dilate(mask, dilate1, element1);
        std::vector<cv::Rect> rects;
        std::vector<std::vector<cv::Point> > contours;
        std::vector<cv::Vec4i> hierarchy;
        std::vector <cv::Point> center_point;
        cv::findContours(dilate1, contours, hierarchy, cv::RETR_EXTERNAL, cv::CHAIN_APPROX_SIMPLE);
        if (contours.size() == 0)
        {
                return result_rects;
        }

        // get the Horizontal outer rectangular boxs of this contours
        for (int i = 0; i < contours.size(); i++)
        {
```

图 7 修改的内容

随后将代码重新 build，并传送到 Atlas200 中，如图所示：

```
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite$ cd src
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite/src$ vim object_detect.cpp
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite/src$ cd ..
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite$ cd build/intermediates/host
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite/build/intermediates/host$ make clean
```

```
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite/build/intermediates/host$ cmake ../../../src -DCMAKE_CXX_COMPILER=aarch64-linux-gnu-g++ -DCMAKE_SKIP_R
PATH=TRUE
-- env INC_PATH: /home/yfx/Ascend/ascend-toolkit/latest/arm64-linux
-- env LIB_PATH: /home/yfx/Ascend/ascend-toolkit/latest/arm64-linux/acllib/lib64/stub
-- Configuring done
-- Generating done
-- Build files have been written to: /home/yfx/samples/cplusplus/contrib/HandWrite/build/intermediates/host
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite/build/intermediates/host$ make
Scanning dependencies of target main
[ 11%] Building CXX object CMakeFiles/main.dir/utils.cpp.o
[ 22%] Building CXX object CMakeFiles/main.dir/camera.cpp.o
[ 33%] Building CXX object CMakeFiles/main.dir/model_process.cpp.o
[ 44%] Building CXX object CMakeFiles/main.dir/object_detect.cpp.o
/home/yfx/samples/cplusplus/contrib/HandWrite/src/object_detect.cpp: In member function 'void* ObjectDetect::GetInferenceOutputItem(uint32_t&, aclmdlDataset*,
uint32_t)':
/home/yfx/samples/cplusplus/contrib/HandWrite/src/object_detect.cpp:534:56: warning: 'uint32_t aclGetDataBufferSize(const aclDataBuffer*)' is deprecated: aclGe
tDataBufferSize is deprecated, use aclGetDataBufferSizeV2 instead [-Wdeprecated-declarations]
    size_t bufferSize = aclGetDataBufferSize(dataBuffer);
                                                        ^
In file included from /home/yfx/Ascend/ascend-toolkit/latest/arm64-linux/acllib/include/acl/acl_rt.h:16:0,
                 from /home/yfx/Ascend/ascend-toolkit/latest/arm64-linux/acllib/include/acl/acl.h:14,
                 from /home/yfx/samples/cplusplus/contrib/HandWrite/src/object_detect.cpp:21:
/home/yfx/Ascend/ascend-toolkit/latest/arm64-linux/acllib/include/acl/acl_base.h:276:30: note: declared here
 ACL_FUNC_VISIBILITY uint32_t aclGetDataBufferSize(const aclDataBuffer *dataBuffer);
                              ^~~~~~~~~~~~~~~~~~~~
[ 55%] Building CXX object CMakeFiles/main.dir/dvpp_process.cpp.o
[ 66%] Building CXX object CMakeFiles/main.dir/dvpp_resize.cpp.o
[ 77%] Building CXX object CMakeFiles/main.dir/dvpp_jpege.cpp.o
[ 88%] Building CXX object CMakeFiles/main.dir/main.cpp.o
[100%] Linking CXX executable /home/yfx/samples/cplusplus/contrib/HandWrite/out/main
/usr/lib/gcc-cross/aarch64-linux-gnu/7/../../../../aarch64-linux-gnu/bin/ld: skipping incompatible /usr/local/lib/libprotobuf.so when searching for -lprotobuf
/usr/lib/gcc-cross/aarch64-linux-gnu/7/../../../../aarch64-linux-gnu/bin/ld: skipping incompatible /usr/local/lib/libprotobuf.a when searching for -lprotobuf
/usr/lib/gcc-cross/aarch64-linux-gnu/7/../../../../aarch64-linux-gnu/bin/ld: warning: libascend_hal.so, needed by /home/yfx/Ascend/driver/libmedia_mini.so, not
 found (try using -rpath or -rpath-link)
[100%] Built target main
```

```
yfx@yfx-VirtualBox:~/samples/cplusplus/contrib/HandWrite$ scp -r $HOME/samples/cplusplus/contrib/HandWrite HwHiAiUser@192.168.1.2:/home/HwHiAiUser
HwHiAiUser@192.168.1.2's password:
.build_project                                100%  153     7.7KB/s   00:00
CMakeLists.txt                                100%  222    30.5KB/s   00:00
.project                                      100%  149    19.2KB/s   00:00
model_process.h                               100% 2248   557.9KB/s   00:00
dvpp_process.h                                100% 1226   286.2KB/s   00:00
dvpp_jpege.h                                  100% 1690   518.8KB/s   00:00
dvpp_resize.h                                 100% 2414   516.9KB/s   00:00
dvpp_jpegd.h                                  100% 2634   772.0KB/s   00:00
camera.h                                      100% 2629   498.5KB/s   00:00
utils.h                                       100% 4119     1.9MB/s   00:00
object_detect.h                               100% 2604     1.5MB/s   00:00
README                                        100%  691   165.5KB/s   00:00
__init__.py                                   100%    0     0.0KB/s   00:00
__init__.py                                   100%    0     0.0KB/s   00:00
logging.conf                                  100%  483   208.3KB/s   00:00
config.conf                                   100%  371    85.9KB/s   00:00
__init__.cpython-37.pyc                       100%  171    30.0KB/s   00:00
list.css                                      100% 2677   253.5KB/s   00:00
testvideo.css                                 100% 1345   452.9KB/s   00:00
dialog.css                                    100% 1300   388.2KB/s   00:00
base.css                                      100% 1045   193.5KB/s   00:00
```

图 8 编译并传输

最后在开发板上运行./main，就可以识别深蓝色的字体了！效果如图所示：

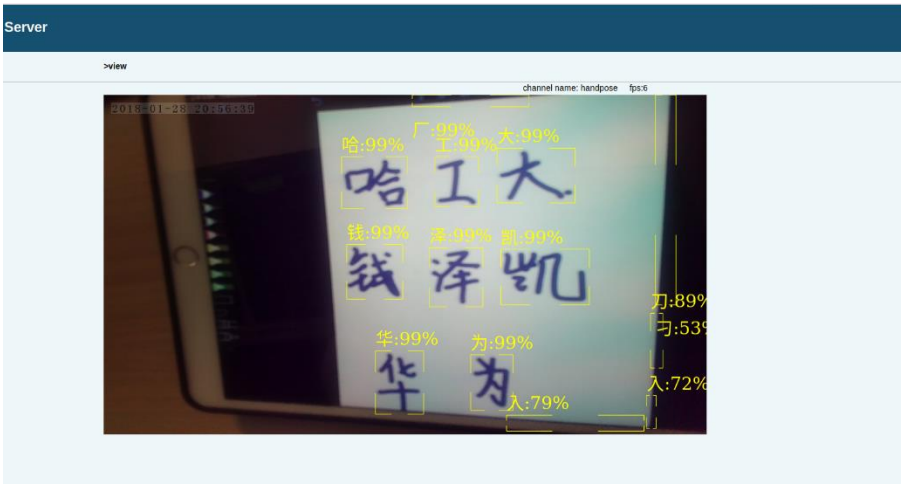


图 9 深蓝色字体识别运行结果

五、实验心得

1. 在搭配虚拟环境时候，要注意电脑 usb 过滤器的存在。

以 virtualbox 6.1.30 为例，在 win10 下顺利安装了它，然后也安装了它的扩展功能，可是在虚拟机中仍无法挂载 U 盘。最终发现问题出在 AMD USB 过滤器上。下面给出解决方法：

1.win+R 输入 regedit 打开注册表工具：

2.依次进入 HKEY_LOCAL_MACHINE\SYSTEM\Curr-entControlSet\Control，点击 Class 找到 {36-FC9E60-C465-11CF-8056-44455354-0000}，点击它，在右侧出现窗口会看到有个 UpperFilter，选中它，鼠标右键-删除。（如图所示，此处我已删除 所以截图中没有了）

3.重启主机电脑，打开 VirualBox 这样就可以在虚拟机中使用 U 盘设备了。

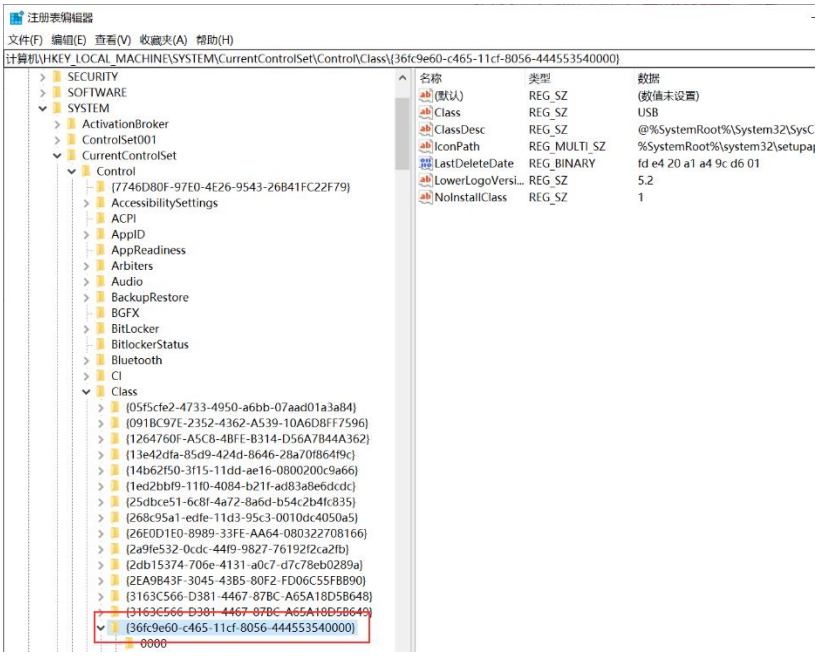

注册表编辑器
文件(F) 编辑(E) 查看(V) 收藏夹(A) 帮助(H)
计算机\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{36fc9e60-c465-11cf-8056-444553540000}
SECURITY
SOFTWARE
SYSTEM
ActivationBroker
ControlSet001
CurrentControlSet
Control
{7746D80F-97E0-4E26-9543-26B41FC22F79}
AccessibilitySettings
ACPI
AppID
AppReadiness
Arbiters
Audio
BackupRestore
BGFX
BitLocker
BitlockerStatus
Bluetooth
CI
Class
{05f5cfe2-4733-4950-a6bb-07aad01a3a84}
{091BC97E-2352-4362-A539-10A6D8FF7596}
{1264760F-A5C8-4BFE-B314-D56A7B44A362}
{13e42dfa-85d9-424d-8646-28a70f864f9c}
{14b62f50-3f15-11dd-ae16-0800200c9a66}
{1ed2bbf9-11f0-4084-b21f-ad83a8e6dcdc}
{25dbce51-6c8f-4a72-8a6d-b54c2b4fc835}
{268c95a1-edfe-11d3-95c3-0010dc4050a5}
{26E0D1E0-8989-33FE-AA64-080322708166}
{2a9fe532-0cdc-44f9-9827-76192f2ca2fb}
{2db15374-706e-4131-a0c7-d7c78eb0289a}
{2EA9B43F-3045-43B5-80F2-FD06C55FBB90}
{3163C566-D381-4467-87BC-A65A18D5B648}
{3163C566-D381-4467-87BC-A65A18D5B649}
{36fc9e60-c465-11cf-8056-444553540000}
0000
名称
类型
数据
(默认)
REG_SZ
(数值未设置)
Class
REG_SZ
USB
ClassDesc
REG_SZ
@%SystemRoot%\System32\SysC
IconPath
REG_MULTI_SZ
%SystemRoot%\system32\setupa
LastDeleteDate
REG_BINARY
fd e4 20 a1 a4 9c d6 01
LowerLogoVersi...
REG_SZ
5.2
NoInstallClass
REG_SZ
1

图 10 注册表修改路径