目录

小孔成像科技

DFace SDK 安装说明书(v1.6.8)

1.0 Linux amd64 系统安装

1.1 环境变量设置

把 DFace SDK 目录下的 lib/ 目录 和 redistribute/lib/ 目录 添加到 LD_LIBRARY_PATH 环境变量

export LD_LIBRARY_PATH = ...

1.2 工具包使用

DFace SDK 在 tools 提供了几个工具可以快速在目标机器展开人脸识别类测试。使用 --help 参数可以查看具体使用说明。

testDetectImage（人脸检测图片工具）

Usage: testDetectImage [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testDetectCamera（人脸检测摄像头工具）

Usage: testDetectCamera [model_path] [num_threads] [min_facesize] [video_width] [video_height]
[model_path]:模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[video_width]: 摄像头分辨率宽
[video_height]: 摄像头分辨率高

testDetectVideo（人脸检测视频工具）

Usage: testDetectVideo [model_path] [num_threads] [min_facesize] [video_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸

小孔成像科技

[video_path]: 视频路径

testRecognize（人脸识别工具）

Usage: testRecognize [model_path] [num_threads] [min_facesize] [accuracy_mode] [img1_path] [img2_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[img1_path]: 图片 1 路径
[img2_path]: 图片 2 路径

testMassiveCompare（大规模 1:N 人脸比对工具）

Usage: testMassiveCompare [model_path] [num_threads] [accuracy_mode] [size]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[size]: 1:N 规模,N 值

testPoseImage（人脸 68 关键点和 3D 姿态角估计工具）

Usage: testPoseImage [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testSexAge（人脸性别和年龄预测工具）

Usage: testSexAge [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testBlur（人脸清晰度判断工具）

Usage: testBlur [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

小孔成像科技

1.3 测试源码编译

DFace SDK 利用 CMake 相关文件快速构建测试项目，测试源码位于 example 目录。提供脚本编译和手动编译的方法，建议手动编译。

编译工具清单:

工具	描述
编译器	gcc/g++ 版本号 4.8 以上，支持 c++11 标准。
构建工具	Cmake, 版本 3.6 以上。
Ide	vim clion qtcreator codeblocks
标准库	libstdc++6 libc6
链接器	ld.so.3

小孔成像科技

手动编译

1. 创建编译目录

mkdir build & cd build

2. 执行 cmake

cmake ..

如果本地未安装 opencv, 可以用 dface_sdk 自带的 opencv 库，例

cmake .. -DOpenCV_DIR=${dface_sdk}/redistribute/share/OpenCV

3. 构建测试项目

make -j 4

构建完之后会在 build 目录生成几个个可执行文件

2.0 Linux ARM 系统安装

2.1 环境变量设置

把 DFace SDK 目录下的 lib 目录 和 redistribute/lib 目录 添加到 LD_LIBRARY_PATH 环境变量

export LD_LIBRARY_PATH = ...

注意：该环境变量设置同时对 ARM 终端设备和交叉编译的宿主机有效。

2.2 工具包使用

DFace SDK 在 tools 提供了 6 个工具可以快速在目标机器展开人脸识别类测试。使用 --help 参数可以查看具体使用说明。

testDetectImage（人脸检测图片工具）

Usage: testDetectImage [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testDetectCamera（人脸检测摄像头工具）

Usage: testDetectCamera [model_path] [num_threads] [min_facesize] [video_width] [video_height]
[model_path]:模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[video_width]: 摄像头分辨率宽
[video_height]: 摄像头分辨率高

testDetectVideo（人脸检测视频工具）

Usage: testDetectVideo [model_path] [num_threads] [min_facesize] [video_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[video_path]: 视频路径

小孔成像科技

testRecognize（人脸识别工具）

Usage: testRecognize [model_path] [num_threads] [min_facesize] [accuracy_mode] [img1_path] [img2_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[img1_path]: 图片 1 路径
[img2_path]: 图片 2 路径

testMassiveCompare（大规模 1:N 比对工具）

Usage: testMassiveCompare [model_path] [num_threads] [accuracy_mode] [size]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[size]: 1:N 规模,N 值

testPoseImage（人脸 68 关键点和 3D 姿态角估计工具）

Usage: testPoseImage [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testSexAge（人脸性别和年龄预测工具）

Usage: testSexAge [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testBlur（人脸清晰度判断工具）

Usage: testBlur [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

小孔成像科技

2.3 测试源码编译

工具	描述
编译器	arm-linux-gnueabihf-g++/aarch64-linux-g++ 版本号 4.8 以上，支持 c++11 标准。
构建工具	cmake, 版本 3.6 以上。
Ide	vim clion qtcreator codeblocks
标准库	libstdc++6 libc6
链接器	ld.so.3

小孔成像科技

DFace SDK 利用 CMake 相关文件快速构建测试项目，测试源码位于 example 目录。由于是 ARM 架构，因此一般在宿主机上执行交叉编译，DFace SDK 提供了 arm 交叉编译的 toolchain。

交叉编译

1. 安装交叉编译环境

sudo apt-get install gcc-arm-linux-gnueabihf

sudo apt-get install g++-arm-linux-gnueabihf

如果是 arm 32 位架构(armv7),还需要安装 32 位的库

sudo apt-get install ia32-libs

sudo apt-get install lib32ncurses5 lib32z1

2. 创建编译目录

mkdir build & cd build

3. 执行 cmake

Cmake .. -DCMAKE_TOOLCHAIN_FILE={DFace SDK 目录}/arm-gnueabi.toolchain.c

make -DOpenCV_DIR={DFace SDK 目录}/redistribute/share/OpenCV

注意: 手动指定了 opencv 库的目录。

4. 构建测试项目

make -j 4

ARM 终端设备运行

参考 2.1 设置环境变量，把交叉编译完的可执行文件拷贝到 ARM 终端设备运行。

注意: Linux arm 由于加密狗的原因需要 root 权限执行。设置加密狗参考 2.4。

2.4 Sentinel 加密狗设置

Linux ARM 的 sentinel 的加密狗需要手动被引导，请将 key/Sentinel/driver/linux_arm 目录下的所有文件拷贝至 /etc/udev/rules.d 目录。root 权限运行。

小孔成像科技

3.0 Windows(64 位)系统安装

3.1 环境变量设置

把 DFace SDK 目录下的 lib 目录 和 redistribute/bins 目录 添加到 PATH 环境变量

3.2 工具包使用

DFace SDK 在 tools 提供了 6 个工具可以快速在目标机器展开人脸识别类测试。使用 --help 参数可以查看具体使用说明。

testDetectImage.exe（人脸检测图片工具）

Usage: testDetectImage.exe [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testDetectCamera.exe（人脸检测摄像头工具）

Usage: testDetectCamera.exe [model_path] [num_threads] [min_facesize] [video_width] [video_height]
[model_path]:模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[video_width]: 摄像头分辨率宽
[video_height]: 摄像头分辨率高

testDetectVideo.exe（人脸检测视频工具）

Usage: testDetectVideo.exe [model_path] [num_threads] [min_facesize] [video_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[video_path]: 视频路径

testRecognize（人脸识别工具）

Usage: testRecognize.exe [model_path] [num_threads] [min_facesize] [accuracy_mode] [img1_path] [img2_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数

小孔成像科技

[min_facesize]: 需要被检测的最小人脸尺寸
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[img1_path]: 图片 1 路径
[img2_path]: 图片 2 路径

testMassiveCompare.exe（大规模 1:N 比对工具）

Usage: testMassiveCompare.exe [model_path] [num_threads] [accuracy_mode] [size]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[accuracy_mode]: 精度模式 0:普通精度 1:高精度(速度稍慢) 2:实时模式(精度稍低)
[size]: 1:N 规模,N 值

testPoseImage（人脸 68 关键点和 3D 姿态角估计工具）

Usage: testPoseImage.exe [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testSexAge（人脸性别和年龄预测工具）

Usage: testSexAge.exe [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

testBlur（人脸清晰度判断工具）

Usage: testBlur.exe [model_path] [num_threads] [min_facesize] [img_path]
[model_path]: 模型目录
[num_threads]: 并行运算线程数
[min_facesize]: 需要被检测的最小人脸尺寸
[img_path]: 图片路径

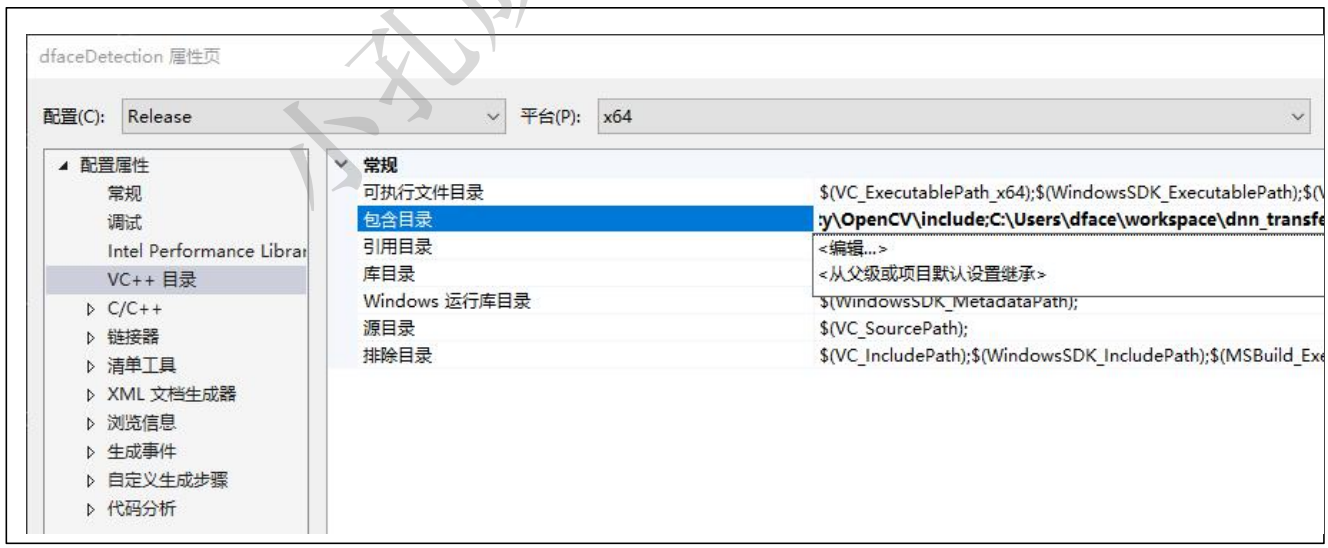
小孔成像科技

3.3 测试源码编译

工具	描述
编译器	Cl(V140) 版本号 V140 以上，需要支持 c++11 标准。
构建工具	Visual studio 2015 以上版本
Ide	Visual studio 2015 以上版本
标准库	
链接器	

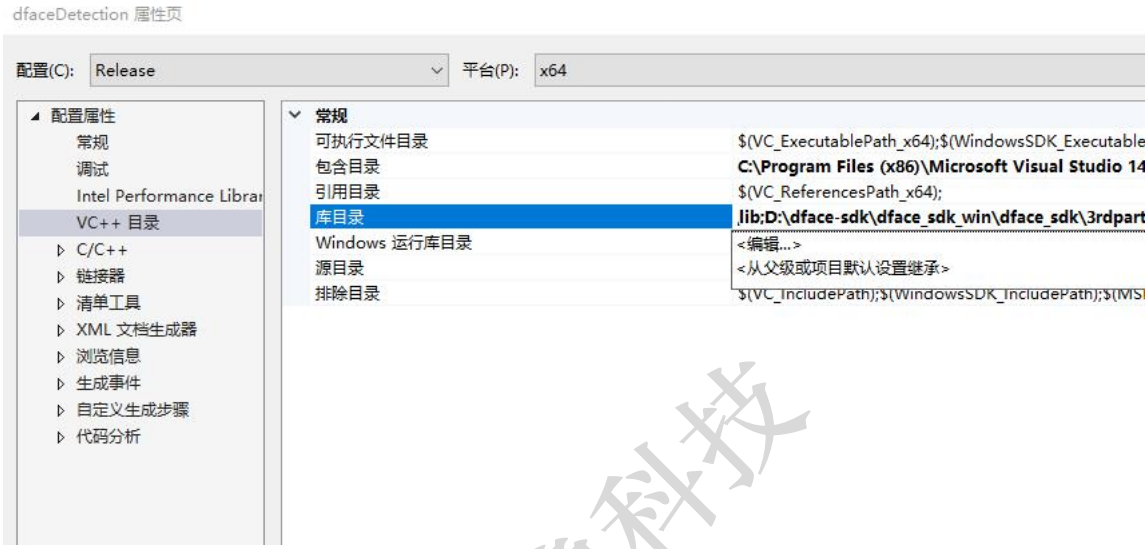
1. 打开 visual studio 2015 新建项目，添加 example 目录下的某个测试文件.

2. 打开 visual studio 项目属性，VC++目录->包含目录->编辑　添加 DFace SDK 目录下的 include 目录和 redistribute\OpenCV\include 目录

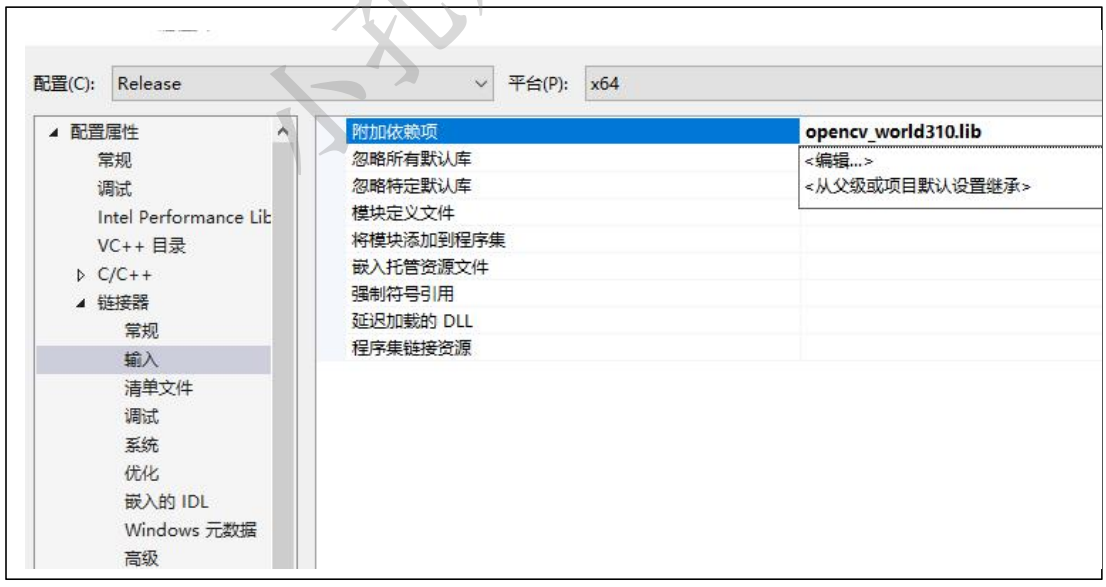


小孔成像科技

3. 打开 visual studio 项目属性，VC++目录->库目录->编辑 把 Dface SDK 目录下的 redistribute\OpenCV\x64\vc14\lib 目录添加到库目录



4. 打开 visual studio 项目属性，连接器->输入->附加依赖项->编辑 添加 opencv_world310.lib



小孔成像科技

4.0 Android 系统安装

4.1 环境变量设置

DFace SDK 安卓版本是静态库生成的，无需设置复杂的环境变量。

4.2 工具包使用

请安装我们的 APK，查看演示。

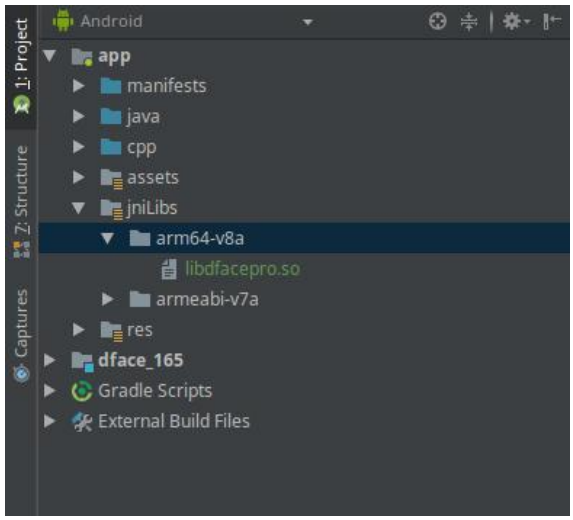
4.3 编译测试

工具	描述
编译器	Clang，最好选择 3.3 以上的版本，支持 c++11 标准。
构建工具	cmake 3.6 以上版本
Ide	Android studio/visual studio
标准库	libc++(clang 的标准库)
链接器	LLVM，最好选择 3.3 以上的版本，支持 c++11 标准。
NDK	我们的 NDK 选用 r16b 的版本

小北成像科技

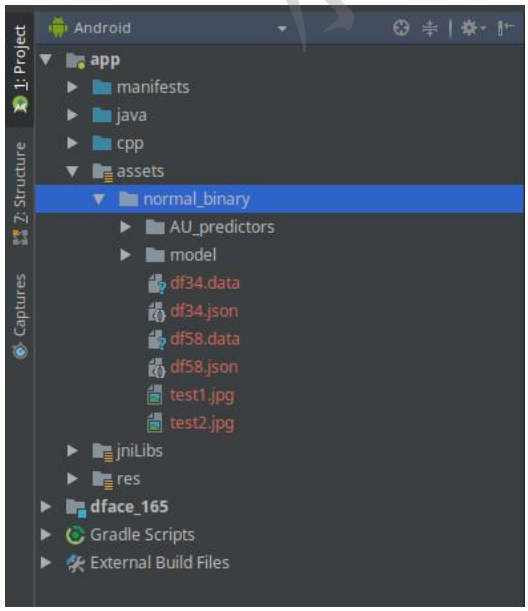
1. 我们在 example/java 目录下提供了几个例子文件。

2. 把 dface_sdk 目录下的 lib/libdfacepro.so 库文件拷贝到 Android studio 项目的 jniLibs 目录下，例如 64 位 sdk 放在 jniLibs/arm64-v8a，32 位 sdk 放在 jniLibs/armeabi-v7a 等。



3. 导入 Jar 包，把 dface_sdk 目录下的 android/jar/dface.jar 导入 android studio。也可以将 dface_sdk 目录下的 android/src/main/java 源码拷贝到自己的项目中。

4. 把 dface_sdk 目录下的 model/normal_binary 目录拷贝至 android studio 目录下的 assets 目录。运行时请自行将该 assets 目录下的模型目录拷贝至设备的 SD 内存卡中。后续的通道初始化过程依赖于该模型目录。



小孔成像科技

5. 调用 SDK 之前需要动态加载 libdfacepro.so 库文件。

例如:

```
static {
        System.loadLibrary("dfacepro");
    }
```

6. 具体调用 SDK 请参考开发文档, dface_sdk/doc 目录。

小孔成像科技

5.0 ROS(机器人操作)系统安装

持续更新中

小孔成像科技