

Opencv3.2 for arm Linux 移植日志

By BobLiang on 20170624/QQ:106030169

一、背景

因为项目要求在全志 H3 板子上运行 Opencv 用来检测图片里的轨迹以实现控制步进电机。目标板的资源稀缺，用来编译 opencv 有些难度。整个移植思路是：在电脑上将 opencv3.2 版本编译成动态链接库(*.so 文件)，然后搬运到目标板的 linux 库路径上。在将 APP 在电脑上编译连接后搬运到目标板上运行。这个过程都需要用到交叉编译。[\(在电脑交叉编译出现各种各样的问题，最后实际在目标板上编译 opencv3.2 so 才可以正常运行，保留电脑交叉编译的日志是因为整个过程和在目标板上的类似\)](#)。

二、开发环境

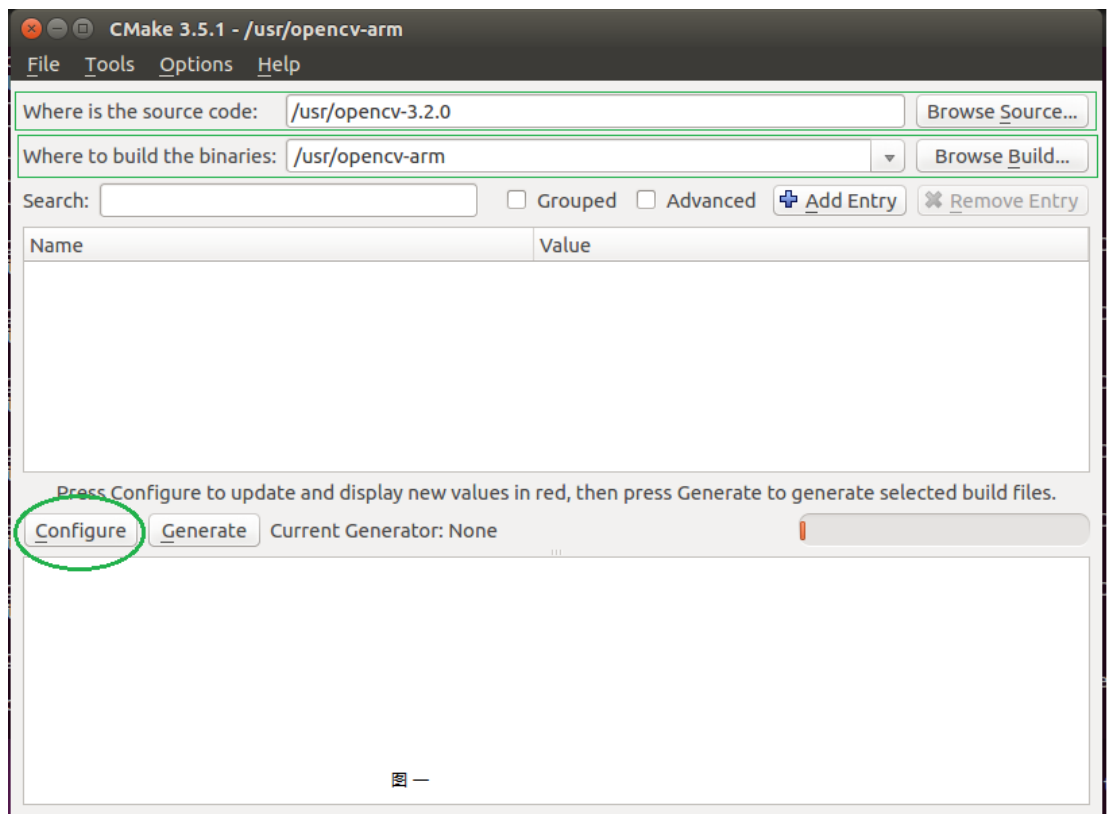
- 1) 在电脑上安装 ubuntu1604 x86 版本；
- 2) 配置国内的更新源（为了更新网速快些）；
- 3) 更新 ubuntu: apt-get update;
- 4) 在电脑上安装交叉编译器:arm-linux-gnueabi-hf-gcc 和 arm-linux-gnueabi-hf-g++ 命令:
apt install gcc-arm-linux-gnueabi-hf
apt install g++-arm-linux-gnueabi-hf
- 5) 增加交叉编译的环境变量(用户级别的~/.bashrc 和生效命令:source ~/.bashrc)
#for arm-linux-gnueabi-hf
export PATH=\$PATH:/usr/arm-linux-gnueabi-hf/bin
- 6) 安装 CMake
在线安装命令: apt-get install cmake
- 7) 安装 cmake-qt-gui
在线安装命令: apt install cmake-qt-gui

三、配置和编译

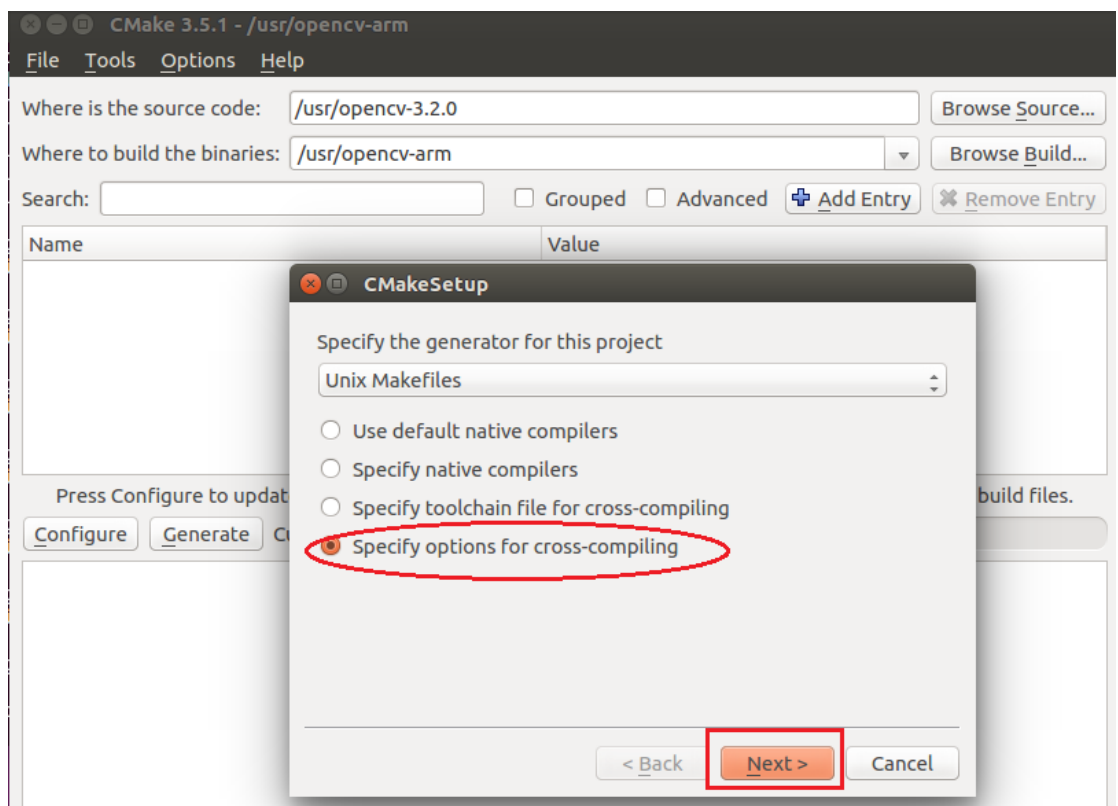
- 1) 下载、拷贝和解压 opencv3.2 到/usr/opencv3.2 目录；
- 2) 新建一个空文件夹/usr/opencv-arm 作为编译路径；
- 3) 新建另一个文件夹/usr/local/arm/lib/opencv3.2(opencv 编译后安装路径)；
- 4) 启动编译配置命令:[cmake-gui](#);

```
root@ubuntu: /usr/opencv-arm#  
root@ubuntu: /usr/opencv-arm#  
root@ubuntu: /usr/opencv-arm#  
root@ubuntu: /usr/opencv-arm# cmake-gui
```

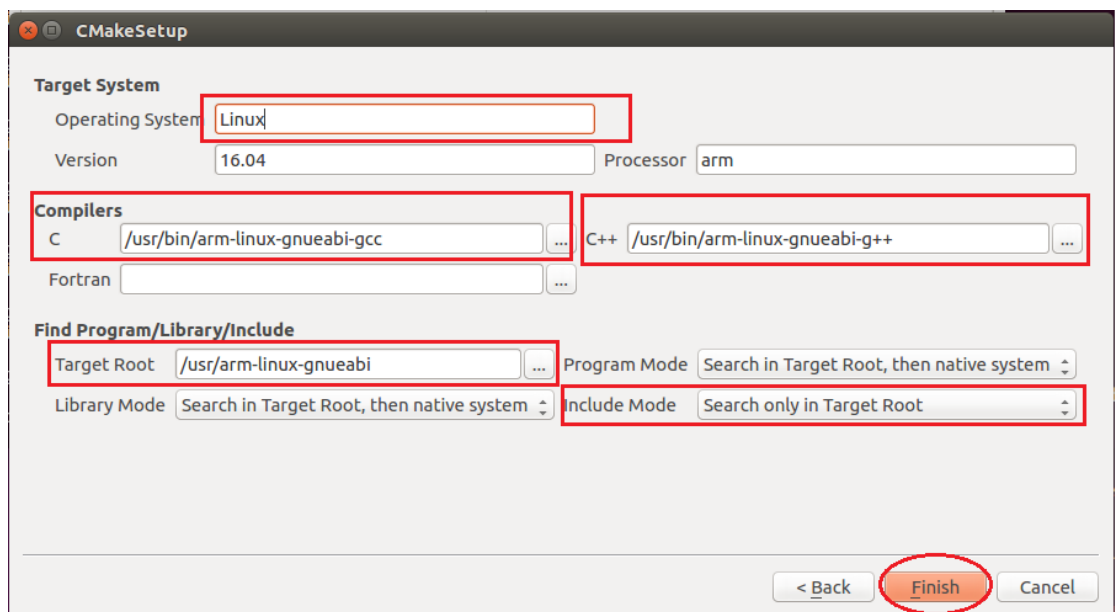
- 5) 在 cmake-gui 界面中配置各个选项(如下图):



6) 选 arm-linux 交叉编译器。点击 Configure 后:

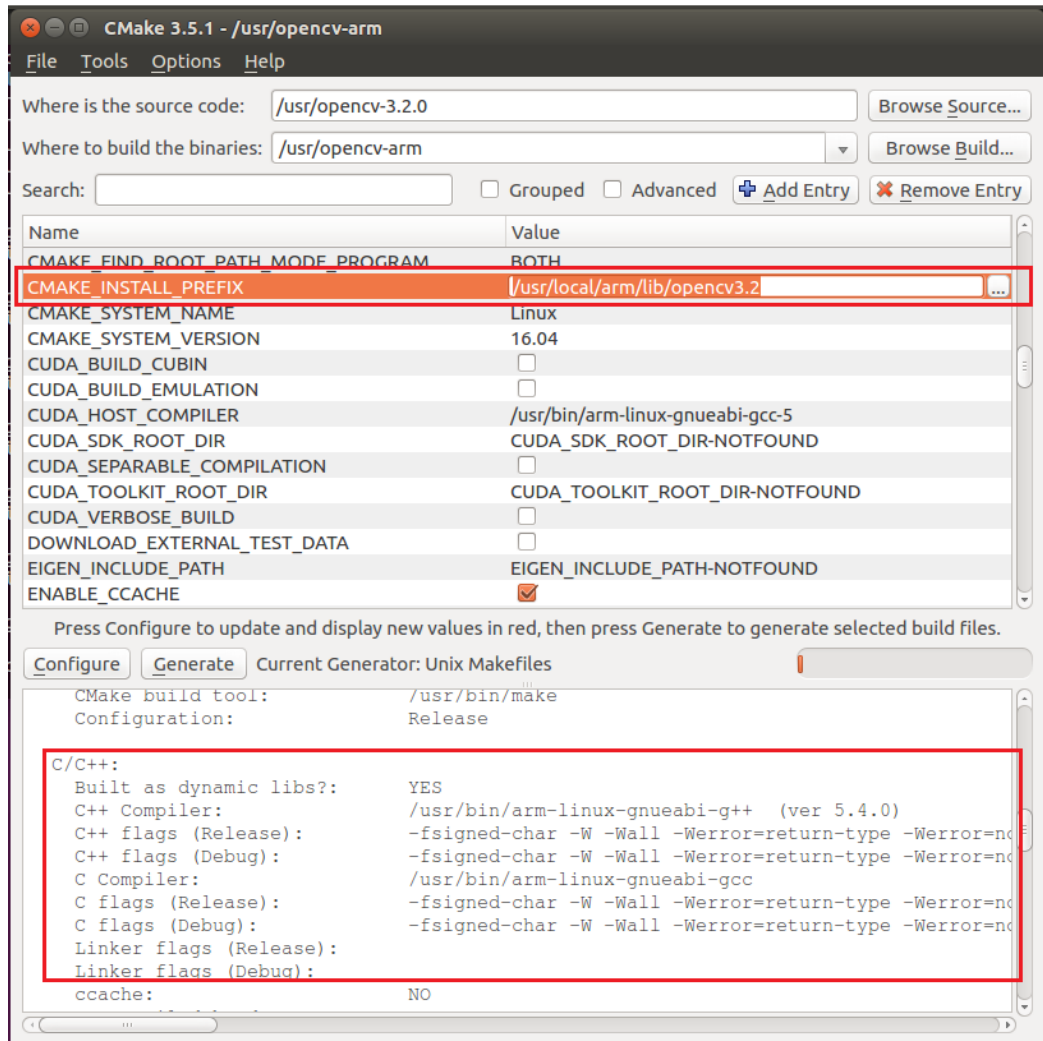


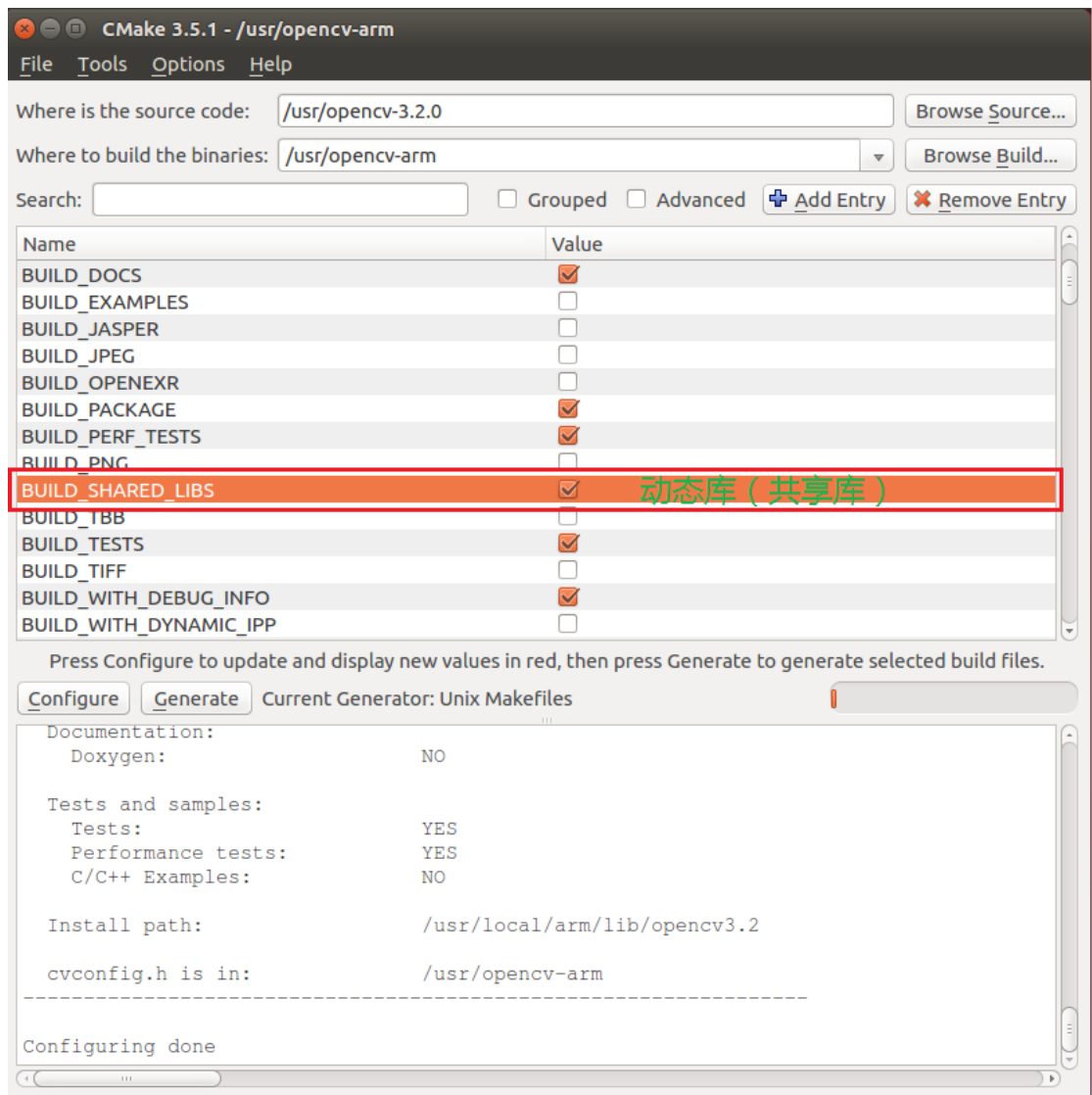
7) 配置编译环境

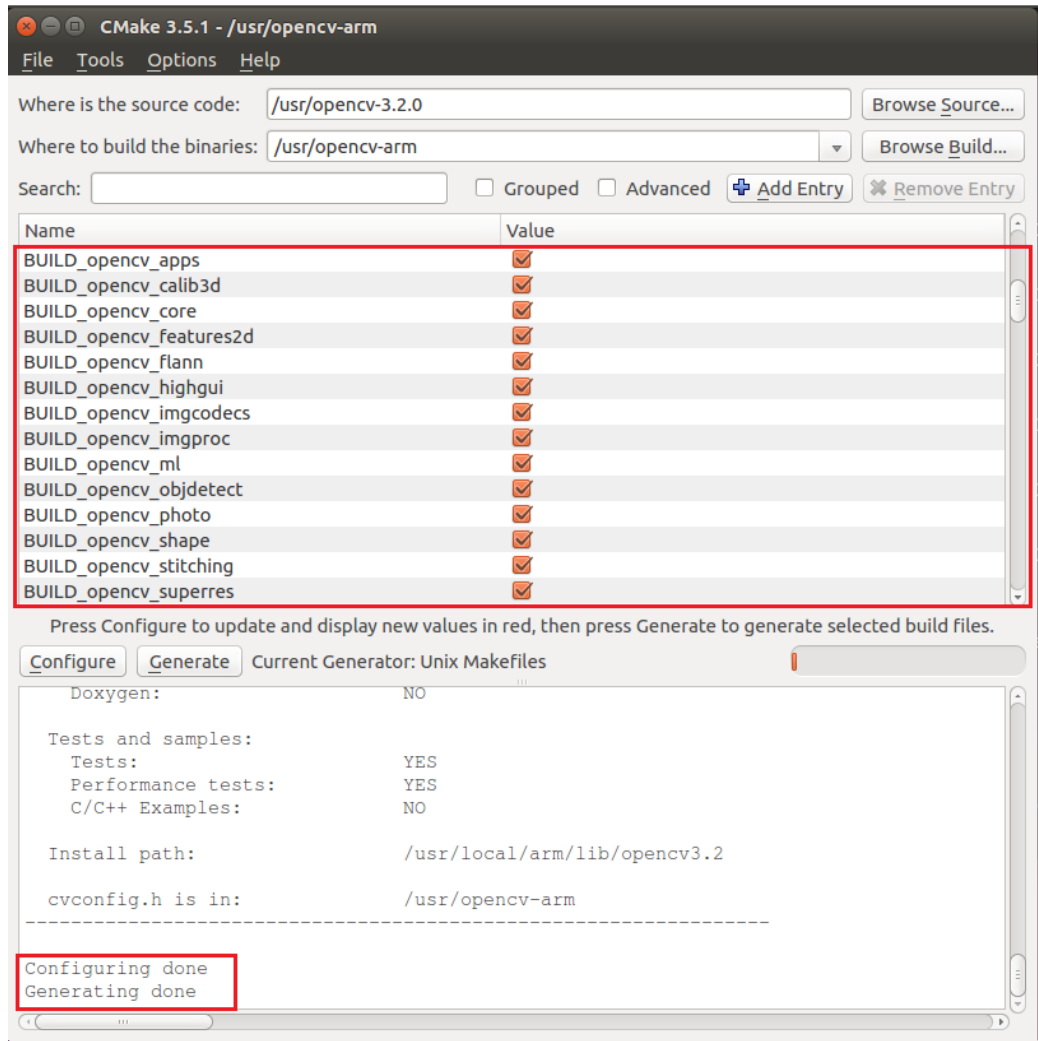


Operation System 要写 Linux，如果 linux 则不会被识别。另外，如果在目标板直接编译似乎可以直接用 gcc/g++ 来编译。

8) 点击 cmake-gui 界面上的 configure 配置生成交叉编译所使用的文件。
这个路径/usr/local/arm/lib/opencv3.2 是编译后用 make install 安装最终结果的路径







以上除了库安装路径外，用默认值即可。

9) 进入编译路径/usr/opencv-arm 开始编译，命令：make 大概 25 分钟编译完毕（看个人电脑速度，我的系统是 VM 虚拟机 ubuntu1604 x86 2G 内存），如下图：

```

root@ubuntu: /usr/opencv-arm
[ 96%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/deblurring.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/frame_source.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/fast_marching.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/motion_stabilizing.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/log.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/inpainting.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/outlier_rejection.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/stabilizer.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/optical_flow.cpp.o
[ 97%] Building CXX object modules/vidstab/CMakeFiles/opencv_vidstab.dir/src/global_motion.cpp.o
[ 98%] Linking CXX shared library ../../lib/libopencv_vidstab.so
[ 98%] Built target opencv_vidstab
Scanning dependencies of target opencv_traincascade
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/features.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/traincascade.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/old_ml_data.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/old_ml_tree.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/old_ml_inner_functions.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/lbpfeatures.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/boost.cpp.o
[ 98%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/old_ml_boost.cpp.o
[ 99%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/HOGfeatures.cpp.o
[ 99%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/haarfeatures.cpp.o
[ 99%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/imagestorage.cpp.o
[ 99%] Building CXX object apps/traincascade/CMakeFiles/opencv_traincascade.dir/cascadeclassifier.cpp.o
[ 99%] Linking CXX executable ../../bin/opencv_traincascade
[ 99%] Built target opencv_traincascade
Scanning dependencies of target opencv_createsamples
[ 99%] Building CXX object apps/createsamples/CMakeFiles/opencv_createsamples.dir/utility.cpp.o
[ 99%] Building CXX object apps/createsamples/CMakeFiles/opencv_createsamples.dir/createsamples.cpp.o
[100%] Linking CXX executable ../../bin/opencv_createsamples
[100%] Built target opencv_createsamples
Scanning dependencies of target opencv_annotation
[100%] Building CXX object apps/annotation/CMakeFiles/opencv_annotation.dir/opencv_annotation.cpp.o
[100%] Linking CXX executable ../../bin/opencv_annotation
[100%] Built target opencv_annotation
Scanning dependencies of target opencv_visualisation
[100%] Building CXX object apps/visualisation/CMakeFiles/opencv_visualisation.dir/opencv_visualisation.cpp.o
[100%] Linking CXX executable ../../bin/opencv_visualisation
[100%] Built target opencv_visualisation
Scanning dependencies of target opencv_version
[100%] Building CXX object apps/version/CMakeFiles/opencv_version.dir/opencv_version.cpp.o
[100%] Linking CXX executable ../../bin/opencv_version
[100%] Built target opencv_version
root@ubuntu: /usr/opencv-arm#

```

10) 运行 `make install`。编译完后，用 `make install` 将 `opencv` 生成的库和头文件安装到目录 `/usr/local/arm/lib/opencv3.2/`。这一步完成后将在目录 `/usr/local/arm/lib/opencv3.2` 得到 `opencv3.2` 基于 `armv7` 的库文件以及头文件等，如下图：

```

root@ubuntu: /usr/local/arm/lib/opencv3.2/lib
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib# ls
libopencv_calib3d.so      libopencv_imgcodecs.so.3.2.0  libopencv_stitching.so.3.2
libopencv_calib3d.so.3.2  libopencv_imgproc.so          libopencv_stitching.so.3.2.0
libopencv_calib3d.so.3.2.0  libopencv_imgproc.so.3.2      libopencv_superres.so
libopencv_core.so         libopencv_imgproc.so.3.2.0    libopencv_superres.so.3.2
libopencv_core.so.3.2      libopencv_ml.so               libopencv_superres.so.3.2.0
libopencv_core.so.3.2.0    libopencv_ml.so.3.2           libopencv_video.so
libopencv_features2d.so    libopencv_ml.so.3.2.0         libopencv_videoio.so.3.2
libopencv_features2d.so.3.2  libopencv_objdetect.so        libopencv_videoio.so.3.2.0
libopencv_features2d.so.3.2.0  libopencv_objdetect.so.3.2    libopencv_video.so
libopencv_flann.so         libopencv_objdetect.so.3.2.0  libopencv_video.so.3.2
libopencv_flann.so.3.2      libopencv_photo.so            libopencv_video.so.3.2.0
libopencv_flann.so.3.2.0    libopencv_photo.so.3.2        libopencv_videostab.so
libopencv_highgui.so       libopencv_photo.so.3.2.0      libopencv_videostab.so.3.2
libopencv_highgui.so.3.2    libopencv_shape.so            libopencv_videostab.so.3.2.0
libopencv_highgui.so.3.2.0  libopencv_shape.so.3.2        pkgconfig
libopencv_imgcodecs.so     libopencv_shape.so.3.2.0
libopencv_imgcodecs.so.3.2  libopencv_stitching.so
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib#

```

11) 查看生成安装包的配置 `pkgconfig` 文件夹下的 `opencv.pc`

```
Text Editor
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib/pkgconfig
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib/pkgconfig# ls
opencv.pc
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib/pkgconfig# ll
total 12
drwxr-xr-x 2 root root 4096 Jun 21 15:39 ./
drwxr-xr-x 3 root root 4096 Jun 21 15:39 ../
-rw-r--r-- 1 root root 654 Jun 21 13:11 opencv.pc
root@ubuntu: /usr/local/arm/lib/opencv3.2/lib/pkgconfig# gedit opencv.pc

opencv.pc (/usr/local/arm/lib/opencv3.2/lib/pkgconfig) - gedit
File Edit View Search Tools Documents Help
Open Save

# Package Information for pkg-config

prefix=/usr/local/arm/lib/opencv3.2
exec_prefix=${prefix}
libdir=${exec_prefix}/lib
includedir_old=${prefix}/include/opencv
includedir_new=${prefix}/include

Name: OpenCV
Description: Open Source Computer Vision Library
Version: 3.2.0
Libs: -L${exec_prefix}/lib -lopencv_shape -lopencv_stitching -lopencv_objdetect -lopencv_superres -
lopencv_videostab -lopencv_calib3d -lopencv_features2d -lopencv_highgui -lopencv_videoio -
lopencv_imgcodecs -lopencv_video -lopencv_photo -lopencv_ml -lopencv_imgproc -lopencv_flann -
lopencv_core
Libs.private: -ldl -lm -lpthread -lrt
Cflags: -I${includedir_old} -I${includedir_new}
```

12) 加入环境变量 PKG_CONFIG_PATH

```
bash.bashrc (/etc) - gedit
File Edit View Search Tools Documents Help
Open Save

        return $?
    elif [ -x /usr/share/command-not-found/command-not-found ]; then
        /usr/share/command-not-found/command-not-found -- "$1"
        return $?
    else
        printf "%s: command not found\n" "$1" >&2
        return 127
    fi
fi

#added by BobLiang on 20170621:opencv for arm-linux
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/arm/lib/opencv3.2/lib/pkgconfig
export PKG_CONFIG_PATH

sh Tab Width: 8 Ln 70, Col 52 INS
```

关闭再重启终端(为了让/etc/bash.bashrc 重新更新)后测试:

命令: pkg-config --cflags opencv

pkg-config --libs opencv

```
root@ubuntu:/home/bob# pkg-config --cflags opencv
-I/usr/local/arm/lib/opencv3.2/include/opencv -I/usr/local/arm/lib/opencv3.2/include
root@ubuntu:/home/bob# pkg-config --libs opencv
-L/usr/local/arm/lib/opencv3.2/lib -lopencv_shape -lopencv_stitching -lopencv_objdetect -lopencv_superres -lopencv_videostab -lopencv_calib3d -lopencv_features2d -lopencv_highgui -lopencv_videoio -lopencv_imgcodecs -lopencv_video -lopencv_photo -lopencv_ml -lopencv_imgproc -lopencv_flann -lopencv_core
root@ubuntu:/home/bob#
```

13) 在电脑交叉编译 app 用来测试:

命令: arm-linux-gnueabi-g++ cam.cpp -o cam `pkg-config opencv --libs --cflags`

因为 app 是 cam.cpp, 需要用 g++ 来编译连接


```

root@ubuntu:/usr/TestCV/cam# arm-linux-gnueabi-g++ cam.cpp -o cam `pkg-config op
encv --libs --cflags`
root@ubuntu:/usr/TestCV/cam# ls
cam cam.cpp Makefile Makefile2.old Makefile.old vars.mk
root@ubuntu:/usr/TestCV/cam#

```

cam.cpp 代码:

```

#include <cv.h>
#include <opencv/highgui.h>
#include <stdio.h>
#include <opencv2/opencv.hpp>
using namespace cv;
int main(int argc, char *argv[])
{
    CvCapture* pCapture = cvCreateCameraCapture(0);
    cvNamedWindow("Video", 1);
    while(1)
    {
        IplImage* pFrame=cvQueryFrame( pCapture );
        if(!pFrame)
        {
            break;
        }
        cvShowImage("Video",pFrame);
        char c=cvWaitKey(33);
        if(c==27)
        {
            break;
        }
    }
    cvReleaseCapture(&pCapture);
    cvDestroyWindow("Video");
    return 0;
}

```

14) 将上面的 opencv3.2 库和测试 app 拷贝到 arm-linux 板子上。

四、在目标板运行

1) 添加动态 opencv3.2 库链接库路径

在/etc/ld.so.conf.d/下建一个 opencv.conf 空文件,再打开 opencv.conf 添加路径 /usr/opencv/lib;

2) 用命令 ldconfig 来使它生效;

3) 尝试运行 cam

注意: 因为 cam 代码里有窗口, 需要在图形界面运行, 因为当时没有保存这个界面信息, 只用串口窗口代替。

```
(COM3,115200) - PuTTY 打开成功
root@OrangePI:/usr/opencv3.2/lib#
root@OrangePI:/usr/opencv3.2/lib# ls cam*
cam
root@OrangePI:/usr/opencv3.2/lib# ./cam
-bash: ./cam: No such file or directory
root@OrangePI:/usr/opencv3.2/lib# ldd cam
        not a dynamic executable
root@OrangePI:/usr/opencv3.2/lib# readelf -l cam |grep ld-linux
[Requesting program interpreter: /lib/ld-linux.so.3]
root@OrangePI:/usr/opencv3.2/lib# readelf -l cam

Elf file type is EXEC (Executable file)
Entry point 0x10d24
There are 9 program headers, starting at offset 52

Program Headers:
Type           Offset  VirtAddr  PhysAddr  FileSiz MemSiz  Flg Align
EXIDX          0x004bcc 0x00014bcc 0x00014bcc 0x00128 0x00128 R  0x4
PHDR           0x000034 0x00010034 0x00010034 0x00120 0x00120 R E 0x4
INTERP         0x000154 0x00010154 0x00010154 0x00013 0x00013 R  0x1
[Requesting program interpreter: /lib/ld-linux.so.3]
LOAD           0x000000 0x00010000 0x00010000 0x04cf8 0x04cf8 R E 0x10000
LOAD           0x004ed8 0x00024ed8 0x00024ed8 0x0019c 0x0029c RW 0x10000
DYNAMIC        0x004ee8 0x00024ee8 0x00024ee8 0x00118 0x00118 RW 0x4
NOTE           0x000168 0x00010168 0x00010168 0x00044 0x00044 R  0x4
GNU_STACK     0x000000 0x00000000 0x00000000 0x00000 0x00000 RW 0x10
GNU_RELRO     0x004ed8 0x00024ed8 0x00024ed8 0x00128 0x00128 R  0x1

Section to Segment mapping:
Segment Sections...
00      .ARM.exidx
01
02      .interp
03      .interp .note.ABI-tag .note.gnu.build-id .gnu.hash .dynsym .dynstr .gnu.version .gnu.version_r .rel.dyn .rel.plt .init .plt .text .fini .rodata .ARM.extab .ARM.exidx .eh_frame
04      .init_array .fini_array .jcr .dynamic .got .data .bss
05      .dynamic
06      .note.ABI-tag .note.gnu.build-id
07
08      .init_array .fini_array .jcr .dynamic
root@OrangePI:/usr/opencv3.2/lib# file ./cam
./cam: ELF 32-bit LSB executable, ARM, EABI5 version 1 (GNU/Linux), dynamically linked, interpreter /lib/ld-linux.so.3, for GNU/Linux 3.2.0, BuildID[sha1]=0df92efc3f85a98bc9051c5a5b4d8b06a03d15c8, not stripped
root@OrangePI:/usr/opencv3.2/lib#
```

尝试运行，但是提示:No such file or directory

4) 定位问题

```
root@OrangePI:/usr/opencv3.2# readelf -d cam |grep NEEDED
0x00000001 (NEEDED)      Shared library: [libopencv_highgui.so.3.2]
0x00000001 (NEEDED)      Shared library: [libopencv_videoio.so.3.2]
0x00000001 (NEEDED)      Shared library: [libopencv_core.so.3.2]
0x00000001 (NEEDED)      Shared library: [libstdc++.so.6]
0x00000001 (NEEDED)      Shared library: [libgcc_s.so.1]
0x00000001 (NEEDED)      Shared library: [libc.so.6]
0x00000001 (NEEDED)      Shared library: [ld-linux.so.3]
root@OrangePI:/usr/opencv3.2# readelf -d hello |grep NEEDED
0x00000001 (NEEDED)      Shared library: [libc.so.6]
root@OrangePI:/usr/opencv3.2#
```

逐个检查和安装上面的动态库后还是无法正常运行!!! 于是放弃这个方案了

五、在 H3 目标板子编译

因为在电脑的 ubuntu16.04 下编译 opencv3.2 无法正常运行，出现各种各样的问题，于是在目标板直接编译 opencv3.2，整个编译过程需要大约 3 个小时。

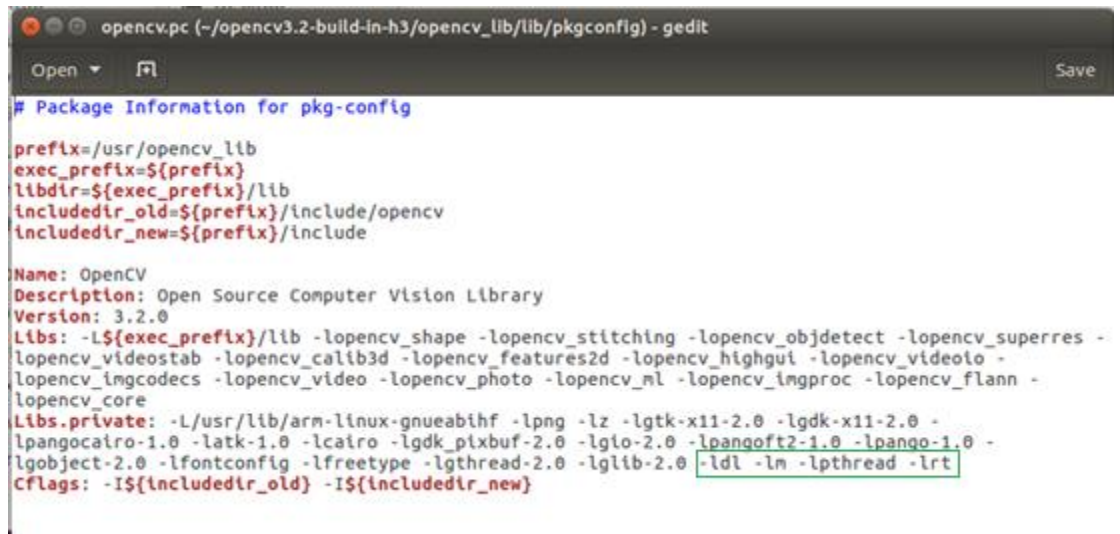
目标板环境：

板子名称：orange pi pc

CPU：全志 H3

系统：orange pi 提供的镜像 即：ubuntu_lxde_desktop_OrangePipc_v0_9_1.img.xz

- 1) 在目标板上安装 gtk2.0
命令: `apt install libgtk2.0-dev`
- 2) 在目标板上编译 opencv3.2, 其编译过程和 cmake-gui 的配置与在电脑上的交叉编译相同, 只是最终结果有差异, 主要体现在 opencv.pc 文件中, 如下图:



```
# Package Information for pkg-config

prefix=/usr/opencv_lib
exec_prefix=${prefix}
libdir=${exec_prefix}/lib
includedir_old=${prefix}/include/opencv
includedir_new=${prefix}/include

Name: OpenCV
Description: Open Source Computer Vision Library
Version: 3.2.0
Libs: -L${exec_prefix}/lib -lopencv_shape -lopencv_stitching -lopencv_objdetect -lopencv_superres -
lopencv_videostab -lopencv_calib3d -lopencv_features2d -lopencv_highgui -lopencv_videoio -
lopencv_imgcodecs -lopencv_video -lopencv_photo -lopencv_ml -lopencv_imgproc -lopencv_flann -
lopencv_core
Libs.private: -L/usr/lib/arm-linux-gnueabi -lpng -lz -lgtk-x11-2.0 -lgdk-x11-2.0 -
lpangocairo-1.0 -latk-1.0 -lcairo -lgdk_pixbuf-2.0 -lgio-2.0 -lpangoft2-1.0 -lpango-1.0 -
lgobject-2.0 -lfontconfig -lfreetype -lgthread-2.0 -lglib-2.0 -ldl -lm -lpthread -lrt
Cflags: -I${includedir_old} -I${includedir_new}
```

比较后发现, 除了 4 个基本库(-ldl -lm -lpthread -lrt)外, 还增加了 17 个其他库。

- 3) 配置动态库环境变量
 - 配置 pkgconfig 的 opencv.pc (参考在电脑编译的相关项);
 - 配置动态库的环境变量 (参考在电脑编译的相关项);
- 4) 在目标板编译 cam.cpp (参考在电脑编译的相关项)。