Rockchip

Linux OopenCV 开发指南

发布版本:**V1.0**

日期:**2019.01**

免责声明

本文档按“现状”提供，福州瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

版权所有 © 2019 福州瑞芯微电子股份有限公司

福州瑞芯微电子股份有限公司
Fuzhou Rockchip Electronics Co., Ltd.
地址: 福建省福州市铜盘路软件园 A 区 18 号
网址: www.rock-chips.com
客户服务电话: +86-591-83991906
客户服务传真: +86-591-83951833
客户服务邮箱: service@rock-chips.com

前言

概述

OpenCV 是一个基于 BSD 许可（开源）发行的跨平台计算机视觉库，实现了图像处理和计算机视觉方面的很多通用算法，本文不再赘述。本文主要介绍 Rockchip 系列芯片如何搭建 OpenCV 环境，以及在此基础上，如何安装/编写 OpenCV 测试 demo，如何调用 RKISP 处理的 Sensor 数据流等等。

本文主要基于 Debian9 系统，简单的来说，OpenCV 通过 Gstreamer 调用到 RK 的 ISP 插件，实现有 3A 的 Camera 预览。

名称	版本
Gstreamer	1.14.4
Python	3.7.2
OpenCV	3.4.5

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

修订记录

日期	版本	作者	审核	修改说明
2019.1.22	V1.0	温暖	蔡枫	添加初始版本

约定

- 代码及命令，本文中代码及命令均用灰色背景填充
- 本文在 RK3399 EVB 板实现，提供的安装编译命令都是在 RK3399 这个 Device 上运行操作
- 命令格式，本文中示例的操作命令， 如遇多行， 则使用转义字符 '\' 将一行命令拆成多行，用户在使用时可以直接复制粘贴，但如果用户是将命令放在一行中， 请去掉转义字符 '\'

目录

目录

插图目录

1 Gstreamer 使用

1.1 使用 Gstreamer 的必要性

OpenCV 中实现了一套 V4L2 取流的程序，我们通过 VideoCapture cap(0);就可以打开/dev/video0 节点的 Camera，但是对于使用 RKISP 的 camera 来说，直接按照此方法是无法预览 camera 的，因为我们使用 ioctl VIDIOC_QUERYCAP 来查询当前 driver 是否合乎规范，OpenCV 中仅仅支持了 V4L2_CAP_VIDEO_CAPTURE，而 RK 平台 RKISP 的 capabilities 为 V4L2_CAP_VIDEO_CAPTURE_MPLANE，所以一种方法是可以参考 camera_engine_rkisp 中 rkisp_demo.cpp 自己写一套调用 camera 的 V4L2 流程，另一种方法是借用 Gstreamer。这两种方法都已经将 ISP 的 3A 处理嵌入其中，Gstreamer 主要以插件(rkisp)的形式实现。

1.2 环境的搭建

首先我们要确保 Device 的 Debian 的环境 Gstreamer 已经是 1.14 版本，可以通过：gst-launch-1.0 --version 这个命令查看，如果是以下版本，一定要升级到 1.14。Device 升级步骤如下：

.移除原先相关的 Gstreamer。

```
apt purge gstreamer1.0-*
apt purge libgstreamer*
```

.添加 Gstreamer 最新版本的源。

```
vi /etc/apt/sources.list
```

.添加下面一行到 sources.list 最后一行，保存退出。

```
deb http://ftp.de.debian.org/debian buster main
```

.更新

```
apt  update
```

.安装 Gstreamer 相关软件

```
apt install gstreamer1.0-plugins-*
apt install gstreamer1.0-libav
apt install libgstreamer1.0*
```

```
    apt-get install libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev \
    libgstreamer-plugins-bad1.0-dev
```

安装完毕，通过 gst-launch-1.0 --version 确认版本。

1.3 Gstreamer 测试

详细可以参考 camera_engine_rkisp 使用文档，本文直接提供相关库的拷贝，以 OV5695 为例，Device 输入以下命令运行：

```
cp librkisp_aec.so /usr/lib/rkisp/ae
cp librkisp_af.so /usr/lib/rkisp/af
cp librkisp_awb.so /usr/lib/rkisp/awb
cp librkisp.so /usr/lib
cp libgstrkisp.so /usr/lib/aarch64-linux-gnu/gstreamer-1.0
cp OV5695.xml /etc/cam_iq.xml
```

编写 camera_rkisp.sh 测试脚本，以 video0 为例， Device 输入以下命令运行：

```
#!/bin/sh
export DISPLAY=:0.0
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib/aarch64-linux-gnu/gstreamer-1.0
#export GST_DEBUG=ispsrc:5
#export GST_DEBUG_FILE=/tmp/2.txt
echo "Start RKISP Camera Preview!"

#echo 7 > /sys/module/video_rkisp1/parameters/debug
su linaro -c " \
gst-launch-1.0 rkisp device=/dev/video0 io-mode=1 analyzer=1 enable-3a=1
path-iqf=/etc/cam_iq.xml ! video/x-raw,format=NV12,width=640,height=480, framerate=30/1 !
videoconvert ! autovideosink"
```

就可以在屏幕上预览 camera 的画面。

2 OpenCV 编译

这一章主要介绍 OpenCV 的编译环境的搭建，编译以及安装。

- 编译环境， 要使用 Gstreamer&Python3，必须包含 Gstreamer&Python3 配置项

2.1 获取并编译 OpenCV

在 https://opencv.org/releases.html 下载最新的 OpenCV 源码包，我们在 Device 下面编译 OpenCV，所以将源码包拷贝到 Device 上解压。

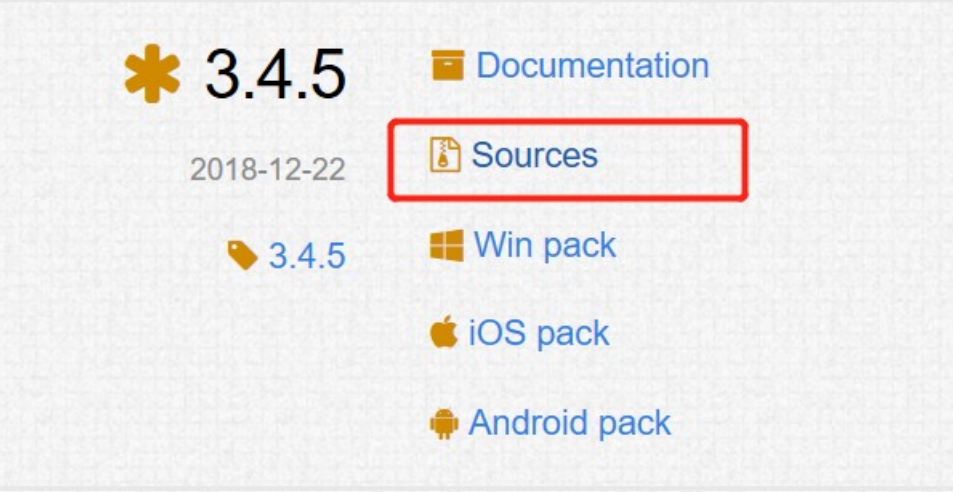

图 2.1 OpenCV Sources

确保 Device 上有 aarch64-linux-gnu-cpp 和 aarch64-linux-gnu-g++命令

```
apt install g++-aarch64-linux-gnu
```

其次安装 OpenCV 的依赖文件，开发板运行下面命令：

```
apt install build-essential make cmake cmake-curses-gui
apt-get install libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev
apt-get install libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff5-dev libdc1394-22-dev
apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev liblapacke-dev
apt install  gfortran python3-numpy python3-dev libgtk-3-dev
apt install  libxine2-dev libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev
libxvidcore-dev libx264-dev libgtk-3-dev libatlas-base-dev gfortran libopenblas-dev
```

注意由于 OpenCV 编译所需要的空间比较大（5,6G 左右），Device 的空间不足，本文将 OpenCV 代码解压在 U 盘（EXT4）中，插入到 Device 编译。

2.2 编译安装 OpenCV

Device 上执行如下命令：

```
cd opencv3.4.5
mkdir build  && cd build
```

Cmake 配置，指明工具链，本文用 Python3 调用 OpenCV，所以也打开了 Python3 的编译选项，Gstreamer 是默认打开，最后安装到/usr/local 目录下

```
cmake  -DCMAKE_CXX_COMPILER=aarch64-linux-gnu-g++ \ -
DCMAKE_C_COMPILER=aarch64-linux-gnu-gcc -DBUILD_opencv_python3=YES \ -
DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=/usr/local ..
```

```
-- General configuration for OpenCV 3.4.5 =====================================
*
*
*
--
--   Video I/O:
--     DC1394:                      YES (ver 2.2.5)
--     FFMPEG:                      YES
--       avcodec:                   YES (ver 58.35.100)
--       avformat:                  YES (ver 58.20.100)
--       avutil:                    YES (ver 56.22.100)
--       swscale:                   YES (ver 5.3.100)
--       avresample:                YES (ver 4.0.0)
--     GStreamer:
--       base:                      YES (ver 1.14.4)
--       video:                     YES (ver 1.14.4)
--       app:                       YES (ver 1.14.4)
--       riff:                      YES (ver 1.14.4)
--       pbutils:                   YES (ver 1.14.4)
--     libv4l/libv4l2:              NO
--     v4l/v4l2:                    linux/videodev2.h
--
--   Parallel framework:            pthreads
--
--   Trace:                         YES (built-in)
--
--   Other third-party libraries:
--     Lapack:                      NO
--     Eigen:                       YES (ver 3.3.7)
--     Custom HAL:                  YES (carotene (ver 0.0.1))
--     Protobuf:                    build (3.5.1)
--
--   OpenCL:                        YES (no extra features)
--     Include path:                /media/linaro/wn/opencv-3.4.5/3rdparty/include/opencl/1.2
--     Link libraries:              Dynamic load
--
--   Python 3:
--     Interpreter:                 /usr/bin/python3 (ver 3.7.2)
--     Libraries:                   /usr/lib/aarch64-linux-gnu/libpython3.7m.so (ver 3.7.2)
--     numpy:                       /usr/lib/python3/dist-packages/numpy/core/include (ver 1.16.0rc2)
--     install path:                lib/python3.7/dist-packages/cv2/python-3.7
--
--   Python (for build):            /usr/bin/python2.7
--
--   Java:
--     ant:                         NO
--     JNI:                         NO
--     Java wrappers:               NO
--     Java tests:                  NO
--
--   Install to:                    /usr/local
-- -----------------------------------------------------------------
```

图 2.2 OpenCV 配置

如图 2-2，Gstraemr&Python3 配置 YES，表示配置正确，接下来是编译，如下

```
make -j4
```

安装，保证你是 root 权限

```
sudo make install
```

3 OpenCV 测试

本章主要介绍 OepnCV 测试 RKISP 的 camera。

编写 Python3 的测试代码 camera.py，并且拷贝到/usr/local/bin

```
import numpy as np
import cv2

cap = cv2.VideoCapture("rkisp device=/dev/video0 io-mode=1 analyzer=1 enable-3a=1 path-iqf=/etc/cam_iq.xml ! video/x-raw,format=NV12,width=640,height=480, framerate=30/1 ! videoconvert ! appsink", cv2.CAP_GSTREAMER)
if not cap.isOpened():
    print('VideoCapture not opened')
    exit(0)
while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
    # Display the resulting frame
    cv2.imshow('Hello RK-OpenCv',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

编写测试脚本 test_opencv_rkisp.sh, 赋予执行权限，并且拷贝到/usr/local/bin

```
#!/bin/sh
export DISPLAY=:0.0
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib/aarch64-linux-gnu/gstreamer-1.0
#export GST_DEBUG=ispsrc:5
#export GST_DEBUG_FILE=/tmp/2.txt

echo "Start RKISP Camera Preview!"
#echo 7 > /sys/module/video_rkisp1/parameters/debug

su linaro -c "python3 /usr/local/bin/camera.py"
```

最终运行 test_opencv_rkisp.sh，预览效果如下

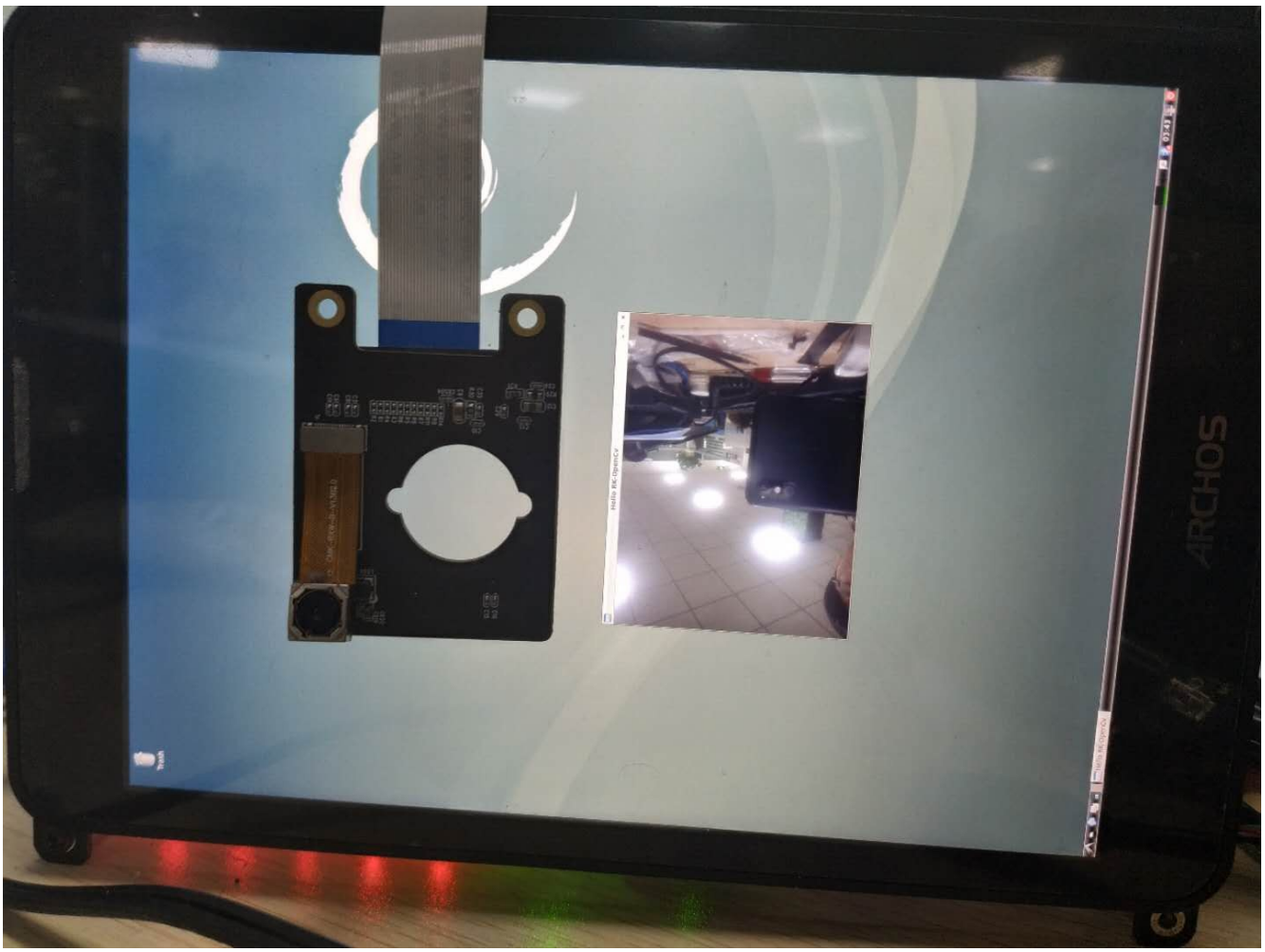

图 3.1 Camera 预览