bookworm OS

BullsEye 이후의 라즈베리파이에서 CSI 카메라 사용의 문제점 및 해결책은 다음 글에서 이미 정리해두었습니다.

- BullsEye-opencv

이 글에서는 라즈베리파이 booklworm에서 2024년 12월 최신 버젼의 OpenCV (버젼 4.10.0)을 bookworm에서 설치하는 스크립트 파일만 수정해서 올립니다.

다음은 libcamera와 기존 raspicam 프로그램간의 비교입니다. BullsEye에서는 libcamera-* 형식의 명령어였지만 bookworm에서 rpicam-*으로 명령어들이 바뀌었습니다. 기존 명령어에 대한 심볼릭 링크만 추가되었기 때문에 기존 명령어(libcamera-*)* 역시 사용할 수 있습니다.

rpicam-apps	**설명**	**대응 raspicam app(libcamera 적용 이전 버젼 명령)**
rpicam-hello	미리보기 창을 이용해 카메라 화면을 보여줍니다.	raspistill -t 5
rpicam-jpeg	jpeg 파일 촬영	raspistill -o image.jpg
rpicam-still	raspistill과 유사하게 rpicam-jpeg보다 정교한 옵션 제공.	raspistill
rpicam-vid	비디오 촬영	raspivid

bookworm OS에서 OpenCV 빌드(32비트, 64비트 공통)

빌드 작업을 하기 전에 반드시 설치된 패키지들을 최신으로 업데이트합니다.

```
apt-get update
apt-get upgrade
```

다음 스크립트는 Install OpenCV on Raspberry 64 OS에서 소개한 내용을 일부 수정한 것입니다.
기존 bullseye에서 사용한 설치 스크립트와 비교하면 Threading Building Blocks (TBB) 라이브러리 설치 방법이 bookworm에서 일부 바뀌었습니다.

그리고 이 스크립트는 설치하고자하는 OpenCV 버젼을 파라미터로 입력받습니다. 2024년 12월 현재 최신 OpenCV는 4.10.0 버젼입니다. 따라서 "./install_opencv.sh 4.10.0"과 같이 사용하면 됩니다.

```
#!/bin/bash
version=$1
if [ $# -eq 0 ]; then
    echo "No OpenCV version is given. run like this : install_opencv.sh 4.10.0"
    exit 0
else
  echo "Installing OpenCV ${version} on your Raspberry Pi 64-bit OS"
  echo "It will take minimal 2.0 hour !"
fi

set -e
cd ~
# install the dependencies
sudo apt-get install -y build-essential cmake git unzip pkg-config
sudo apt-get install -y libjpeg-dev libtiff-dev libpng-dev
sudo apt-get install -y libavcodec-dev libavformat-dev libswscale-dev
sudo apt-get install -y libgtk2.0-dev libcanberra-gtk* libgtk-3-dev
sudo apt-get install -y libgstreamer1.0-dev gstreamer1.0-gtk3
sudo apt-get install -y libgstreamer-plugins-base1.0-dev gstreamer1.0-gl
sudo apt-get install -y libxvidcore-dev libx264-dev
sudo apt-get install -y python3-dev python3-numpy python3-pip
sudo apt-get install -y libv4l-dev v4l-utils
sudo apt-get install -y libopenblas-dev libatlas-base-dev libblas-dev
sudo apt-get install -y liblapack-dev gfortran libhdf5-dev
sudo apt-get install -y libprotobuf-dev libgoogle-glog-dev libgflags-dev
sudo apt-get install -y protobuf-compiler
sudo apt-get install -y libtbbmalloc2 libtbb-dev libdc1394-dev gstreamer1.0-
libcamera
#If you want to build with QT, install libqt6core5compat6-dev
sudo apt-get -y install libqt6core5compat6-dev

# download the latest version
cd ~
sudo rm -rf opencv*
echo "**** Downloading opencv ****"
wget -O opencv.zip
https://github.com/opencv/opencv/archive/refs/tags/${version}.zip
echo "**** Downloading opencv_contrib ****"
wget -O opencv_contrib.zip
https://github.com/opencv/opencv_contrib/archive/refs/tags/${version}.zip

# unpack
unzip opencv.zip
unzip opencv_contrib.zip
```

```
# some administration to make live easier later on
mv opencv-${version} opencv
mv opencv_contrib-${version} opencv_contrib

# clean up the zip files
rm opencv.zip
rm opencv_contrib.zip

cd opencv
mkdir build
cd build

# run cmake
cmake -D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
-D ENABLE_NEON=ON \
-D WITH_OPENMP=ON \
-D WITH_OPENCL=OFF \
-D BUILD_TIFF=ON \
-D WITH_FFMPEG=ON \
-D WITH_TBB=ON \
-D BUILD_TBB=ON \
-D WITH_GSTREAMER=ON \
-D BUILD_TESTS=OFF \
-D WITH_EIGEN=OFF \
-D WITH_V4L=ON \
-D WITH_LIBV4L=ON \
-D WITH_VTK=OFF \
-D WITH_QT=OFF \
-D OPENCV_ENABLE_NONFREE=ON \
-D INSTALL_C_EXAMPLES=OFF \
-D INSTALL_PYTHON_EXAMPLES=OFF \
-D PYTHON3_PACKAGES_PATH=/usr/lib/python3/dist-packages \
-D OPENCV_GENERATE_PKGCONFIG=ON \
-D BUILD_EXAMPLES=OFF ..

# run make
make -j4
sudo make install
sudo ldconfig

# cleaning (frees 300 MB)
make clean
sudo apt-get update

echo "Congratulations!"
echo "You've successfully installed OpenCV ${version} on your Raspberry Pi 64-
bit OS"
```

위 내용으로 install_opencv.sh 파일을 만들어 실행하면 됩니다.
빌드가 끝나면 자동으로 설치까지 하도록 스크립트가 만들어져 있습니다. 그리고 가장 중요한 것은 빌드 옵션에 "-D WITH_GSTREAMER=ON " 가 추가되어 있다는 것입니다. 약 1시간 이상의 빌드 시간이 필요합니다.

```
# 만약 pip으로 설치한 버젼이 있으면 삭제후 설치한다.
$ pip3 uninstall opencv-python
$ chmod 755 install_opencv.sh
$ ./install_opencv.sh 4.10.0
```

GStreamer 지원 OpenCV 확인

빌드 스크립트는 설치까지 해주기 때문에 이제 다시 빌드 정보를 확인해봅니다.

```
spypiggy@raspberrypi:~ $ python3
Python 3.11.2 (main, Sep 14 2024, 03:00:30) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> print(cv2.getBuildInformation())

General configuration for OpenCV 4.10.0 =====================================
  Version control:               unknown

  Extra modules:
    Location (extra):            /home/spypiggy/opencv_contrib/modules
    Version control (extra):     unknown

  Platform:
    Timestamp:                   2024-12-25T00:41:58Z
    Host:                        Linux 6.6.62+rpt-rpi-v8 aarch64
    CMake:                       3.25.1
    CMake generator:             Unix Makefiles
    CMake build tool:            /usr/bin/gmake
    Configuration:               RELEASE

  CPU/HW features:
    Baseline:                    NEON FP16
      required:                  NEON
    Dispatched code generation:  NEON_DOTPROD NEON_FP16 NEON_BF16
      requested:                 NEON_FP16 NEON_BF16 NEON_DOTPROD
      NEON_DOTPROD (1 files):    + NEON_DOTPROD
```

```
      NEON_FP16 (2 files):       + NEON_FP16
      NEON_BF16 (0 files):       + NEON_BF16

  C/C++:
    Built as dynamic libs?:      YES
    C++ standard:                11
    C++ Compiler:                /usr/bin/c++  (ver 12.2.0)
    C++ flags (Release):         -fsigned-char -W -Wall -Wreturn-type -Wnon-
virtual-dtor -Waddress -Wsequence-point -Wformat -Wformat-security -Wmissing-
declarations -Wundef -Winit-self -Wpointer-arith -Wshadow -Wsign-promo -
Wuninitialized -Wsuggest-override -Wno-delete-non-virtual-dtor -Wno-comment -
Wimplicit-fallthrough=3 -Wno-strict-overflow -fdiagnostics-show-option -pthread
-fomit-frame-pointer -ffunction-sections -fdata-sections    -fvisibility=hidden
-fvisibility-inlines-hidden -O3 -DNDEBUG  -DNDEBUG
    C++ flags (Debug):           -fsigned-char -W -Wall -Wreturn-type -Wnon-
virtual-dtor -Waddress -Wsequence-point -Wformat -Wformat-security -Wmissing-
declarations -Wundef -Winit-self -Wpointer-arith -Wshadow -Wsign-promo -
Wuninitialized -Wsuggest-override -Wno-delete-non-virtual-dtor -Wno-comment -
Wimplicit-fallthrough=3 -Wno-strict-overflow -fdiagnostics-show-option -pthread
-fomit-frame-pointer -ffunction-sections -fdata-sections    -fvisibility=hidden
-fvisibility-inlines-hidden -g  -O0 -DDEBUG -D_DEBUG
    C Compiler:                  /usr/bin/cc
    C flags (Release):           -fsigned-char -W -Wall -Wreturn-type -Waddress
-Wsequence-point -Wformat -Wformat-security -Wmissing-declarations -Wmissing-
prototypes -Wstrict-prototypes -Wundef -Winit-self -Wpointer-arith -Wshadow -
Wuninitialized -Wno-comment -Wimplicit-fallthrough=3 -Wno-strict-overflow -
fdiagnostics-show-option -pthread -fomit-frame-pointer -ffunction-sections -
fdata-sections    -fvisibility=hidden -O3 -DNDEBUG  -DNDEBUG
    C flags (Debug):             -fsigned-char -W -Wall -Wreturn-type -Waddress
-Wsequence-point -Wformat -Wformat-security -Wmissing-declarations -Wmissing-
prototypes -Wstrict-prototypes -Wundef -Winit-self -Wpointer-arith -Wshadow -
Wuninitialized -Wno-comment -Wimplicit-fallthrough=3 -Wno-strict-overflow -
fdiagnostics-show-option -pthread -fomit-frame-pointer -ffunction-sections -
fdata-sections    -fvisibility=hidden -g  -O0 -DDEBUG -D_DEBUG
    Linker flags (Release):      -Wl,--gc-sections -Wl,--as-needed -Wl,--no-
undefined
    Linker flags (Debug):        -Wl,--gc-sections -Wl,--as-needed -Wl,--no-
undefined
    ccache:                      NO
    Precompiled headers:         NO
    Extra dependencies:          dl m pthread rt
    3rdparty dependencies:

  OpenCV modules:
    To be built:                 aruco bgsegm bioinspired calib3d ccalib core
datasets dnn dnn_objdetect dnn_superres dpm face features2d flann freetype
fuzzy gapi hdf hfs highgui img_hash imgcodecs imgproc intensity_transform
line_descriptor mcc ml objdetect optflow phase_unwrapping photo plot python3
quality rapid reg rgbd saliency shape signal stereo stitching structured_light
superres surface_matching text tracking ts video videoio videostab
wechat_qrcode xfeatures2d ximgproc xobjdetect xphoto
    Disabled:                    world
    Disabled by dependency:      -
```

```
    Unavailable:                 alphamat cannops cudaarithm cudabgsegm
cudacodec cudafeatures2d cudafilters cudaimgproc cudalegacy cudaobjdetect
cudaoptflow cudastereo cudawarping cudev cvv java julia matlab ovis python2 sfm
viz
    Applications:                perf_tests apps
    Documentation:               NO
    Non-free algorithms:         YES

  GUI:                           GTK3
    GTK+:                        YES (ver 3.24.38)
      GThread :                  YES (ver 2.74.6)
      GtkGlExt:                  NO

  Media I/O:
    ZLib:                        /usr/lib/aarch64-linux-gnu/libz.so (ver
1.2.13)
    JPEG:                        /usr/lib/aarch64-linux-gnu/libjpeg.so (ver 62)
    WEBP:                        /usr/lib/aarch64-linux-gnu/libwebp.so (ver
encoder: 0x020f)
    PNG:                         /usr/lib/aarch64-linux-gnu/libpng.so (ver
1.6.39)
    TIFF:                        build (ver 42 - 4.6.0)
    JPEG 2000:                   build (ver 2.5.0)
    OpenEXR:                     OpenEXR::OpenEXR (ver 3.1.5)
    HDR:                         YES
    SUNRASTER:                   YES
    PXM:                         YES
    PFM:                         YES

  Video I/O:
    DC1394:                      YES (2.2.6)
    FFMPEG:                      YES
      avcodec:                   YES (59.37.100)
      avformat:                  YES (59.27.100)
      avutil:                    YES (57.28.100)
      swscale:                   YES (6.7.100)
      avresample:                NO
    GStreamer:                   YES (1.22.0)
    v4l/v4l2:                    YES (linux/videodev2.h)

  Parallel framework:            TBB (ver 2021.11 interface 12110)

  Trace:                         YES (with Intel ITT)

  Other third-party libraries:
    Lapack:                      NO
    Custom HAL:                  YES (carotene (ver 0.0.1, Auto detected))
    Protobuf:                    build (3.19.1)
    Flatbuffers:                 builtin/3rdparty (23.5.9)

  Python 3:
    Interpreter:                 /usr/bin/python3 (ver 3.11.2)
    Libraries:                   /usr/lib/aarch64-linux-gnu/libpython3.11.so
```

```
(ver 3.11.2)
    Limited API:                 NO
    numpy:                       /usr/lib/python3/dist-
packages/numpy/core/include (ver 1.24.2)
    install path:                /usr/lib/python3/dist-packages/cv2/python-3.11

  Python (for build):            /usr/bin/python3

  Java:
    ant:                         NO
    Java:                        NO
    JNI:                         NO
    Java wrappers:               NO
    Java tests:                  NO

  Install to:                    /usr/local
-----------------------------------------------------------------
```

Video I/O에서 GStreamer가 YES(1.22.0)로 되어있는 것을 알 수 있습니다. 이제 우리가 빌드한 OpenCV는 GStreamer를 사용할 수 있습니다.

```
# install the remaining plugins
$ sudo apt-get install libgstreamer1.0-dev \
     libgstreamer-plugins-base1.0-dev \
     libgstreamer-plugins-bad1.0-dev \
     gstreamer1.0-plugins-ugly \
     gstreamer1.0-tools
# if you have Qt5 install this plugin
$ sudo apt-get install gstreamer1.0-qt5
```

이제 GStreamer가 카메라를 제대로 제어하는지 확인해보겠습니다. GStreamer 파이프라인이 libcamerasrc 에서 시작하는 것을 알 수 있습니다.
라즈베리파이5에서는 GStreamer 파이프라인에 반드시 format을 넣어줘야 합니다. 아마도 라즈베리파이 5 에서 디스플레이포트와 카메라포트가 통합되면서 바뀐 HW 구조 때문인 것 같습니다.
NV12 또는 RGBx 포맷을 Rpi5에서는 반드시 포함해야 작동합니다. Rpi4에서는 이 옵션이 없어도 작동합니다.

```
pi@raspberrypi:~ $ gst-launch-1.0 libcamerasrc  ! video/x-raw,format=NV12,
width=1280, height=720, framerate=30/1 ! videoconvert ! videoscale !
clockoverlay time-format="%D %H:%M:%S" ! video/x-raw, width=640, height=360 !
autovideosink
```

```
Setting pipeline to PAUSED ...
[0:10:17.079545711] [2644]  INFO Camera camera_manager.cpp:325 libcamera
v0.3.2+99-1230f78d
[0:10:17.090998750] [2650]  INFO RPI pisp.cpp:695 libpisp version v1.0.7
28196ed6edcf 29-08-2024 (16:33:32)
[0:10:17.102686550] [2650]  INFO RPI pisp.cpp:1154 Registered camera
/base/axi/pcie@120000/rp1/i2c@88000/imx296@1a to CFE device /dev/media2 and ISP
device /dev/media0 using PiSP variant BCM2712_C0
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
[0:10:17.105845581] [2653]  INFO Camera camera.cpp:1197 configuring streams:
(0) 1280x720-NV12
[0:10:17.106016139] [2650]  INFO RPI pisp.cpp:1450 Sensor:
/base/axi/pcie@120000/rp1/i2c@88000/imx296@1a - Selected sensor format:
1456x1088-SBGGR10_1X10 - Selected CFE format: 1456x1088-PC1B
Redistribute latency...
WARNING: from element
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0/GstXvImageSink:autovideo
sink0-actual-sink-xvimage: Pipeline construction is invalid, please add queues.
Additional debug info:
../libs/gst/base/gstbasesink.c(1249): gst_base_sink_query_latency ():
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0/GstXvImageSink:autovideo
sink0-actual-sink-xvimage:
Not enough buffering available for  the processing deadline of
0:00:00.015000000, add enough queues to buffer  0:00:00.015000000 additional
data. Shortening processing latency to 0:00:00.000000000.
WARNING: from element
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0/GstXvImageSink:autovideo
sink0-actual-sink-xvimage: Pipeline construction is invalid, please add queues.
Additional debug info:
../libs/gst/base/gstbasesink.c(1249): gst_base_sink_query_latency ():
/GstPipeline:pipeline0/GstAutoVideoSink:autovideosink0/GstXvImageSink:autovideo
sink0-actual-sink-xvimage:
Not enough buffering available for  the processing deadline of
0:00:00.015000000, add enough queues to buffer  0:00:00.015000000 additional
data. Shortening processing latency to 0:00:00.000000000.
^Chandling interrupt.
Interrupt: Stopping pipeline ...
Execution ended after 0:00:06.352084308
Setting pipeline to NULL ...
Freeing pipeline ...
```

그리고 화면에 다음과 같이 카메라 화면이 출력됩니다. 참고로 라즈베리파이 OS에서 Legacy Camera는 비활성화된 상태입니다.

GSttreamer가 제대로 libcamera를 이용해서 처리하는 것을 확인했습니다. 이제 드디어 OpenCV에서 GStreamer 파이프라인을 이용해 카메라를 열어보도록 하겠습니다.

OpenCV에서 libcamera 스택을 이용한 카메라 제어

다음은 GStreamer 파이프라인을 이용하는 파이썬 OpenCV 코드입니다.

```
import cv2
import numpy as np
import sys
connstr = 'libcamerasrc ! video/x-raw,format=NV12, width=640, height=480,
framerate=30/1 ! videoconvert ! videoscale ! clockoverlay time-format="%D
%H:%M:%S" ! appsink'
cap = cv2.VideoCapture(connstr, cv2.CAP_GSTREAMER)
if cap.isOpened() == False:
    print('camera open Failed')
    sys.exit(0)


while True:

    succes, img = cap.read()
    if succes == False:
        print('camera read Failed')
        sys.exit(0)

    k = cv2.waitKey(1)
    if k == ord('q'):
        break

    cv2.imshow('Img',img)

cap.release()
cv2.destroyAllWindows()
```

앞에서 gst-launch-1.0를 이용한 GStreamer 테스트에서 사용했던 파이프라인과 거의 동일합니다. 마지막 sink 부분만 appsink로 바뀌었습니다.

```
spypiggy@raspberrypi:~ $ python3 preview.py
[2:14:32.817020022] [20131]  INFO Camera camera_manager.cpp:325 libcamera
v0.3.2+99-1230f78d
[2:14:32.848758221] [20138]  WARN RPiSdn sdn.cpp:40 Using legacy SDN tuning -
please consider moving SDN inside rpi.denoise
[2:14:32.850976268] [20138]  WARN RPI vc4.cpp:393 Mismatch between Unicam and
CamHelper for embedded data usage!
```

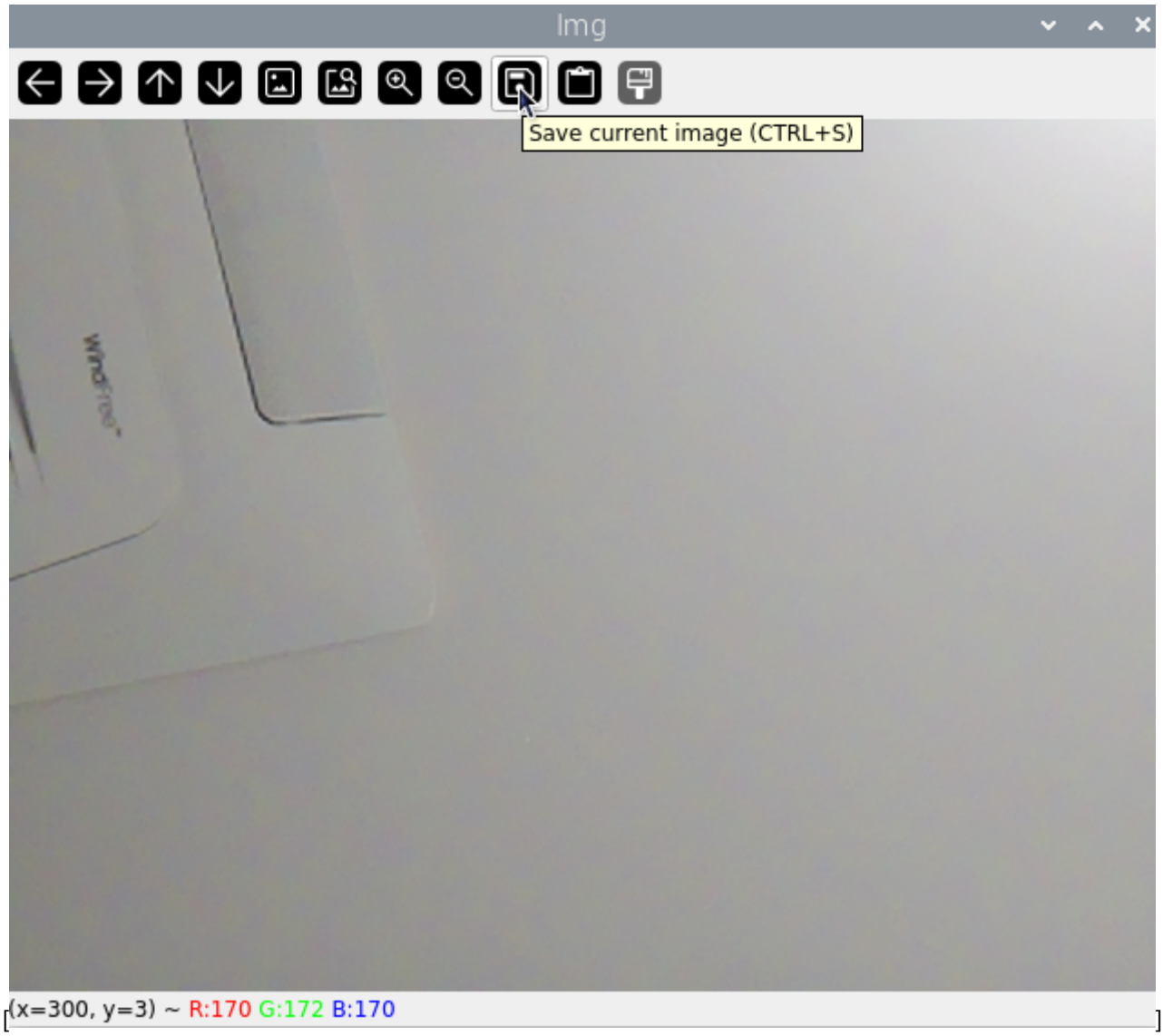
```
[2:14:32.851667921] [20138]  INFO RPI vc4.cpp:447 Registered camera
/base/soc/i2c0mux/i2c@1/imx219@10 to Unicam device /dev/media2 and ISP device
/dev/media0
[2:14:32.862533624] [20141]  INFO Camera camera.cpp:1197 configuring streams:
(0) 640x480-NV21
[2:14:32.862976985] [20138]  INFO RPI vc4.cpp:622 Sensor:
/base/soc/i2c0mux/i2c@1/imx219@10 - Selected sensor format: 640x480-
SBGGR10_1X10 - Selected unicam format: 640x480-pBAA
[ WARN:0@0.543] global cap_gstreamer.cpp:1754 open OpenCV | GStreamer warning:
unable to query duration of stream
[ WARN:0@0.543] global cap_gstreamer.cpp:1777 open OpenCV | GStreamer warning:
Cannot query video position: status=0, value=-1, duration=-1
```

정상 작동합니다.

QT 지원

앞에서 install_opencv.sh 스크립트에서 cmake 옵션 중 WITH_QT 값은 OFF입니다. 즉 QT를 제외한 opencv를 빌드하겠다는 의미입니다.
QT 옵션을 ON으로 바꾸면 OpenCV에서 GUI 기능을 추가할 수 있습니다.
예를 들어 Preview 창에서 현재 프레임을 이미지로 저장하거나 zoom in, out 기능을 구현할 수도 있습니다.
QT 옵션을 활성화 하려면 반드시 libqt6core5compat6-dev 패키지를 미리 설치해야 합니다.
install_opencv.sh 스크립트에서는 이 패키지를 설치하도록 해두었습니다. QT 옵션을 활성화한 패키지는 약간의 성능저하가 발생할 수 있습니다. QT GUI 기능이 필요한 경우에만 활성화 후에 빌드하는 것이 좋습니다.

```
sudo apt-get -y install libqt6core5compat6-dev
```



위 그림은 다음과 같이 단순한 프로그램입니다. 하지만 Window 상단의 버튼들은 위에서 설명한 여러가지 기능을 GUI 버튼을 통해 지원합니다.

```
#preview.py
import cv2
import numpy as np
import sys
connstr = 'libcamerasrc ! video/x-raw, format=RGBx, width=640, height=480,
framerate=30/1 ! videoconvert! video/x-raw, format=(string)BGR ! videoscale !
appsink'
cap = cv2.VideoCapture(connstr, cv2.CAP_GSTREAMER)
if cap.isOpened() == False:
    print('camera open Failed')
    sys.exit(0)

while True:
    succes, img = cap.read()
    if succes == False:
        print('camera read Failed')
```

```
            sys.exit(0)

        k = cv2.waitKey(1)
        if k == ord('q'):
            break

        cv2.imshow('Img',img)

    cap.release()
    cv2.destroyAllWindows()
```

결론

라즈베리파이 OS가 BullsEye에서 bookworm으로 업데이트되면서 tbb관련 라이브러리가 바뀌면서 설치하는 패키지 이름이 살짝 바뀌었습니다. 따라서 스크립트 파일에도 약간의 변경이 생겼지만 크게 바뀐 부분은 없습니다.
카메라에서 프레임을 읽은 방법을 반드시 OpenCV를 이용할 필요는 없습니다. PiCamera2의 캡쳐 기능을 이용하고 나머지 화면 출력, 이미지 프로세싱 등의 기능을 OpenCV에서 사용해도 됩니다. 이 방법은 OpenCV를 빌드할 필요없이 pip으로 설치한 opencv-python을 이용할 수 있다는 장점이 있습니다.
하지만 OpenCV를 이용해서 CSI 카메라 입력을 받아야만 하는 경우하면 위의 빌드 과정으로 OpenCV를 빌드해서 사용하시면 됩니다.