All Posts

Ivan Nemov Jul 4 3 min read

Install and troubleshoot PyQt5 on Raspberry Pi

Overview

When it comes to developing GUI for an autonomous project on Raspberry Pi, PyQt5 could be a good choice thanks to its versatility and broad user community: for sure you are not going to stay alone in its troubleshooting.

If your Raspberry Pi project involves computer vision, my suggestion is to read first PyImageSearch posts about OpenCV installation, and then install PyQt5 on top of it in the same virtual environment.

Unlike some other libraries, PyQt5 cannot be simply installed on Raspberry Pi virtual environment by:

```
pip install PyQt5
```

It needs to be compiled from source. During my search, I came across following useful posts. They helped me a lot, and this post is just a refined version of them:
https://raspberrypi.stackexchange.com/questions/62939/pyqt5-on-a-raspberry-pi
https://stackoverflow.com/questions/51693993/pyqt5-error-pycapsule-getpointer-called-with-incorrect-name

Installation process

First of all, it is very easy to "spoil" overall setup by trying to install different versions of PyQt5 and SIP over and over. The best way to avoid frustration is to take SD card backup before

installing a new library and just start from scratch if it didn't run well.

For compatibility and troubleshooting easiness, it is good to have the same versions of PyQt5 on Raspberry Pi and on your laptop/desktop, so check PyQt5 version there:

```
from PyQt5.Qt import PYQT_VERSION_STR
print("PyQt version:", PYQT_VERSION_STR)
```

riverbankcomputing.com resource contains all previous PyQt5 version sources, you need to download right one. It also provides SIP sources, and you have to download the latest one. When two tar archives are downloaded, place them where you like on your Raspberry Pi SD card. If you use Windows on your laptop/desktop, you probably find Paragon Software convenient for accessing Linux file system on SD card.

From terminal on Raspberry Pi go to the directory where achieves are located:

```
cd /home/pi
```

Unzip the archives using:

```
tar -xzvf PyQt5_gpl-5.12.3.tar.gz
tar -xzvf sip-4.19.14.tar.gz
```

Go to virtual environment, in my case it is "cv":

```
workon cv
```

Install QT Core:

```
sudo apt-get install qt5-default
```

Configure SIP:

```
cd /home/pi/sip-4.19.14
python configure.py --sip-module PyQt5.sip
```

Build and install SIP make:

```
make
sudo make install
```

Configure PyQt5:

```
cd /home/pi/PyQt5_gpl-5.12.3
python configure.py
```

Build and install PyQt5 make:

```
make
sudo make install
```

Building PyQt5 make takes 2 hours on Raspberry Pi 3 A+, installing takes another 15 min. SIP takes just few minutes for make and make install.

Troubleshooting

1. *Be aware that OpenCV imshow when used in PyQt5 GUI on Raspberry Pi can result in following error and freeze PyQt5 GUI*:

```
(python3:959): GLib-GObject-WARNING **: 16:34:34.421: cannot register
existing type 'GdkDisplayManager'
(python3:959): GLib-CRITICAL **: 16:34:34.421: g_once_init_leave:
assertion 'result != 0' failed
```

```
(python3:959): GLib-GObject-CRITICAL **: 16:34:34.421:
g_object_new_with_properties: assertion 'G_TYPE_IS_OBJECT
(object_type)' failed
```

Read this post to troubleshoot it. In short, solution is to edit qt5ct.conf file:

```
sudo nano /etc/xdg/qt5ct/qt5ct.conf
```

remove:

```
style=gtk2
```

and add:

```
style=gtk3
```

2. *Use of cv2.imshow() and of some other OpenCV functions in a thread other than main is not possible.* For example, below class to display video stream will not work in a thread started from main application:

```
import cv2
import time
from threading import Thread
from picamera import PiCamera
from picamera.array import PiRGBArray

class video(Thread):
    def __init__(self):
        Thread.__init__(self)

    def run(self):
        camera = PiCamera()
        w = 640
        h = 480
        camera.resolution = (w, h)
        cap = PiRGBArray(camera, size=(w, h))
        time.sleep(1)
        for frame in camera.capture_continuous(cap, format='bgr',
use_video_port = True):
            image = frame.array
            cv2.imshow('Video', image)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
            cap.truncate(0)
        camera.close()
```

Instead, send QImage object as a signal from the thread to the main application:

```
import cv2
import time
from threading import Thread
from picamera import PiCamera
from picamera.array import PiRGBArray
from PyQt5 import QtCore, QtGui

class video_message(QtCore.QObject):
    video_frame_signal = QtCore.pyqtSignal(QtGui.QImage)

class video(Thread):
    def __init__(self, _video_message):
        Thread.__init__(self)
```

```
            self._video_message = _video_message

        def run(self):
            camera = PiCamera()
            w = 640
            h = 480
            camera.resolution = (w, h)
            cap = PiRGBArray(camera, size=(w, h))
            time.sleep(1)
            for frame in camera.capture_continuous(cap, format='bgr',
    use_video_port = True):
                image = frame.array
                Q_Image = QtGui.QImage(image.data, image.shape[1],
    image.shape[0], QtGui.QImage.Format_RGB888).rgbSwapped()
                self._video_message.video_frame_signal.emit(Q_Image)
                cap.truncate(0)
            camera.close()
```

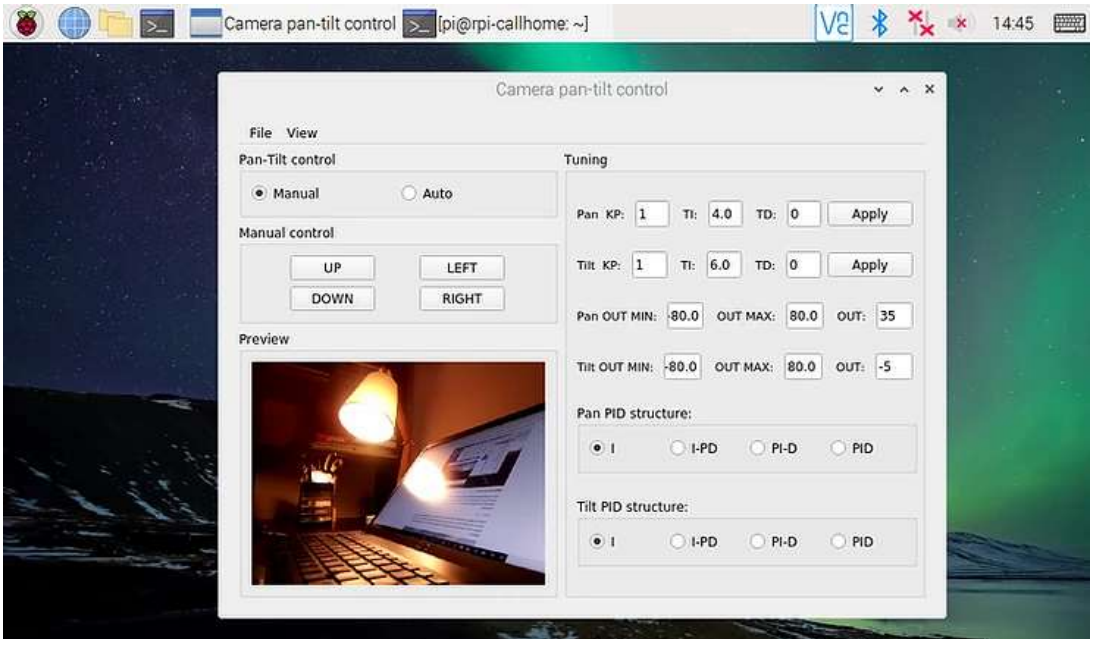
For the main application to be able to receive the messages, the class should be called in the following way:

```
import video_class # this is py file where the class is saved
...
_video_message = video_class.video_message()
_video_thread = video_class.video(_video_message)
_video_message.video_frame_signal.connect(self.update_video_frame)
_video_thread.start()
```

Finally, define a function which updates image in QtWidgets.QLabel widget:

```
def update_video_frame(self, Q_image):
        convertToQtFormat = QtGui.QPixmap.fromImage(Q_image)
        convertToPixmap = QtGui.QPixmap(convertToQtFormat)
        self.VideoFrame.setPixmap(convertToPixmap.scaled(w, h))
```

Both self.update_video_frame function and self.VideoFrame widget belong to FormWidget class of the main application.



For reference:

Hardware setup:

Raspberry Pi 3A+

Software:

Raspbian Buster full, ref. date 2020-02-14
Python 3.7.3
OpenCV 4.1.1
PyQt5 5.12.3
Sip 4.19.14

#PyQt5 #RaspberryPi #OpenCV

Recent Posts

See All

Setup right click for Raspberry Pi Waveshare …

4 Write a comment

Log in to leave a comment.