Sadaan Tahir - 25100224

Integration of LANKA® Full HD 1080P Car Dash Cam on Ubuntu 20.04 with ROS Noetic on Odroid XU4.

Step 1. Connect camera to **Odroid XU4** via USB and run the following command to check if it is recognised:

```
ls /dev/video*
```

Step 2. Check the video device details by running the following command to list video device capabilities:

```
v4l2-ctl --list-devices
```

You should get a terminal output like this:

```
root@odroid:~# v4l2-ctl --list-devices

GENERAL - UVC : GENERAL - UVC (usb-xhci-hcd.8.auto-1.2):
      /dev/video0
      /dev/video1
      /dev/media0
```

Step 3. Run this command to see what video formats the camera supports:

```
v4l2-ctl --device=/dev/video0 --list-formats
```

You should get an output like this:

```
root@odroid:~# v4l2-ctl --device=/dev/video0 --list-formats

ioctl: VIDIOC_ENUM_FMT
    Type: Video Capture

    [0]: 'MJPG' (Motion-JPEG, compressed)
```

Step 4. Install **FFmpeg** (includes **ffplay**) using the following commands:

```
sudo apt update
sudo apt install ffmpeg -y
```

FFMpeg
Step 5. With installed, run the following command to view the video stream:

```
 ffplay -f v4l2 -input_format mjpeg -i /dev/video0
```

Note: *The steps below are optional and can be used to set up a ROS Node for the Dash Cam.*

Sadaan Tahir - 25100224

Step 6. Make sure that **OpenCV** and **cv_bridge** is installed:

```
sudo apt update
sudo apt install ros-noetic-cv-bridge ros-noetic-image-transport
python3-opencv -y
```

Step 7. Navigate to your ROS workspace (assuming its **catkin_ws**) and create a package specifically for the Dash Cam:

```
 cd ~/catkin_ws/src
catkin_create_pkg dashcam_publisher rospy std_msgs sensor_msgs cv_bridge
image_transport
cd ~/catkin_ws
catkin_make
```

Step 8. Inside the package, create a **scripts** folder and create a python node:

```
 cd ~/catkin_ws/src/dashcam_publisher
mkdir scripts
touch scripts/dashcam_publisher.py
chmod +x scripts/dashcam_publisher.py
```

Step 9. Now edit the **dashcam_publisher.py** file:

```
#!/usr/bin/env python3

import cv2
import rospy
from sensor_msgs.msg import Image
from cv_bridge import CvBridge

def publish_video():
    rospy.init_node('dashcam_publisher', anonymous=True)
    pub = rospy.Publisher('/dashcam/image_raw', Image, queue_size=10)
    bridge = CvBridge()

    cap = cv2.VideoCapture('/dev/video0', cv2.CAP_V4L2)
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
    cap.set(cv2.CAP_PROP_FPS, 30)

    if not cap.isOpened():
        rospy.logerr("Failed to open camera")
        return

    rospy.loginfo("Publishing video stream from dashcam...")
    rate = rospy.Rate(30)  # 30 FPS
    while not rospy.is_shutdown():
        ret, frame = cap.read()
        if not ret:
            rospy.logerr("Failed to capture frame")
            break

        msg = bridge.cv2_to_imgmsg(frame, encoding="bgr8")
        pub.publish(msg)
        rate.sleep()

    cap.release()

if __name__ == '__main__':
    try:
        publish_video()
    except rospy.ROSInterruptException:
        pass
```

Step 10. Modify **CMakeLists.txt** as shown:

```
find_package(catkin REQUIRED COMPONENTS
  rospy
  std_msgs
  sensor_msgs
  cv_bridge
  image_transport
)

catkin_package()
```

Step 11. Build the workspace again and run the ROS node:

```
cd ~/catkin_ws
catkin_make
source devel/setup.bash
rosrun dashcam_publisher dashcam_publisher.py
```

Step 12. Lastly, verify that the ROS node is working:

```
rostopic list
```