How to Guide on Using the Logitech C270 Camera and the BeagleY-AI to Take Pictures

Overview:

This guide walks you through the steps needed to be able to take pictures using a Logitech C270 camera connected to the BeagleY-AI. First, this guide will show you how to connect and set up your Logitech C270 and BeagleY-AI to enable you to take photos. Second, this guide will show you how to create and cross-compile, from your host to the BeagleY-AI, a simple C program that lets you take photos. Third, this guide will show you how to modify the C file of this program to be able integrate it into the same executable with any of your projects.

Hardware Required:

- BeagleY-AI
- Logitech C270 Webcam
- Host running on Linux (Debian 12)

Step 1: Connecting and Setting up Your Camera and BeagleY-AI

a. Plug your Logitech C270 webcam to any of the USB-A ports on your BeagleY-AI.

b. Check that your BeagleY-AI recognizes your Logitech C270 Webcam using the following commands, ensuring that your BeagleY-AI produces the same outputs [1].

i. `(byai)$ lsusb`

```
jostin@jae6-beagle:~$ lsusb
Bus 002 Device 002: ID 0451:8140 Texas Instruments, Inc. TUSB8041 4-Port Hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 002: ID 0451:8142 Texas Instruments, Inc. TUSB8041 4-Port Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

ii. `(byai)$ ls/dev/video3`

```
jostin@jae6-beagle:~$ ls /dev/video3
/dev/video3
```

Troubleshooting:

If your BeagleY-AI does not recognize your Logitech C270, try these steps in the given order, running the commands in **(b)** after each step.

I. Test that your webcam works on another computer
II. Plug in your webcam to another USB-A port on the BeagleY-AI
III. Reboot your BeagleY-AI
IV. Remove other peripherals or run the BeagleY-AI off the wall outlet for more power
V. Check if your USB-A ports are working by plugging in another peripheral and checking if your BeagleY-AI recognizes this new peripheral
VI. Replace your webcam

Step 2: Creating C Program to Take Pictures and Cross-compiling it

a. Clone Derek Molloy's BoneCV repository https://github.com/derekmolloy/boneCV. We will be using and modifying the grabber.c file later on in this step [1].
   i. `(host)$ git clone https://github.com/derekmolloy/boneCV.git`
b. Install libv4l-dev on your BeagleY-AI [1]
   i. `(byai)$ sudo apt install libv4l-dev`
c. Install libv4l-dev:arm64 on your host for cross-compiling
   i. `(host)$ sudo apt install libv4l-dev:arm64`
d. Create a new directory called grabber on your host's work directory and copy the grabber.c file from the cloned BoneCV repository.
e. Modify the following lines
   i. Line 60
      Before: `char *dev_name = "/dev/video0";`
      After: `char *dev_name = "/dev/video3";`
   ii. Line 73, 74
      Before:
      `fmt.fmt.pix.width = 1920;`

```
fmt.fmt.pix.height = 1080;
```

After (Highest resolution that the camera supports):

```
fmt.fmt.pix.width = 1280;
fmt.fmt.pix.height = 960;
```

f. Cross-compile code to your host's NFS directory using the following command

```
(host)$ aarch64-linux-gnu-gcc grabber.c -lv4l2 -o
~/cmpt433/public/myApps/grabber
```

g. Navigate to your remote NFS directory on your BeagleY-AI and run the program to take pictures! This will capture 20 photos (in .ppm format) and save them onto your NFS directory.

```
(byai)$ ./grabber
```

Step 3: Incorporating This C Program Into any Project

a. Copy the grabber directory from step 2 into your project
b. Create a header file grabber.h and a CMake file in your grabber directory
   i. Your project file structure should look similar to this

```
|-Project
      |-app
            |- ...
      |-hal
            |- ...
      |-grabber
            |-grabber.c
            |-grabber.h
            |-CMakeLists.txt
      |-CMakeLists.txt
      |- ...
```

c. In grabber.c replace the main function with a thread

   Before: `int main(int argc, char **argv)`

   After: `static void* grabber_thread()`

   Ensure that the new function returns NULL

d. In grabber.c create a function that creates and detaches from this thread. Call this function to take pictures
e. In grabber.h create a function prototype for the function in **(d)**
f. Add the following lines into your CMakefiles
   i. In the CMake file in the app directory add:

```
target_link_libraries(<Your project name> LINK_PRIVATE v4l2)
target_link_libraries(<Your project name> LINK_PRIVATE grabber)
```

ii. In the CMake file in the grabber directory add:

```
add_library(grabber STATIC grabber.c)

# Remove extra compiler options that were set in base.
# Ref:
https://discourse.cmake.org/t/how-to-disable-pedantic-compiler-opti
on-for-a-specific-library/1575
get_target_property(target_options grabber COMPILE_OPTIONS)
list(REMOVE_ITEM target_options "-Wpedantic")
list(REMOVE_ITEM target_options "-Wextra")
list(REMOVE_ITEM target_options "-Werror")
list(REMOVE_ITEM target_options "-Wall")
set_property(TARGET grabber PROPERTY COMPILE_OPTIONS
${target_options})

# Setup the include paths for where modules that use this library
will search.
target_include_directories(grabber PUBLIC .)
include_directories(grabber)

add_compile_options(-D_BSD_SOURCE)
```

iii. In the CMake file in the project directory add:

```
add_subdirectory(grabber)
```

g. Clear your project's build cache and build

You should now be able to take pictures when the function defined in grabber.h is called.

More Information

For more information on the grabber code or the BoneCV repository in general check out this video by Derek Molloy (the author of this repository) https://www.youtube.com/watch?v=8QouvYMfmQo.[2]

References

[1] J. Vazquez, M. Fraser, C. Rossiter, and D. Tufail, “Recording Webcam Videos with the BeagleBone Black”.

[2] Derek Molloy, *Beaglebone: Video Capture and Image Processing on Embedded Linux using OpenCV*, (May 25, 2013). Accessed: Apr. 08, 2025. [Online Video]. Available: https://www.youtube.com/watch?v=8QouvYMfmQo