TIMELAPSE WORKSHOP

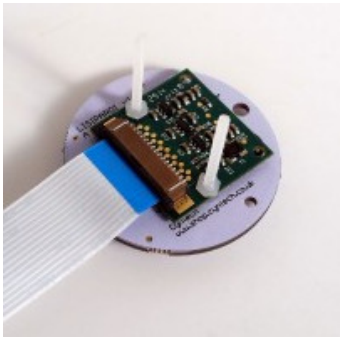
This workshop will guide you through setting up a Lisiparoi InfraRed LED module with a Raspberry NoIR Camera module and connecting them up to your Pi. You will then work through the Python code required to create a nightvision infrared timelapse animation.

Lisiparoi Installation

When you purchase a Lisiparoi LED module you need to solder a set of pins to it before you can use it. The modules you are using today have already been soldered. Full instructions can be found on their website here: http://www.lisiparoi.com/how-to-use/

Attaching to the camera module

1. Push the bolts through the LISIPAROI and screw the nut on to the back
2. Add the camera module, and screw the remaining 2 nuts to hold it all in place



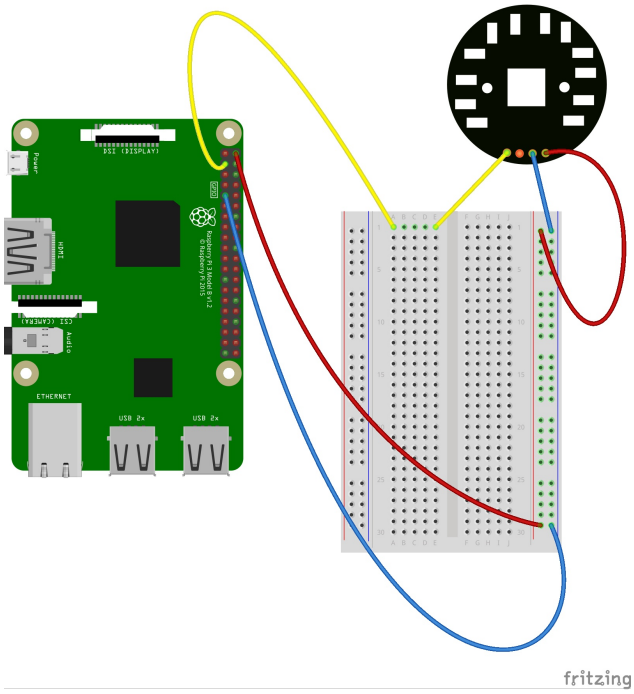
STATIC WARNING!

The camera is a static sensitive component. Try to avoid touching the circuit board itself (although holding it at the edges is fine). Also, whilst the camera is active, try to avoid letting the camera board come into contact with the Pi.

Wiring up to the Raspberry Pi

If you are using the Lisiparoi Kit then you can cable directly to the pins on the Raspberry Pi as it comes with female to female jumper leads. Otherwise you will need to make use of a breadboard to connect the LED module to your Raspberry Pi. Connect your LED module to your Raspberry Pi as follows:

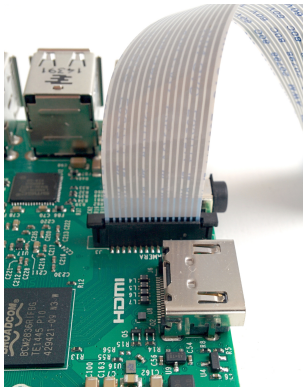
1. Connect the 5v on the Lisiparoi to the + rail on the breadboard
2. Connect GND on the Lisiparoi to the - rail on the breadboard
3. Connect GPIO on the Lisiparoi to row 1, column e on the breadboard
4. Connect row 1, column a on the breadboard to the second pin down on the inside row on the Raspberry Pi (Pin 2 GPIO2)
5. Connect the other end of the + rail on the breadboard to the first pin down on the outside row of the Raspberry Pi (Pin 40 5V)
6. Connect the other end of the - rail on the breadboard to fifth pin down on the inside row of the Raspberry Pi (Pin 5 Ground)



TIMELAPSE WORKSHOP

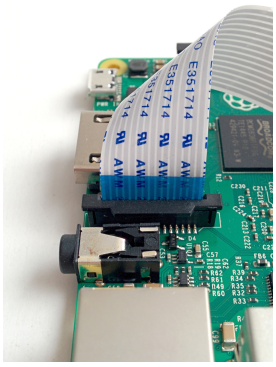
Camera Installation

Now we have set up the LED module we need to connect our camera module too. Carry the next set of steps out with your Raspberry Pi still switched off!!



Attaching the camera

1. The camera port on the Raspberry Pi is the thin black port next to the Ethernet port
2. Lift the tab on the top of connector
3. Place the camera's connecting strip in the connector with the blue side facing the Ethernet port
4. Ensuring the strip stays in the connector, push the tab back down



Activating the camera

Switch your Raspberry Pi on now!
You may find that this has already been done on your Raspberry Pi, but it is always worth following these steps if you are unsure:

1. Open the menu and open "Accessories > Terminal"
2. In the window which appears enter the command sudo raspi-config
3. In the menu system which opens, move down to Enable camera, using your cursor keys and press Enter
4. Move right to highlight Finish and press Enter
5. Select Yes to reboot the Pi

Testing the camera and LED Module

Now we have installed both pieces of equipment, we need to test that they are working and we can control them. Open the terminal again and this time type the command raspistill -o test.jpg
If everything is working correctly you will see a preview of the camera view on your screen and after a short delay the camera should capture its view and save it as test.jpg in your home directory.

Capturing an image

Now we are going to write some code to control both our LED and camera. We are going to use the python programming language for this. From the menu open "Programming > Python 3 (IDLE)". In the window which opens, select File > New Window. Save this new file as camera-test.py and then enter this code.

Press the F5 button on your keyboard to run your code.

```
from time import sleep
from gpiozero import LED
from picamera import PiCamera

cam = PiCamera()
led = LED(2)
cam.start_preview(alpha=192)
led.on()
sleep(5)
cam.capture('/home/pi/Desktop/image.jpg')
led.off()
cam.stop_preview()
```

TIMELAPSE WORKSHOP

Taking multiple images

If we want to create a timelapse animation we need to take a series of images of a period of time. The easiest way to do this is to use a loop. Consider the following code:

```
from time import sleep
from gpiozero import LED
from picamera import PiCamera

cam = PiCamera()
led = LED(2)
led.on()
for i in range(10):
    cam.capture('/home/pi/Desktop/image.jpg')
led.off()
```

This code will turn on our LED and then enter the loop. Whilst in the loop it will capture an image called "image.jpg" and repeat this 10 times. Can you see the problem here?

We are overwriting the image each time! To carry out a timelapse we need to have a unique filename each time we go round the loop. There are a number of ways we can do this, but the following method will allow us to easily stitch our images back together in the correct order later on.

Adjust your camera-test.py file so that it is the same as the following code. We are using Python String Formatting here to pass the value from i (the counter for our for loop) into the filename of the photo we are saving. The {0:04d} part tells Python to take the number passed from the format(i) part and add it to the filename string ensuring that it is 4 digits long, adding leading zeros where necessary.

```
from time import sleep
from gpiozero import LED
from picamera import PiCamera

cam = PiCamera()
led = LED(2)
led.on()
for i in range(10):
    cam.capture('/home/pi/Desktop/image{0:04d}.jpg'.format(i))
led.off()
```

Press the F5 button to run your code.

You should now find ten images saved to your Desktop named image0000.jpg through to image0009.jpg

Making a GIF

Now we can take multiple photos we need a way to turn those pictures into an animated GIF sequence. To do this we are going to make use of the ImageMagick program. You won't need to install this at the Jam as we have done it already, but to install it at home you would type the following two commands into the terminal:

```
sudo apt-get update
sudo apt-get install imagemagick -y
```

ImageMagick is a command line program which lets us manipulate images. It can do a great many things, but there is one particular function we are interested in today: creating GIF animations.

To make use of a command line program inside our Python code we will make use of the os library.

TIMELAPSE WORKSHOP

Edit your camera-test.py file to match the following code:

```
from time import sleep
from gpiozero import LED
from picamera import PiCamera
from os import system

cam = PiCamera()
led = LED(2)
led.on()
for i in range(10):
    cam.capture('/home/pi/Desktop/image{0:04d}.jpg'.format(i))
led.off()

system('convert -delay 10 -loop 0 /home/pi/Desktop/image*.jpg  \
/home/pi/Desktop/animation.gif')
print("Done!")
```

Press F5 to run your code again and after some time you should see "Done" returned in the Python shell.

This will most likely take some time and you will find that your animation.gif file is very large (around 10mb).

There are two more things to do to our code; we need to make the file size smaller and we also need to add a pause between each image being taken.

To reduce the file size we are going to set the camera resolution in our code and we will make use of the Python sleep function. Adjust your code to match this:

```
from time import sleep
from gpiozero import LED
from picamera import PiCamera
from os import system

cam = PiCamera()
cam.resolution = (1024, 768)
led = LED(2)
led.on()
for i in range(10):
    cam.capture('/home/pi/Desktop/image{0:04d}.jpg'.format(i))
    sleep(60)
led.off()

system('convert -delay 10 -loop 0 /home/pi/Desktop/image*.jpg \
/home/pi/Desktop/animation.gif')
print("Done!")
```

Here we have reduced the camera resolution to 1024 x 768 pixels and we have added a 60 second pause between each picture being taken.

Run your code again by pressing F5 and check everything works!

Our Secret Project

For our top secret project things need to be a little different!

We don't really want to reduce the file size (so the final animation can be as high quality as possible) and there is little point having the Pi make the animation automatically; this can be done later on a more powerful computer. However we do need to think about a few things:

1. How long does it need to run for?
2. How many images will it take?
3. How much storage space will this require?