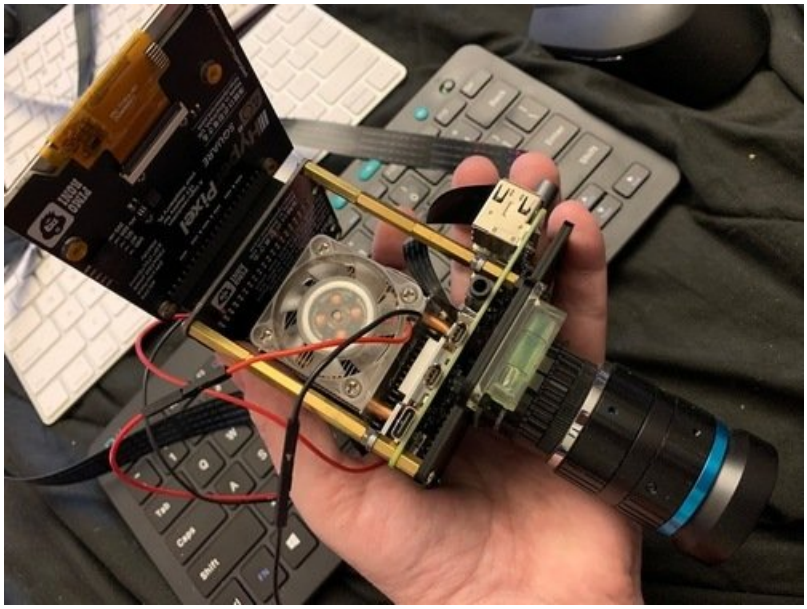
adafruit learning system

Raspberry Pi Low-Light Long-Exposure Photography

Created by Philip Moyer



Last updated on 2021-05-18 05:20:47 PM EDT

Guide Contents

Overview

Have you ever wished you could take pictures at night with the sweet Raspberry Pi High Quality (HQ) Camera?

Using the *raspistill* command line program is tricky because the auto setting determination usually results in a black picture.

Now, though, with a little Python, you can take pictures up to six seconds long at ISO 800! That'll turn a dark night lit only by a few house lights into a bright, well-saturated picture!

This image was taken at one in the morning with a Pi4 and HQ Camera with 6mm lens.

There are a wide range of options you can use to implement this. I wanted a compact, portable system to do astrophotography, so I built the system specifically to do that. If you want, you can just use the code with an HQ Camera setup you already have. It's totally up to you how sophisticated you want to be. (For example, I don't go into motorized star tracker equatorial mounts even though I'm doing astrophotography.) The mounting solution is up to you, again, depending on your use cases!

Parts

Raspberry Pi 4 Model B - 4 GB RAM

The Raspberry Pi 4 Model B is the newest Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 4 better...

$55.00

In Stock

Add to Cart

Raspberry Pi High Quality HQ Camera

Snap, snap! There's a new official camera board released by the Raspberry Pi Foundation!The Raspberry Pi High Quality Camera is the latest camera accessory...

$50.00

In Stock

Add to Cart

16mm 10MP Telephoto Lens for Raspberry Pi HQ Camera

This is the 16mm 10MP Telephoto Lens, an essential for the Raspberry Pi High Quality Camera. It has a wide...

$50.00

In Stock

Add to Cart

Pimoroni Mini Black HAT Hack3r - Fully Assembled

A nifty little breakout from our friends at Pimoroni that lets you access all of the GPIO pins while also running a

$14.95

In Stock

Add to Cart

Pimoroni HyperPixel 4.0 Square - Hi-Res Display for Raspberry Pi

It's hip to be square with the Pimoroni's HyperPixel 4.0 Square Display for Raspberry Pi computers. This display has all the great features of their

Out of Stock

Out of Stock

---

1 x 10000 mAh Li-Ion Battery Pack

USB Battery Pack for Raspberry Pi - 10000 mAh

Add to Cart

---

1 x USB A to USB C Cable

USB Type A to Type C Cable - approx 1 meter / 3 ft long

Add to Cart

---

1 x Stacking GPIO Header

Stacking 2x20 Extra Tall Header

Add to Cart

---

1 x Keyboard

Official Raspberry Pi Keyboard

Add to Cart

---

1 x Mouse

Official Raspberry Pi Mouse

Add to Cart

---

1 x Male/Female Jumper Wires

Premium Male-Female Jumper Wires, 3 inches

Add to Cart

---

1 x ICE Tower Cooling System

GeeekPi Raspberry Pi 4 Cooling Tower

Buy Now

---

1 x M2.5 Brass Standoff Kit

M2.5 Brass Standoff Kit

Buy Now

---

1 x Raspberry Pi Mounting Plate Pro

Raspberry Pi Mounting Plate for High Quality Camera

Buy Now

---

1 x 128GB Flash Drive

Samsung 128GB Flash Drive

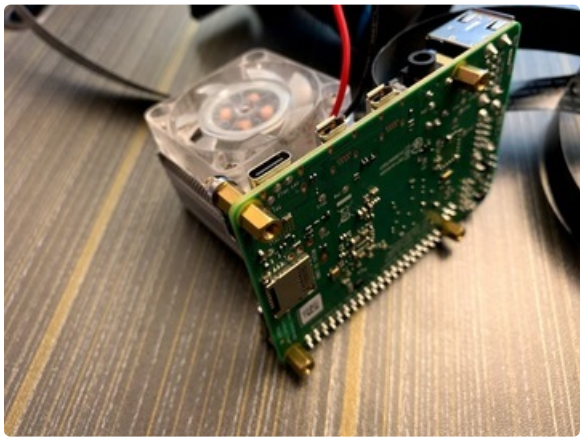
Buy Now

Assembly

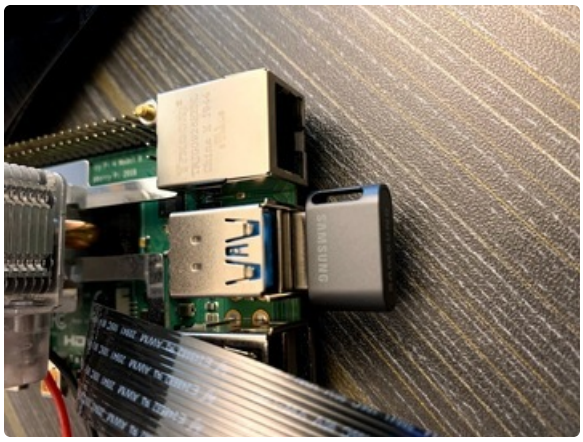
Collect all the parts in one place, and it's time to start the build. (You'll notice some of the parts in the picture are already partially assembled.)

First, attach the included ribbon cable to the HQ Camera, or use one you have handy. Then attach the HQ Camera to the HQ Camera Mount Pro. You can put the lens on now, or wait a while, whichever you prefer. It doesn't get in the way. The official HQ Camera Mount Pro assembly instructions are available here (https://adafru.it/Sha) (they're helpful for finding the right bolts to use).

Attach the camera ribbon cable to the Pi 4. Remember, the silver connectors face *towards* the HDMI ports on the Pi.

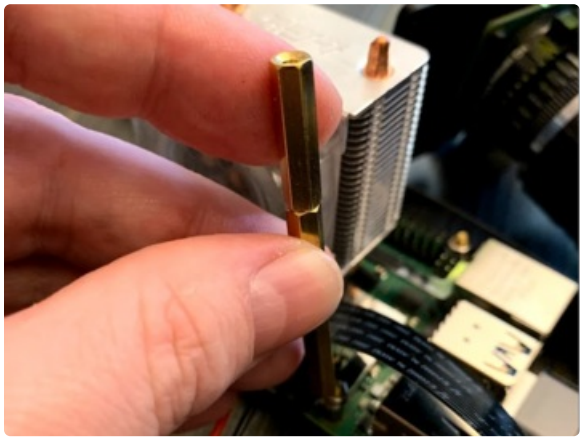


Attach the ICE Tower to the Raspberry Pi 4 following the enclosed instruction booklet. Leave off the lower base plate, though, since that's where we'll attach the camera mount.



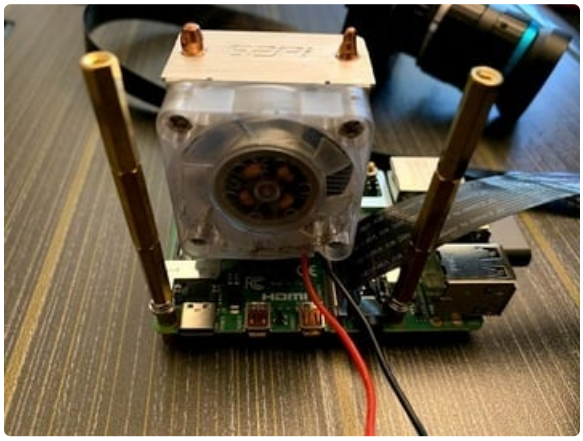
Now take a break and use the Raspberry Pi Imager on a separate computer to burn the Raspberry Pi OS onto the flash drive. I recommend using the 32-bit release. The Pi 4 now supports booting from USB, so this will work with new Pi 4s. If you have an older Pi 4 without the updated firmware, you'll need to follow a procedure to update the firmware yourself (not documented here). Spoiler: it's a lot easier to use a new Pi 4.

Once the flash drive is ready, put it in one of the two USB-C ports on the Pi 4. Those are the ones with the blue spacers.

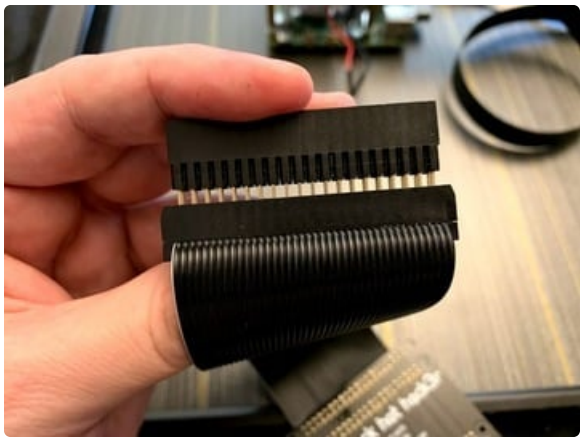


Now that the bits are in place, we can go back to assembling.

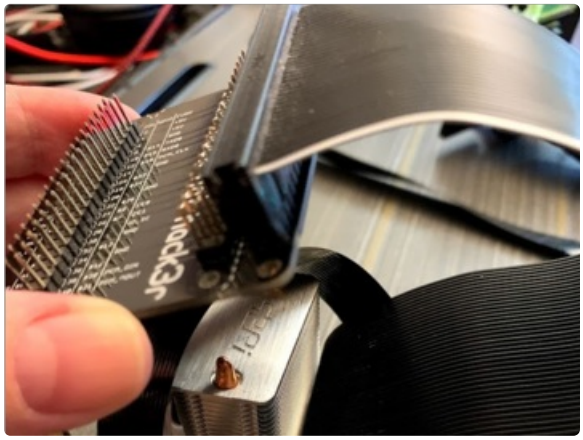
Connect three 20mm brass standoffs together so they are long enough for the Hack3r plate to clear the fan. Make two of these.



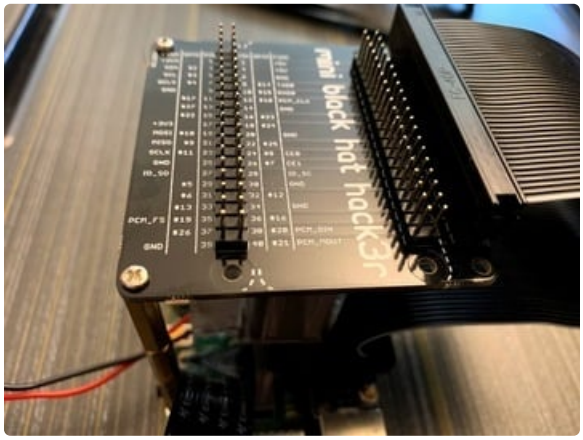
Screw those standoffs onto the two ICE Tower mounting bolts *opposite* the GPIO headers.



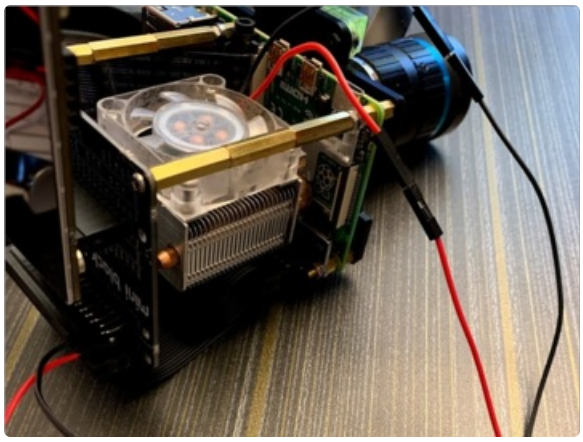
Things are coming together now! Press the tall Pi headers into one end of the Black Hat Hack3r ribbon cable. This provides the clearance to connect the Hack3r to the Pi.



Attach the other end of the cable to the Hack3r plate.



Attach the Hack3r plate to the standoffs in the orientation shown.



Next, attach the Hyperpixel display to the Hack3r panel. It will protrude to the left in the above picture. Be very careful not to press down on the glass. Grasp the PCB at the sides and slowly wiggle it onto the pins. You can see the final position here.

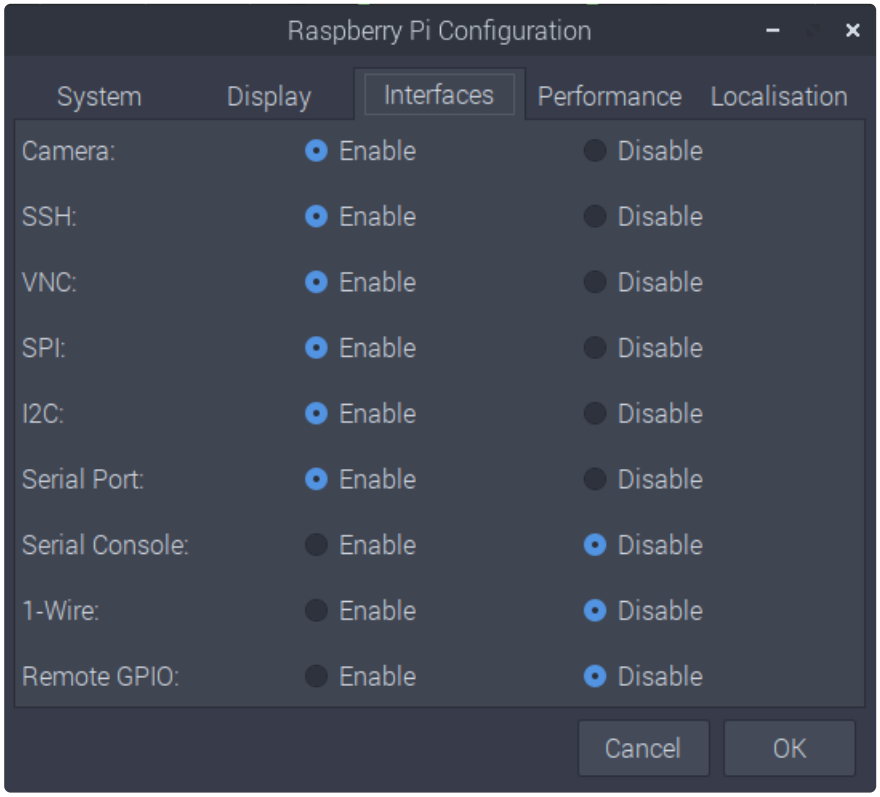
Last assembly step! Use a red and a black wire to extend the fan power connections to the Hack3r plate header row that's still open.

That's it! Now you have a camera that, with the power bank and USB-C cable, is portable to the locations you want to shoot.

Software Configuration

Step one is to configure the operating system (https://adafru.it/Shb) on the Pi. Connect the external monitor, and mouse and keyboard. Then apply power and configure as you normally would for a Raspberry Pi. The Pi should boot automatically from the flash drive.

Follow the on-screen instructions. When you have the install done, run Preferences-> Raspberry Pi Configuration (https://adafru.it/Shc). I recommend changing the Pi's name. Then configure the interfaces. Be sure to enable SPI, I2C, and Camera.



When you click OK you should reboot the Pi.

Step two is to configure the Hyperpixel 4 display. Run this command (https://adafru.it/Shd) as the user pi, not root (i.e., don't use sudo):

```
curl https://get.pimoroni.com/hyperpixel4 | bash
```

When it runs, it will ask for your display. Pick the correct one. If you are following this guide to the letter, you'll want the Pi 4 Square option.

When the script has finished running, it will ask if you want to reboot. Say yes.

Step three is to make the system usable. The display will be upside down when the Pi comes back up, which makes it a bit difficult to use. Fortunately, there is a command that will fix it. You'll only need this once:

```
hyperpixel4-rotate inverse
```

Now the display will be right side up and you can use it like any other Raspberry Pi.

Taking Pictures

Copy the following text to a file called **lowlight.py**. You can click the **Download Project Bundle** link to get a zip file containing the Python file:

```
from picamera import PiCamera
import time
from fractions import Fraction
import datetime

cur_time = datetime.datetime.now()
stub = cur_time.strftime("%Y%m%d%H%M_low")

camera = PiCamera(framerate=Fraction(1,6))

# You can change these as needed. Six seconds (6000000)
# is the max for shutter speed and 800 is the max for ISO.
camera.shutter_speed = 1750000
camera.iso = 800

time.sleep(30)
camera.exposure_mode = 'off'

outfile = "%s.jpg" % (stub)
camera.capture(outfile)

camera.close()
```

Run the program on your py to take a picture. It will save a jpg in the current directory with a timestamp.

$ ./lowlight.py

Adjust the shutter speed to get the exposure you want.

Python Code

Open up your favorite editor and save the following code as **lowlight.py**

First we import the libraries we need. You don't need to install anything special since these libraries are included with Raspberry Pi OS.

```
from picamera import PiCamera
import time
from fractions import Fraction
import datetime
```

Create the timestamp string used to build the output file name.

```
cur_time = datetime.datetime.now()
stub = cur_time.strftime("%Y%m%d%H%M_low")
```

Next, create an instance of the `Camera` object. This is where the magic is, since it allows us to change the default frame rate to 1/6 of a second.

```
camera = PiCamera(framerate=Fraction(1,6))
```

Now we can set the shutter speed and ISO in the `Camera` object instance. The shutter speed is in milliseconds, so for a six second exposure you would use 6000000.

```
# You can change these as needed. Six seconds (6000000)
# is the max for shutter speed and 800 is the max for ISO.
camera.shutter_speed = 1750000
camera.iso = 800
```

Next, we let the camera collect available light to auto-detect other camera settings. We also turn off the exposure mode so the auto-detection won't change our shutter speed and frame rate.

```
time.sleep(30)
camera.exposure_mode = 'off'
```

Now we set the output file name and take a picture!

```
outfile = "%s.jpg" % (stub)
camera.capture(outfile)
```

The last step is to close the `Camera` object so the operating system properly releases the camera

controls.

```
camera.close()
```

Full Example Code

```
from picamera import PiCamera
import time
from fractions import Fraction
import datetime

cur_time = datetime.datetime.now()
stub = cur_time.strftime("%Y%m%d%H%M_low")

camera = PiCamera(framerate=Fraction(1,6))

# You can change these as needed. Six seconds (6000000)
# is the max for shutter speed and 800 is the max for ISO.
camera.shutter_speed = 1750000
camera.iso = 800

time.sleep(30)
camera.exposure_mode = 'off'

outfile = "%s.jpg" % (stub)
camera.capture(outfile)

camera.close()
```

Future Work

There are some things we could do to enhance the project.

1. To enhance portability, we can configure the Raspberry Pi as a WiFi access point. This would create a local wireless network so you can control the system with an iPad with a VNC Client installed.
2. Modify **lowlight.py** so you can specify the shutter speed and ISO with options.
3. Build a small LED light and use it for light painting.
4. Use velcro and piggyback it on a motorized telescope, then do astrophotography.

Use your imagination!