



Capstone Project

Edge computing device programming for AI projects

Lecture 6

Dr. Wilton Fok &

Carol Chen

Recap: Pose estimation

Inference image

- `$ /home/nvidia/jetson-inference/build/armch64/bin/posenet YourImagePath ResultSavePath`

Inference video

- `$ /home/nvidia/jetson-inference/build/aarch64/bin/posenet /dev/video0`

Get the keypoint

load the pose estimation model

- `net = jetson.inference.poseNet(opt.network, sys.argv, opt.threshold)`

perform pose estimation (with overlay)

- `poses = net.Process(img, overlay=opt.overlay)`
- `for pose in poses:`
- `print(pose.Keypoints) #print keypoints id, x, y. (x,y) is the coordinate`

Recap: Use the keypoint to calculate angle

```
def calculate_angle(id1, id2, id3):  
    point1_x = np.array(pose.Keypoints[id1].x)  
    point1_y = np.array(pose.Keypoints[id1].y)  
    point2_x = np.array(pose.Keypoints[id2].x)  
    point2_y = np.array(pose.Keypoints[id2].y)  
    point3_x = np.array(pose.Keypoints[id3].x)  
    point3_y = np.array(pose.Keypoints[id3].y)  
    v1 = (point1_x - point2_x, point1_y - point2_y)  
    v2 = (point3_x - point2_x, point3_y - point2_y)  
    cosine_angle = np.dot(v1, v2) / (np.linalg.norm(v1) * np.linalg.norm(v2))  
    angle = np.arccos(cosine_angle)  
    return np.degrees(angle)
```

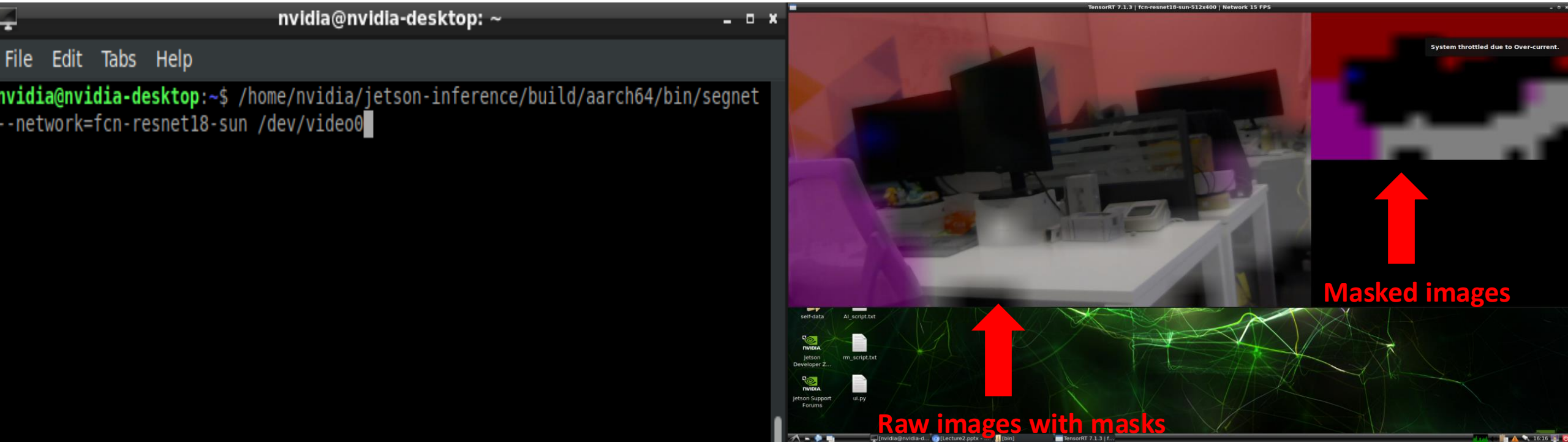
Calculate the angle among 3 keypoints:
Get three keypoints' coordinates, and
calculate the two vectors:

$$\theta = \cos^{-1} [(\mathbf{a} \cdot \mathbf{b}) / (|\mathbf{a}| |\mathbf{b}|)]$$

The python code is:

```
cosine_angle = np.dot(v1, v2) /  
    (np.linalg.norm(v1) * np.linalg.norm(v2))  
angle = np.arccos(cosine_angle)  
return np.degrees(angle)
```

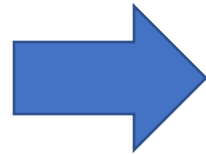
Recap: Object segmentation



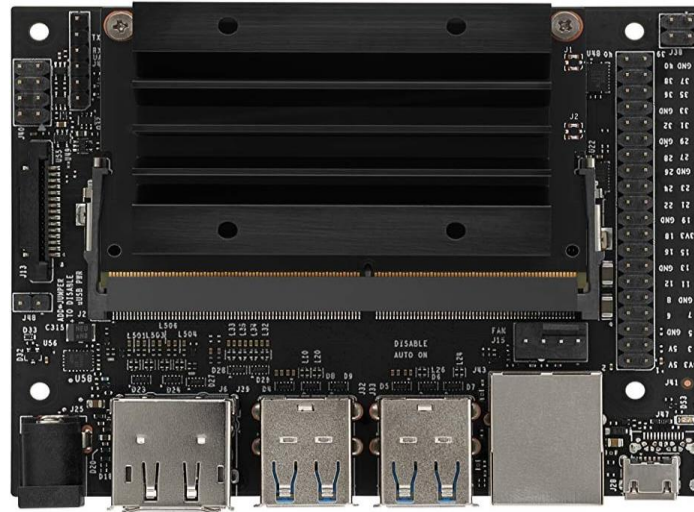
```
$ /home/nvidia/jetson-inference/build/aarch64/bin/segnet --network=fcn-resnet18-voc /dev/video0
```

Computer vision/ Video analytic

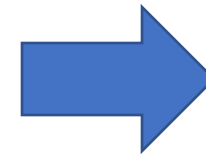
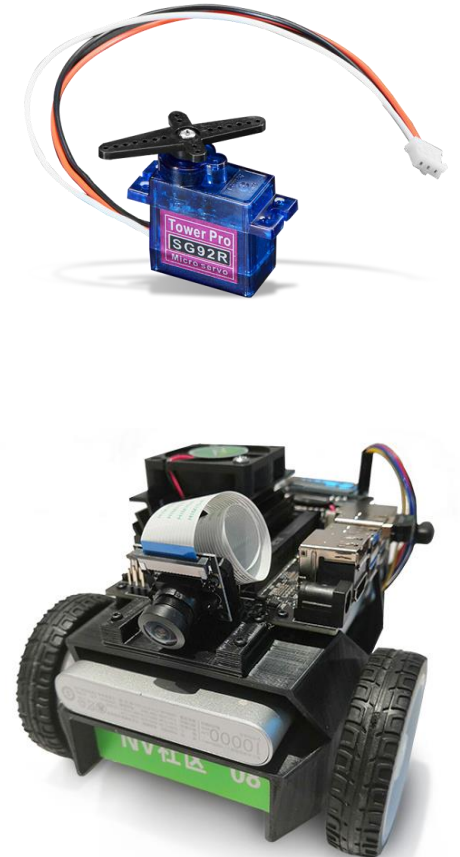
Input



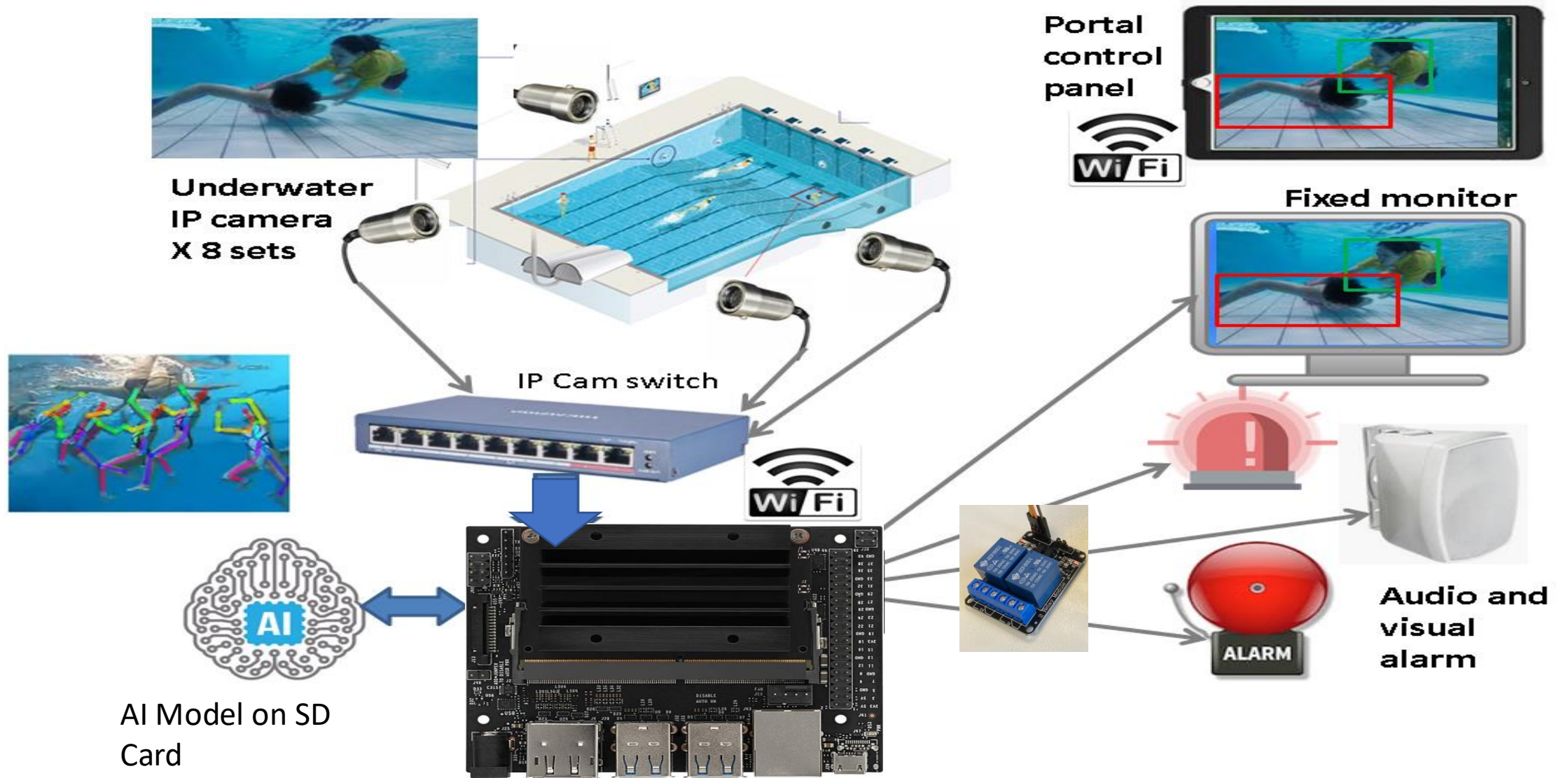
Process



Output

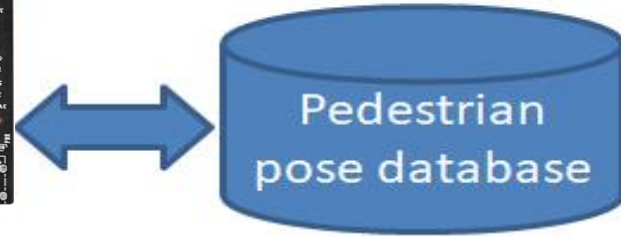
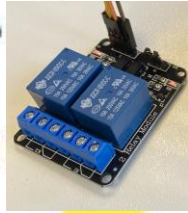


Drowning detection



Smart traffic – safety in bus terminal

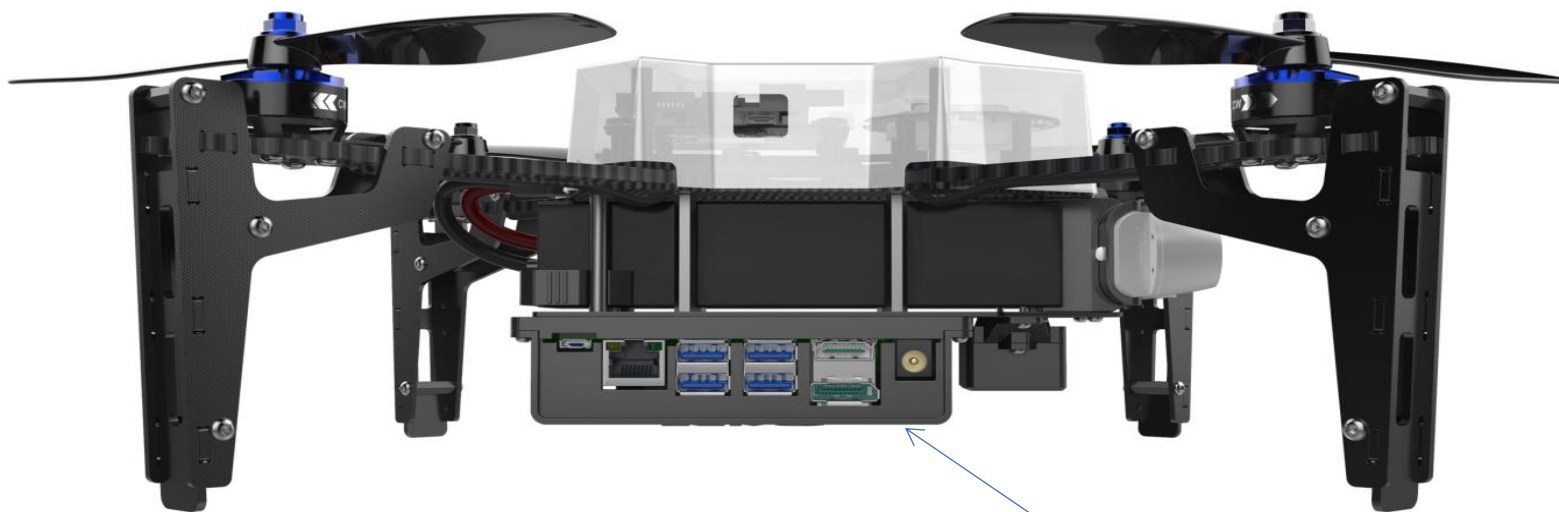
Wireless signal to
bus driver that
pedestrian is in a
area in the bus
terminal



Local alert



Control drone



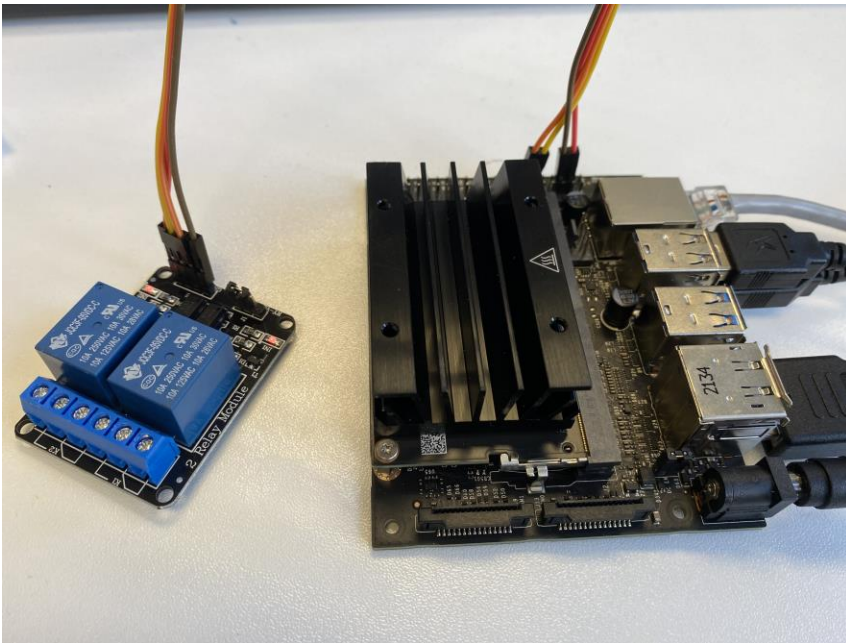
Jetson nano



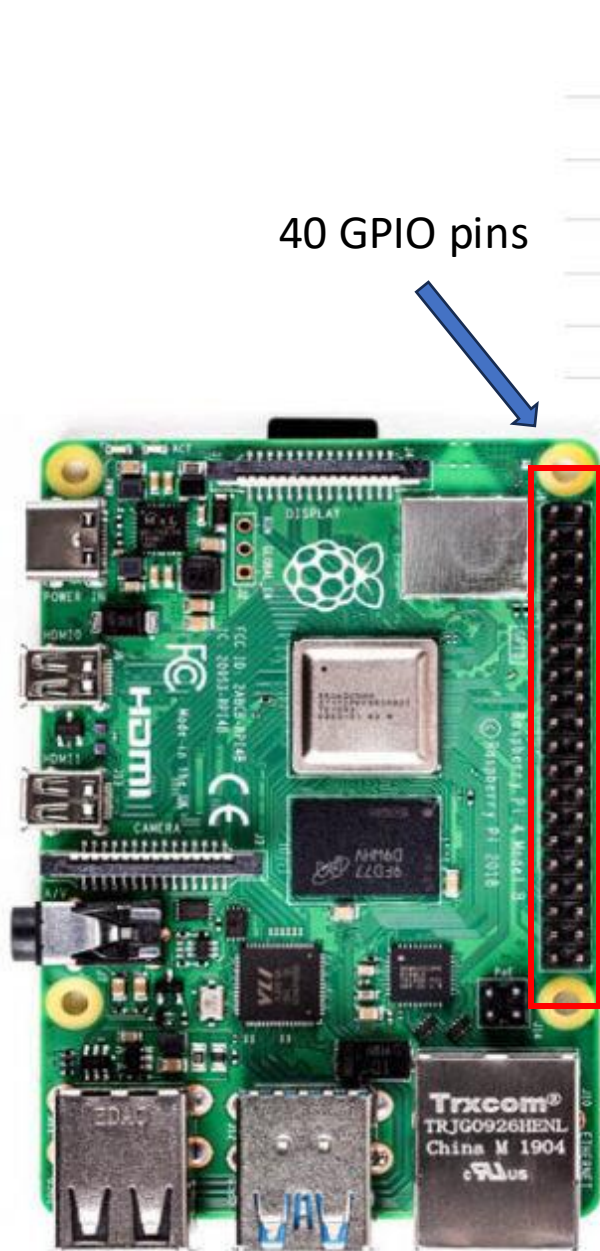
What is Jetson GPIO?

Jetson Nano development boards contain a 40 pin GPIO header, similar to the 40 pin header in the Raspberry Pi. These GPIOs can be controlled for digital input and output using the Python library provided in the Jetson GPIO Library package.

With these GPIO pins, we can control motors, LEDs and other electronic components.



Here is an example
of nano connecting
to a relay module.



Broadcom SOC channel

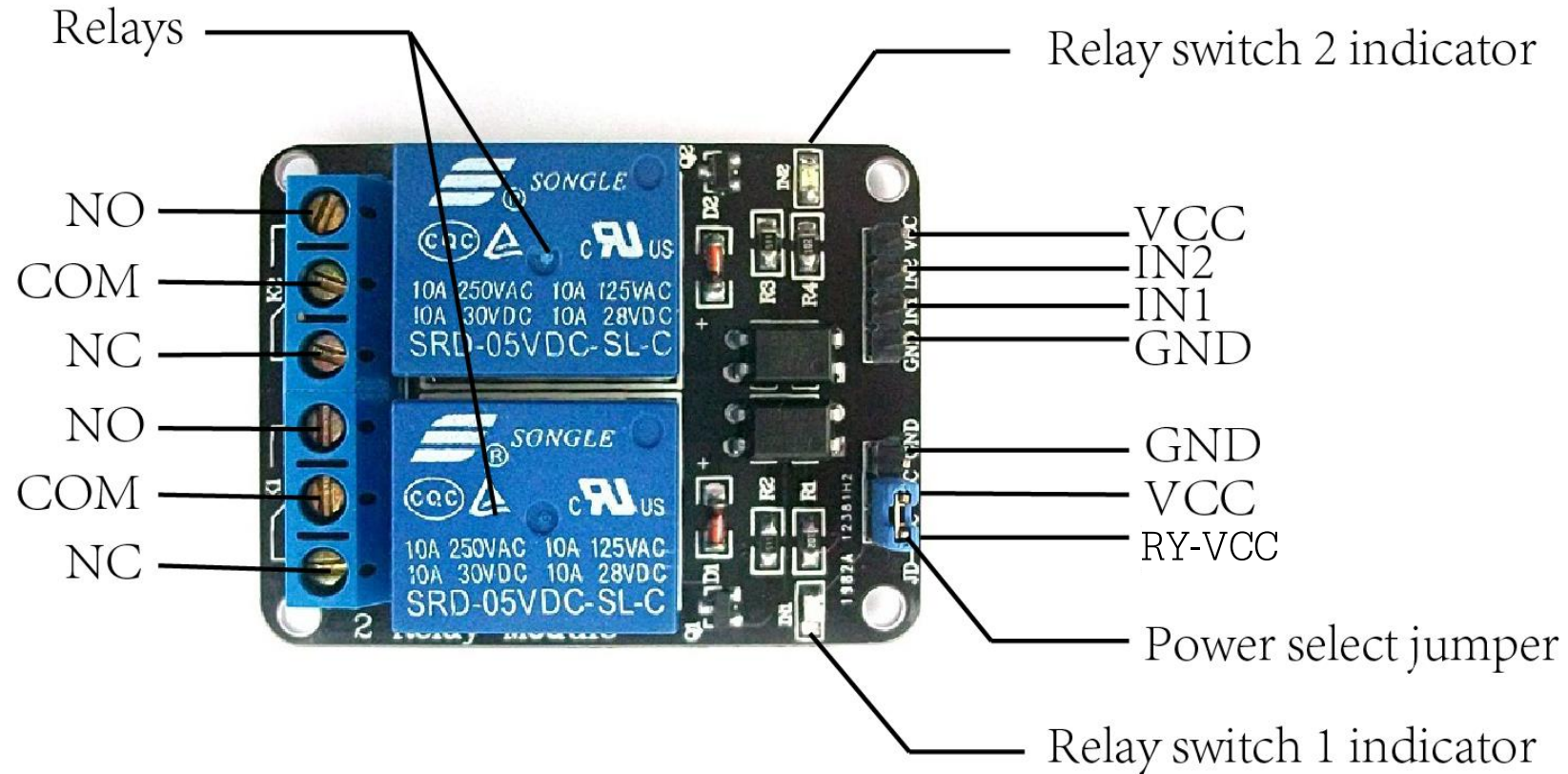
Physical Pins

Function	BCM	pin#	pin#	BCM	Function
3.3 Volts		1	2		5 Volts
GPIO/SDA1 (I2C)	2	3	4		5 Volts
GPIO/SCL1 (I2C)	3	5	6		GND
GPIO/GCLK	4	7	8	14	TX UART/GPIO
GND		9	10	15	RX UART/GPIO
GPIO	17	11	12	18	GPIO
GPIO	27	13	14		GND
GPIO	22	15	16	23	GPIO
3.3 Volts		17	18	24	GPIO
MOSI (SPI)	10	19	20		GND
MISO(SPI)	9	21	22	25	GPIO
SCLK(SPI)	11	23	24	8	CEO_N (SPI)
GND		25	26	7	CE1_N (SPI)
RESERVED		27	28		RESERVED
GPIO	5	29	30		GND
GPIO	6	31	32	12	GPIO
GPIO	13	33	34		GND
GPIO	19	35	36	16	GPIO
GPIO	26	37	38	20	GPIO
GND		39	40	21	GPIO

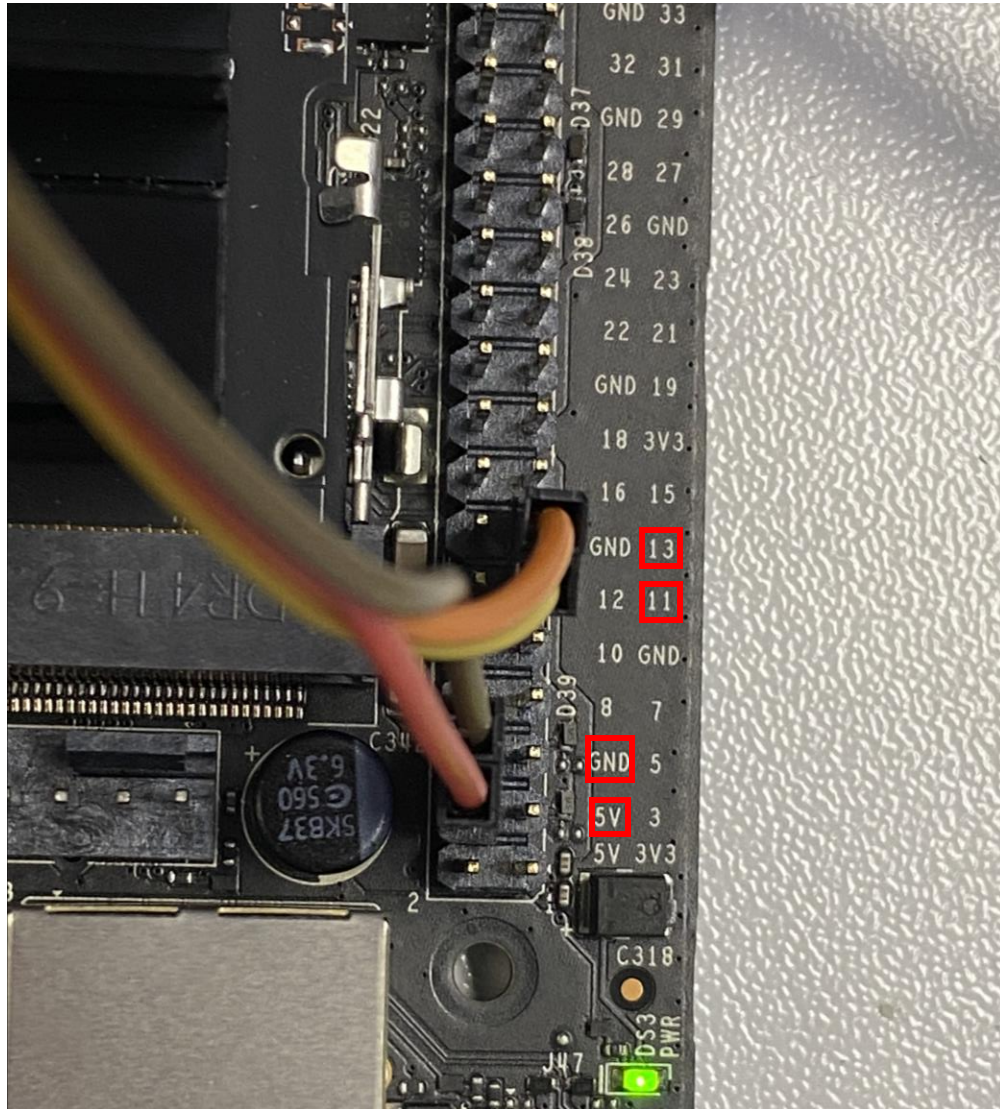
the BCM number is different from the physical pins, because in the initial boards of nano, there was a lesser number of pins. As new boards have launched, more pins are added, and the BCM number remains the same due to which the overall alignment of pin numbers gets disturbed.

In our later application, we will use the BCM number!

Introduction of the relay module



Link the Jetson nano and relay module



Please remember always link the DuPont cables with the power **NOT** linking with nano, all linking pins action must be carrying out in power-off mode.

There are two group DuPont cables, one group has brown, red, orange, yellow cables. The other group has black, white, gray and purple cables.

Link the DuPont cables with nano as follows:

VCC(5V, the last second at left): red or purple cable.

GND(The last third at left): brown or black cable.

11: yellow cable or gray cable.

13: orange cable or white cable.

Link the Jetson nano and relay module

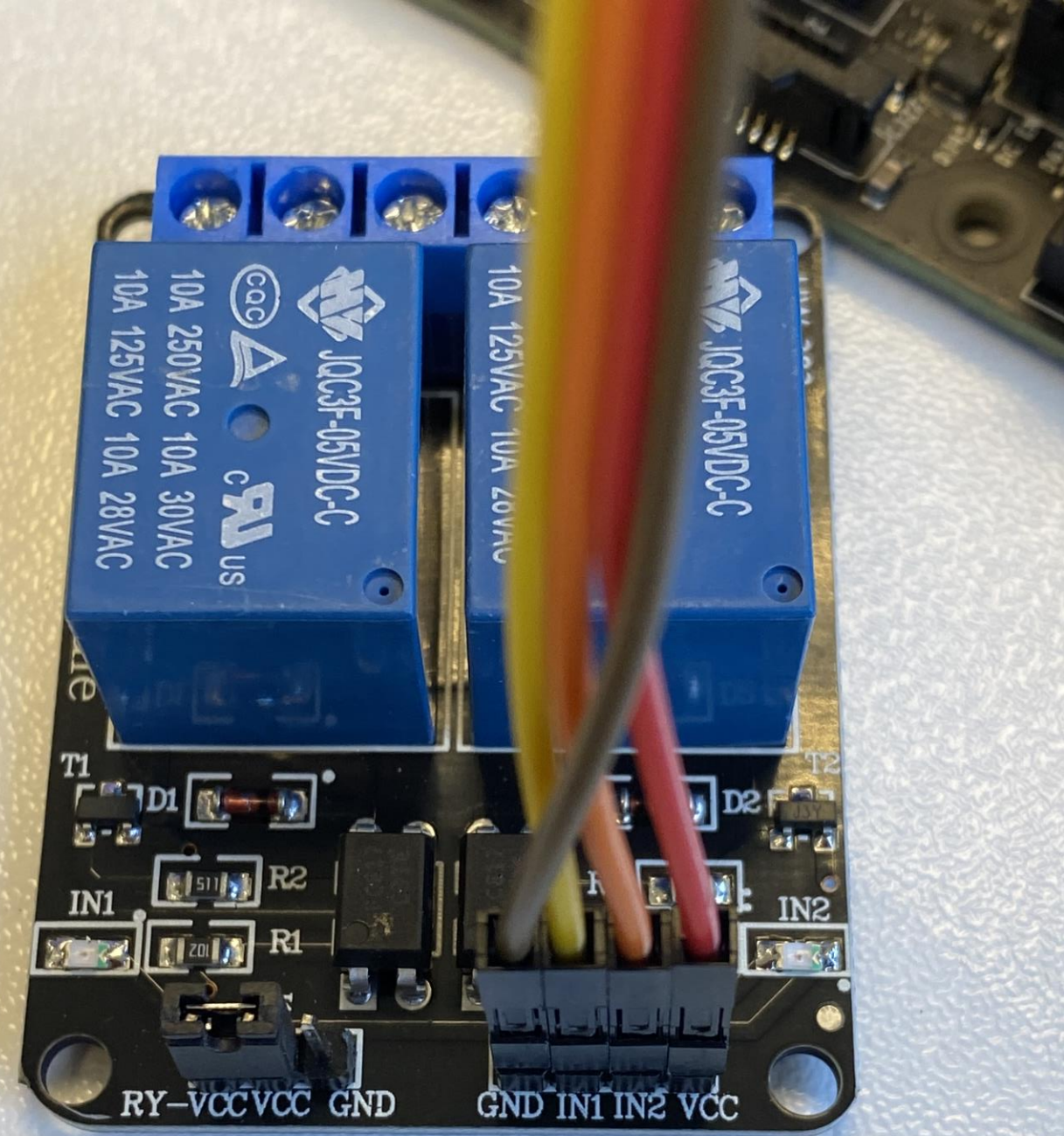
Then link the DuPont cables with relay module:

VCC: red or purple cable.

GND(The last third at left): brown or black cable.

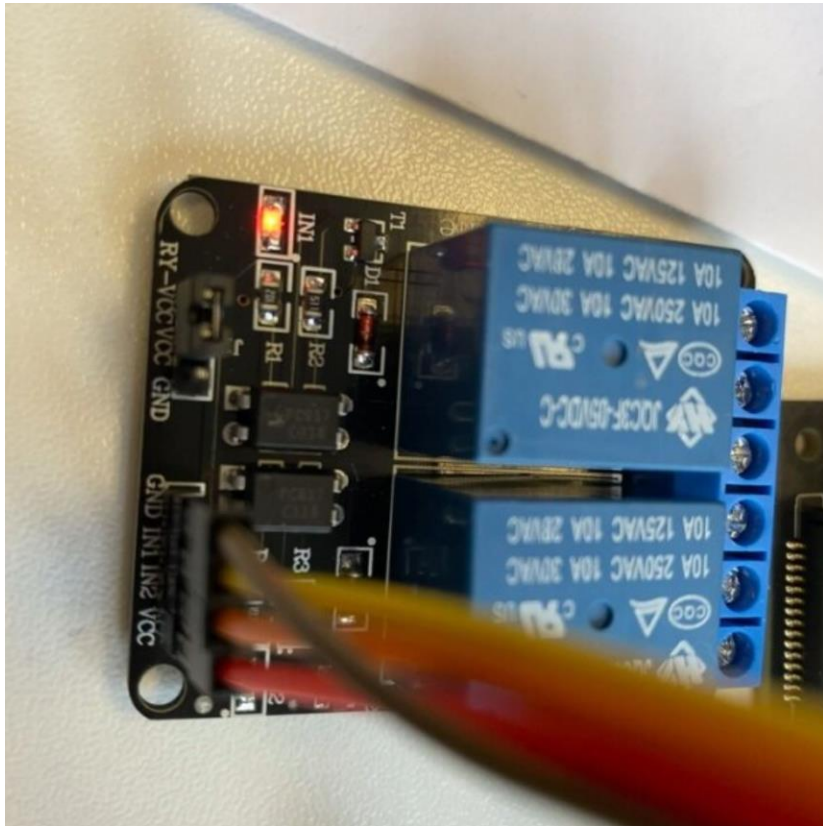
IN1: yellow cable or gray cable.

IN2: orange cable or white cable.



Jetson GPIO Samples— Simple out

- Open terminal, run the command:
`$ git clone https://github.com/NVIDIA/jetson-gpio.git`
`$ cd jetson-gpio/samples`
- Open `simple_out.py`, modify the `output_pin=17`
Pin Definitions
`output_pin = 17 # BCM pin 17, BOARD pin 11`
- Save changes, run `simple_out.py` in terminal use the command:
`$ python3 simple_out.py`



```
nvidia@nvidia-desktop:~/jetson-gpio/samples$ python3 simple_out.py
Starting demo now! Press CTRL+C to exit
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
Outputting 0 to pin 17
Outputting 1 to pin 17
```

Jetson GPIO Samples—Simple out

- We can see the IN1 light kept turn on or turn off with the different signals 1 0
- In the same way, you can modify the `output_pin = 27`(the BCM number of pin 13) to see the IN2 light.

Jetson GPIO Samples—Test all pins

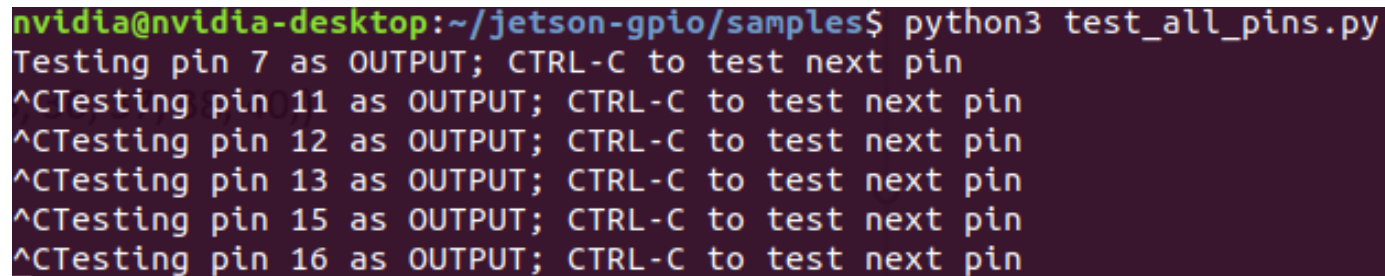
We can use `test_all_pins.py` to check which pin is the output. Skip all `input_only` pins, we set `all_pins` as:

```
all_pins = (7, 11, 12, 13, 15, 16, 18, 19, 21, 22, 23, 24, 26, 29, 31, 32, 33, 35, 36, 37, 38, 40,)
```

Then open a terminal and run commands:

```
$ cd jetson-gpio/samples
```

```
$ python3 test_all_pins.py
```

A terminal window with a dark purple background. The prompt is 'nvidia@nvidia-desktop:~/jetson-gpio/samples\$'. The command 'python3 test_all_pins.py' has been executed. The output shows the program testing pins 7, 11, 12, 13, 15, and 16 as outputs. Each line is followed by '^C' indicating a Ctrl-C interrupt, and the program continues to the next pin.

```
nvidia@nvidia-desktop:~/jetson-gpio/samples$ python3 test_all_pins.py
Testing pin 7 as OUTPUT; CTRL-C to test next pin
^CTesting pin 11 as OUTPUT; CTRL-C to test next pin
^CTesting pin 12 as OUTPUT; CTRL-C to test next pin
^CTesting pin 13 as OUTPUT; CTRL-C to test next pin
^CTesting pin 15 as OUTPUT; CTRL-C to test next pin
^CTesting pin 16 as OUTPUT; CTRL-C to test next pin
```

Press CTRL-C in the terminal to test next pin. In our case, when we pin 11 and 13 as output, the signal light of relay will be flashed alterately.

Use Jetson GPIO output the detection result

We can use Jetson GPIO output the result whether detected the person in the image.

- Firstly download [detectnet-gpio.py](#) and [test.jpg](#), then put them in the path: /home/nvidia/jetson-gpio/samples

- Run the command:

```
$ cd jetson-gpio/samples
```

```
$ python3 detectnet-gpio.py test.jpg result.jpg
```

- Then you can see the IN2 flashed twice: Turn on-Turn off-Turn on-Turn off, it means the 'person' class is successfully detected.
- You can also download an image without person to see what will happen.

How to output detection result on relay?

1. Import GPIO and time:

```
import Jetson.GPIO as GPIO
```

```
import time
```

2. Define pin number and setup

```
GPIO.setmode(GPIO.BOARD)
```

```
relay_pin = 13 # Replace with the actual GPIO pin number you are using
```

```
GPIO.setup(relay_pin, GPIO.OUT)
```

3. Write if condition

```
if detection.ClassID == 1:
```

```
    GPIO.output(relay_pin, GPIO.HIGH) # Activate the relay
```

```
    time.sleep(0.5)
```

```
    GPIO.output(relay_pin, GPIO.LOW)
```

```
    time.sleep(0.5)
```

```
GPIO.output(relay_pin, GPIO.HIGH)
```

```
time.sleep(0.5)
```

```
GPIO.output(relay_pin, GPIO.LOW)
```

```
time.sleep(0.5)
```

```
else:
```

```
    GPIO.output(relay_pin, GPIO.LOW) # Deactivate  
the relay
```

4. GPIO cleanup

```
GPIO.cleanup()
```

How to output detection result on relay?

```
[image] loaded 'test.jpg' (750x1125, 3 channels)
detected 1 objects in image
<detectNet.Detection object>
  -- ClassID: 1
  -- Confidence: 0.993652
  -- Left: 201.965
  -- Top: 200.226
  -- Right: 536.865
  -- Bottom: 1050.29
  -- Width: 334.9
  -- Height: 850.067
  -- Area: 284687
  -- Center: (369.415, 625.259)
```

As for class "person", we can see the ClassID is 1 in the terminal print out result. So we set the if condition
`detection.ClassID == 1`

Assignment (Group Project)

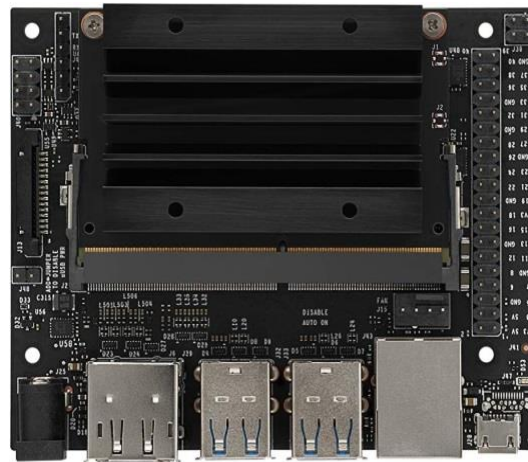
Use detectnet_gpio.py to detect other category objects.

You can download an image on the internet and select one another category in the ssd-mobilenet-V2 model.

Also, you can change the relay signal as you like.

Design an application with AI Model and GPIO in real-time

Once you finish this assignment, please raise your hand!



A



B



Innovate your creative project idea



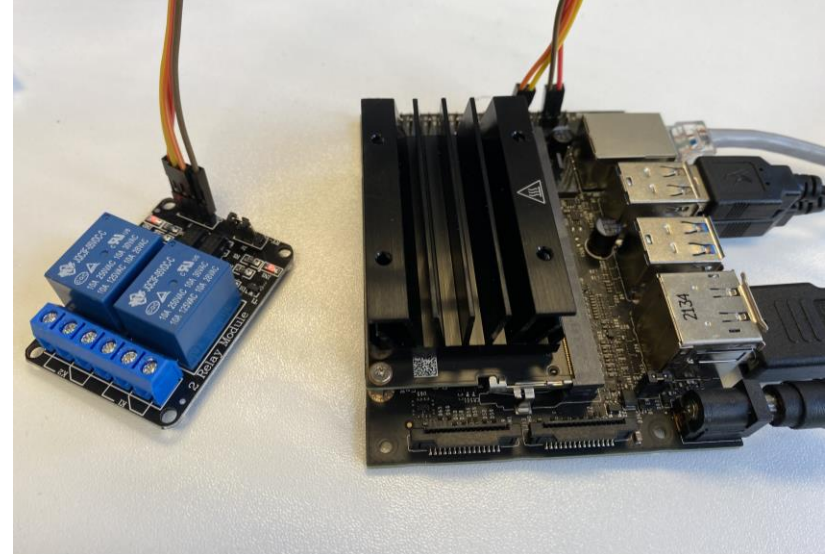
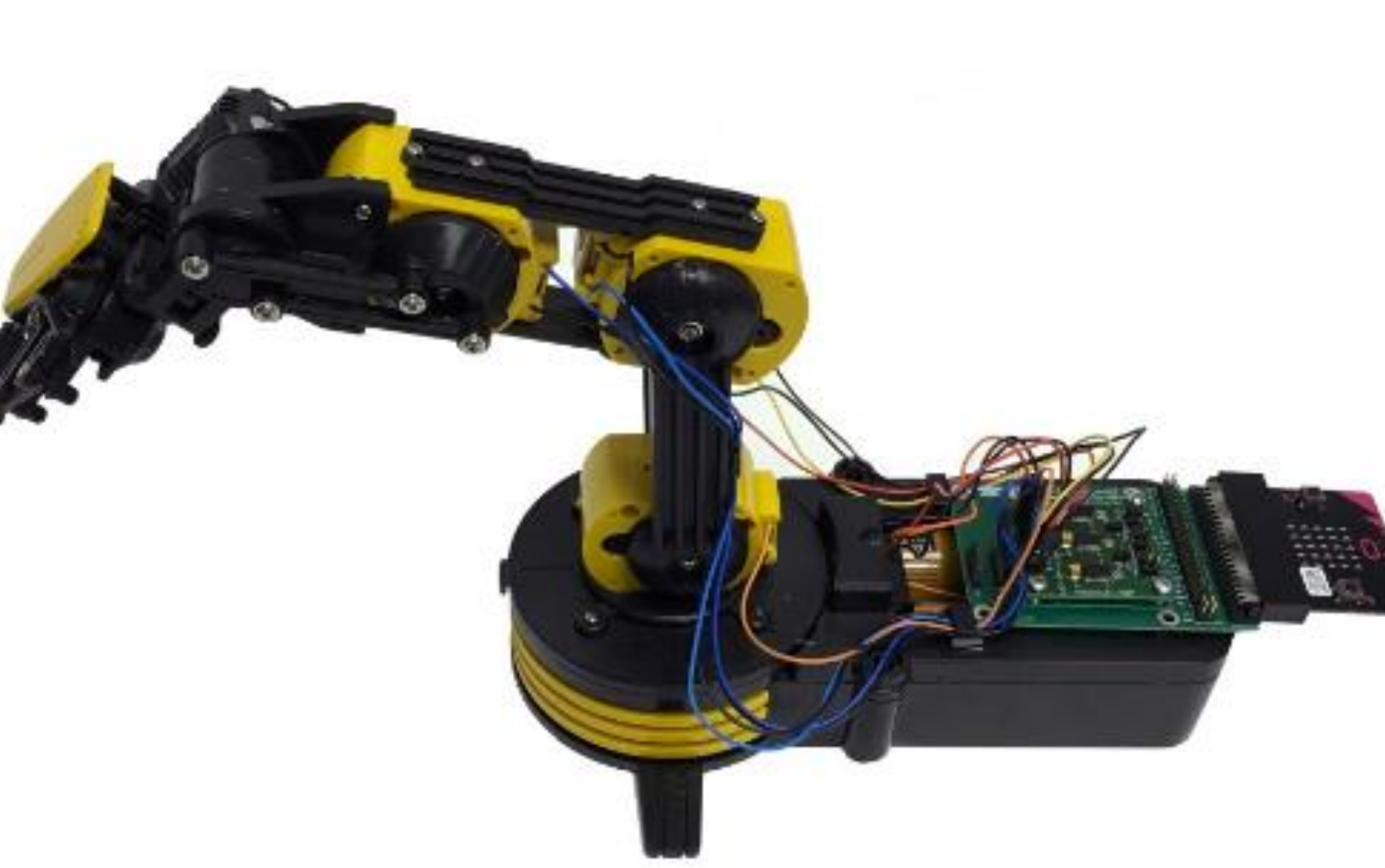
Jetson Nano GPIO has a wide range of utilities, search for some utilities online and choose the one you are most interested in.



Let's innovate how to use the Nano GPIO and some other components to realize the function



We also encourage some ideas for nano GPIO applications based on existing components and technologies.

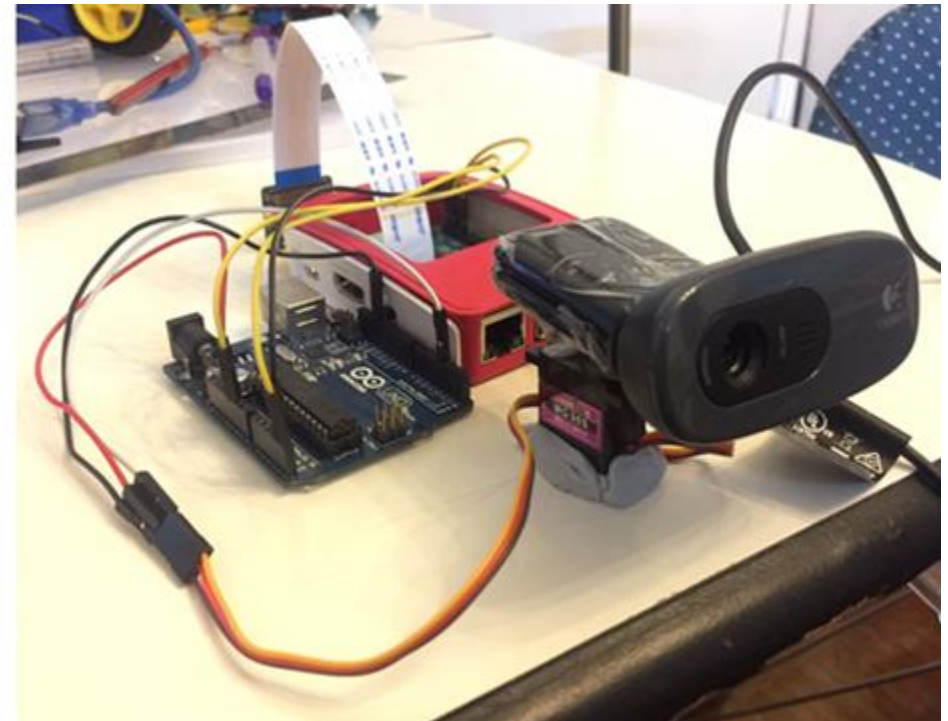
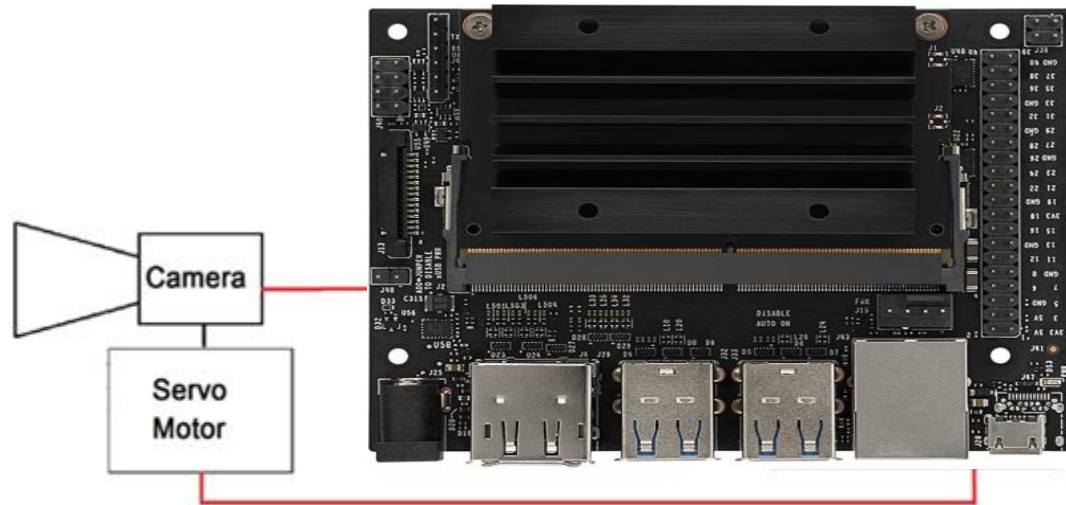


Applications

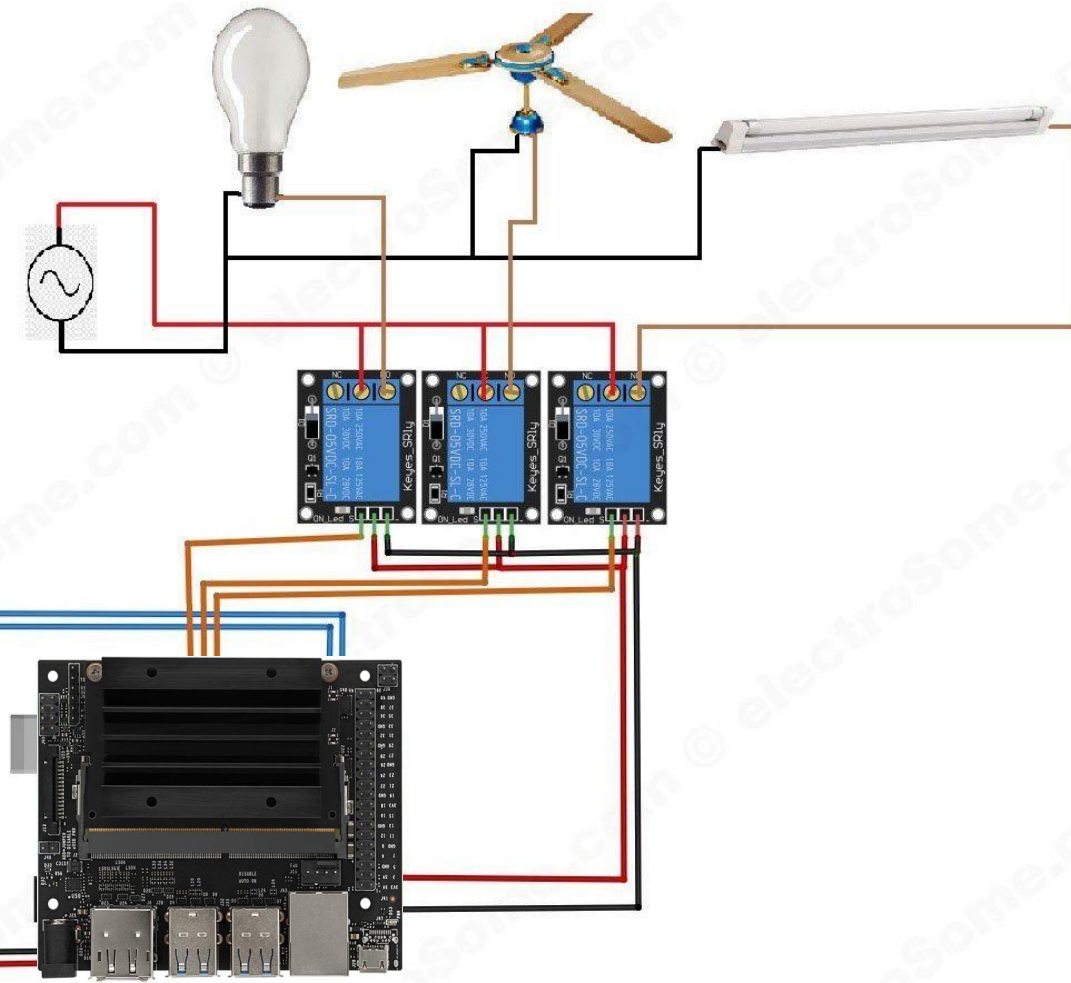
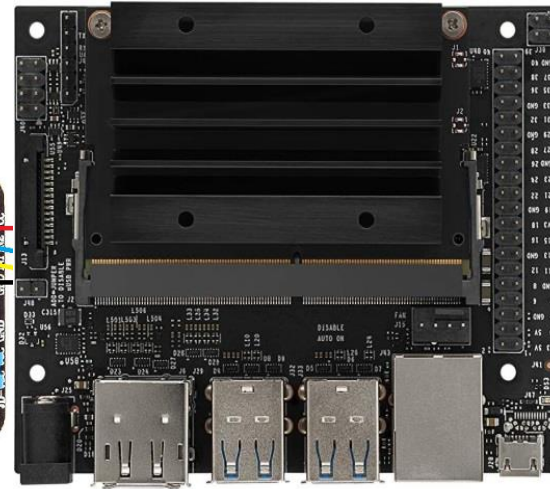
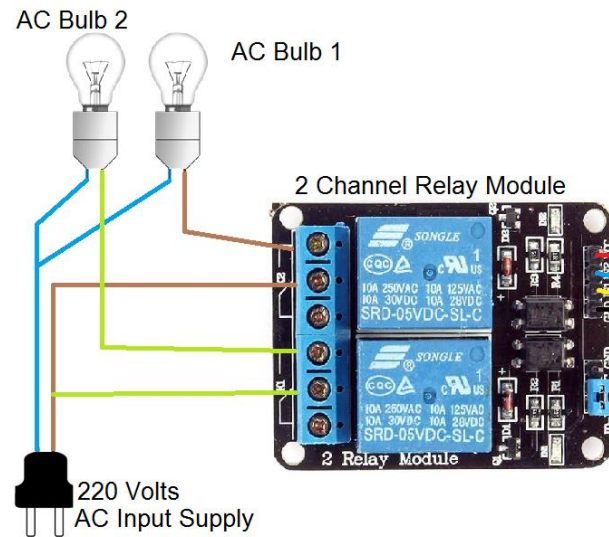
Image tracking

For example : face tracking (to keep the face in the middle)

IF the face is on the left, THEN move the servo to the left , ELSE move the servo to the right



For smart home



Course Closing: Check by TA

Delete files together

Desktop/TeachingMaterial,

Delete Assignment 2 your own dataset and model

Delete Assignment 3 your own my-detection.py file

you'll save it on your host device under **home/jetson-inference/examples/my-detection.py**,

Delete Assignment 4 your own .py and test_image

Download [new-posenet.py](#) [test_image](#) and put them in the path:
/home/nvidia/jetson-inference/python/examples