Gesture Recognition Based on Coral USB Accelerator

TAIWAN TECH National Taiwan University of Science and Technology

Shanq-Jang Ruan, Ph. D.

Distinguished Professor

Dept. Electronic and Computer Engineering

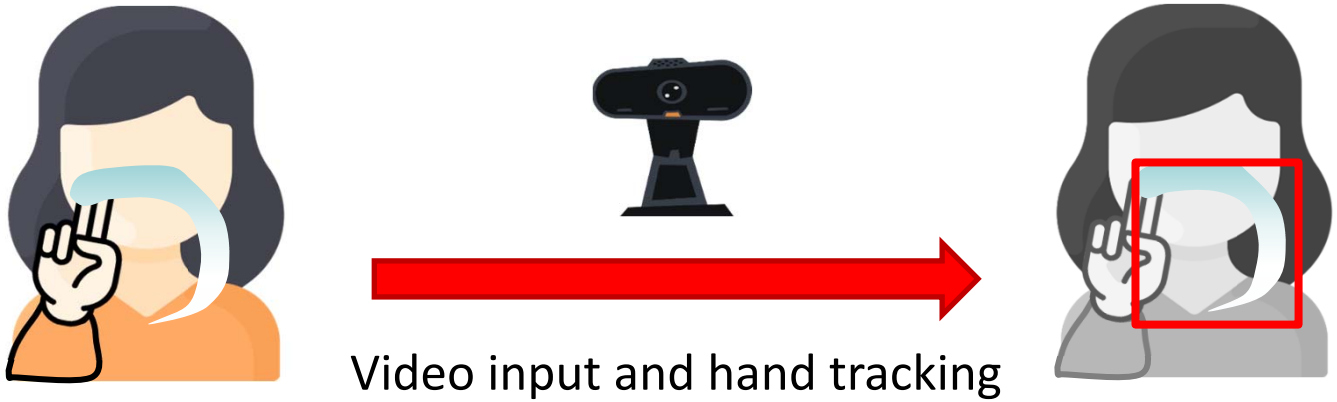
National Taiwan University of Science and Technology

Outline

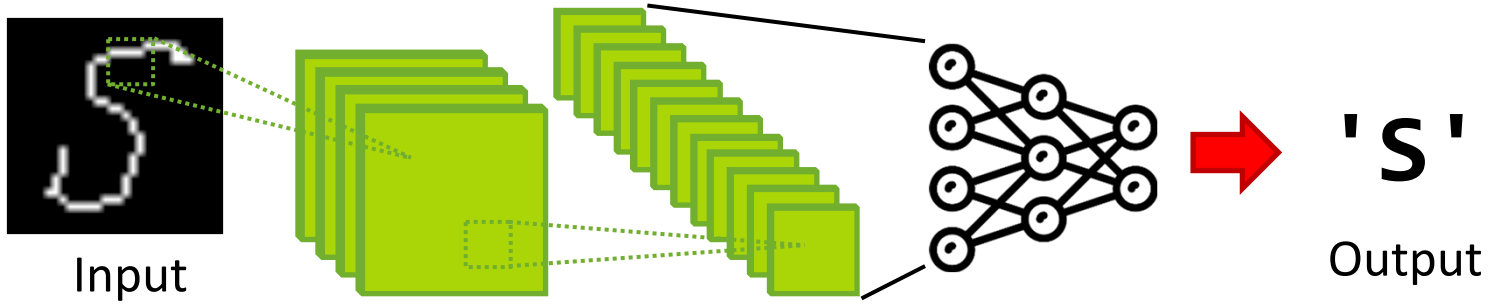
- Gesture Recognition
  - Hand Tracker
  - Gesture Classification
- Raspberry Pi
- Google Coral
- Conclusion

Gesture Recognition

- Hand Tracker

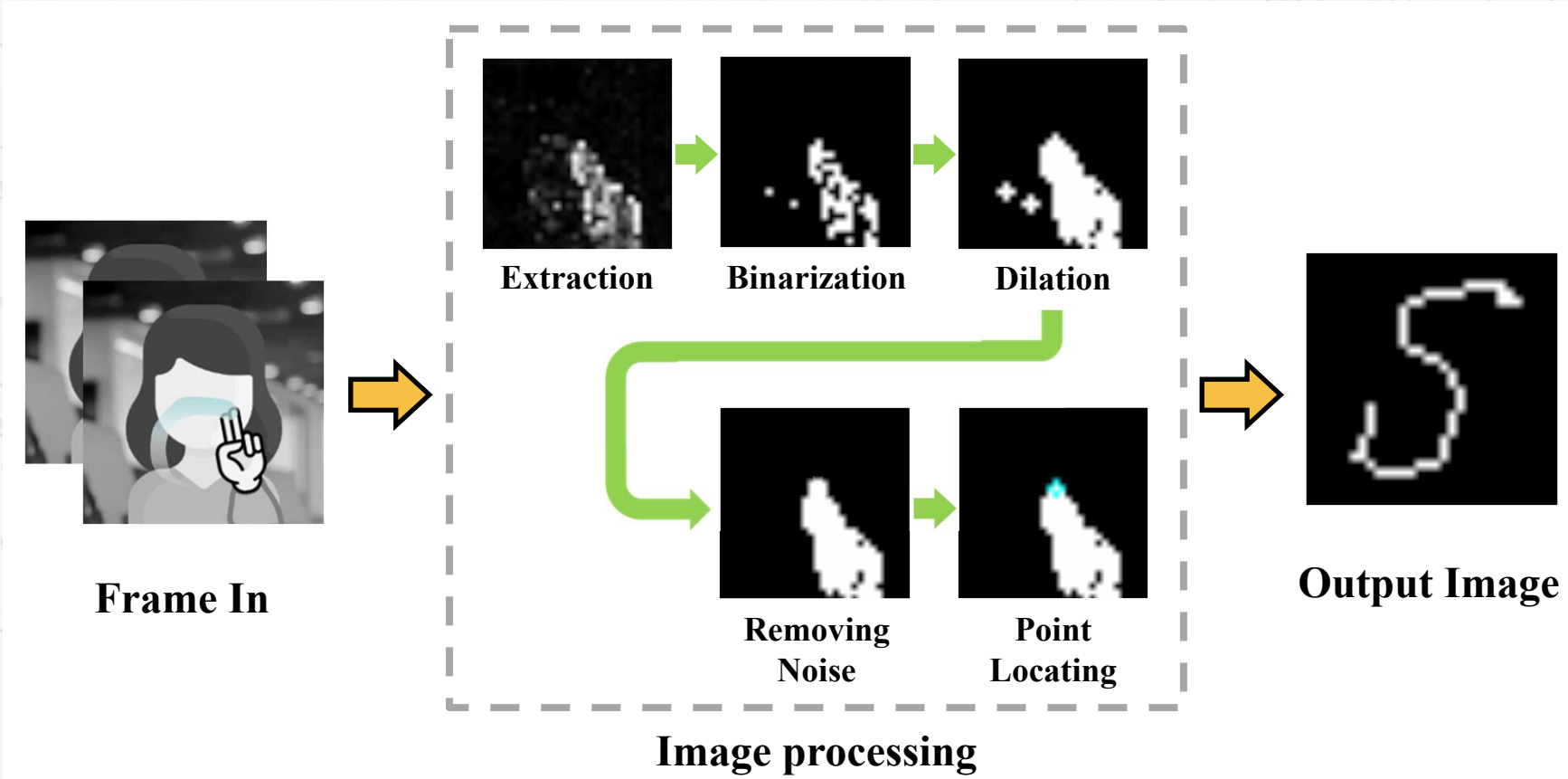


- Gesture Classification



Hand Tracker

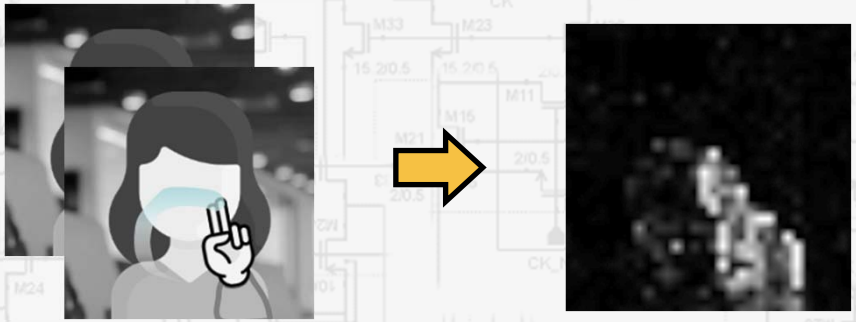
- Output an image every 30 frame.



Hand Tracker

- Extraction
  - Extract differences between continuous frames to capture the movement of the hand.

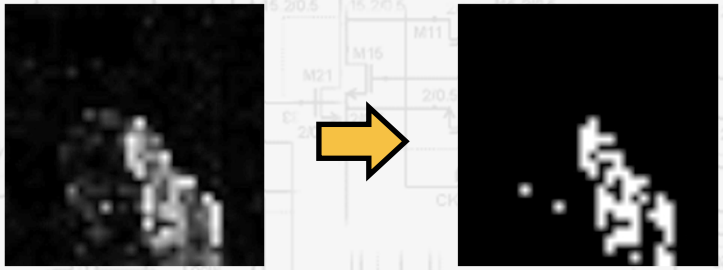
```
1       import cv2
2
3       cap = cv2.VideoCapture(0)
4       ret, frame_RGB = cap.read()
5
6       frame_gray = cv2.cvtColor(frame_RGB, cv2.COLOR_BGR2GRAY)
7
8       frame_absdiff = cv2.absdiff(frame_gray, frame_gray_old)
9       frame_gray_old = frame_gray
```



Hand Tracker

- Binarization
  - Convert a grayscale image to a binary image.
    - Global fixed threshold binarization
    - Local adaptive threshold binarization
    - **Otsu binarization**

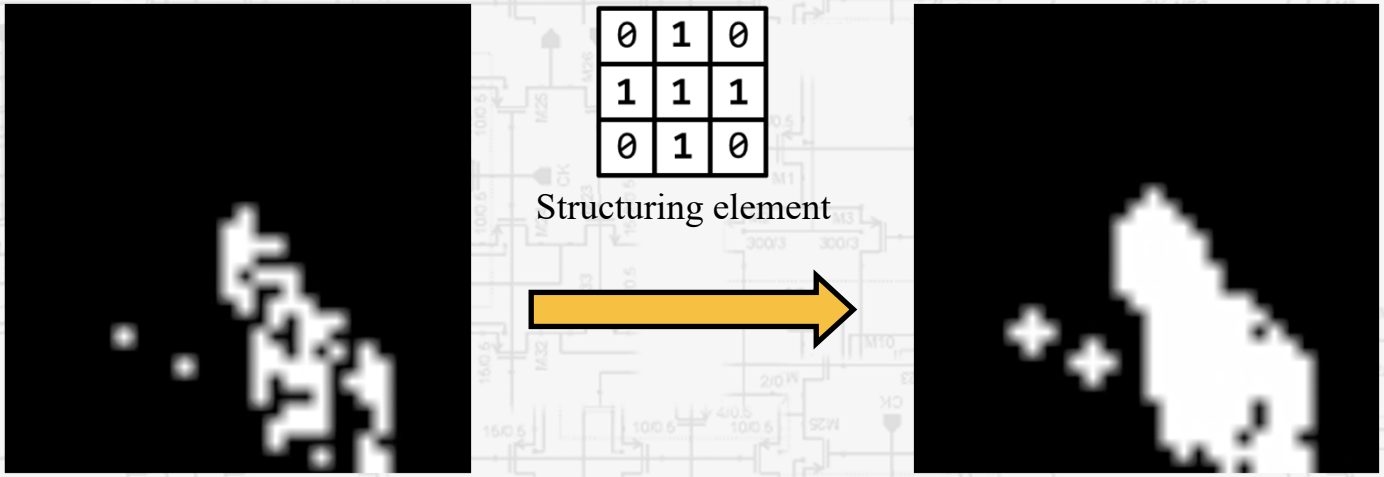
```
1    # Global fixed threshold
2    ret1, th1 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
4    # Local adaptive threshold
5    th2 = cv2.adaptiveThreshold(img, 255, cv2.ADAPTIVE_THRESH_MEAN_C,
6                                cv2.THRESH_BINARY, 11, 2)
8    # Otsu threshold
9    ret2, th3 = cv2.threshold(img, 0, 255, cv2.THRESH_BINARY+ cv2.THRESH_OTSU)
```



Hand Tracker

- Dilation
  - Dilation adds pixels to the boundaries of objects in an image.

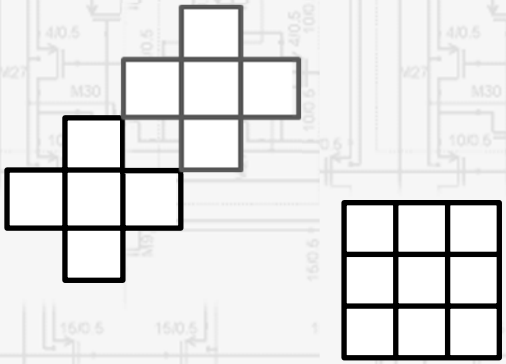
```
1    kernel = cv2.getStructuringElement(cv2.MORPH_CROSS, (3,3))
2    dilation = cv2.dilate(img, kernel, iterations = 1)
```

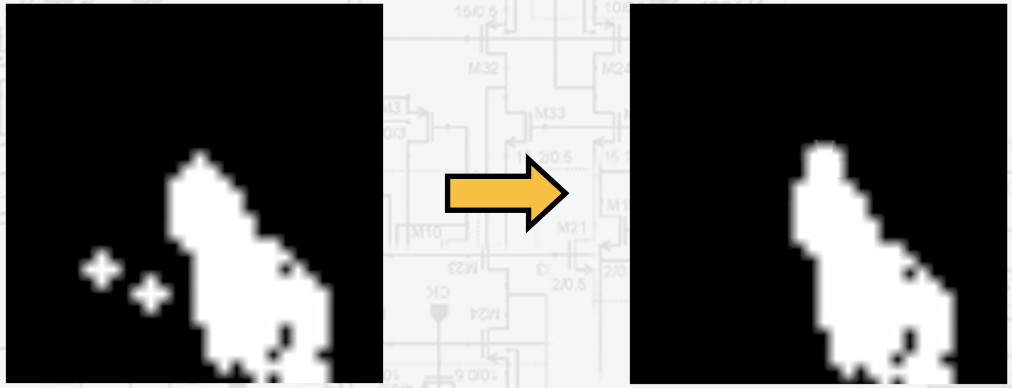


Hand Tracker

- Removing Noise
  - Find max connected-component to remove noice.
  - Connected-component Labeling (CCL)
    - 4-connected
    - 8-connected

```
1  num, labels, stats, centroids = cv2.connectedComponentsWithStats(img,
2                                                                   connectivity=4)
```

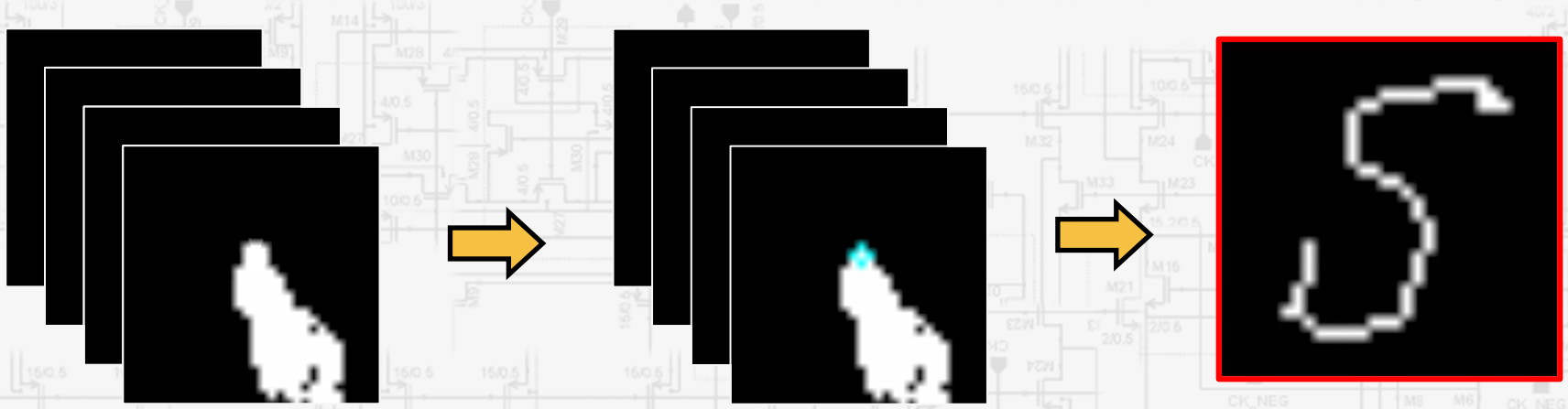




Hand Tracker

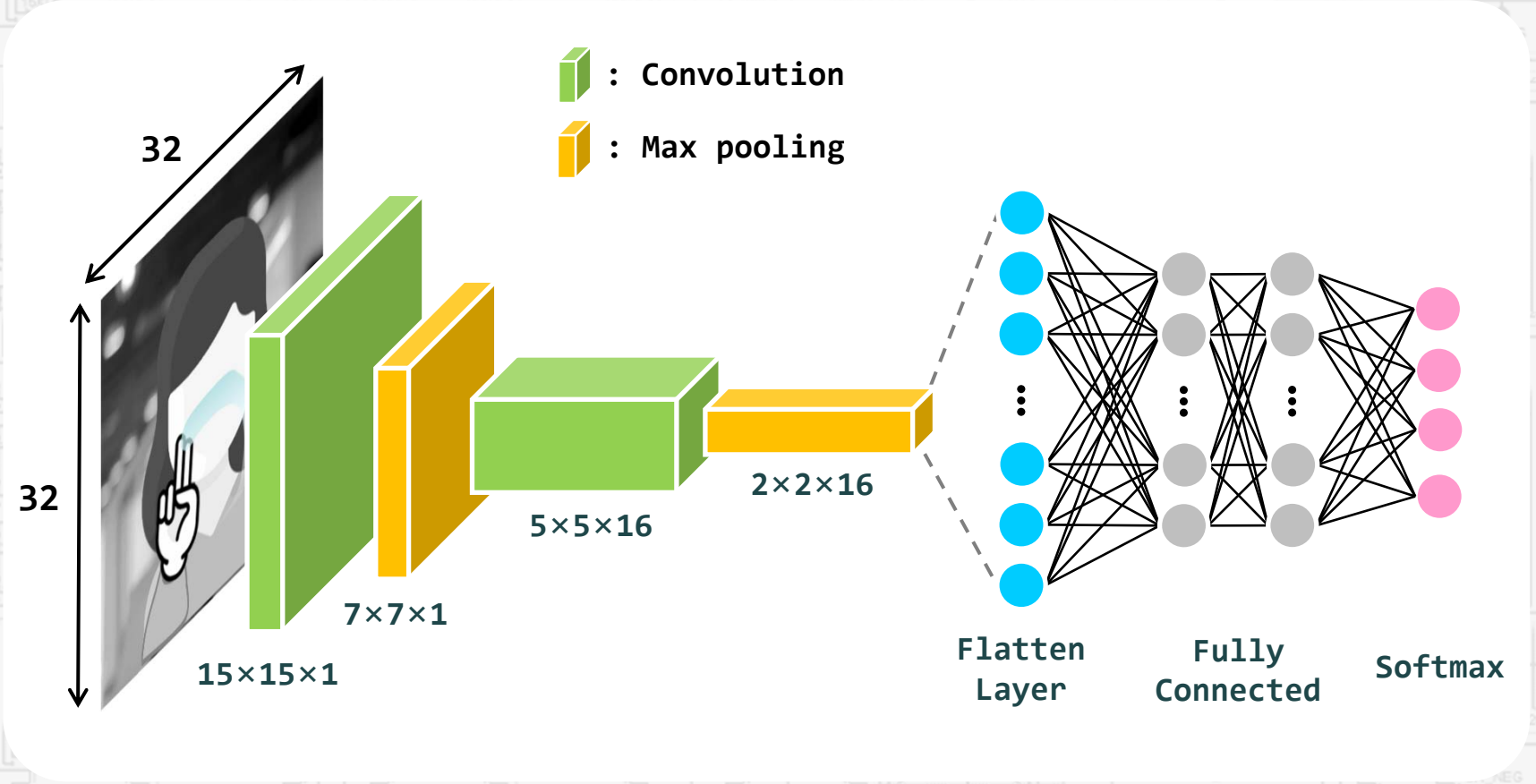
- Hand Localization
  - Find the top-right point of the object in each frame and draw lines between these points.

```
1    x = stats[sorted(range(len(sizes)), key=sizes.__getitem__)[-2]][0]
2    y = stats[sorted(range(len(sizes)), key=sizes.__getitem__)[-2]][1]
3
4    line = cv2.line(canvas, (pre_x, pre_y), (x, y), (0, 255, 255), 1)
5    pre_x, pre_y = x, y
```



Gesture Classification

- Architecture

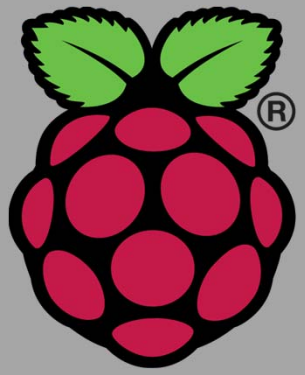


Gesture Classification

- Architecture
  - Convolution layer
  - Max pooling
  - Fully connected layer
  - Activation function
    - ReLU
  - Output layer
    - Softmax

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 15, 15, 1)	10
max_pooling2d (MaxPooling2D)	(None, 7, 7, 1)	0
conv2d_1 (Conv2D)	(None, 5, 5, 16)	160
max_pooling2d_1 (MaxPooling2	(None, 2, 2, 16)	0
flatten (Flatten)	(None, 64)	0
dense (Dense)	(None, 16)	1040
activation (Activation)	(None, 16)	0
dense_1 (Dense)	(None, 16)	272
activation_1 (Activation)	(None, 16)	0
dense_2 (Dense)	(None, 16)	272
activation_2 (Activation)	(None, 16)	0
dense_3 (Dense)	(None, 4)	68
activation_3 (Activation)	(None, 4)	0

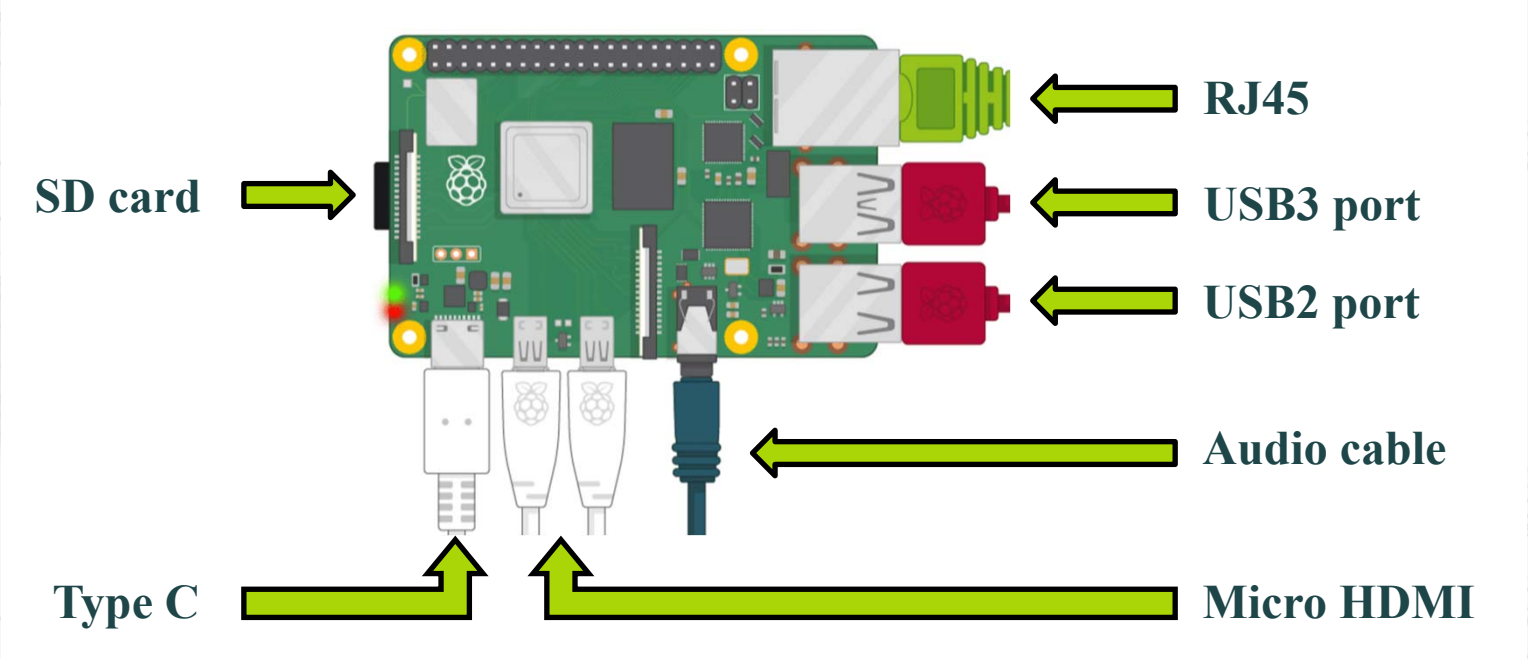
Total params: 1,822
Trainable params: 1,822
Non-trainable params: 0



Raspberry Pi

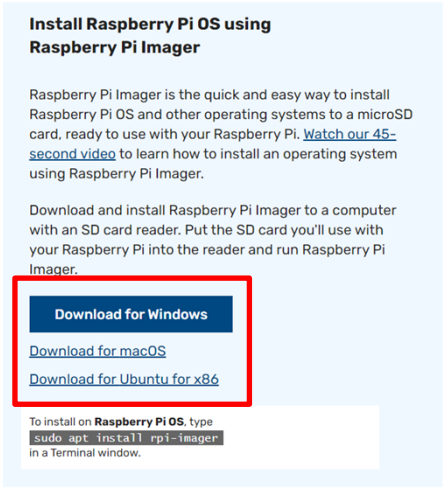
Raspberry Pi

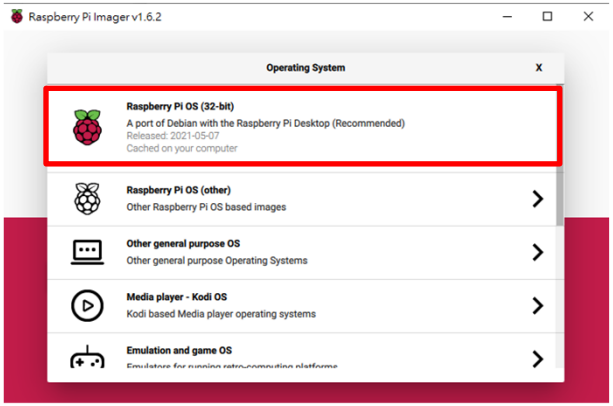
- The Raspberry Pi is a tiny computer that runs Linux.
- It employs SoCs, which combine the CPU and GPU onto a single integrated circuit, with the RAM, USB ports, and other components soldered to the board.



Raspberry Pi

- Raspberry Pi set up:
  - Install OS: Raspberry Pi downloads page





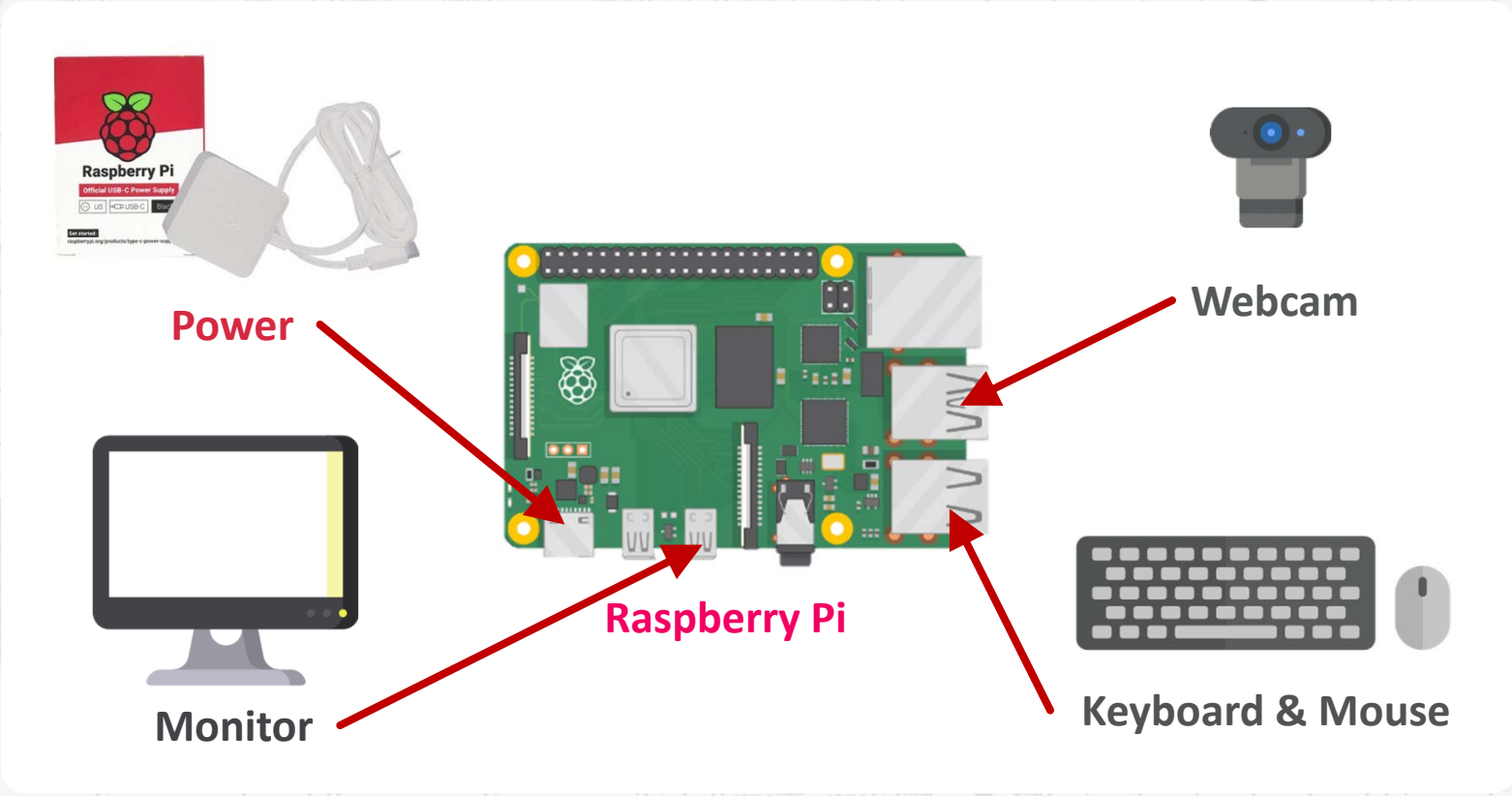
Step 1: Select the Raspberry Pi Imager which matches your OS

Step 2: Launch Raspberry Pi Imager and choose your settings

Step 3: Press "WRITE" to write image to the SD cards

Raspberry Pi

- Raspberry Pi set up:



Raspberry Pi

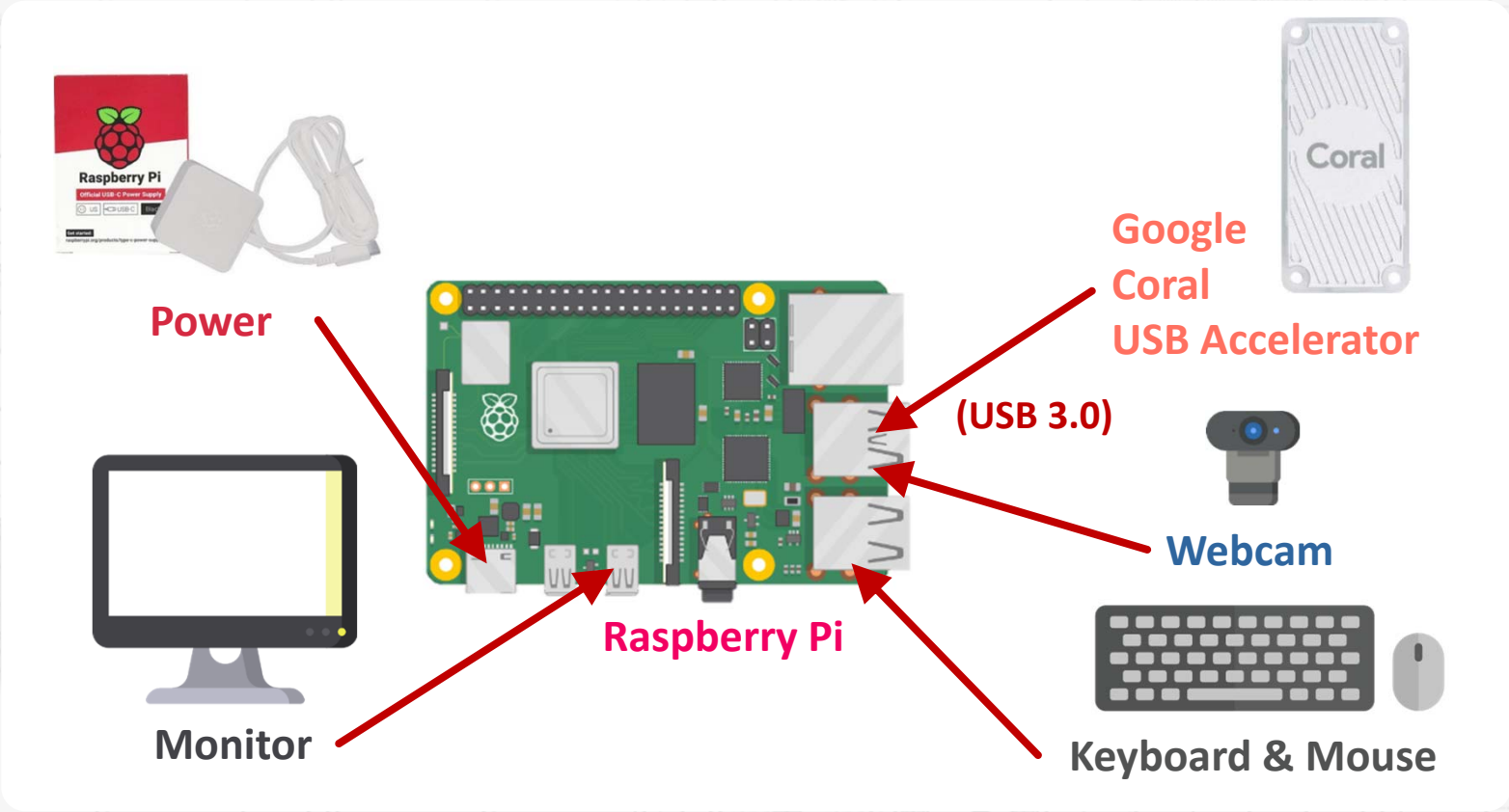
- Install Packages:

```
# upgrade the tools
$> pip3 install --upgrade pip
$> sudo apt-get -y update
$> pip3 install --upgrade setuptools
# install TensorFlow
$> pip3 install tensorflow-2.8.0-cp39-cp39-linux_aarch64.whl wrapt --upgrade
   --ignore-installed
# keras
$> pip3 install keras==2.8.0
# opencv
$> pip3 install opencv-contrib-python
$> pip install protobuf==3.20.*
```

Coral

Google Coral

- Coral set up:



Google Coral

- Install the Edge TPU runtime
  - Add Debian package repository to your system

```
echo "deb https://packages.cloud.google.com/apt coral-edgetpu-stable main" | sudo tee
/etc/apt/sources.list.d/coral-edgetpu.list

curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

sudo apt-get update
```

  - Install the Edge TPU runtime

```
sudo apt-get install libedgetpu1-std
```

  - Connect the Coral USB Accelerator to the computer using the provided USB 3.0 cable.

Google Coral

- Edge TPU Compiler
  - Download the compiler on your Linux system

```
curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

echo "deb https://packages.cloud.google.com/apt coral-edgetpu-stable main" | sudo tee
/etc/apt/sources.list.d/coral-edgetpu.list

sudo apt-get update

sudo apt-get install edgetpu-compiler
```

  - Compile the TensorFlow Lite model (.tflite file) into a file that's compatible with the Edge TPU.

```
edgetpu_compiler model/yolov4-tiny-relu-int8.tflite
```

Kneron USB Accelerator

- Link: https://youtu.be/4aPVb8fm_o4

Gesture Recognition
Based on Coral USB Accelerator

Implementation

Gesture Recognition

- Load TensorFlow Lite model

```
1  import tensorflow as tf
2
3  model_path = "model_quant_NIST_model_edgetpu.tflite"
4  interpreter = tf.lite.Interpreter(model_path,
5  experimental_delegates=[tf.lite.experimental.load_delegate('libedgetpu.so.1')])
6
7  interpreter.allocate_tensors()
8
9  input_details = interpreter.get_input_details()
10 output_details = interpreter.get_output_details()
11
12 input_shape = input_details[0]["shape"]
13 input_data = img.reshape(input_shape).astype("float32")
14 interpreter.set_tensor(input_details[0]["index"], input_data)
15 interpreter.invoke()
16 output_data = interpreter.get_tensor(output_details[0]["index"])
```

Conclusion

- **Gesture Recognition**
  - We developed a hand gesture tracking and recognition method using image processing algorithms and deep learning.
- **Raspberry Pi**
  - Low cost, credit-card sized computer.