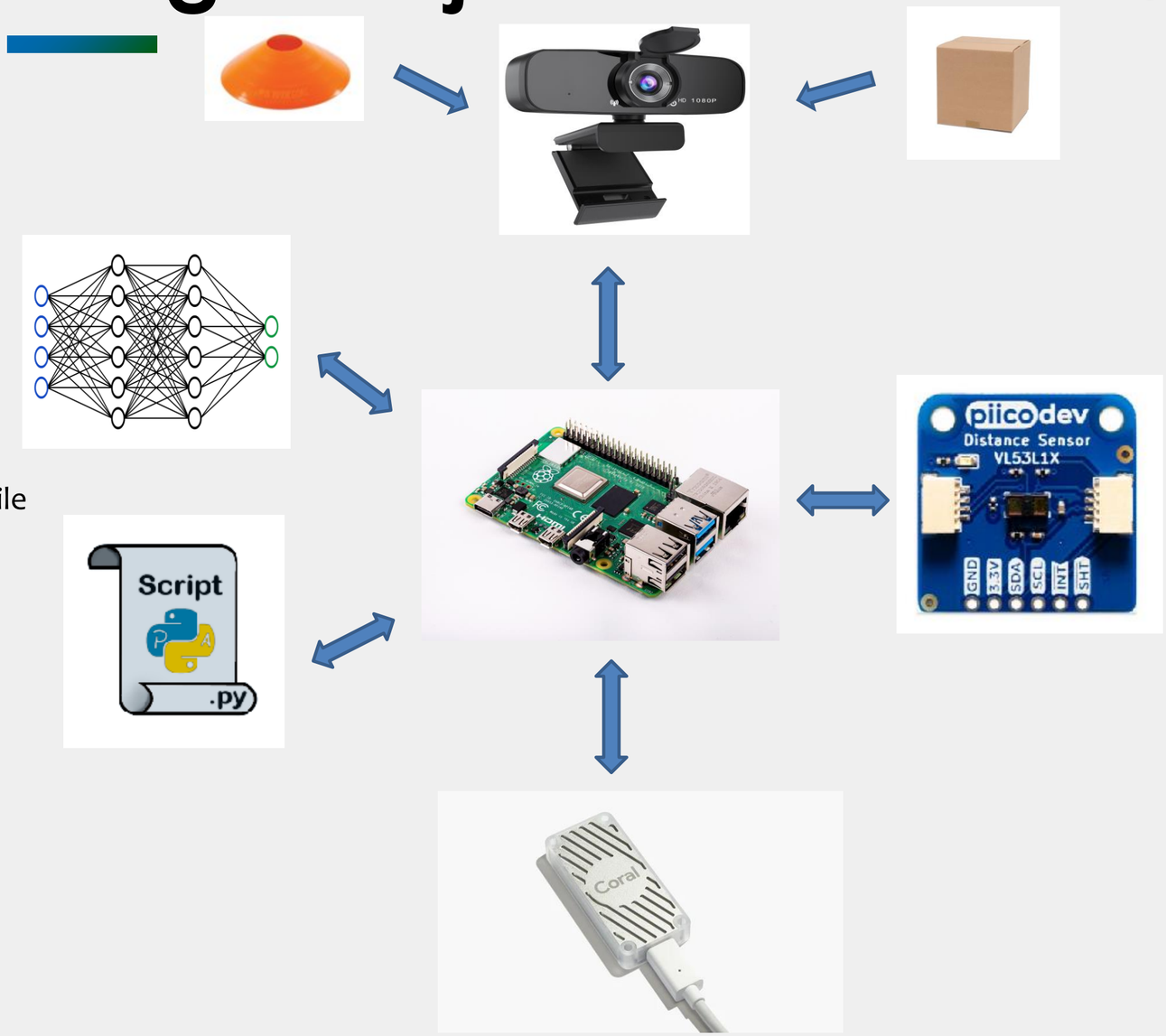
ERU High-Level Processing – Object Detection

- **Hardware/Software tools utilized**
  - Raspberry Pi 4, HD USB webcam (CV2), Google coral edge TPU
  - Ubuntu 20.04, Google Colab, TFLite, LabelIMG, Python3.8
  - PiicoDev Laser Distance Sensor

- **Network**
  - Efficientdet_lite_0(current), possibly 0~2 can be used.
  - Originally trained on COCO(Common Objects in Context) 2017 dataset, optimized for TFLite, designed for performance on mobile CPU, GPU, and EdgeTPU.
  - Retrained through Google Colab using custom dataset (hiker, package, cone, and box).

- **Implementation**
  - Upon detecting obstacle, I/O signal is sent from inference python script
  - FPS(Frames per second) and Distance (in mm) printed on live camera screen
  - Signals will be packaged and interpreted via ROS



ERU High-Level Processing – Object Detection

```python
import time
import cv2
from tflite_support.task import core
from tflite_support.task import processor
from tflite_support.task import vision

#piicodev libraries for sensor
from PiicoDev_VL53L1X import PiicoDev_VL53L1X
from time import sleep
import utils
```

-> Imports

```python
def run() -> None:
    cap = cv2.VideoCapture(0)
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

    #CHANGE NAME OF FILE YOU'RE USING HERE AND SET USE_CORAL TO TRUE IF USING EDGETPU LINE 16
    base_options = core.BaseOptions(file_name="android_edgetpu.tflite", use_coral=True, num_threads=4)
    detection_options = processor.DetectionOptions(max_results=3, score_threshold=0.3)
    options = vision.ObjectDetectorOptions(base_options=base_options, detection_options=detection_options)
    detector = vision.ObjectDetector.create_from_options(options)

    #declare visualization parameters
    row_size = 20  # pixels
    left_margin = 24  # pixels
    right_margin = 300
    text_color = (0, 0, 255)  # red
    font_size = 1
    font_thickness = 1
    fps_avg_frame_count = 10
    counter, fps = 0, 0
    start_time = time.time()
```

-> Set-up

```python
    distSensor = PiicoDev_VL53L1X()
    while cap.isOpened():
        #sensor#############################################################
        success, image = cap.read()
        dist = distSensor.read()  # read the distance in millimetres

        distance_text = 'Distance = {:.1f}'.format(dist) + 'mm'
        text_location = (left_margin, row_size)
        cv2.putText(image, distance_text, text_location, cv2.FONT_HERSHEY_PLAIN, font_size, text_color, font_thickness)
        #sleep(0.1)
        #FPS#################################################
        counter += 1
        if counter % fps_avg_frame_count == 0:
            end_time = time.time()
            fps = fps_avg_frame_count / (end_time - start_time)
            start_time = time.time()
```

-> Laser Sensor

-> FPS calculation

```python
        # Show the FPS
        fps_text = 'FPS = {:.1f}'.format(fps)
        text_location = (right_margin, row_size)
        cv2.putText(image, fps_text, text_location, cv2.FONT_HERSHEY_PLAIN, font_size, text_color, font_thickness)
        ######################################################

        input_tensor = vision.TensorImage.create_from_array(image)
        detection_result = detector.detect(input_tensor)
        #image = utils.visualize(image, detection_result)
        numObj = len(detection_result.detections)
        for i  in range(numObj):
            label_halfx = detection_result.detections[i].bounding_box.width//2
            label_halfy = detection_result.detections[i].bounding_box.height//2
            label_x = label_halfx + detection_result.detections[i].bounding_box.origin_x
            label_y = label_halfy + detection_result.detections[i].bounding_box.origin_y
            cv2.putText(image, "L", (label_x,label_y), cv2.FONT_HERSHEY_PLAIN,10,(0,0,255),10)


        if cv2.waitKey(1) == 27:
            break
        cv2.imshow('object_detector', image)

    cap.release()
    cv2.destroyAllWindows()

run()
```

-> Displaying FPS

-> Image feed & Coordination

ERU High-Level Processing – Object Detection

- **Wireless Video Transmission using NDI**
  - FPS measured: About 15~17, max 20

- **NDI Protocol Specifications**
  - By default, it transmits using UDP
  - Many endpoints can watch the video source on the network
  - Utilizes mDNS (multicast Domain Name System) to allow for zero configuration discovery

Demo video Link:

https://youtu.be/-JaYwJnfG48