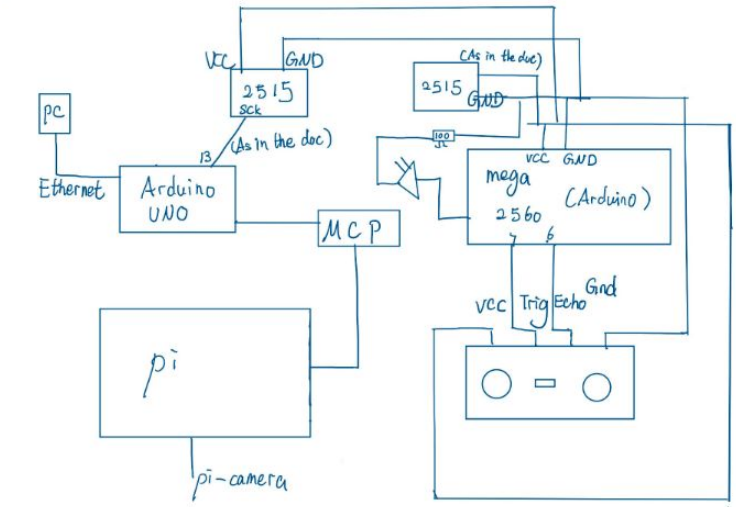
CS498 IoT lab1 report

Group Members Names: Yifei Liu, Xingzhi Liu, Siyu Niu, Han Jiang
Member's netids: yifeil6, xingzhi2, siyuniu2, hanj2

Besides the demo video, you should also submit a **report** containing:

1) **A topology specifying the wiring of all your devices.** (We recommend drawing the graph using Fritzing to show detailed wirings, including the choice of pins on the Arduino.)



2) **Your design considerations (briefly) for each part of the lab**

Briefly design:
We have four parts: Part 1 is the Raspberry Pi. Part 2 is the Arduino micro-controller. Part 3 is CAN to IP. Part 4 is the head unit.

Part 1
Raspberry Pi:
We decide to use Pi as a video stream to send real-time information of images data to the head unit.

1. Would hardware acceleration help in image processing? If so, have the packages mentioned above leveraged it? If not, how could you properly leverage hardware acceleration?
Raspberry Pi has limited calculation ability so the inference speed would be extremely slow. It will cause a delay of about several seconds in a real-time

video when an object is close to the detector. Even a small model needs about 22-27M brandwidth in transmission, which is way out of Raspberry Pi's ability. So we decided to use Raspberry Pi as a video streamer and implement the object detector in the head unit by getting data from Raspberry Pi's detection. By using a Head Unit to infer with the GPU acceleration, the speed of inference has been largely optimized.

2. Would multithreading help increase (or hurt) the performance of your program?
Since the multithread package in Python is not bug-free due to GIL, we are not using a multithread method in our program.
We found some code to make the video from the camera to speed up, but we don't find that the speed of detection has been increased since the most time-consuming part is the inference so images from the real-time video are delayed largely. This part of the pipeline has been already implemented by the TensorFlow library, so there is not much we can do.

3. How would you trade-off between the frame rate and detection accuracy?
As we are using the ssd_mobile_app model created by Google, which is a good model for pretending. Also, as we use pi as a video streamer, and use the computer to deal with the operation, the frame rate is not a big problem. It turns out to be fluent as well as accurate.

4. why quantized models are better for resource-constrained devices?
Quantized models are integers, not floats, so they can be sent and received faster.

Part 2
Micro-controller:
We learned the Arduino code for detecting the distance between an object and the detector online. In a loop with three lines of code, we consistently calculate the distance as long as detecting an object within the detection range.

you may calculate the distance between the sensor and objects using simple Physics (Could you show how this can be done in the report?).

We calculate the distance with the formula: distance = velocity * time. We get the time from our Arduino, which give out the sound wave and record the difference of time between send and receive.

Part3
CAN to IP gateway:
Can - > either net shell,
Set a static IP address and port number for the CAN, Pi, and our computer. By this way, every component can receive and UDP and TCP packets.

UDP or TCP?
We decided to use UDP for sending distance from the detector to the detected objected because the size of the distance data is relatively small. In this case, UDP is a good-fitting enough protocol for data transmission.
By Numpy an image will be transferred to a 2D array. This kind of data can be considerably large for some objects. For sending and accepting images from the video of detection, we decided to use TCP because the network is usually unreliable and might be disconnected in between when an image is being sent. So one image might be separated into different parts during the transmission. UDP does not ensure the order of the packets been sent, so finally, we decided to use TCP as the protocol to transfer images of all detection.TCP can ensure every part of the image has been sent orderly and entirely.

Part 4:
Head Unit

For the Head Unit, we use plain HTML and some CSS style to display three kinds of data: the image of real-time detection, the distance between the detector and the object and the status of LED (this information can be received from the distance calculation.)

Although the Pi might already be too occupied to send any packet rather than processing the video stream, you should at least update the dashboard once every 5 seconds. (Can we make the time interval smaller? Also, you should mention how sending data over the network influence the Pi's performance in the report.) Also, since we have three Ethernet Devices (the Pi, the CAN to IP

gateway, and the Head Unit), how should we interconnect them in the Ethernet network?

Can we make the time interval smaller? Yes, we can.
Since in our implementation, the Raspberry Pi acts as a streamer, the output image gathering is remarkably smooth like a normal video, we decided not to consider more performance optimization.
The data has been sent to the head-unit for visualization, so the time calculation is only for sending. Because of GPU acceleration, the speed has been remarkably increased, and therefore we have a fluent and comfortable experience from the user's perspective.

We interconnect Pi, the CAN to IP gateway, and the Head Unit.
We connect the CAN to IP gateway through the Ethernet shell to our computer and the Pi. By sending packets of images via TCP and data of distances (&LED status) via UDP, we display the data on our head unit with plain HTML and some CSS styles.

3) **Measurement of the end-to-end performance in your obstacle detection system in terms of video frame rate and detection accuracy. Observe at least 3 different scenarios (for example, people moving towards the camera, moving away from the camera, etc) with 5 data points for each scenario.**

detection accuracy (mAP): 22mAP
Video frame rate: 40 fps

Final data
Scenario 1- people moving towards the camera, 2-moving away from the camera
3. People staying in the same distance

4) **Discuss above questions in bold text in your report.**
   They are answered on question 2).