**1.Slide 1**

Today I’m going to talk about my last research, that is distance-measurement and object segmentation applying Mask-RCNN using Intel D435i Depth camera.

**2.Slide 2**

This is the flowchart of the program. First I get color frame and depth frame, combine these two data flow from these two thread to get the distance of every point in the color frame. In that time I use the Mask-RCNN model to detect and get object’s segments in the color frame. From this now we are able to calculate the average distance of each object in the frame.

**3.Slide 3**

About the camera I use for this research, that is Intel Realsense Depth Camera instead of the Kinect camera that I used for my last Final year project, because it gives me better depth image, better in point cloud calibrating. Moreover, it is easier to manipulate with (it has Python library), and finally is is possible to be used in industrial environments.

**4.Slide 4**

Now comes to the program’s procedure. Like I said to calculate the distance of object, both color BGR frame and depth frame are required. Both frames is configured to have exact resolution (in this code I set it to 1280x720). And we can see depth frame is black-likely because it only contain depth value of every points in the frame, and OpenCV transform the value to black to display.

**5.Slide 5**

The depth value is in 720x1280 array-type. For example, the first “289” value is the depth in millimeter of the (1,1) position in the frame (the top-left one).

**6.Slide 6**

For easier of understanding, let me show you the example. This is the depth of (100,250) position point.

**7.Slide 7**

Next I want to briefly explain to you about the deep learning Mask-RCNN model that I use in this program. mrcnn is a form of instance segmentation. The differences between object detection and object-segmentation is that in object detection, we only find out where the object is in the image by finding the bounding box of the object through the Return On Investment (ROI) formula, and in object-segmentation we divide or partition the image into various parts called segments, choose which partition is most likely to contain objects to detect. So object-segmentation can more accurately indicate the shape of an object, serving more purposes.

(Nếu có thằng nào dở chứng hỏi ROI là gì thì trả lời như sau)

‍What is ROI?

Machine learning and data science projects are an investment to each person involved and comes with associated risks. ROI stands for ‘return on investment’ and is used as a performance metric, usually expressed as a ratio or percentage. In its most basic formula,

ROI = Net gains/ Investment cost

In this program RoI is used to predict object classes and bounding boxes.

**8.Slide 8**

The Mask R-CNN framework is built on top of Faster R-CNN. Simply put, when we feed an image in addition to returning the label and bouding box of each object in an image, it will add us object masks.

(Nếu có thằng dở hơi nào hỏi về mạng thì trả lời như sau)

1. It will first use ConvNet to extract features from the input image.

2. These features will then be passed through a Region Proposal Network (RPN) to predict whether the object is in that region or not, which will then return bounding boxes at regions that may have objects of different sizes. That is when we get the ROI calculation.

3. And these features are passed to a fully connected layer for classification and the output is a bounding box for each object.

4. We get the ROI based on the calculation so what we need to do now is add a mask branch to the current architecture.

**9.Slide 9**

In conclusion, this is the output we get. You can see the mask in my smart phone was a little bit faded because of the same color with the smartphone, but look closely they still have masks. We can see it more clearly in the next slide.

**10.Slide 10**

About collecting and training process, I used the dataset of COCO website, which has number of class is 80 and train it until our model’s losses get small enough.

**11.Slide 11**

And this is my recorded result video.

**12.Slide 12**

I get those result, that is the program can get the color frame and depth frame, segment objects using mask-rcnn architecture, and most important, it can measure objects’ distance to the camera. But come with these results are some difficulties. The current program is running very slow because it only use CPU for the computation,

the training time for mask-rcnn architecture is very long compared to other architectures, and finally is how to connect between object detection/measurement program and robot control program.

Here are a few workarounds that I can think of at the moment, that is finding a way to use the GPU for the program, using a good enough set of pre-trained parameters or learning about lighter architectures . And the final solution is to find a way to connect the modules together in the ROS environment.