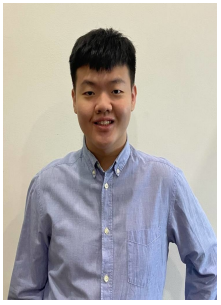
UNIVERSITY OF MALAYA

WID3005 INTELLIGENT ROBOTICS
SEM II 2021/2022
Littering Observation Intelligent (LOI)
Project Report
Group: 12

Leader	Member 1	Member 2	Member 3	Member 4
				
Gun Hong Shen	**Tan Chee Lam**	**Koh Shi Hui**	**Lian Zhi Chun**	**Soh Zhi Chen**
17206426/1	17207181/1	17205189/1	17204715/1	17204929/1
gunhongshen@gmail.com	tancheelam2@gmail.com	shihui360@gmail.com	zhichun1602@gmil.com	wid190050@siswa.um.edu.my

Introduction

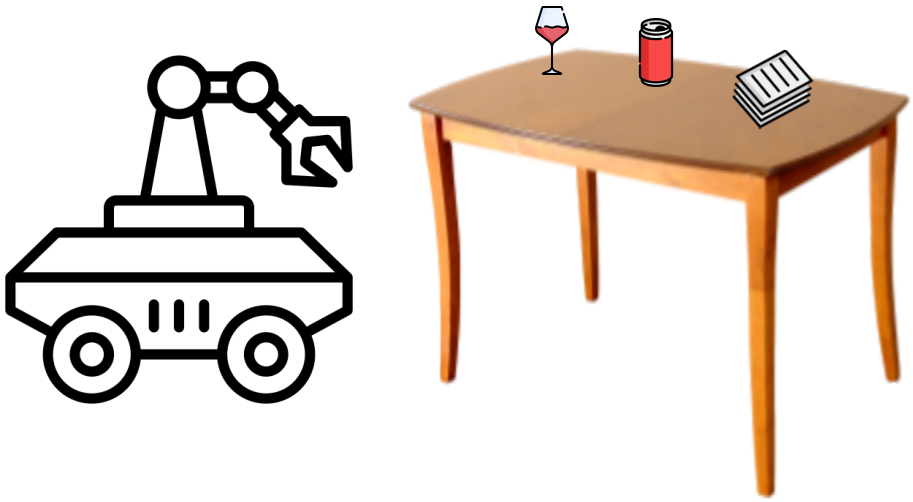


Figure 1: Environment setup of LOI

Littering Observation Intelligent (LOI) is a robot that helps people clean up the environments by collecting rubbish and promote recycling by classifying rubbish into recyclable categories. It is equipped with advanced object recognition capability with integrated Artificial Intelligence (AI) module to detect wastes and classify them into different types of garbage. Besides, it is able to manipulate its arm to grab different types of rubbish. Table 1 shows the classification of recyclable items.

To do our demo and Proof of Concept (PoC), the team has decided to use FSKTM lab room as the indoor environment to showcase the capabilities of the robot. We require a little space with only one table with some recyclable item on it such as can, glass, and newspaper for our robot to classify the item into their categories such as aluminum and plastic, paper, and glass as shown in the Figure 1 above.

As shown in Figure 2 below, our LOI robot will first detect and identify the rubbish location and determine whether the rubbish is recycleble or non-recycleble. Then, it will move to that location and picked up the rubbish using the robot arm. At the same time, it will classify the rubbish into correct recycling categories and throw the rubbish into the recycle bin according to the color of the recycle item.

Figure 2: Storyboard for the LOI demo

1. Pick up rubbish

2. Recognise & classify the type of rubbish

3. Throw rubbish according to the type of recycle bin

Table 1: Target objects to be recognized

Recyclable Items			Non-Recyclable
Paper	**Can / Plastic**	**Glass**	
Newspaper	Plastic bag	Glass bottle	Food waste
Color paper	Soda can	Wine glass	Snack packaging
A4 paper	Plastic cup	Liquor bottles	
Manilla card	Detergent bottle	Bulb	
Tissue paper	Plastic utensils	Glass cup	
Cardboard	Plastic container	Mirror	

Robot Features

1. Pick up rubbish(Robot Arm)

The robot will be able to pick up the rubbish and remove it. Our aim is to implement 3R (reduce, reuse, recycle) in our environment, so the robot should be able to pick up the rubbish in order to reduce it in our environment.

2. Detect and classify type of recyclable rubbish (Image Processing)

Robot will be able to detect 4 types of recyclable rubbish, which are Paper, Plastic, Metal and Glass. The functions will be performed through computer vision. The computer vision will be trained with these 4 types of recyclable rubbish and it will be implemented into the robot system.

Technical Content

Pickup the rubbish using robot arm

We used the jupiterobot_arm_bringup package which is a Python library that can run on the Robot Operating System (ROS) to control the arm movement of Jupiter robot to implement object manipulation and to move object from one point to another. In our case, we want our robot able to reach its arm to the rubbish and grab the rubbish.

Besides controlling the arm, we also used the mobile_base command package to move the robot from one point to another. In our case, we want our robot to move from where it grabs the rubbish to the targeted recycle bin.

To implement all these actions together and automatically, we coded a series of actions we wanted and compiled to a script file. The whole script is referenced from the official documentation of Jupyter Robot. As a result, our robot is able to grab the rubbish, move to the recycle bin and finally throw the rubbish into the recycle bin in a series of actions.

Rubbish detector

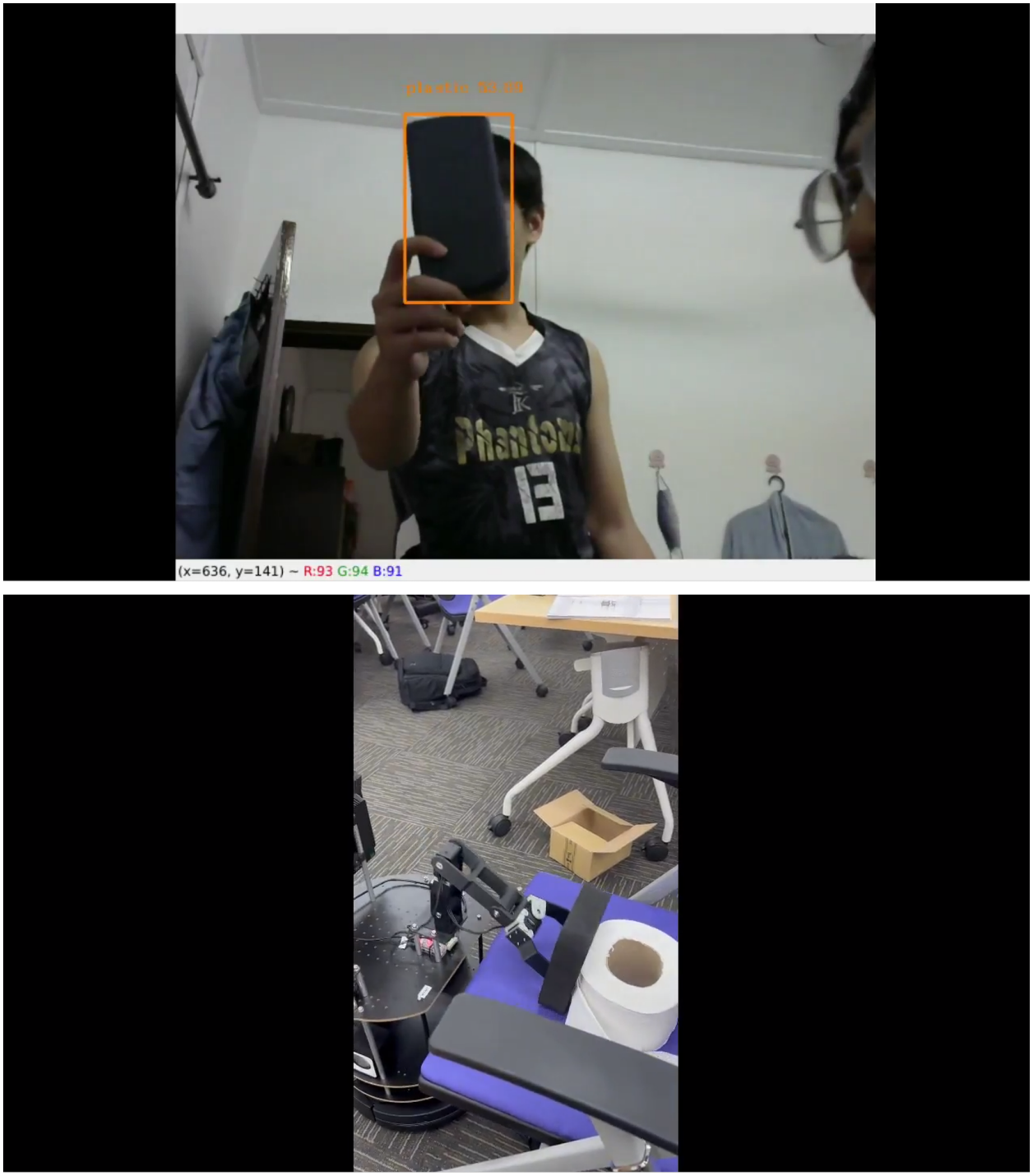
Our rubbish detector system is implemented with transfer learning, which is a pre-trained CNN model and we further fine tune the model with our additional data. The pretrained model that we used is called YoloV5x which is the one of the YOLO architecture family. This model is recently released in October 2021. To implement this model, we use PyTorch, an AI framework to load the model from the model hubs.

To get the image stream from the Jupyter robot camera, we used Open Source Computer Vision Library (OpenCV Python) library to create a video capture device and extract all the image captured. Then, we use the loaded YoloV5x model for inference and detecting possible objects in the camera. We convert the result obtained into Python dictionary for the ease of information extracted and map each predicting label to respective recycle categories. For example, paper cup is mapped to paper category with color blue, while tupperware is mapped to plastic category with color orange We also constraint and limit the model to draw bounding box and focus on recyclable items only.

Another implementation that we tried is modified the `robot_vision_openvino` module to suit our use case. We modified the labels of the model used in yolo_ros launch file and mapped it according to the recycle bin categories. Then we used the command catkin_make to rebuilt the package with the latest changes. After that, we ran the command `roslaunch robot_vision_openvino yolo_ros.launch` to show that the robot able to detecting objects correctly and map them according to the recycle categories. This implementation is fast and easy but limited to the trained labels only.

Proof of Demo

The following link is the video of our trash detector model and our LOI robot demonstration on picking rubbish. Do take note that both of the demo are edited into 1 video as show below.



https://drive.google.com/file/d/16eHE4CqN81ozgB1VQ0a1EjPv4qGrF031/view?usp=sharing

Source Code

The github contains the trash detector model and the script for our rubbish picking demontration. For the script, it is named as arm_movement.sh .

arm_movement.sh

```
# roslaunch jupiterobot_arm_bringup joints_bringup.launch

rostopic pub -1 /elbow_controller/command std_msgs/Float64 -- 0.0                                                    # Move the robot elbow to default origin position
rostopic pub -1 /hand_controller/command std_msgs/Float64 -- 1.0                                                     # Open the robot hand

rostopic pub -1 /elbow_controller/command std_msgs/Float64 -- 2.0                                                    # Move the robot elbow with position set to 2.0

rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[0.5, 0.0, 0.0]' '[0.0, 0.0, 0.0]'           # Command to move the robot foward

rostopic pub -1 /hand_controller/command std_msgs/Float64 -- -0.8                                                    # Closs the hand of the robot to take something

rostopic pub -1 /elbow_controller/command std_msgs/Float64 -- 1.0                                                    # Move the robot elbow upward at position 1.0

rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[-0.5, 0.0, 0.0]' '[0.0, 0.0, 0.0]'          # Command to move the robot backward

rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[0.0, 0.0, 0.0]' '[0.0, 0.0, -1.5]'          # Turn the robot for 15 degree
rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[0.0, 0.0, 0.0]' '[0.0, 0.0, -1.5]'          # Turn the robot for another 15 degree
rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[0.0, 0.0, 0.0]' '[0.0, 0.0, -1.5]'          # Turn the robot for another 15 degree


rostopic pub -1 /mobile_base/commands/velocity geometry_msgs/Twist -- '[0.5, 0.0, 0.0]' '[0.0, 0.0, 0.0]'           # Command to move the robot foward

rostopic pub -1 /elbow_controller/command std_msgs/Float64 -- 2.0                                                    # Move the robot elbow downward


rostopic pub -1 /hand_controller/command std_msgs/Float64 -- 1.0                                                     # Release the item on hand

```

src > trash_detector.py > ...

```
# import the opencv library
from calendar import c
from statistics import mode
from unittest import result
from matplotlib import image
from detector import model
import cv2
import numpy as np
# import rospy


if __name__ == "__main__":
    # rospy.init_node('trash_detector', anonymous=True)

    # Import labels
    with open('labels.txt', 'r') as file:
        labels = list(map(lambda x: x.strip(), file.readlines()))

    # define a video capture object
    vid = cv2.VideoCapture(2)

    while(True):

        # Capture the video frame
        # by frame
        ret, frame = vid.read()

        # Input the image into the model to get prediction result
        pred = model(frame)

        # Extracted all the predicted bounding boxes
        bboxes = pred.pandas().xyxy

        # Convert pandas dataframe into dictionary
        dict_result = bboxes[0].to_dict()

        # Obtained the number of predicting objects
        n_results = len(dict_result['name'])

        annotated_img = frame

        for i in range(n_results):
            # Get the ith bbox start point
            start_point = (int(dict_result['xmin'][i]), int(dict_result['ymin'][i]))
            # Calculate the ith font location
            font_point = (int(dict_result['xmin'][i]), int(dict_result['ymin'][i]) - 20)
            # Get the ith bbox end point
            end_point = (int(dict_result['xmax'][i]), int(dict_result['ymax'][i]))
            # Get the ith prediction result
            pred_class = dict_result['class'][i]
            # Get the label mapping
            pred_label = labels[pred_class]
            # Get the confidence score
            accuracy = dict_result['confidence'][i] * 100

```

https://github.com/LobbeyTan/trash_detector