Technical Assignment

Prerequisite: Please let us have your GitHub account to enable you to contact us on the issue and to clone the repository used for Task 2 deployment. If you have any questions, please contact us on the issue page of our GitHub repository!
https://github.com/roms-jp/TechnicalAssignment_J

Task 1.

In this task we would like to hear your hypothesis (rough planning) on a part of the robot picking systems pipeline.
/TechnicalAssignment_E

Problem definition: In Robot Picking systems the goal is to be able to pick various objects and place them in predefined places. Since those items are in boxes, you must avoid collisions with the boxes when picking. To do this, we need to recognize the three-dimensional position and orientation of the box and know where the walls of the box are located. We assume that we know in advance the intrinsic parameters of the camera capturing the target boxes and products, and the CAD models of the boxes are given. It is also assumed that an RGB image and a depth image (aligned with the color image) are acquired at each pick.

Q.1) Knowledge: Based on the above assumptions, how would you generate the point cloud? If using Python's Open3D, which function would you execute?

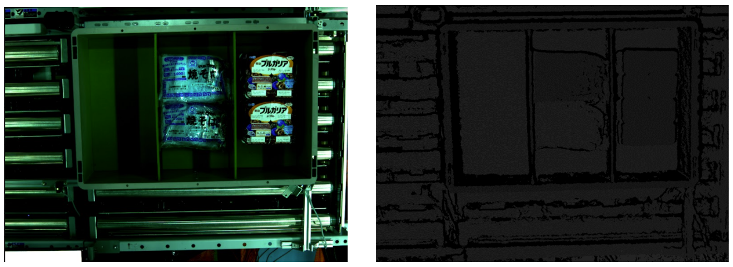
Q.2) Design: How would you estimate the 3D position and orientation of the target box? Please propose the following two approaches using a brief flowchart, considering that products can be fulfilled in the box. But it should not be beyond the top of the box. It's enough that you can share concepts of each approach.

a) From RGB and Depth image b) From Point cloud

Q.3) Assessment and Decision: Please explain the Pros/Cons of the above two approaches. In addition, please describe how you would prioritize the development of the two approaches presented in Q.2)? Please also explain why.

Materials:

- This is a set of examples of captured images (left: RGB, right: depth).



- You can assume the following intrinsics are given:
  - cx, cy: Center of image in x and y coordinates
  - fx, fy: Focal length in x and y coordinates
  - w, h: Image width and height

Note: This task does not require full coding or model training/evaluation.

Task 2. **Object (car and pedestrian) detection.**

Test dataset: 100 images containing cars and pedestrians. Please feel free to use any images you could find online (ref. open images).

Annotation: You can either find the test dataset together with their annotations or you can simply annotate by yourself.

Model: Take any appropriate model you could find online. Please, do not spend time on training (or fine-tuning) the model. The objective of this task is not the model accuracy. The objective is to assess the evaluation and deployment ability.

Q.1) Evaluation: Given a model and the test dataset (100 images), evaluate the model performance in terms of detection performance (please decide on the metrics by yourself) and the inference time.

Q.2) Assessment and Decision: Usually, the developed AI pipeline should be modified considering the business requirements. Let's assume that the business team has two requirements:

1. Not to miss any pedestrians and to have highest possible precision in detecting cars.
2. Detecting the pedestrians as accurately as possible, i.e. the predicted bounding box should fit the ground truth bounding box as much as possible (>0.8 IoU). For "cars", however, rough detection/localization should be fine (0.5 IoU).

Considering the business requirements, please decide on the most appropriate confidence threshold and IoU threshold for each object (4 thresholds in total). After deciding on the thresholds, please report the corresponding precision and recall values for each object (cars and pedestrians).

Q.3) Deployment: We use GitHub for software development, and Docker for the environment. Please share your software so that it meets the following requirements.

a) Please build an environment so that the inference program to a test image implemented in Q.2 can run on Docker. docker-compose is also acceptable.
b) Please clone the following repository and create a branch whose name is "yourname/feature-task2": https://github.com/roms-jp/TechnicalAssignment_J
c) Please share your Dockerfile and the programs needed to run them on GitHub so that we can run them, and include instructions for running them in the README.
d) Please create a Pull Request on GitHub from your branch to the master branch with instructions on which document we are supposed to follow to reproduce your environment.