Yashveer Jain
Tharun Puthanveettil
Pavan Mantripragada

Midterm Proposal

ENPM808X
October 12, 2022

1 Overview

In recent years, because of covid pandemic, there are problems related to the shortage of staff, and maintaining the social distancing norm in restaurant, and it becomes challenging for a restaurant business to sustain. So, to mitigate the problem we introduce the "Robo-Butler". Robo-butler is a cost-efficient fully autonomous robot that serves food to the customers' table from the kitchen in a restaurant with minimal contact thereby ensuring maximum hygiene. It localizes and track human motion to prevent potential collisions which are possible in a social environment like at the restaurants.

1.1 Assumptions

The following are the assumptions considered:
1. Detected human height is 6ft, to estimate depth
2. Robot base frame coordinates are known
3. Resolution of the camera suffice the requirements of the algorithm
4. Occlusion between entities(humans) are absent

1.2 Deliverables

1. A pipeline for obstacle detection.
2. A pipeline to track detected obstacles.
3. Localizing the tracked objects w.r.t. the robot's frame of reference.

2 Process

We extract key requirements from our UML activity and class diagrams to create an initial backlog. Stages of design and testing for the project are done in parallel using a V-model. We will test our detection model using existing datasets to measure accuracy of our model. We only accept the model when its accuracy is greater than 80%. To ensure a successful real-time deployment we will test for FPS above a certain threshold. We will use manually labelled data for tracking error based on difference in centroids and areas of bounding boxes. The google test suite will be employed for designing unit tests to ensure full code coverage and proper operation. For version control and tracking we will be using Git.

3 Organization

Throughout this project we adhere to the guidelines of Agile Iterative Process (AIP). Our team consists of 3 members, with each member having a specific role. The entire development

Tharun Puthanveettil	Driver	writes and maintains code
Pavan Mantripragada	Navigator	guides the driver during coding
Yashveer Jain	Design Keeper	creates the high-level design

Table 1: Team Member Roles

is done in a pair-programming fashion with roles mentioned in the Table 1, and it is done in a span of two week-long sprints. Each of us is expecting to work for 10 hours a week. So our iteration capacity is 30 hours. In the first sprint, requirements from the initial backlog will be implemented, bug catches, and new features will be tracked. The backlog is updated with these new requirements and will be handled in the subsequent sprint. We will discuss the project project progress, resolve conflicts between commits, and clarify ambiguities in changes that are needed in our daily meetings. At the end of each sprint an iteration review is held to evaluate the code and product backlog in preparation for the next sprint.

4 Technologies

The primary language of implementation will be C++ version 11 or above. Code packaging and test would be achieved with the help of CMake. For performing video and image processing , OpenCV version 4.5.6 or above with Apache 2 license will be used. In order to load, customize and modify the outputs of the pre-trained object detection model, Tensorflow 1.6 and Pytorch 1.10 or above will be used. Version control will be incorporated with Git 2.3.

5 Management

We will use Azure DevOps to track the progress of our sprint, the product backlog, and time logs. An iteration backlog chart is maintained to keep track of our adherence to the planned schedule. To effectively measure our code quality and reliability we will monitor change failure rate after every commit. The main elements of the project are segregated into features, these are then sub-divided into requirements to generate our product backlog. We intend to use Team velocity and Sprint burndown charts to track our progress in a sprint.

6 Technical Process

Human detection and tracking as the name suggests necessitate the need for locating humans in the input image feed and identifying each detected person with a unique label. The detection will be performed using Single-Shot Detection models like YOLO or any of its variants. Further to which a Kalman filter based tracking system will be used to ensure continuous

detection and association of detected objects(humans) over subsequent frames. Cost metrics like Mahalanobis Distance and Intersection Over Union(IOU) can be used to evaluate the quality of association and tracking of detected humans over frames. Other detection algortiths like Histogram of Gradient(HoG) and tracking algorithms like Medianflow, Goturn or Lucas Kanade would be tested in parallel. The mentioned would be chosen as a functional alternative in case where the original pipeline fails to generate the expected outcomes.

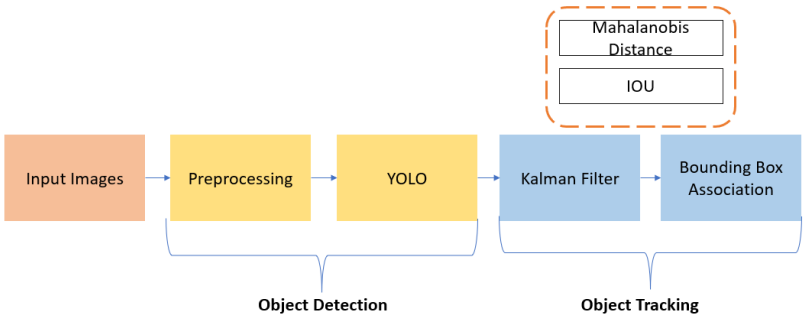

Figure 1: Implementation pipeline

Upon receiving the bounding box coordinates for the detected human in the frame of consideration, it can be converted to the robot's frame of reference by constructing similar triangles as in the Fig 2[**?**]. The computation is possible given the assumption that parameters like Field of View(FoV), Focal length, width and height of humans are known prior.

7 References

1. Software project management plan
2. Rajlich, V. (n.d.). Software Engineering - The Current Practice.
3. Shon Cortes, Sameer Pusegaonkar - Proposal for Vision System for MediBot

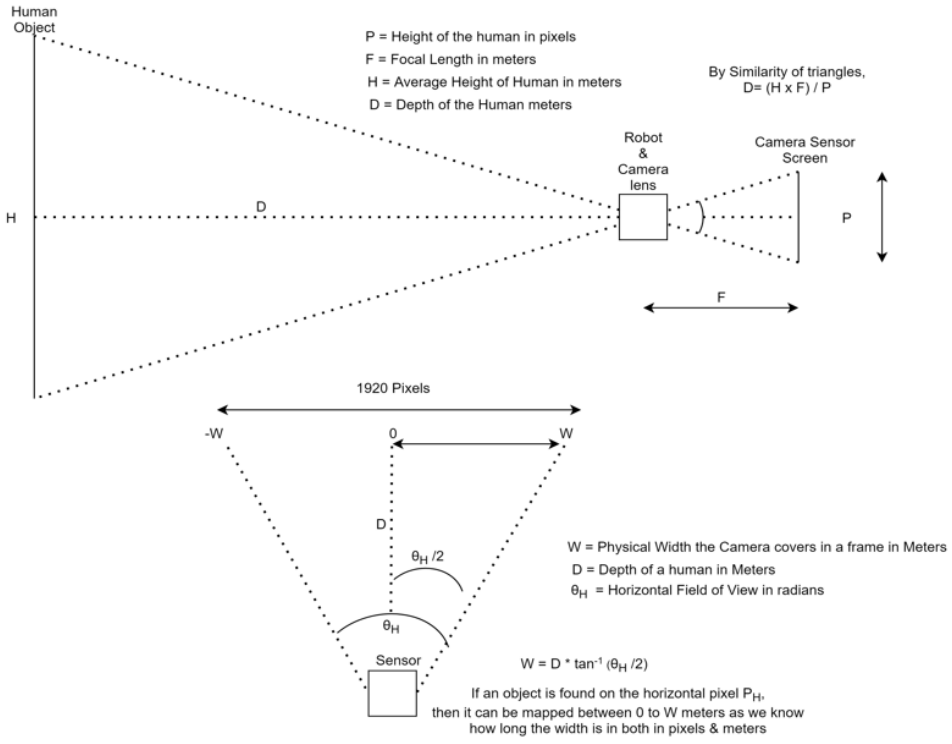

Figure 2: Coordinate conversion from camera-frame to robot-frame