ENPM700: Mid-Term Project Proposal

"Human Obstacle Detection and Tracking System for Warehouse Robots"

Rishie Raj *Uthappa Suresh Madettira*
120425554 *120305085*

1 Introduction

We propose designing a human obstacle detection and tracking system for Acme's warehouse robot, aimed at improving safety and efficiency in material handling operations. This system is vital in environments where human workers coexist with robots, ensuring that the robot can detect and track multiple humans within its operational range and adjust its path accordingly.

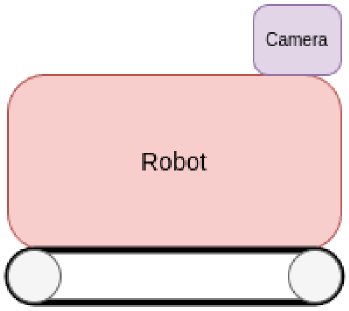

Figure 1: Robot Layout (Side View)

The system will use a monocular video camera to detect and track human presence, outputting positional information directly in the robot's reference frame. This will allow the robot to navigate effectively and safely while avoiding potential collisions with workers.

2 Importance and Usage

An object detection and tracking system is critical for modern robots as they often need to operate in crowded environments. A robust perception system helps the robots to plan a path avoiding obstacles. This particular module is built in a self-contained manner such that it can be integrated directly into any robot system as a package and run as part of the whole system.

The software module designed by us will be used in the Acme warehouse robot for safe navigation on the shop-floor for which it requires detection and tracking of human obstacles.

3 System Overview

3.1 Tools, Technologies and Algorithms

In order to build our module, we will be needing the following tools and algorithms.

- **YOLOv5 for Human Detection:** An efficient and robust object detection model used for real-time use cases.
- **OpenCV for Localization:** A open source library used to perform real-time computer vision tasks. (licensed under Apache 2.0)
- **C++:** The primary programming language for real-time robotics applications.
- **ONNX:** To load the pre-trained YOLO model into the system (licensed under Apache 2.0).

- **CMake:** For building the system.

3.2 Component Functionality

There are the following two most critical components of the module we are designing in terms of functionalities:

- **Human Detection:** The essential part of the perception pipeline is detecting humans in the warehouse shop-floor so that the robot can avoid them during navigation. This is being achieved by using a pre-trained YOLO model which will process each frame from the video feed and detect humans through bounding boxes.
- **Tracking and Localization:** We will be using functions from the OpenCV library for detecting the objects in reference to the robot's camera frame. The coordinate transformation is achieved using OpenCV library functions.

4 Design and Development Process

We will follow the Agile Iterative Process (AIP) for development. Therefore the project will be divided into 2 iterations based on the timeline provided for implementation. Each iteration will improve upon the previous implementation. We are also using the the pair programming approach coupled with test-driven development. So the number of work hours will be divided equally between the two programmers. *Rishie Raj* will be implementing the detection aspect of the project along with designing the unit tests while *Uthappa Madettira* will be implementing the tracking aspect and making sure that the unit tests are satisfied.

5 Risks/Bottlenecks

A major potential bottleneck will be synchronizing the camera feed and the detection module in such a way that there is no lag between them, leading to inaccurate position updates. We plan to ensure this be using open-source functions and algorithms optimized for real-time performance and optimized integration code.

It is also important to make our module robust to disturbances in the environment so that there are no unexpected behaviours of the tracking algorithm. We can ensure this by building proper test cases for our implementations.

6 Final Deliverables

The final deliverable to Acme would include the complete software module that when built and executed would take the camera feed as input and provide the locations of the human obstacles in the robot's reference frame in real-time. The source code will be passing all test cases with a code coverage of greater than 90%, along with proper Doxygen documentation and Google style formatting.

7 References

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 779–788).

[2] OpenCV Official Documentation:`https://docs.opencv.org/4.x/`