Senior Design
ENG EC 464

To: Professor Pisano

Team: 10

Date: 4/10/2025

Subject: Final Testing Test Plan

Final Testing Test Plan

Draper Convoy

By

Team 10

Team Members

Sadman Kabir: kabirs@bu.edu
Benjamin Hsu: bhsu25@bu.edu
Nikhil Krishna: nikhilk@bu.edu
Hugo Silmberg: hsilmber@bu.edu
Michael Moran: mmoran3@bu.edu

Required Materials

Hardware:

- Computer running Ubuntu
- Physical Turtlebot4 Lite:
    - OAK-D-LITE (Camera)
    - RPLIDAR-A1 (Lidar)
    - Raspberry Pi 4B
- Hi-VIS vest
- Mounted tennis ball

Software:

- ROS2 installed on the desktop computer
    - Using ROS2 Jazzy distribution
    - Allows data to be accessed through TurtleBot sensors and communicate with other software
- Vest Detection Node
    - YOLO object detection algorithm
        - YOLOv8n model retrained to identify Hi-VIS vests
    - Identifies Hi-VIS vests beings detected within camera view and outputs information about the target's position relative to the robot
    - Highly accurate, even from long distances and figure partially cut off
    - Vest's design allows it to function seamlessly with any human leader
- Tennis Ball Detection Node
    - YOLOv8n model retrained to identify a tennis ball with greater accuracy
    - Allows the second robot in the convoy to identify the robot that it should be following
    - Takes advantage of the ball's uniform structure by using bounding area to return an accurate distance measurement
- Motor Control Node

        - Accepts data from the vest detection node and outputs information to the control coordinator to control the TurtleBot's motors
        - Used for close-up following and smooth turning
- Path Planner Node:
    - Integrates Nav2 into turtlebot movement, performing obstacle avoidance and handles "going around a corner" cases to redirect the bot efficiently
    - Used for long range following and redirection cases
- Control Coordinator Node
    - Switches between navigation algorithms and rudimentary motor control based on the target's distance. This allows the robot to navigate difficult environments.
    - Velocity output is multiplexed based on distance to ensure correct data is published to motor input and not overwritten

Set-Up

1. Remove Turtlebot from the dock and make sure it starts up correctly. Restart the robot if it does not start up right the first time.

2. On the desktop, set up a new terminal environment and launch our nodes using the launch_robot.sh bash script:

```
./launch_follower_bot.sh
./launch_leader_bot.sh
```

3. Once all nodes are fully started, simply press P in the motor control node terminal to start human tracking.

Pre-Testing Set-Up Procedure

- Undock TurtleBots.
- Run our bash scripts to run and launch all required processes for the system.
- Switch from manual control to tracking mode by pressing p.
- Place Turtlebots into an environment with targets available.
    - Targets should include a marked bot (marked with tennis ball) and a designated human target wearing a HI VIS vest.

Testing Procedure

- Place lead bot (Bot 2) in front of follow bot (Bot 1)
- Place an obstacle in the environment
- Have target wear vest and stand a few meters away from the lead bot
- Launch necessary nodes using 2 bash scripts
- Once tracking, have target move away from the bot and around a corner
- Target proceeds away from the bot while it works around the corner
- Observe the bot navigate the obstacle
- Target proceeds through the environment for as long as possible
- Monitor robot's navigation through the environment during FTL behavior by observing the rviz display and other information terminals

Measurable Criteria

I. The Convoy's ATR system should be able to detect designations, HI VIS vests for humans, and tennis balls for bots, and draw bounding boxes around the targets in the camera frame.
II. Information regarding the relative angle and distance of the detected targets should be published in ROS for other modules to read from.
III. The motor control and path planning nodes should be able to use information from ATR to effectively track and follow targets.
IV. The motor control and path planning nodes should be able to avoid obstacles dynamically during follow-the-leader behavior.
V. The TurtleBot should be able to follow human leaders around corners

Score Sheet

Functionality	**Working? (Y/N)**
Identifies designated targets	
Bounding box around model w/ confidence level, angle calculation, and centroid	
Target seeking behavior	
Stop when close to the target	
Dynamically avoid obstacles	
Follows leader around a corner	
Convoy behaviour (multiple bots) working	