Project 2 with Demo

Al Waleed Al Saleemi ,Rwaida Alssadi , Sara Zuiran.

Introduction

This project involves a robotic system that autonomously navigates and interacts with its environment using an iRobot Create® 3 platform. The primary objective of the robot is to follow a wall, count three distinct doors on its right side, pause at the third door to simulate a cargo acceptance process, and then return to its starting position.
The robot utilizes a combination of wall-following algorithms and collision-based door detection to achieve this goal. The system is designed to function independently, responding dynamically to obstacles and collisions, and providing visual feedback on its current state through lights. Key functionalities include wall-following using PID control, door detection, and side-switching behavior after reaching specific checkpoints.

System Overview

The robotic system is designed with the following core functionalities to accomplish its objectives:

- **Wall Following:** The robot uses Proportional-Integral-Derivative (PID) control to maintain a specific distance from the wall while moving forward. This ensures smooth and accurate wall-following behavior, allowing the robot to stay close enough to detect and respond to door frames on its right side.
- **Door Detection:** The robot detects doors by interpreting collision events as it moves along the wall. Each time the robot collides with an object within a certain time threshold, it considers this as encountering a new door. The robot increments its door counter with each detected door until it reaches the third one.
- **Side Switching**: After counting three doors on the right side, the robot pauses briefly, indicating a state change. It then switches to the left side, resuming its wall-following behavior. This behavior simulates completing one cycle of the assigned task and allows the robot to return to its starting position.
- **Stopping Condition**: The robot continues counting doors on the left side until it encounters the third door again. At this point, it halts entirely, marking the end of its task. This stopping condition ensures that the robot completes the required route in a controlled manner.
- **Collision Handling:** To ensure the robot handles obstacles safely and efficiently, it adapts its behavior based on the side it is following. Collisions trigger predefined responses, helping the robot navigate around obstacles while maintaining its route.
- **Visual Feedback:** The robot uses LED lights to communicate its current state. For instance, it displays different colors to indicate wall-following, door detection, and switching sides, enhancing user understanding and debugging.

Code Structure and Explanation

The code is organized into several functions and handlers to control the robot's movement, door detection, and collision handling. Each key component is outlined below:

- *main_function():*

Purpose: Initializes the robot's navigation and starts the wall-following process on the right side.

Flow: Resets the robot's position and orientation, then initiates the wall-following behavior on the right side, transitioning to left-side following after detecting three doors.

- *wall_following_control(side='right'):*

Purpose: The main control loop for wall-following, which uses PID control to maintain a set distance from the wall.

Parameters: side - specifies whether the robot is following the wall on its right or left.

Functionality: Continuously monitors distance from the wall and adjusts wheel speeds to correct any deviation. Detects doors based on collision events and switches sides after counting three doors.

- *handle_collision(robot):*

Purpose: Responds to collision events to increment the door counter and handle obstacle encounters.

Functionality: Increments the door counter when a new collision is detected, with timing checks to avoid counting repeated bumps as separate doors. Adjusts behavior based on the side being followed.

- *pid_distance(d1, d2, side='right'):*

Purpose: Calculates the PID correction to maintain the robot's distance from the wall.

Parameters: d1 and d2 are distance readings from the robot's proximity sensors; side indicates the wall-following side.

Returns: A correction value for adjusting wheel speeds to maintain the desired wall distance.

- *switch_sides():*

Purpose: Executes the maneuvers necessary to switch wall-following from one side to the opposite side.

Functionality: Turns the robot 90 degrees left, moves it forward a short distance, then turns another 90 degrees left to align it with the opposite wall.

- *set_lights():*

Purpose: Provides visual feedback by setting the LED lights based on the robot's current state.

Functionality: Lights change color according to the robot's state (e.g., wall-following, waiting, switching sides).

- *backoff(side):*

Purpose: Helps the robot move away from obstacles after a collision.

Parameters: side - specifies the side the robot is following, affecting the backoff direction.

Functionality: Moves the robot back slightly, adjusts its angle to avoid the obstacle, and then resumes wall-following.

- *cleanup():*

Purpose: Final cleanup function that runs when the program terminates.

Functionality: Outputs the total count of detected doors and other summary data before closing the program.

Bayesian Network Conditional Probability Tables (CPTs)

Conditional Probability Table: Door Detected | Collision, Wall Proximity

This table shows the probability of a door being detected given the collision status and wall proximity.

Collision	Wall Proximity Close	P(Door Detected = True)	P(Door Detected = False)
True	True	0.9	0.1
True	False	0.7	0.3
False	True	0.3	0.7
False	False	0.1	0.9

Collision Event (C)

This table represents the probability of a collision occurring, which may be informed by prior observations or testing data.

Collision Event (C)	**Probability**
True	0.4
False	0.6

Wall Proximity (W)

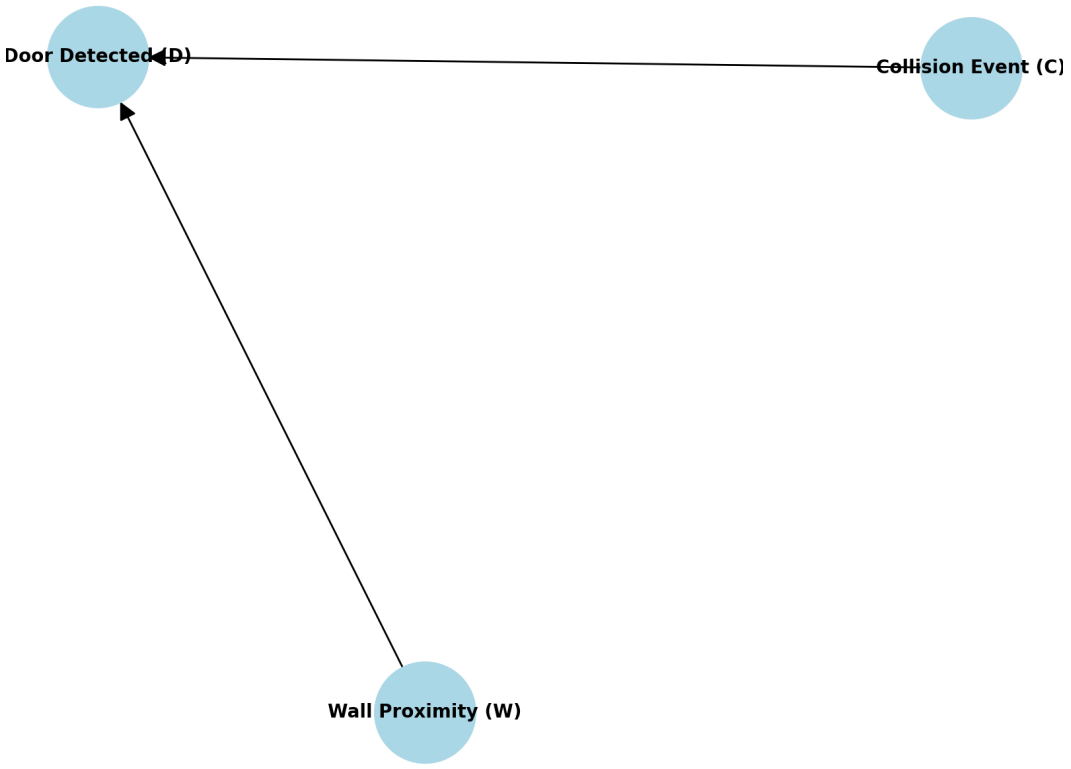
This represents the probability of the robot being close to the wall, based on sensor readings.

Wall Proximity (W)	**Probability**
Close	0.7
Far	0.3

Door Detected (D) | Collision Event (C) and Wall Proximity (W)

This CPT specifies the probability of detecting a door based on the state of collision and proximity to the wall.

Collision (C)	**Wall Proximity (W)**	**Door Detected (D = True)**	**Door Detected (D = False)**
True	Close	0.9	0.1
True	Far	0.7	0.3
False	Close	0.3	0.7
False	Far	0.1	0.9


Door Detected (D)
Collision Event (C)
Wall Proximity (W)