# Autonomous Trash Detection and Navigation Robot

## 1. Project Overview

### Problem Statement

This project focused on the design and implementation of **Botzilla 03**, an autonomous mobile robot built to **detect trash zones, follow walls, avoid obstacles, and navigate an environment using multiple sensors**. The goal was to integrate reactive control, sensor fusion, and structured decision-making into a single, reliable system.

### Objectives

- Detect and respond to “trash zones” represented by black floor markings.
- Switch navigation behaviors when special markers (blue lines) are encountered.
- Maintain robust wall-following with sonar feedback.
- Avoid obstacles dynamically using sonar and recovery maneuvers.
- Track robot position using odometry for analysis and debugging.

---

## 2. Hardware and Sensors

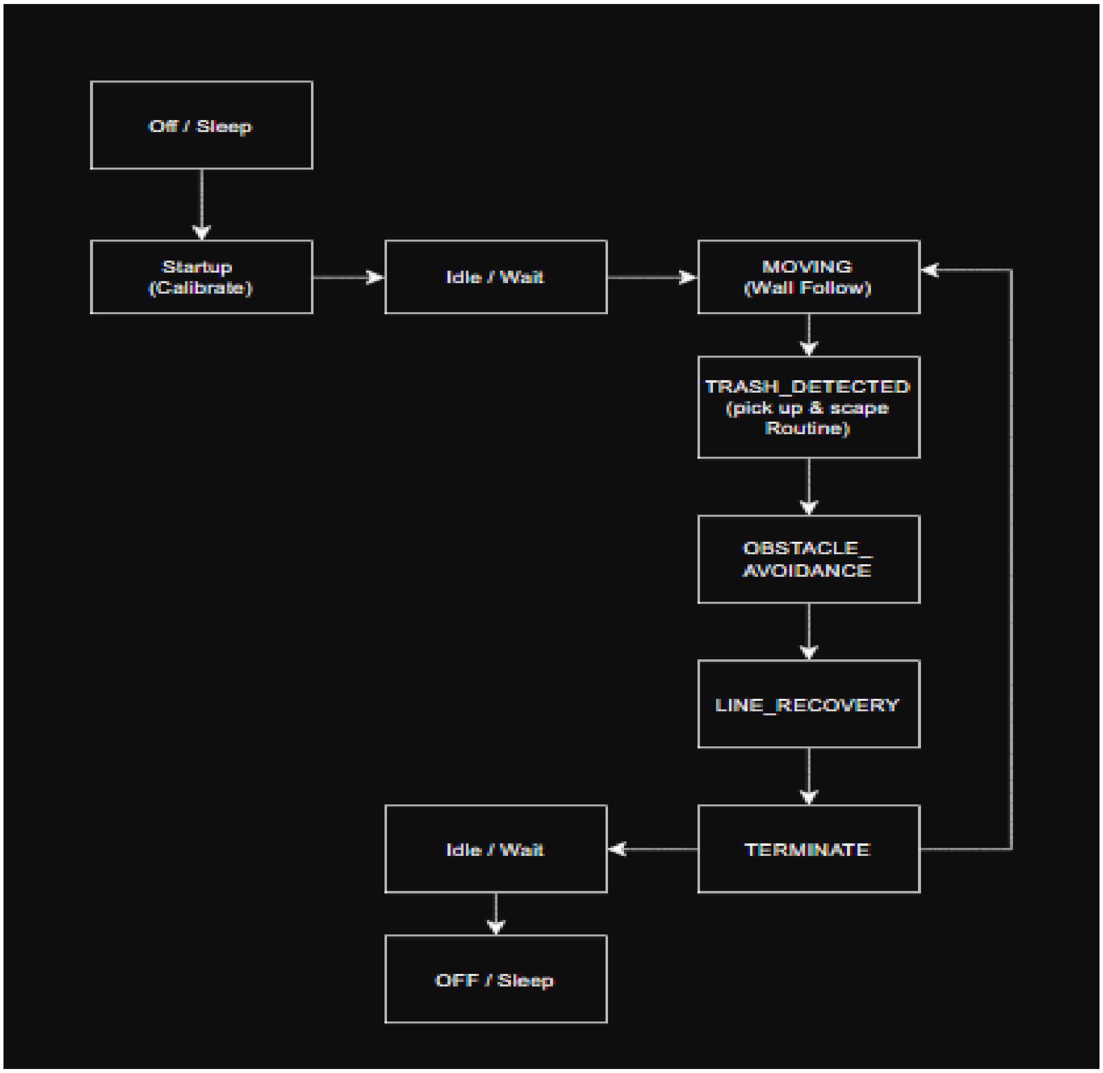
- **Pololu Reflectance Line Sensor Array** – Detects black zones (trash) and blue lines (wall-switch markers).
- **Ultrasonic Sonar (servo-mounted)** – Provides distance measurements in front, left, and right directions for wall following and obstacle detection.

- **Wheel Encoders** – Track displacement and orientation (x, y, θ).
- **Pololu 3pi+ Platform** – Main chassis with DC motors and built-in controllers.

---

# 3. Finite State Automaton (FSA)

## FSA Diagram



### High-Level States

1. **Startup / Calibration** – Initializes sensors and odometry.
2. **Idle / Wait** – Standby mode before starting mission or after recovery.
3. **Moving** – Main navigation loop, wall following via PD control.
4. **Trash Detected** – Executes escape routine to prevent double-counting.
5. **Obstacle Avoidance** – Avoids obstacles detected by sonar.
6. **Rotating (Angle PID)** – Adjusts heading when commanded or required.
7. **Line Recovery** – Corrective behavior when robot drifts off path.
8. **Terminate** – Marks end of navigation mission.

### Transitions (examples)

- **Idle** → **Moving** when mission starts.
- **Moving** → **Trash Detected** when black line is detected.
- **Moving** → **Obstacle Avoidance** when sonar distance < 11 cm.
- **Moving** → **Rotating** on angle correction request.
- **Any active state** → **Failsafe** on sensor fault.
- **Failsafe** → **Idle** after error handling.

**Moving** → **Terminate** when mission is complete.

---

# 4. Control Architecture

### Architecture Type

A **Hybrid Reactive / Behavior-Based** architecture was used:

- **Reactive layer** – Immediate responses to sonar and line sensor input.
- **FSM layer** – Structured state-based decision-making.

### Controllers

- **PD Controller** – Maintains lateral wall distance using sonar feedback.
- **PID Controller** – Manages angular rotation and orientation corrections.

---

## 5. Testing and Validation

### Strategies

- Logged sensor data to tune thresholds.
- Tested wall-following on left and right walls.
- Simulated trash detection with black squares.
- Introduced artificial obstacles to validate recovery logic.

### Metrics

- Accuracy of line detection.
- Distance error from wall (target: 8 cm).
- Number of correct trash detections.

- Time to recover from obstacle/trash events.

---

## 6. Challenges and Solutions

- **Blue Line Detection Failure** – Floor/sensor contrast was too small. Solution: fallback to manual wall switch commands.

- **Double Counting Trash** – Solved by implementing timed escape routines after detection.

- **Odometry Drift** – Mitigated by periodic recalibration with line sensors.

---

## 7. Innovation and Creativity

- Designed a **trash escape protocol** preventing double-count.

- Integrated **servo scanning sonar** for richer environmental awareness.

- Structured FSA design inspired by **commercial cleaning robots (Roomba)**.

- Considered future extension: **vision-based trash classification**.

---

## 8. Personal Contribution & Teamwork

Although this started as a course project, I've refined and expanded it as a **personal robotics portfolio project**.

- I designed the **state machine** and implemented obstacle avoidance and edge-case handling.

- Collaborator Umar tuned wall-following PD control and line detection.

- Together, we debugged transitions, adjusted thresholds, and validated recovery behaviors.

---

## 9. Future Work

- Add **camera-based trash recognition** (computer vision + ML).
- Improve **blue line detection** with color-filtered sensors.
- Implement **path memory** to optimize coverage.
- Extend FSA with **battery-aware docking** similar to Roomba models.