System Design Document

December 5, 2024

Table of Contents

1. EXECUTIVE SUMMARY 3

1.1 PROJECT OVERVIEW 3
1.2 PURPOSE AND SCOPE OF THIS SPECIFICATION 3

2. PRODUCT/SERVICE DESCRIPTION 3

2.1 PRODUCT CONTEXT 3
2.2 USER CHARACTERISTICS 3
2.3 ASSUMPTIONS 4
2.4 CONSTRAINTS 4
2.5 DEPENDENCIES 4

3. REQUIREMENTS 4

3.1 FUNCTIONAL REQUIREMENTS 5
3.2 SECURITY ..... **ERROR! BOOKMARK NOT DEFINED.**
3.2.1 Protection 5
3.2.2 Authorization and Authentication 6
3.3 PORTABILITY 6

4. REQUIREMENTS CONFIRMATION/STAKEHOLDER SIGN-OFF 6

5. SYSTEM DESIGN 6

5.1 ALGORITHM 6
5.2 SYSTEM FLOW 7
5.3 SOFTWARE 7
5.4 HARDWARE 7
5.5 TEST PLAN 8
5.6 TASK LIST/GANTT CHART 8
5.7 STAFFING PLAN 8

1. Executive Summary

1.1 Project Overview

This project, from the CS104's Robotics Triathlon, challenges our ability to program a robot, the Sphero, to complete a series of tasks. Endurance (circumnavigation of a room), Accuracy (repeated navigation through a figure eight staying within the path), and Agility (run successfully through obstacle courses). The robot is required to use specified signals, sounds, and actions from the start to the finish of each task. The intended audience are the students and instructor in the course to observe the run and outcome of the robotic project.

1.2 Purpose and Scope of this Specification

The purpose of this specification is to state the requirements, design, and testing procedures for the robotics project. It will cover the system's flow, algorithmic design, hardware and software requirements, and a test plan. This specification is intended for user/observer(s) on the project.

In scope

- Requirements
- Designs
- Programs
- Test Plan
- Gantt Chart

Out of Scope

Any additional enhancements past the specified tasks.

2. Product/Service Description

2.1 Product Context

This project's product, the robot, relies solely on its onboard sensors, pre-loaded software, and internal hardware components to perform the required tasks making it highly independent versus some products that require being remotely controlled to do certain actions. The robot also operates on its own power supply and only needs to be charged. This allows it to move freely without being tethered to a power outlet.

System Interconnections and External Interfaces

- **Sensors**: Proximity sensors for obstacle detection, gyroscope for path following.
- **LED Indicators and Audio Module**: To provide visual and auditory feedback based on task requirements.
- **Code Repositories**: Used for coding and collaboration (e.g., GitHub, Sphero app).

2.2 User Characteristics

Instructor Profile:

- Role: Mentor, Evaluator, and Project Facilitator
- Experience: Experience in robotics, software engineering, and project-based learning. Familiar with guiding students with hands-on projects involving problem solving.
- Technical Expertise: Proficiency in programming, robotics, and system design.

- Other characteristics: Provides support to students

Student(s) Profile:

- Role: Students are the primary users of the product, developing, designing, and testing the robot as part of their project.
- Experience: Varies, ranging from beginner to having experience with some coding in robotics and programming. Some may be familiar with basic coding but new to applied robotics.
- Technical Expertise: Students should have some foundational programming skills (Python, Javascript, etc.), an understanding of algorithms.
- Other Characteristics: Depending on each student, some may not often work in teams, making it difficult to divide tasks and communicate with one another.

2.3 Assumptions

- Room Availability (Assume the room is available when team members meet)
- Laptops present (Assume laptops/tablets are present to program the robot)
- User Expertise (Assume users have basic understanding of how to instruct the robot through code)
- Operating System (Assumes compatibility with the robot's onboard OS and any integrated sensors)

2.4 Constraints

- Objects in Room (may limit how the robot moves if desks, chairs, etc. are in the way)
- Damaged Tape (may affect robot from staying on the path or determining which direction to go)
- Power (Robot must complete tasks within battery limits without needing to constantly recharge)
- Other design constraints (e.g., design or other standards, such as programming language or framework)

2.5 Dependencies

- Room setup for each task (e.g., paths, obstacles) must be consistent for each test run.
- Robot must be fully charged to perform task uninterrupted.
- Robot must be aimed in the correct direction of the path for each test run.

3. Requirements

Priority Definitions

The following definitions are intended as a guideline to prioritize requirements.

- Priority 1 – The requirement is a “must have” as outlined by policy/law
- Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
- Priority 3 – The requirement is a “nice to have” which may include new functionality

3.1 Functional Requirements

Req#	Requirement	Comments	Priority
ENDUR_01	The robot must successfully circumnavigate the perimeter of the room following the specified path, starting and ending in the designated square.	The robot must stop in the starting square and avoid collisions with objects.	1
ENDUR_02	The robot must stop in the starting square and avoid collisions with objects.	Lights and speech should align with task events for proper evaluation.	1
ENDUR_03	The robot must turn right at the center of each yellow tile and must return to its starting position without deviating from the path.		1
ACC_01	The robot must complete five figure-eight navigations while staying within the defined path.		1
ACC_02	Upon completion of the task, the robot must flash multicolored lights for five seconds and speak the phrase “I am the winner.”	Ensure visual and auditory feedback are integrated into the programming.	2
AGIL_01	The robot must avoid three obstacles during the course without making contact.		1
AGIL_02	The robot must successfully traverse a ramp without stopping or losing balance.	Ensure motor power and speed are calibrated to maintain smooth movement over the ramp.	1
AGIL_03	The robot must knock over as many pins as possible during the final segment of the course.	Strategy for pin-knocking should consider the robot's speed and trajectory.	2

3.2 Security

3.2.1 Protection

Encryption

- All communication between the robot and its controlling device (e.g., laptop or smartphone) are encrypted to prevent interception or tampering.
- Use Secure Bluetooth communication protocols with a passkey to ensure only authorized devices can pair with the robot.

Activity Logging and Historical Data Sets

- The robot maintains activity logs for each task, capturing:
  - Sensor data (e.g., proximity readings, gyroscope data).
  - Motor activity logs (speed, direction, ramp adjustments).
- Historical logs are stored securely in a central repository (e.g., GitHub) to prevent loss and enable post-task analysis.

Restrictions on Intermodular Communications

- Modules (e.g., sensors, motors, and LED indicators) interact through predefined, restricted APIs to prevent unauthorized or unintended interactions.
- Intermodular communication is validated by the robot’s central controller to ensure commands originate only from the approved task execution algorithm.

3.2.2 Authorization and Authentication

User Authentication

- Only authorized team members and the instructor should have access to the programming environment and the robot's operational interface.
- Only Device Bluetooth connected to the robot may operate it.

3.3 Portability

The system is designed to ensure ease of portability and adaptability across various host environments, hardware platforms, and development setups. The following attributes contribute to the portability of the robot's system:

Host-Independent Design

- The software is written in a portable programming language using JavaScript that is widely supported across multiple operating systems, including Windows, macOS, and Linux.

Environment-Independent Performance

- The robot operates independently of external environments. All core functionalities (e.g., movement, sensor integration) rely solely on onboard hardware and pre-loaded software.
- External programming tools (e.g., Sphero app or GitHub) are used only for development and debugging, ensuring the system's behavior remains consistent regardless of the development environment.

Ease of Porting Across Physical Locations

- The robot's lightweight design and battery-powered operation allow it to function without reliance on external infrastructure, making it easy to transport and deploy in diverse locations.

4. Requirements Confirmation/Stakeholder sign-off

Include documentation of the approval or confirmation of the requirements here. For example:

Meeting Date	Attendees (name and role)	Comments
12/06/24	Connor - Role: Team Leader/Coder, Ziad - Role: Planner/Coder, & Ashley - Role: Documentation	confirmed

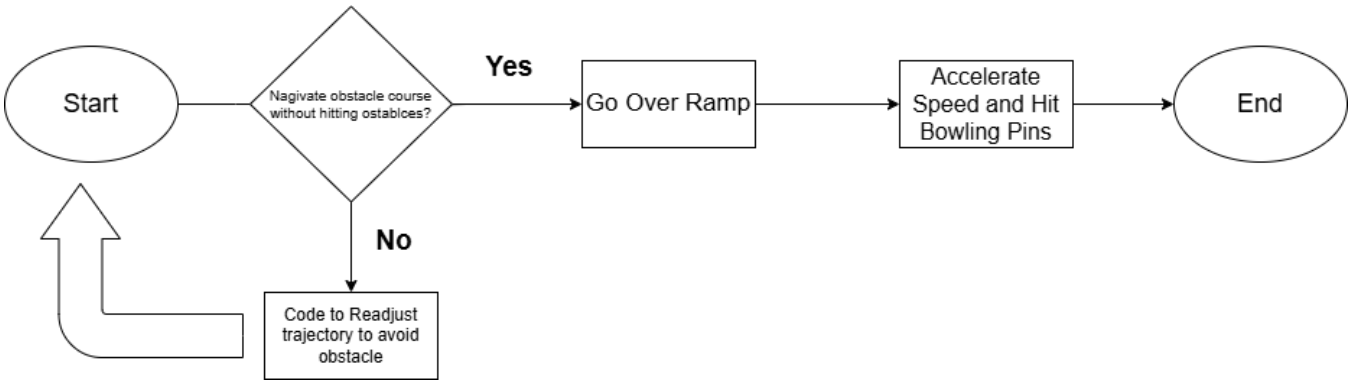
5. System Design

5.1 Algorithm

Agility Task Algorithm:

- **Input**: Proximity sensor data, ramp detection, and pin location feedback.
- **Steps**:
    1. Start
    2. Navigate to avoid obstacles using proximity sensor feedback:
        - If an object is detected within a predefined range, adjust trajectory to bypass.
    3. Adjust motor power to maintain smooth traversal over the ramp.
    4. Accelerate toward the pins and optimize trajectory for maximum knock-down.
- **Output**: Successful completion of the agility course.

5.2 System Flow

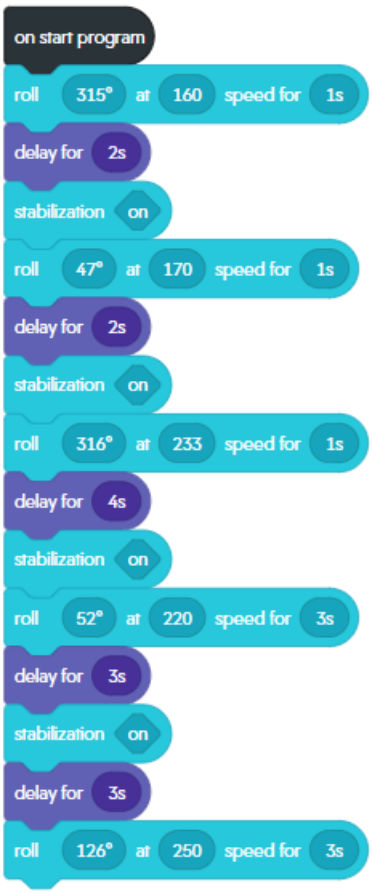


5.3 Software

Sphero Edu App: This platform is used to write and execute JavaScript code directly on the robot. The app provides tools for testing, debugging, and real-time monitoring of the robot's performance.

The **Block Coding** in the Sphero Edu App is used for prototyping and visual programming. It provides an accessible way to implement complex logic by arranging blocks instead of writing raw code. Block code can be converted into **JavaScript** for advanced editing.

Block Code:



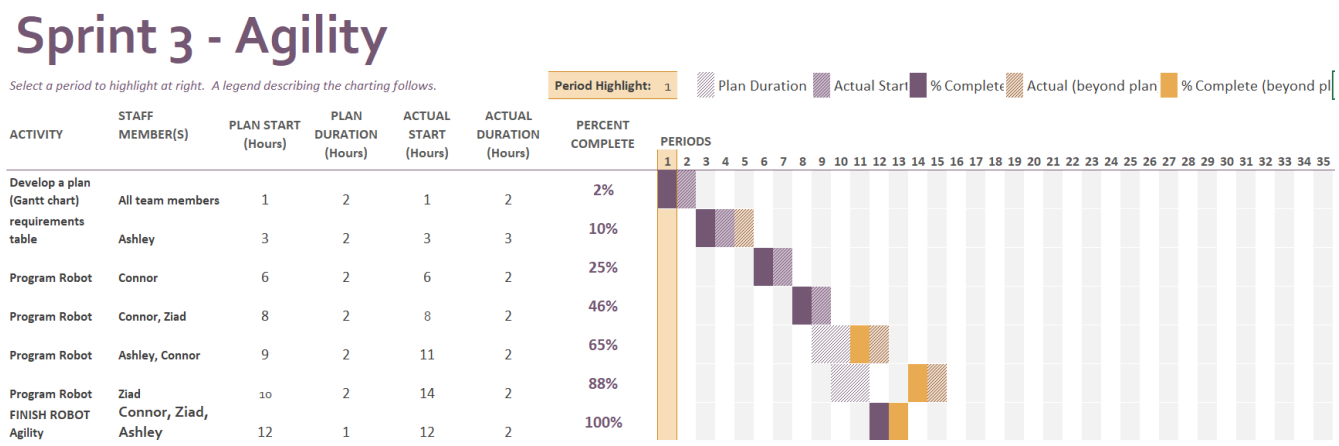
5.4 Hardware

This project used a robot equipped with motors, sensors, and LED lights, capable of rotating, flashing colors, and giving audio feedback.

5.5 Test Plan

Reason for Test Case	Test Date	Expected Output	Observed Output	Pass/Fail
Course Navigation	11/25/24	Correct Navigation	Strayed Off Course	Fail
Course Navigation	11/25/24	Correct Navigation	Stayed on Course	Pass
Obstacle Avoidance	11/26/24	No collisions	No collisions	Pass
Ramp Traversal	11/26/24	Fast incline/pass ramp completely	Successfully passed mounting the ramp	Pass
Pin Knocking	12/5/24	All pins knocked down	10 out of 10 pins knocked	Pass

5.6 Task List/Gantt Chart



5.7 Staffing Plan

Name	Role	Responsibility	Reports To
Connor	Coding	Instructing Robot	
Ziad	Planning & Scheduling/Coding	Creating Schedule/Gantt Chart and Test Plan, Instructing Robot	Connor
Ashley	SDD	Writing SDD	Connor