Sina Nazari, Sina Zaker, Nima Nazari

Abstract

The **Smart Home League** is a prestigious robotics competition that focuses on developing programming, problem-solving, and critical thinking skills in participants. Held within the **Webots simulation environment**, the competition challenges participants to create efficient algorithms for autonomous robots capable of performing specific tasks in a virtual smart home.

The competition fosters innovation by encouraging participants to solve real-world challenges in a simulated environment. Robots are programmed to navigate multiple rooms, avoid obstacles, manage resources like battery life (in U19), and optimize cleaning tasks. Additionally, participants engage in technical challenges and interviews to demonstrate their understanding of robotics and algorithms.

Key Features of the Smart Home League:

- **Educational Objectives:**
    - Enhances algorithmic thinking and programming proficiency.
    - Provides hands-on experience with automation and smart technologies.
    - Encourages teamwork and collaboration in a competitive yet supportive setting.
- **Competition Scope:** The competition focuses on developing autonomous cleaning robots that efficiently navigate and clean home environments. Robots must accurately identify rooms, track cleaning progress, and avoid obstacles while optimizing performance. Participants are challenged to integrate smart home technologies, improve path planning, and enhance cleaning efficiency in dynamic environments.
- **Age Categories:**
    - **First Step** (Elementary School)
    - **U14** (Primary)
    - **U19** (Secondary)
- **Programming Environment:**
  The competition exclusively uses the **Webots simulation platform**, a widely

used tool in robotics research and education. Participants must write their programs in **Python**, a versatile language ideal for both beginners and advanced users.

1. Robots and Sensors

The robots used in the **Smart Home League** are virtual autonomous vacuum robots simulated in the **Webots environment**. Each robot is equipped with a variety of sensors and functional components, enabling it to navigate, clean tiles, and interact with the environment effectively. This section provides a detailed overview of the robots and their sensors.

1.1 Robot Features

- **Autonomous Behavior:**
  The robots operate autonomously, controlled by programs written in Python by the participants. They execute tasks based on sensor inputs and pre-designed algorithms.
- **Mobility:**
  Equipped with wheels, the robots can move in all directions (forward, backward, and turn). The movement is controlled by setting velocity values for the left and right wheels.
- **Task-Oriented Design:**
    - Robots are optimized for cleaning floor within a limited time frame.
    - Efficiency and collision avoidance are key to scoring high points.

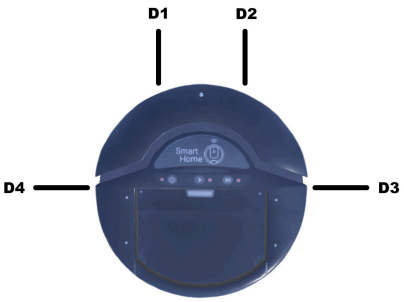
- **Energy Management (U19 only):**
    - U19 robots include a simulated battery system. Participants must consider energy consumption when designing their algorithms.

1.2 Sensors Overview

The following sensors are integrated into the robots to assist in navigation and task execution:

1.2.1 Distance Sensors:

- **Purpose:** Detect obstacles, walls, and furniture in the environment.
- **How They Work:**
  These sensors emit signals and measure the time taken for the signals to reflect back, providing distance data.
- **Placement:** Located around the robot:
    - **Front:** D1 → Front Left, D2 → Front Right
    - **Sides:** D3 → Right, D4 → Left
- **Application:**
    - Prevent collisions by detecting nearby obstacles.
    - Enable path planning and safe navigation.



1.2.2 Bumper Sensor:

- **Purpose:** Detects physical contact or close proximity to obstacles, Prevents collisions and damage to smart home robots or automated systems.
- **How It Works:**
  The bumper sensor detects physical contact or nearby obstacles and sends a signal to the control system.

- The system then stops or changes direction to avoid collisions and ensure safe movement.
- **Application:**
    - Prevent collisions with walls and furniture while cleaning.
    - Detect and avoid obstacles during navigation.

1.2.3 Compass Sensor:

- **Purpose:** Provide directional data relative to the virtual north.
- **How It Works:**
  The compass outputs angular values, allowing the robot to orient itself accurately in the environment.
- **Application:**
    - Determine and maintain direction during movement.
    - Adjust the robot's heading toward a specific target.

1.2.4 Room Number (U14 only):

- **Purpose:** The Room Number sensor tracks which room the robot is in, ensuring efficient and organized cleaning.
- **How It Works:** Each room is assigned a unique ID, and when the robot enters, it detects the corresponding room number, and The system updates the robot's location in real-time to help users monitor progress.
- **Application:**
    - Allows users to see which room is being cleaned at any moment.
    - Helps the robot follow an optimized cleaning path.

1.2.5 Room Cleaning Percentage (U14 only):

- **Purpose:** The Room Cleaning Percentage sensor measures how much of a room has been cleaned. It helps track progress and ensures full coverage of each space.
- **How It Works:** The robot calculates the total area of the room, As it moves, it tracks the cleaned sections and updates the percentage in real-time.
- **Application:**
    - Helps optimize cleaning efficiency, allowing the robot to focus on remaining dirty spots.
    - Can trigger notifications when the cleaning is complete or when certain

areas need extra attention.

1.2.6 Color Sensor (U14 only):

- **Purpose:** Detect surface colors and variations in floor markings within the simulation environment.
- **How It Works:** The color sensor emits light and measures the reflected intensity across red, green, and blue (RGB) channels. Based on the reflected values, it determines the color of the surface beneath or in front of the robot.
- **Application:**
    - Classify floor types (e.g., carpet vs. tile) to adjust cleaning mode.
    - Trigger specific behaviors when a certain color is detected.

1.2.7 GPS Sensor (U19 only):

- **Purpose:** Track the robot's precise position in the simulation.
- **How It Works:** Outputs the robot's coordinates in the environment, enabling location-based decision-making.
- **Application:**
    - Implement point-to-point navigation.
    - Optimize routes for cleaning and returning to charging stations.

1.2.8 Charging System (U19 only):

- **Purpose:** Monitor the robot's remaining energy level.
- **How It Works:** Outputs the current battery level, allowing participants to manage tasks based on energy consumption.
- **Application:**
    - Prioritise tasks based on available energy.
    - Plan for recharging when battery levels are low.

1.3 Customization

- Participants are required to design their algorithms to fully utilise the sensors and motors provided.

- Creativity in combining sensor data (e.g., compass and distance sensors) can result in more efficient navigation and scoring.

2. Competition Platform

The **Smart Home League** operates within a **Webots simulation environment**, allowing participants to design, program, and test autonomous robots in a realistic smart home setup. This platform is essential for simulating real-world scenarios in a controlled virtual environment.

2.1 About Webots

- **Purpose:** Webots provides a detailed simulation of robots, enabling participants to focus on algorithm development without requiring physical hardware.
- **Features:**
    - High-precision physics engine for realistic robot behaviour.
    - Integrated sensor and actuator support, including distance sensors, compasses, and GPS.
    - Easy integration with Python, the programming language used in this competition.

2.2 Required Version

a. Participants must download and use **Webots version 2025a** to ensure compatibility with the competition files and configurations. Download for macOS / Download for Windows

b. **Python Version 3.8 or higher:** To bring our robots to life, we'll need to write controllers using Python. Download

2.3 Access Competition Files:

a. Download the competition files from the **Smart Home GitHub** Navigate to the folder Smart Home\game\worlds.
b. Launch Webots, load the appropriate world file, and test your robot using the base code provided.

2.4 Additional Resources

If you're unfamiliar with Webots or need help setting up, refer to the **Smart Home League Website** for detailed installation guides and tutorials.

---

3. Game Structure

The **Smart Home League** challenges participants to program and control autonomous robots that navigate and perform cleaning tasks in a simulated smart home environment. Teams must design algorithms that enable their robots to move safely and efficiently while detecting and avoiding walls, furniture, and other obstacles within the home. As the robot moves across the field, the surface it covers is considered cleaned, and the final score is primarily based on the percentage of area successfully covered within the given time limit. The competition is structured into three sub-leagues of increasing complexity, allowing participants to progressively demonstrate their skills in navigation, strategy, sensor integration, and energy management.

3.1 First Step Sub-League

- The robot must autonomously navigate the field, avoid walls, furniture, and obstacles, and maximize the cleaned surface within the time limit.
- Scoring is based solely on the total cleaning percentage, with no bonus elements included.

  Final Score = (Cleaning Percentage × K) − (Number of Relocates × K)

  K = Scoring factor (for example k = 40)

3.2 U14 Sub-League

- The objectives are the same as in the First Step category, the robot must autonomously navigate the field, avoid walls, furniture, and obstacles, and maximize the cleaned surface percentage within the time limit.
- Robots are equipped with additional sensors, including a **Color sensor**, **Room Number** and **Room Cleaning Percentage**.
- One or more Boost Stations are placed on the field to provide additional

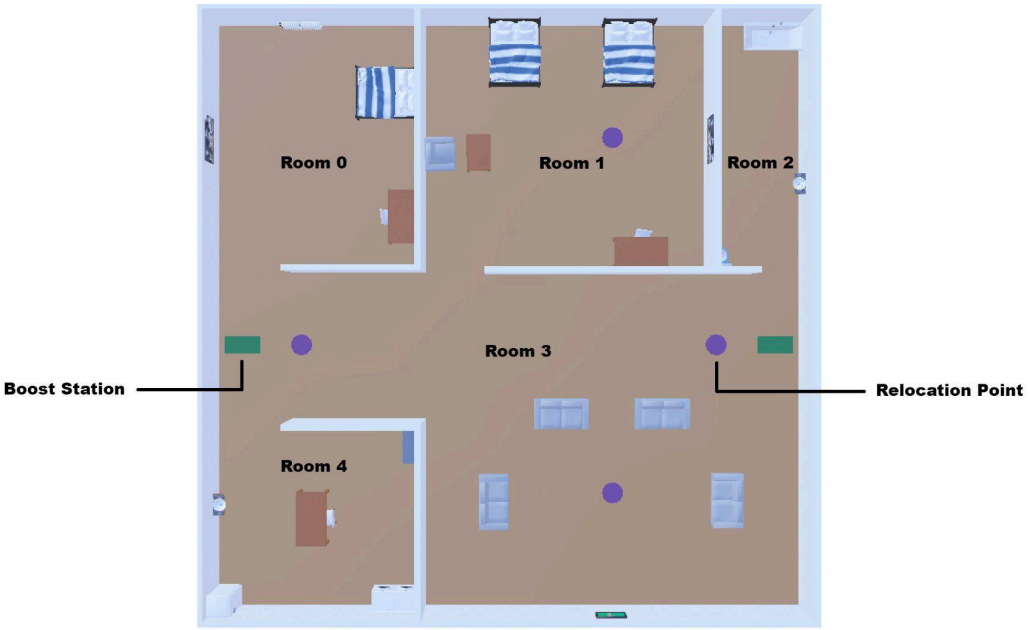
scoring opportunities; when the robot enters a Boost Station area, it receives bonus points, but each station grants bonus points only once per robot, and re-entering the same Boost Station will not provide additional points.

- The robot starting position may change dynamically in each round of the competition; however, the starting position will be identical for all teams within the same round to ensure fairness. Additionally, the field may be presented in **Light View mode**, where only the general structure of the environment is known in advance, and the exact locations of walls, rooms, and Boost Stations are not fully specified.
- The final score is primarily determined by the total cleaning percentage achieved during the match; additional bonus points may be awarded for activating Boost Stations, while penalty points are deducted for each relocate, affecting the overall score.

Final Score = (Cleaning Percentage × K) − (Number of Relocates × K) +

(Number of Boost Activations × 4K)

K = Scoring factor (for example k = 40)



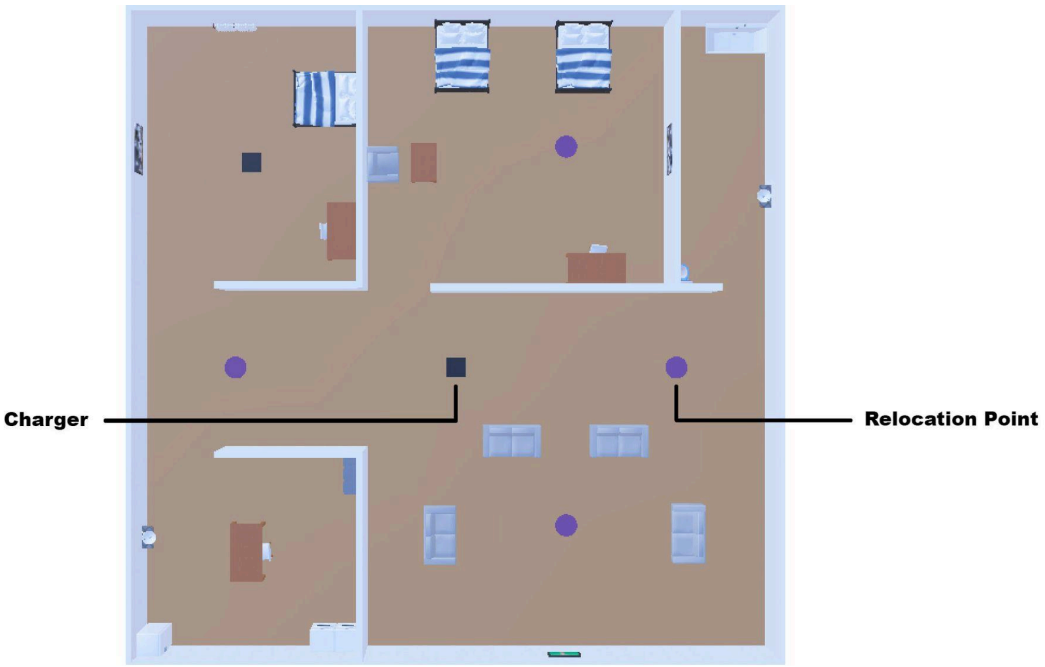
3.3 U19 Sub-League

- The primary objective remains the same as in the lower categories: the robot must autonomously navigate the field, avoid collisions with walls, furniture, and obstacles, and maximize the cleaned surface percentage within the allocated match time.

- In this category, robots operate under an advanced energy management challenge. Each robot has a limited battery capacity and must manage its energy efficiently throughout the match.
- Designated **Charging Stations** are placed on the field. The robot must autonomously navigate to a charging station when necessary to recharge its battery. Strategic timing of charging is essential to maintain optimal cleaning performance.
- The competition map may be provided in **Light View mode**. In this mode, only the general structure of the house is known in advance. The exact positions of walls, doors, charging stations, and even the robot's starting point may vary and are not fully specified beforehand. Additionally, connection issues may occur in certain rooms of the field; in such areas, the robot's **GPS sensor may become temporarily unavailable or non-functional**.
- Teams must design robust and adaptive algorithms that handle partial map information and intermittent sensor disruptions while balancing cleaning efficiency and energy consumption. The map and sensor conditions are **fixed for all teams during a round** and do not change between matches.

Final Score = (Cleaning Percentage × K) − (Number of Relocates × K)

K = Scoring factor (for example k = 40)



3.4 Scoring

3.4.1 Cleaning Percentage

- Points are awarded based on the percentage of clean surface.
- **Objective:** Cover as much of the environment as possible within the time limit.

3.4.2 Technical Challenges

- Participants are given **specific tasks** that test their programming skills and creativity beyond standard robot behavior.
- These tasks are designed to ensure that robots perform actions that are **not part of their usual cleaning functions**. Examples include:
    - **Wall Following:** The robot follows the perimeter of a room while maintaining a consistent distance from the wall.
    - **Room Navigation:** Moving from **Room A** to **Room Z**, stopping for a specific duration (e.g., 10 seconds) before proceeding.
    - **Dynamic Obstacle Avoidance:** Detecting and avoiding moving objects placed in the environment.
- **Objective:** Solve challenges that highlight participants' ability to design and implement advanced algorithms.

3.4.3 Technical Interview

- Participants undergo a technical interview where judges evaluate:
    - **Understanding of the Code:** Participants must explain **all** parts of their algorithms clearly.
    - **Innovation and Problem-Solving:** Creativity in addressing unique challenges.
    - **Team Collaboration:** How responsibilities and tasks were distributed within the team.
    - **External Libraries Understanding:** All team members must know what external libraries they use in their algorithm and have to explain what the library does and how it works.
    - **Extension Tools Understanding:** All team participants must have and explain the source code of every helping tool they use (if any).

4. Rules and Guidelines

This section outlines the comprehensive rules and guidelines for the Smart Home League, covering all aspects of team participation, competition setup, scoring, penalties, and additional considerations.

4.1 General Rules

4.1.1 Teams must consist of **2-6 participants**.

4.1.2 Robots must operate **autonomously** using the Python programs written by the participants.

4.1.3 All participants must adhere to the competition's schedule and instructions from the judges.

4.1.4 Judges' decisions are **final** and binding. Any disputes must follow official written procedures and be submitted professionally.

4.1.5 Teams must **respect other teams, judges, and organisers** throughout the event.

4.1.6 Every object in the world is dynamic and may change and reposition in the competition.

4.1.7 If a team changes its algorithm during the competition, they have to inform the technical committee. If not, all team members will be subject to a severe technical interview or even disqualification.

4.1.8 The Technical Committee is not responsible for any software issues in the teams. It only ensures that at least one laptop in each team can run the webots and the latest version of the smart home software in the U14 and FS sub-leagues.

4.1.9 There is no limit in using any external helping software during the competitions. But the external tool will be subject to section **3.3.3**.

4.1.10 There is no limit in using external libraries. These libraries will be subject to section **3.3.3**.

4.2 Team Requirements

4.2.1 Teams' Code Submission:

Teams must submit their source code **at the exact designated time** before their match.

- Teams are required to place their USB drives containing the files into the **designated container** at the submission point.
- **Late submission** or **incorrect naming** on the USB drive will incur penalties outlined in every competition.

File Naming Requirements:

- Before each match, judges will announce the required naming format for submitted files. The format typically includes:
    - The **match round number**, the **team name**, and the `.py` file extension.
- Example:
    - If the team **Sweeper** is playing in **Round 2**, their file name should be:
      R2-Sweeper.py

4.2.2 Quarantine Rules

- Teams will get a full version of that round's map at the start of each quarantine before every round.
- Teams will have a fixed time to write or modify their code in a controlled quarantine environment before their match. This time varies in every competition.
- During quarantine:
    - **No communication devices** (phones, tablets, smart-watches, internet) or external resources are allowed.
    - Any violations, including communication with individuals outside the quarantine, will result in penalties or disqualification.
    - Entering or exiting the environment is forbidden.

4.2.3 Equipment Requirements

- Teams must bring:

    - **Power strips (multi-plugs)** to ensure smooth operation of their devices.
    - Laptops that meet the following criteria:
        - Fully charged with a **functional charger**.

4.2.4 USB Drive Requirements

- USB drives used for submission must:
    - Be **formatted** and ready for use.
    - Match the team name displayed on the scoreboard.
    - Have a **label/sticker** with the team's name clearly written.
    - It only includes the code related to the round.

4.2.5 Mentor Behavior

- Mentors must not communicate with students during quarantine.
- Any violations will result in penalties, including disqualification of the team for that round.
- Complaints or objections must be submitted **in writing** and professionally. **Unprofessional behaviour may result in mentor suspension or team disqualification.**

4.3 Scoring System

4.3.1 **Task Performance**

- **Scoring Across Rounds:**
  The scores from all rounds are **accumulated**, meaning the total score is the sum of the points earned across multiple rounds (This calculations may vary in every competition).
- **Scoring Based on Cleaning Percentage:**
  Points are awarded based on the **percentage of cleaning** in each round. This is automatically calculated by the system.

4.3.2 **Technical Challenge**

- Refer to Section 3.3.2 for details.

4.3.3 **Technical Interview**

- Refer to Section 3.3.3 for details.

4.4 Relocation and Restart Rules

4.4.1 Each team is allowed up to **3 relocation requests** per match (varies in every competition).

4.4.2 Relocation is permitted only if the robot is stuck against a wall or in a looped position, verified by the judges.

4.4.3 Teams must notify the judges **immediately** to request relocation.

4.4.4 **Special Case:**

- If the robot sinks into the ground or its wheels get stuck due to simulation issues, teams can:
    - Request the referee to pull the robot out at the same position.
    - Request a full match restart for one time, subject to the opinion of the Technical Committee.

4.5 Super Team Event (Optional)

4.5.1 If time permits, an optional **Super Team Event** may be organised.

4.5.2 Teams will collaborate with others to form larger **super teams** and solve a shared task announced on competition day.

4.5.3 Tasks may differ from standard challenges and will test creativity and teamwork.

4.5.4 The winning super team will receive a **certificate of achievement**.

4.6 Penalties

4.6.1 **Unauthorised Intervention:**

- Any **unauthorised intervention** or **modification of the server or game settings** will result in **severe penalties** for the team in that round.

4.6.2 **Late Submission:**

- Teams that fail to submit their files to the **judges** on time will lose **5% of their score per minute delayed** (may vary in each competition).

4.6.3 USB Issues:

Any violation of these rules will result in a disqualification for the relevant round:

- The team name on the USB drive does not match the name displayed on the **scoreboard**.
- The USB drive does not have a **label/sticker** with the team name.
- The USB drive contains **viruses**.
- The USB drive is **not empty** or includes irrelevant files.

4.6.4 **Exceeding Relocation Limit:**

- Each additional relocation beyond the allowed 3 will result in a **5% deduction** from the total score.

4.7 Additional Notes

4.7.1 Teams must use only the official competition files and environments provided by the organisers.

4.7.2 Judges' decisions are **final and cannot be contested**.

- Any team that submits an objection in an unprofessional or inappropriate manner may face **suspension** at the discretion of the judges.

4.7.3 Participants are encouraged to review the rules thoroughly and clarify any doubts before the competition.

4.8 Ways to Contact the Technical Committee

For support, updates, and assistance, participants can connect with the competition's technical committee through the following channels:

1. **Discord:**
    - A dedicated server is available for real-time announcements, technical queries, and community discussions.
2. **Telegram:**
    - Join the official group for quick updates, reminders, and direct communication during the competition.
3. **Website:**
    - Visit the official **Smart Home League website** for:
        - Access to rules and resources.
        - Tutorials and guides.
        - Downloads for competition files and tools.