Smart Home League

Official Rulebook — English Edition

VERSION 3.1 · FEBRUARY 2026

Nima Nazari · Sina Zaker · Sina Nazari · Arshia Sadeghi

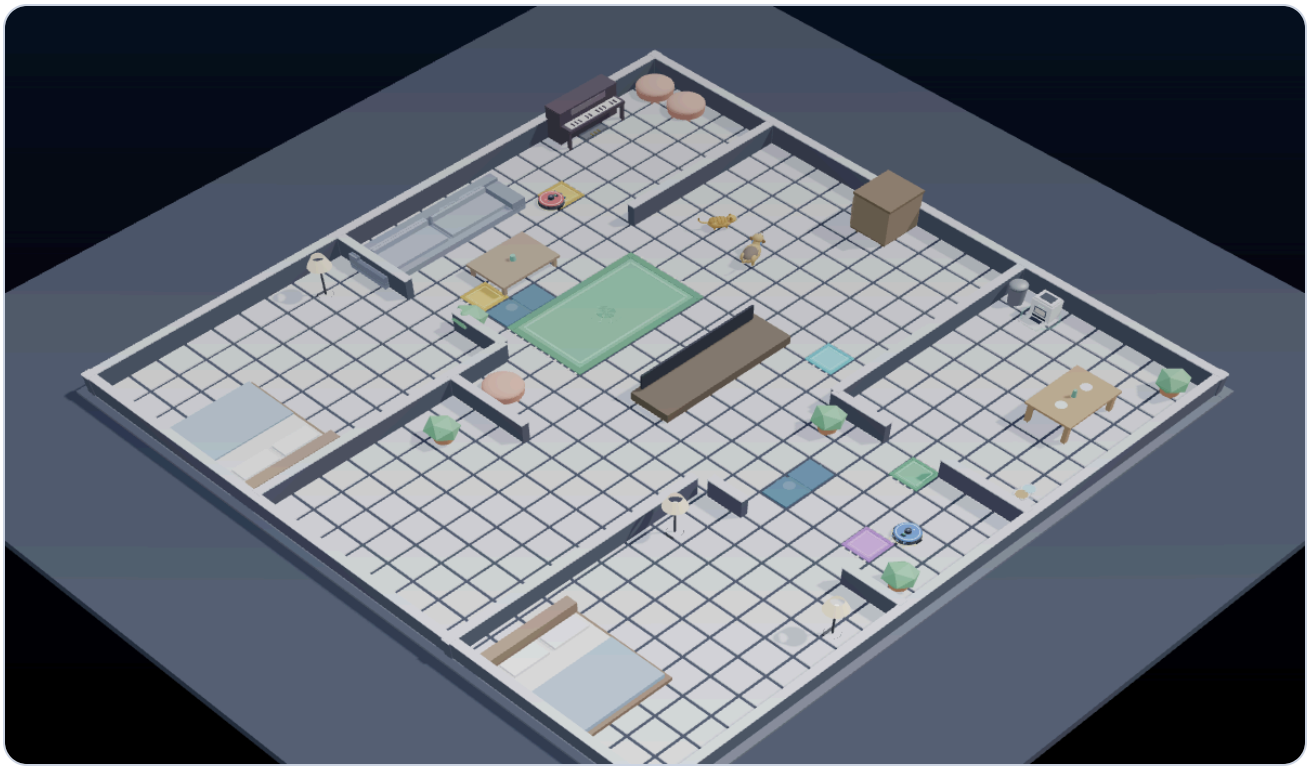

Figure 1 — The official competition field (rendered by the official simulator)

Abstract

The **Smart Home League** is an educational robotics competition that introduces students to the world of **smart homes and the Internet of Things (IoT)**: participants program autonomous robots inside a virtual smart home and learn what it takes to make a home "smart". Its goal is to develop **algorithmic thinking, programming, problem-solving and teamwork**.

The league has several sub-leagues, and the **Smart Vacuum** is one of them — the sub-league whose rules this book describes. The Smart Home League runs at **FIRA** competitions and has so far been held at FIRA **Iran** and FIRA **Canada**.

Everything runs **entirely virtually**, inside the league's own official simulator — the application delivered in your competition folder. The robot, the house, the sensors, the automatic refereeing and the match report

all live in that single program. The robot must navigate several rooms on its own, avoid obstacles, manage its battery in U19, and clean as much of the house as it can. Alongside the matches, participants also take part in a **technical challenge** and a **technical interview**.

Key features

- **Educational objectives:** algorithmic thinking and programming proficiency; hands-on experience with smart-home automation and IoT concepts; teamwork and collaboration in a competitive yet supportive setting.
- **Competition scope:** autonomous cleaning robots that navigate a home efficiently — identifying rooms accurately, tracking cleaning progress, and avoiding obstacles.
- **Divisions:** First Step (newcomers) · U14 (Primary) · U19 (Secondary).
- **Programming language: Python**.
- **Environment:** the league's official simulator — on Windows and macOS, with nothing else to install.

1 · Robots and Sensors

The robots are virtual autonomous vacuum robots. In every sub-league the robot is provided **ready-made**: teams select a team colour and write the control program only.

1.1 Robot features

- **Autonomous behaviour:** the robots operate autonomously, controlled exclusively by the Python programs written by the participants; they execute tasks based on sensor inputs and the team's algorithms.
- **Mobility:** equipped with two driven wheels, the robots move in all directions (forward, backward, turning). Movement is controlled by setting velocity values for the left and right wheels (`wheelleft` / `wheelright`).
- **Task-oriented design:** the robots are optimised for cleaning floor area within a limited time frame; efficiency and collision avoidance are key to a high score.
- **Energy management (U19 only):** U19 robots include a simulated battery system; participants must account for energy consumption in their algorithms.

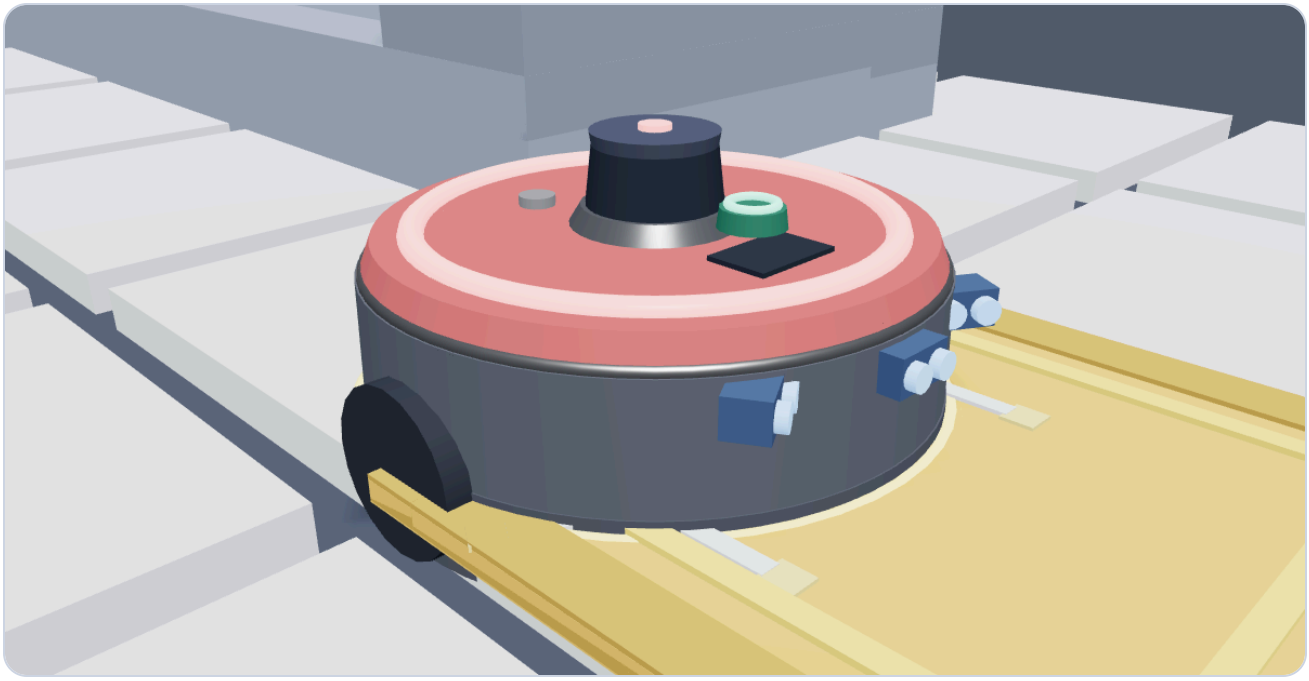

Figure 2 — The competition robot on the official field

1.2 Sensors overview

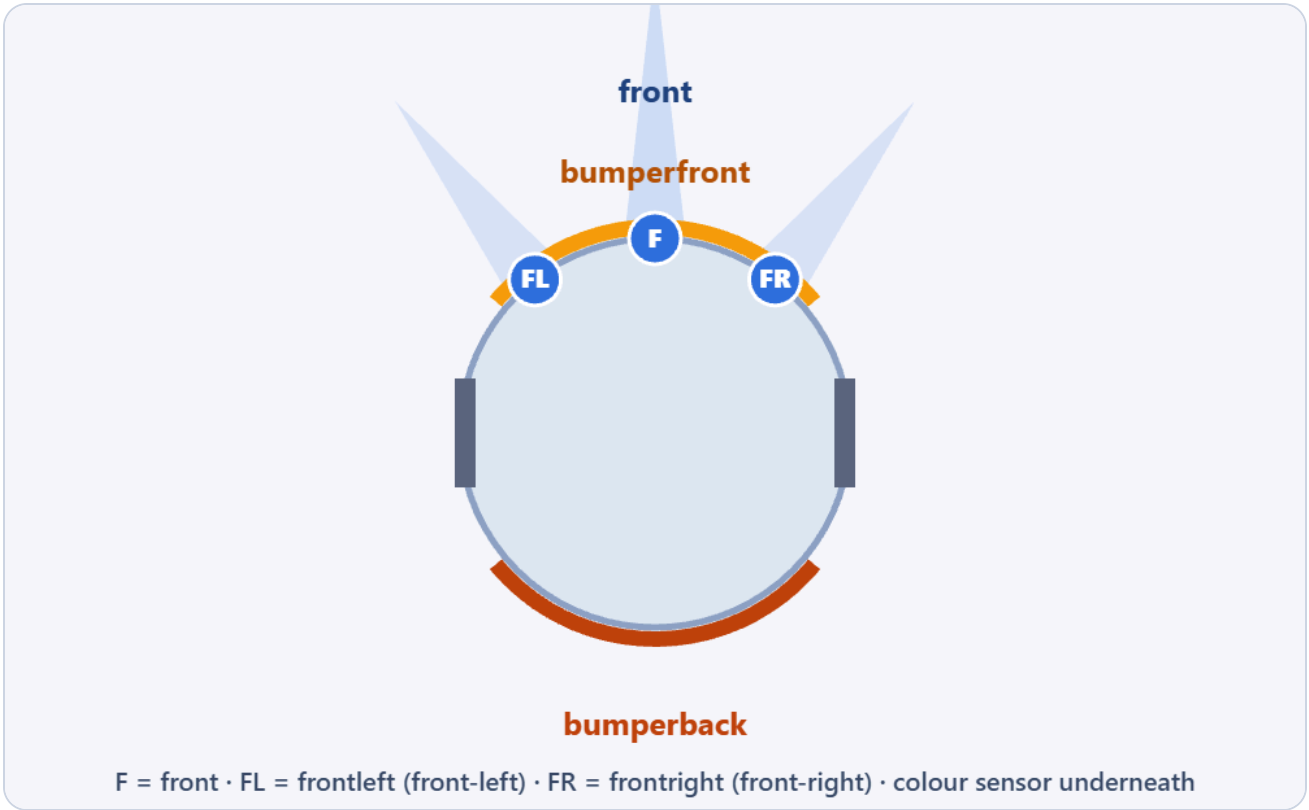

Figure 3 — Sensor placement: the three ultrasonic distance sensors (`front` , `frontleft` , `frontright`), the bumper ring in its two halves (`bumperfront` and `bumperback`) and the drive wheels

Sensor	Function · variable names	Application	Leagues
1.2.1 Distance sensors (Ultrasonic Sensor — US)	Emit an ultrasonic pulse and measure its return time to compute distance. The robot carries **three** of them, acting as its eyes: `front` looks straight ahead, `frontleft` to the front-**left** and `frontright` to the front-**right** — all in centimetres (a smaller number means closer).	Collision prevention; choosing a safe path.	All
1.2.2 Bumper sensor	Detects physical contact or immediate proximity to obstacles and reports it to the control program. Names: `bumper, bumperfront, bumperback` (0/1).	Collision handling with walls and furniture; obstacle avoidance.	All
1.2.3 Compass sensor	Provides directional data relative to the virtual north as an angular value. Name: `heading` (0 = east, 90 = north, 180 = west, 270 = south).	Maintaining direction; orienting toward a specific target.	U14 · U19
1.2.4 Room number	Each room carries a unique identifier; upon entry the robot reads the corresponding number in real time. Name: `room` (0 hall, 1 kitchen, 2/3/4 bedrooms, 5 bathroom).	Organised cleaning; following an optimised room-by-room path.	U14
1.2.5 Room cleaning percentage	Measures how much of each room has been cleaned, updated in real time. Names: `clean1 … clean5` (%).	Tracking progress; leaving a finished room for the next one.	U14
1.2.6 Colour sensor	Measures the reflected RGB intensity beneath the robot and reports the surface colour. Name: `color`, with the predefined constants `white, green, purple, black, red, blue, orange, cyan`.	Classifying floor markings; triggering behaviour on a detected colour.	U14 · U19
1.2.7 GPS sensor	Reports the robot's precise coordinates. Names: `x, y` (cm), together with the navigation functions `goto(x, y)`, `distto(x, y)`, `angleto(x, y)`.	Point-to-point navigation; route optimisation, including trips to the charging station.	U19
1.2.8 Charging system	Reports the remaining energy level. Names: `battery` (%), and the station coordinates `dockx, docky`, provided to both robots from the start of the match.	Task prioritisation by available energy; recharge planning.	U19

1.3 Customisation. Participants are expected to design algorithms that make full use of the sensors and motors provided. Creativity in combining sensor data — for example the compass together with the distance

sensors — results in more efficient navigation and a higher score.

2 · Competition Environment

- **2.1 The league's official simulator.** Matches run entirely virtually inside the league's own simulator: the physics engine, every sensor, the automatic refereeing (relocations, penalties, overtime) and the end-of-match report all live in that one application.
- **2.2 Running it.** The folder you received contains two sub-folders, **Windows** and **Mac**. Double-click `Smart Home League.exe` on Windows, or `Smart Home League (Mac).command` on macOS. Nothing else needs to be installed.
- **2.3 Programming language. Python**. The base code, the sample robots and the code-building helpers are inside the application itself (Base code and Helper on each division's front page).
- **2.4 Learning material.** The in-app Tutorial is step-by-step and runnable; each division also receives its own printed lesson booklet.

3 · Game Structure

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.

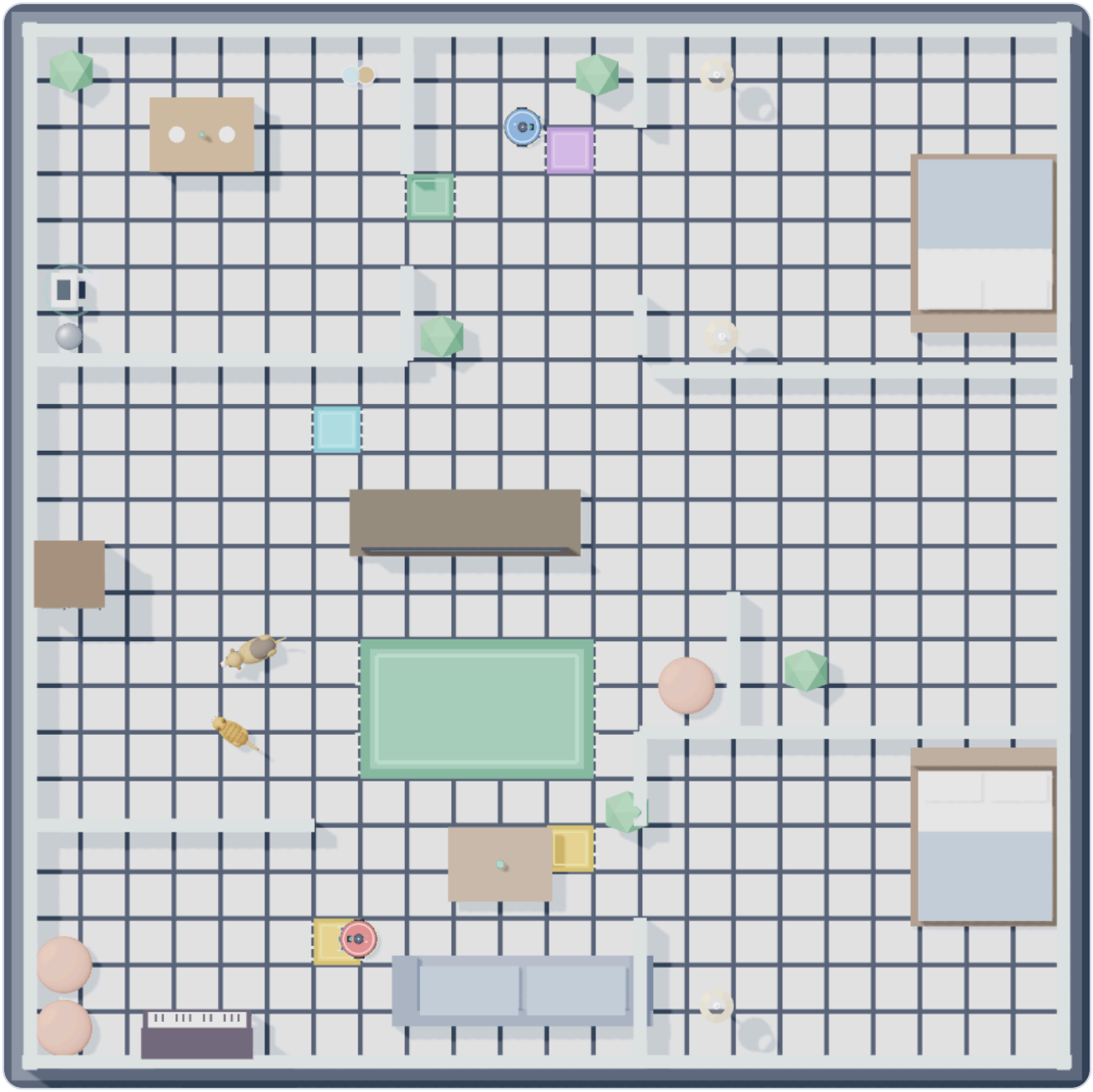

Figure 4 — The official field, orthographic top view

3.1 First Step sub-league

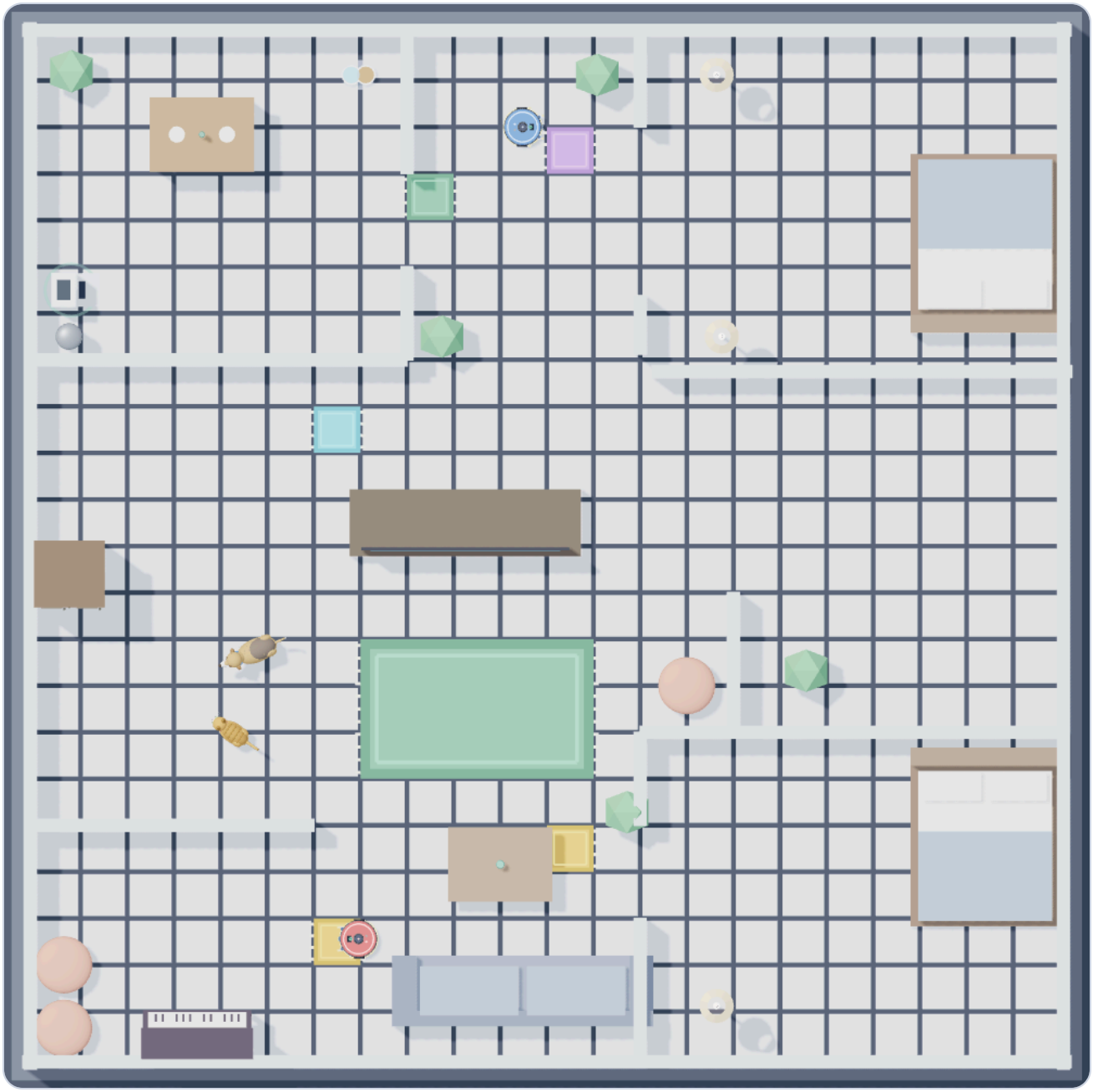

Figure 5 — The First Step field: coloured marker rugs (purple, orange, cyan) identify each doorway. **The house itself is exactly the U14 house** (Figure 6); what differs is the **sensors the robot carries** — First Step has no compass, U14 does — and that **U14's rooms are numbered**, so its robot can read `room` and `clean1..clean5`. In First Step those numbers do not exist.

- The robot must autonomously navigate the field, avoid walls, furniture, and obstacles, and maximise the cleaned surface within the time limit.
- Scoring is based solely on the total cleaning percentage; no bonus elements are 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 identical to First Step; in addition, the robots are equipped with the **colour sensor**, the **room number**, and the **room cleaning percentage** sensors.

- **Boost Stations.** One or more Boost Stations are placed on the field. Entering a Boost Station area grants bonus points; each station grants its bonus **only once per robot** — re-entering the same station yields no additional points.
- **Dynamic starting position.** The starting position may change in each round; within a given round it is identical for all teams to ensure fairness.
- **Light View mode.** The field may be presented in Light View, in which only the general structure of the environment is known in advance; the exact locations of walls, rooms, and Boost Stations are not fully specified.

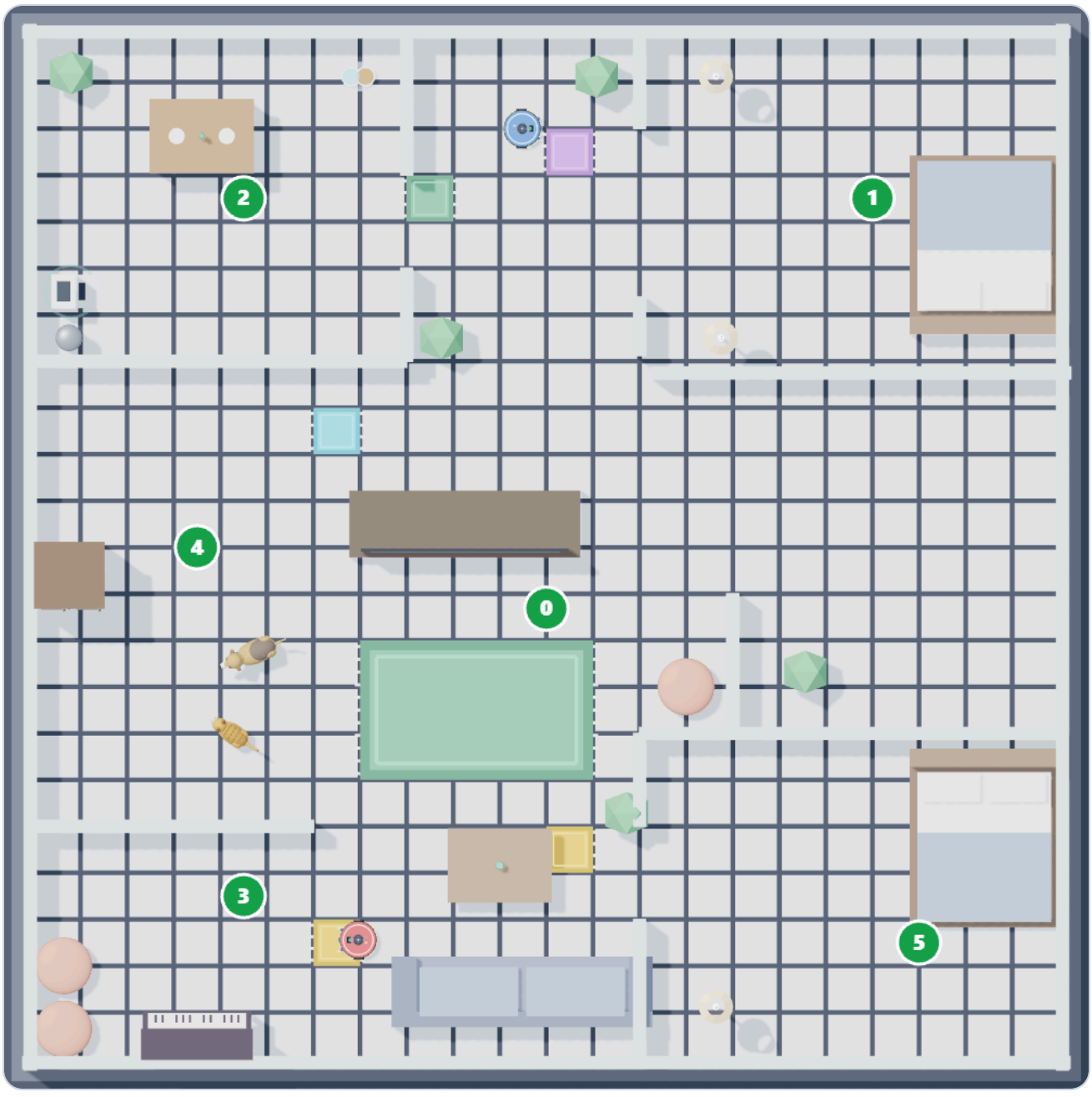

Figure 6 — The U14 field: numbered rooms (0 hall, 1 kitchen, 2/3/4 bedrooms, 5 bathroom) with doorway marker rugs

```
Final Score = (Cleaning Percentage × K) − (Number of Relocates × K) + (Number of Boost Activations × 4K)
```

3.3 U19 sub-league

- The primary objective remains unchanged; in addition, the robots operate under an advanced **energy-management challenge**: each robot has a limited battery capacity that must be managed efficiently throughout the match.
- **Charging stations.** Designated charging stations are placed on the field. The robot must autonomously travel to a charging station when necessary; strategic timing of recharging is essential for optimal cleaning performance.
- **Light View mode.** The map may be provided in Light View: only the general structure of the house is known in advance, and the exact positions of walls, doors, charging stations, and even the starting point may vary.
- **GPS disruptions.** Connection issues may occur in certain rooms; in such areas the GPS sensor may become temporarily unavailable. Teams must design robust, adaptive algorithms that handle partial map information and intermittent sensor disruption while balancing cleaning efficiency against energy consumption.
- The map and sensor conditions are fixed for all teams within a round and do not change between matches of that round.

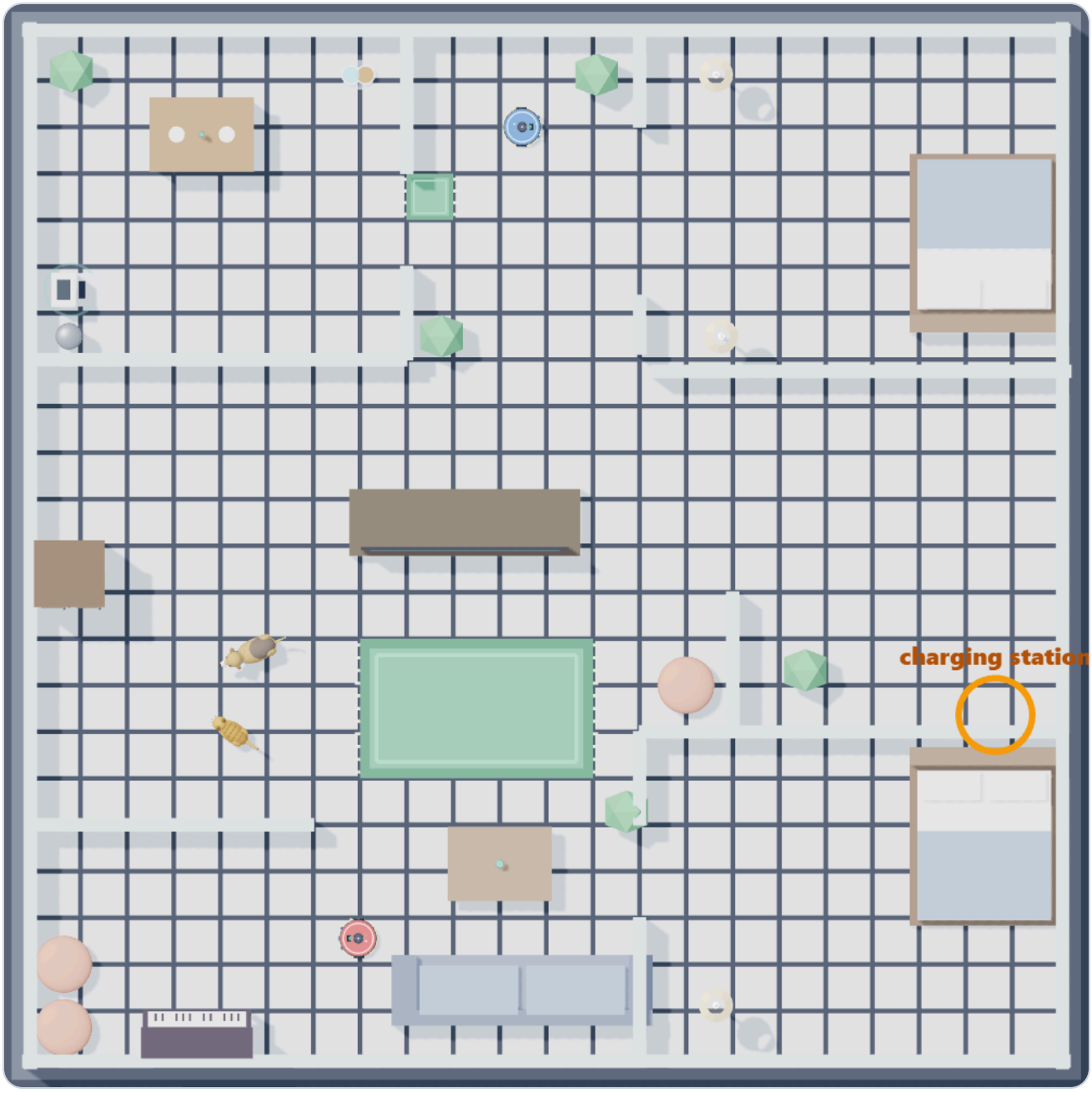

Figure 7 — The U19 field: no doorway marker rugs (navigation is coordinate-based); the charging station is highlighted

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

Simulator field rules (U19). A battery at 0% does not terminate the robot: it continues at 15% speed and resumes charging if it reaches the pad. Queueing behind an occupied charging pad is legitimate: a low-battery robot waiting beside the pad is not considered stuck, whereas a fully charged robot occupying the pad is subject to the automatic 15-second relocation watchdog. Waste disposal is currently disabled.

3.4 Scoring

3.4.1 Cleaning percentage

Points are awarded based on the percentage of cleaned surface. The objective is to cover as much of the environment as possible within the time limit; the value is computed automatically by the system.

3.4.2 Technical challenges

Participants are given specific tasks that test programming skill and creativity beyond standard robot behaviour. Examples include:

- **Wall following:** the robot follows the perimeter of a room while maintaining a consistent distance from the wall.
- **Room navigation:** travelling from Room A to Room Z, stopping for a specified duration (for example, ten seconds) before proceeding.
- **Dynamic obstacle avoidance:** detecting and avoiding moving objects placed in the environment.

3.4.3 Technical interview

- **Understanding of the code:** participants must be able to explain every part of their algorithm clearly.
- **Innovation and problem-solving:** creativity in addressing unique challenges.
- **Team collaboration:** how responsibilities and tasks were distributed within the team.
- **External libraries:** all team members must know what each external library used in their algorithm does and how it works.
- **Extension tools:** all team members must be able to present and explain the source code of every helping tool they use, if any.

4 · Rules and Guidelines

4.1 General rules

1. Teams must consist of **2–6 participants**.
2. Robots must operate autonomously, using the Python programs written by the participants.
3. All participants must adhere to the competition schedule and the instructions of the judges.
4. The judges' decisions are final and binding; any dispute must follow the official written procedure and be submitted professionally.
5. Teams must respect other teams, the judges, and the organisers throughout the event.
6. Every object in the world is dynamic and may change or be repositioned during the competition.
7. A team that changes its algorithm during the competition must inform the Technical Committee; failure to do so subjects all team members to a rigorous technical interview or disqualification.
8. The Technical Committee is not responsible for software issues within the teams; it only ensures that at least one laptop per team can run the latest version of the Smart Home software in the U14 and First Step sub-leagues.
9. The use of external helping software is not restricted; any such tool is subject to section 3.4.3.
10. The use of external libraries is not restricted; such libraries are subject to section 3.4.3.

4.2 Team requirements

4.2.1 Code submission

- Teams must submit their source code at the exact designated time before their match, placing the USB drive containing the files into the designated container at the submission point.
- Late submission or incorrect naming on the USB drive incurs the penalties defined for the event.
- **File naming.** Before each match the judges announce the required naming format, typically comprising the round number, the team name, and the `.py` extension. Example: the team *Sweeper* playing in Round 2 submits `R2-Sweeper.py`.

4.2.2 Quarantine rules

- Teams receive the full map of the round at the start of each quarantine.
- Teams have a fixed period, which varies per event, to write or modify their code in a controlled quarantine environment before their match.
- During quarantine, no communication devices (phones, tablets, smart watches, internet access) or external resources are permitted. Violations, including communication with individuals outside the quarantine, result in penalties or disqualification. Entering or leaving the environment is forbidden.

4.2.3 Equipment requirements

- Teams must bring power strips (multi-plugs) and laptops that are fully charged and accompanied by a functional charger.

4.2.4 USB drive requirements

- USB drives used for submission must be formatted and ready for use; must match the team name displayed on the scoreboard; must carry a clearly written label with the team's name; and must contain only the code related to the round.

4.2.5 Mentor behaviour

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

4.3 Scoring system

- **Scoring across rounds:** the scores from all rounds are accumulated; the total score is the sum of the points earned across the rounds (the exact calculation may vary per event).
- **Cleaning percentage:** computed automatically by the system for each round.
- Technical challenge and technical interview: refer to sections 3.4.2 and 3.4.3.

4.4 Relocation and restart rules

1. Each team is allowed up to **3 relocation requests** per match (this may vary per event).
2. Relocation is permitted only if the robot is stuck against a wall or trapped in a looped position, as verified by the judges.

3. Teams must notify the judges immediately to request a relocation.
4. **Special case.** If the robot sinks into the ground or its wheels become stuck owing to a simulation issue, the team may request that the referee extract the robot at the same position, or request a full match restart once, subject to the decision of the Technical Committee.

Automatic watchdog (simulator). A robot that fails to progress beyond a small circle for **15 seconds** is automatically relocated to an open position with the applicable penalty. Exceptions: a robot taking on charge, and a low-battery robot queueing beside the charging pad.

4.5 Super Team event (optional)

If time permits, an optional Super Team event may be organised. Teams collaborate to form larger super teams and solve a shared task announced on the day of the competition; the tasks may differ from the standard challenges and are designed to test creativity and teamwork. The winning super team receives a certificate of achievement.

4.6 Penalties

Violation	Consequence
Unauthorised intervention in, or modification of, the server or game settings	Severe penalties for the team in that round
Late file submission	Loss of **5% of the score per minute** of delay (may vary per event)
USB violations: name not matching the scoreboard; missing label; the drive contains viruses; the drive is not empty or includes irrelevant files	**Disqualification for the relevant round**
Exceeding the relocation limit	Each additional relocation beyond the allowed three: **5% deduction from the total score**

4.7 Additional notes

- Teams must use only the official competition files and environments provided by the organisers.
- The judges' decisions are final and cannot be contested; a team that submits an objection in an unprofessional or inappropriate manner may face suspension at the discretion of the judges.
- Participants are encouraged to review the rules thoroughly and clarify any doubts before the competition.

4.8 Contacting the Technical Committee

- **Discord:** a dedicated server for real-time announcements, technical queries, and community discussion — discord.gg/Pf4ND7bbHR

- **Telegram:** the official group for updates, reminders, and direct communication during the competition — t.me/firasmarthome
- **Website:** the official Smart Home League website — rules and resources, tutorials and guides, and downloads of the competition files and tools — smarthomeleague.ir