Introduction

SCRAP is a modular software platform, specifically designed for the development of autonomous robotic systems. Its distributed and scalable architecture allows it to handle a wide range of complexities, from automating simple household tasks.

System Architecture

SCRAP is structured into three main modules, each with a well-defined role:

- SCRAP.Android: Acts as the user interface, providing an intuitive API for human interaction. The Android app includes speech-to-text (STT) functionality for more natural interaction and a graphical user interface for viewing the robot's status and configuring parameters.
- SCRAP.Net: The computational heart of the platform. It implements:
  - Motion planning: Uses advanced pathfinding algorithms (A*) to generate optimal paths in dynamic and static environments, considering constraints of space, time, and energy.
  - Environment interaction: Handles perception of the environment through sensors (LiDAR, cameras, etc.) and physical interaction with it through actuators (motors, grippers, etc.).
  - Communications: Uses communication protocols (WebSocket, HTTP) to exchange data with other modules and external systems.
  - LLM interface: Integrates large language models to interpret natural language commands and generate complex actions.
- SCRAP.Arduino: The module dedicated to interfacing with the robot's hardware. Manages communication with microcontrollers (typically Arduino) and connected sensors/actuators. Provides a simplified API to control hardware devices and acquire data from sensors.

Command Execution Flow

1. User request: The user sends a command through the Android interface.
2. Command processing: SCRAP.Net receives the command and analyzes it.
3. Planning: The actions necessary to execute the command are planned, taking into account the current state of the robot and the surrounding environment.
4. Execution: Commands are sent to SCRAP.Arduino, which executes them directly on the hardware.
5. Feedback: The robot sends feedback on the execution status to SCRAP.Net, which in turn communicates it to the Android interface.

Environment Mapping and Navigation

SCRAP.Net uses algorithms to build detailed maps of the environment in real time. Maps are used for:

- Localization: Determining the robot's exact position in space.
- Navigation: Planning safe and efficient paths, avoiding static and dynamic obstacles.
- Odometry: Estimating the robot's displacement based on sensor readings.

Hardware Emulator

The emulator integrated into SCRAP.Net allows you to simulate the behavior of the robot in a virtual environment. This feature is essential for developing, debugging, and testing new algorithms and functionalities, without the need for a physical robot.

Technologies Used

- SCRAP.Net: C# and .NET Core for portability and performance.
- SCRAP.Android: Kotlin for Android app development.
- SCRAP.Arduino: C++ for programming Arduino microcontrollers.
- Communications: WebSocket, HTTP, serial.
- Motion planning: A*.

Namespace MainRobot

Classes

Configuration

Program

RobotConfiguration

ServiceRegistration

StatusFile

StatusLogInfo

StatusRobot