an environment is sufficient to constitute embodiment. Others [5] think that more information is needed, such as a certain understanding of an environment and a robot's own "body". RAI follows the latter view by enabling the creation of an agent's embodied identity from versatile data sources. The proposed RAI framework enables users to experiment with different types and configurations of embodiment mechanisms. The dedicated `RAI_whoami` package provides an easy-to-configure way to incorporate embodying information into the *Agents*. This package includes features to automatically convert text, PDFs, and other documents into a vector database. RAI supports multiple vector databases and was deployed with Faiss [3] as a robust solution. The vector database can then enable LLMs with Retrieval Augmented Generation (RAG). The functionality to store other embodiment-related data, such as images or URDFs, is also provided. All of this data is accessible to a RAI *Agent* and can be used to semi-automate the creation of system prompts. It can also be used during the *Agent's* runtime to retrieve additional information, e.g. when handling a user requesting information about the robot's capabilities.

2.3 Example of system configuration

To better understand these abstractions, consider a simple embodied deployment illustrated in Figure 1. In the illustrated example, we have access to a robot, its documentation, and a ROS 2 service that uses an open-set segmentation model GroundingDINO [13]. These can be easily connected to create an embodied *Agent* using RAI. Such an *Agent* would be designed to have two ROS 2 *Connectors* – one for connection with the robot and one with the GroundingDINO ROS 2 Node. The first *Connector* would be coupled with *Tools* designed for robot control, and the second one with a *Tool* for parsing GroundingDINO output into LLM understandable input. The *Agent* could be a simple conversational LLM, or a more advanced system (see Section 3 for more complex examples).

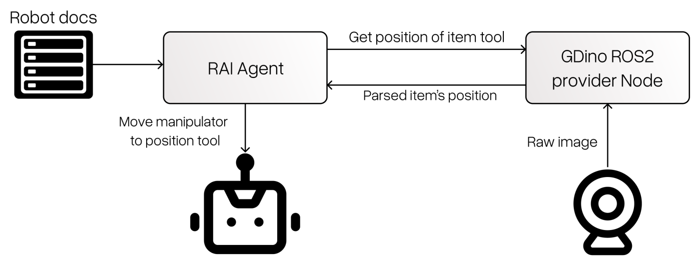

Fig. 1: Simple embodied agent setup

2.4 Flexibility and extensibility

To simplify the deployment of abstractions from Section 2.1, RAI comes with a library of configurable components. Currently, there are four pre-packaged deployable agents:

- `VoiceRecognition` – coupled with configurable models for speech processing pipelines, enabling automatic speech recognition and transcription for human-robot interaction, as well as microphone integration;
- `TextToSpeech` – equipped with direct speaker integration and coupled with text-to-speech (TTS) model;
- `Conversational` (Figure 2) – an agent based on ReAct [25] LLM architecture which utilizes connectors to send and receive multimodal inputs;
- `StateBased` (Figure 3) – an agent based on a finite state machine – a mechanism similar to SPADE's [16] **FSMBehaviour**, which integrates LLM based reasoning with procedural approaches, which can be utilized for more complex tasks.

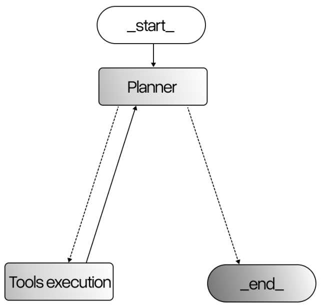

Fig. 2: Conversational Agent

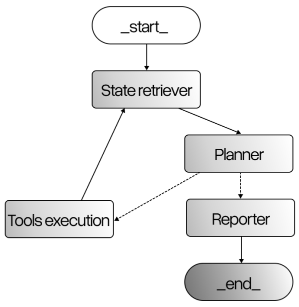

Fig. 3: StateBased Agent

The provided *Agents* come with a set of connectors that enable interoperability through communication protocols (e.g. `ROS2Connector`) and direct peripheral access (e.g. `SoundDeviceConnector`). Connectors are coupled with tools for low-level communication e.g. `CallROS2Service` or `ReceiveROS2Message`. The library also includes *Tools* for data processing, like `GetDistanceToObjects` for spatial analysis of camera and depth data. It should be noted that the RAI components library is under active and continuous development, so the pool of available components is constantly growing. In Section 4 plans for further work are described.

3 Applications

3.1 Navigation and Human-Robot-Integration: Autonomous Mobile Robot

Overview The Husarion ROSBot XL was selected for indoor navigation and HRI experiments. It is an autonomous mobile robot platform for indoor applications. It comes equipped with an Intel RealSense Camera, a Slamtec RPLIDAR S3, and a Bosch BNO055 IMU. Navigation is controlled through the ROS 2 nav2 [14] stack. Two environments were used for the tests: a physical office setting (Figure 4) and a simulated household implemented in Open 3D Engine (O3DE) [4] (Figure 5) with a digital twin of the robot. The double deployment accelerated prototyping and enabled testing of the system's robustness during the transfer from simulation to the real world. In addition, the MAS deployment included a flexible human-robot interface that supports both S2S and text-based interactions via a web user interface.

In the experiments, RAI *Agents* were equipped with open-set detection and generic ROS 2 tools used by the *Agents* for the discovery of robot interfaces and the execution of navigation objectives. To achieve embodiment, the system prompt was initialized with the robot's identity, and a tool for RAG querying was provided utilizing the mechanism described in Section 2.2.

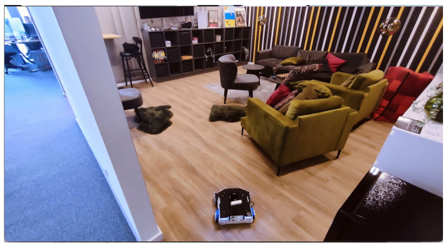
Fig. 4: Physical office setting

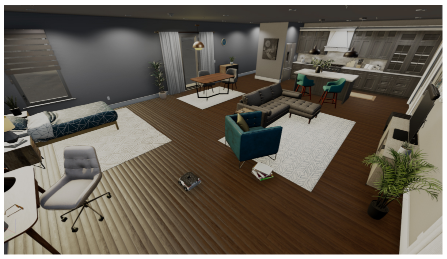
Fig. 5: Household with the digital twin

Configuration A series of experiments was conducted using different architecture configurations. Initially, a single state-based agent (described in Section 2.4) was implemented to handle human interaction, camera image interpretation, and tool invocation. This configuration resulted in advantages such as lower latency, full context retention, and easier debugging. However, it also showed drawbacks: while the robot was performing a mission, the HRI interface was less responsive, as the LLM risked losing track of the mission goal when handling concurrent user queries. To overcome these issues, a multi-agent setup was applied – shown in

[4] `https://o3de.org`