Adding a Cognitive Architecture to ROS 2 for Interactive Task Learning in TurtleBots

Johannes Lopitz
Study program Robotics
and Autonomous Systems
University of Lübeck
Lübeck, Germany
Email: johannes.lopitz@student.uni-luebeck.de

Thomas Sievers
Institute of Information Systems
University of Lübeck
Lübeck, Germany
Email: t.sievers@uni-luebeck.de

Nele Russwinkel
Institute of Information Systems
University of Lübeck
Lübeck, Germany
Email: nele.russwinkel@uni-luebeck.de

***Abstract*—Autonomous mobile robots operating in real-world environments must handle incomplete knowledge and dynamic obstacles. This work presents an empirical study of an integrated cognitive model that combines ROS 2 actions and sensory data with the cognitive architecture ACT-R (pyactr) for a Turtlebot4 robot. The robot operates in a discrete matrix-based environment containing both solid and fake obstacles, while initially only assuming the existence of unknown obstacles. Path planning is performed using the A* algorithm, and encountered obstacles are evaluated through physical interaction. Based on repeated collision experiences, the robot adapts its navigation strategy by switching from shortest-path strategy to a risk-aware one that avoids all obstacles, even unknown ones. Experimental results demonstrate that the cognitive extension enables adaptive strategy selection and memory-based learning of obstacle properties. These findings highlight the potential of cognitive approaches for interactive task learning in mobile robots.**

I. Introduction

Autonomous mobile robots are expected to operate in complex and at least partially unknown environments, where predefined maps and static assumptions are insufficient. For example in emergency scenarios such as building fires, where robots may be required to guide humans to safety despite impaired visibility caused by smoke and dynamically changing environmental conditions. Modern robotic middleware such as the Robot Operating System 2 (ROS 2) provides a modular infrastructure for perception, communication, and navigation [22]. ROS is a cornerstone of modern robotics research and development, accelerating robotics research across different domains with freely available components and a modular framework that robots share, as well as a modular, scalable, and reliable architecture for sensing, planning, mobility, and autonomy [25, 19, 15]. ROS 2 was introduced to address the limitations of the first generation, ROS 1, in terms of real-time performance, safety, and scalability. However, standard ROS 2 navigation pipelines primarily focus on geometric path planning and reactive obstacle avoidance, often assuming that environmental properties are known in advance or can be inferred from sensor data alone [1, 6, 3]. This limits the robot's ability to learn from interaction and adapt its behavior in ambiguous environments.

Classical navigation approaches treat such situations as reactive events without integrating experience into a persistent internal representation, causing robots to repeat the same mistakes and reducing robustness [1, 6]. Cognitive architectures address this limitation by providing structured mechanisms for memory, learning, and decision-making [16]. Unlike sub-symbolic approaches, they explicitly represent past experiences and use this knowledge to guide future behavior [35].

The Adaptive Control of Thought-Rational (ACT-R) cognitive architecture provides simplified models of complex neural and behavioral processes that model cognition through declarative memory, procedural rules, and perceptual-motor processes, making them suitable for learning from interactions in sequential tasks [2]. ACT-R is particularly renowned for its focus on memory and learning, utilizing a declarative memory system for storing facts and a procedural memory system for skills. It has proven itself in applications that require the simulation of cognitive tasks and can therefore be employed to equip artificial agents with flexibility and adaptability for tasks in changing environments, as well as with human-like memory functions.

Since ACT-R's strengths lie in memory modeling, symbolic reasoning, and rule-based decision-making, controlling the robot's perception and motors via the corresponding ACT-R modules would potentially be difficult for effective interaction with the real world in real time, where sensory input and decision-making must be closely interlinked and processed quickly [21, 34]. Therefore, our ACT-R cognitive model focuses on interactive task learning, while the robot's perceptions and motor control are handled via ROS 2 and Python programming on the Turtelbot4 Standard robot [24].

An important feature of intelligent agents like autonomous mobile robots is the balance between reactivity and proactivity [33]. Reactivity refers to the ability to perceive the environment and respond to what is perceived. Proactivity, on the other hand, is the ability to initiate action to achieve goals. If an agent is only reactive, it will have difficulty implementing measures to achieve long-term goals, as it merely responds to the environment. If an agent is purely proactive and goal-oriented, it will take measures to achieve the goals, but will rarely check whether the conditions for selecting these goals still exist, which can lead to pursuing a goal that is no longer possible or relevant. Therefore, the robot must develop a strategy for how it can best achieve its goals in an only partially known environment.

To achieve this, we augmented a ROS 2-based navigation system with a cognitive layer implemented using pyactr to enable interactive task learning in Turtlebot4 robots [4, 9]. The robot incrementally constructs an internal representation of its environment by physically interacting with obstacles. Navigation toward a goal is performed using A* path planning based on currently available information. Path planning enables autonomous agents such as robots or self-driving vehicles to navigate from a starting point to a destination while avoiding obstacles and complying with operational restrictions [23, 32]. Path planning follows an interaction of the robot with the environment which serves as an information-gathering action. Non-traversable obstacles are identified through collision feedback and stored as "solid", while fake obstacles are remembered as "passable" and excluded from future avoidance. This distinction allows the robot to refine its internal map and reduce unnecessary detours.

The cognitive approach further enables adaptive strategy selection based on experience. After repeated interactions with solid obstacles, the robot adopts a conservative strategy that prioritizes safety over optimization, thereby avoiding collisions with unchecked objects. Once the target is reached, the robot returns to its starting position and repeats the task using the shortest-path strategy, exploiting learned knowledge and potentially discovering shortcuts.

This work investigates whether augmentation of an existing robot middleware such as ROS 2 with a cognitive architecture enables interactive task learning through remembering previous experiences to adapt strategies in dynamic and partially unknown environments.

II. Related work

In order for a robot to act autonomously with human-like cognitive abilities, it must be able to accurately perceive its environment, recall relevant past experiences, and make context-based decisions. This is a demanding and complex challenge that requires the use of advanced cognitive architectures that integrate perception, memory, and decision-making processes, thereby improving the robot's adaptability and intelligence [21]. Frameworks like ROS enable the standardization of representations and processes for robotic systems through abstraction and encapsulation of perception and motor functions. This facilitates research into problems in less clearly defined and open areas such as reasoning in complex domains, which were previously considered the domain of cognitive science. Such problems are now being researched under the term "cognitive robotics," which is best accomplished through the use of cognitive architectures specifically designed for general problem solving and modeling human cognition [17].

Such cognitive architectures are now an essential part of the development of intelligent autonomous robots, as they enable learning, adaptation to dynamic environments, and an adaptive strategy selection. There is a growing trend toward hybrid architectures that combine symbolic thinking with reactive responsiveness [11]. ROS has become a de facto standard that facilitates reproducibility and knowledge transfer between research groups and commercial applications in robotics.

Silva et al. investigated the use of cognitive agents in conjunction with ROS for programming intelligent mobile robots, utilizing the capabilities of the ROS layers for the robot to interact with its sensors and actuators [29]. They developed an agent-oriented programming language for their cognitive agents, which controlled simulated unmanned aerial vehicles (UAVs). Gonzáles-Santamarta et al. used the MERLIN2 cognitive architecture integrated into ROS 2, enhanced by the reasoning capabilities of LLMs, to enable planning and reasoning in autonomous robots [10]. They also presented SAILOR, a framework for symbolic anchoring integrated into ROS 2 [12]. Symbolic anchoring enables cognitive-based robots to obtain symbolic knowledge from the perceptual information generated by sensors and maintain the link between that knowledge and the sensory data.

An extension of the ACT-R cognitive architecture for embodied use was introduced by Trafton et al. to enable researchers to create embodied models of humans and predict how a person will behave in different situations [31]. Smart et al. integrated communication between ACT-R cognitive models and the Unity game engine for virtual robots that can move in a 3D environment and interact with various objects [30]. Information from the robot's sensor systems was forwarded to the ACT-R model, which used this information to control the robot's behavior in order to find target objects and deposit them at a specific destination.

In the field of social robotics, cognitive architecture can help robots develop social skills from social encounters with humans by storing experiences from human-robot interaction (HRI) in declarative memory [13, 28]. This requires the ability to perceive and respond to their environment in such a way that they can recognize and understand the social context and learn from experience.

In order to overcome limitations in search and planning algorithms for agents that are required to move from a starting point to a destination in a shared environment, research has recently shifted toward decentralized and learning-based approaches [8]. Such multi-agent reinforcement learning algorithms learn through experience in their environment and thus find good solutions through interactions with that environment.

The complexity of path planning algorithms depends on robot parameters such as motor speed or sensor range, and on external parameters including map accuracy, environment size, and number of obstacles [27]. To achieve the best result for the shortest path, Dijkstra's is a well-known search algorithm, but A* can perform the search for the shortest path faster, although it does not always deliver optimal results [5]. To address the issues of low search efficiency, excessive node expansion, and poor real-time capability in the traditional A* algorithm, an improved A* algorithm for mobile robot path planning has been proposed [18, 7]. However, these methods do not optimize the route based on experience and memory.

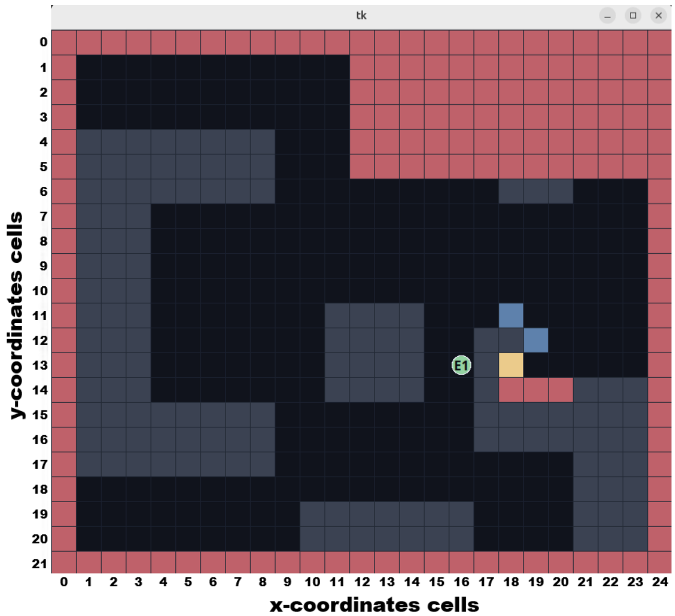

Fig. 1. Discrete two-dimensional matrix map from the parallel running simulation. Each colored square represents an obstacle (red: solid, grey: unknown, blue: passable, yellow: target). The circle represents the robot

III. Methods and Materials

A. System Architecture

The proposed system consists of a layered architecture combining low-level robot control and a second cognitive decision-making layer as well as an adapter for the second layer [16]. ROS 2 is responsible for sensor processing, motion execution, and communication, while the cognitive level is implemented using pyactr, a Python package for creating and executing ACT-R cognitive models that supports symbolic and sub-symbolic processes in ACT-R and covers all basic cases of ACT-R modeling [14]. In addition, an extension is used for multi-agent simulations, which simplifies the connection of powerful external modules, such as robot controllers, to the cognitive core system of pyactr [20]. All low-level functionalities, including bumper sensing, odometry, and velocity control, are implemented as ROS 2 nodes.

The environment is represented in an internal simulation as a discrete two-dimensional matrix map (see Fig. 1). Cells can be either free or obstructed, but at the beginning of each run, the robot has no knowledge about the true nature of obstacles in the environment. All obstacles are therefore initially treated as "unknown" obstacles.

The cognitive layer consists of multiple states and phases, as can be seen in Fig. 2. It maintains a persistent memory from which information about previously encountered obstacles and their evaluated properties can be retrieved. This memory is implemented using pyactr's declarative memory, continuously updated during navigation, and persists across multiple planning cycles within a task. After initially locating the robot's position, pathfinding begins by attempting to reach the destination using the shortest path strategy around unknown obstacles.

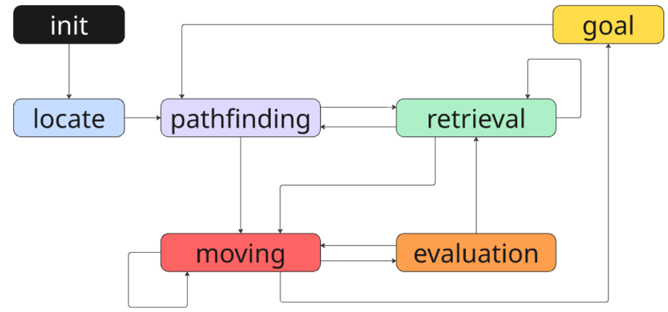

Fig. 2. Cognitive model consisting of phases the robot will experience throughout its journey to the goal

Knowledge about the actual nature of the obstacles along the way is gained through successful passages or, otherwise, collisions. This knowledge is constantly updated as chunks in the declarative memory, and the map from the simulation keeps track of the path. By explicitly representing obstacle knowledge at the cognitive level, the robot can base future decisions not only on its current sensory input, but also on accumulated experience from prior interactions.

B. Navigation and Path Planning

Navigation within the matrix-based environment is performed using the A* algorithm, which computes the shortest path from the robot's current position to a specified goal cell. The A* (A-Star) algorithm is a direct search method for calculating the shortest path in a static traffic network [23, 32]. Due to its robust search capability, completeness, and optimal efficiency, it is used in many areas, such as autonomous driving, UAVs, disaster relief operations, and mine exploration [26]. The path-planning operates on the discrete grid representation and uses a heuristic based on Manhattan distance to estimate the remaining cost to the target. At the start of each new run, only the start and goal positions are known, while all other cells are assumed to be traversable.

This initial assumption reflects the robot's lack of prior knowledge about the environment and encourages exploratory behavior. The robot follows the planned path by executing a sequence of discrete motion commands, each corresponding to a fixed translation or rotation. Movement commands are deliberately kept simple to ensure a clear correlation between planned actions, the simulation represented for the cognitive level, and the movements performed.

When the robot encounters an obstacle, a collision is detected through the bumper sensors. The robot estimates the previously executed translation and rotation based on odometry data and command history. Using this information, the robot performs a corrective maneuver to return to its last valid position before the collision occurred. This maneuver serves a dual purpose. On a functional level, it allows the robot to recover from navigation failures and continue operating. On a cognitive level, the collision is interpreted as an interaction with the environment that provides information about the encountered obstacle.

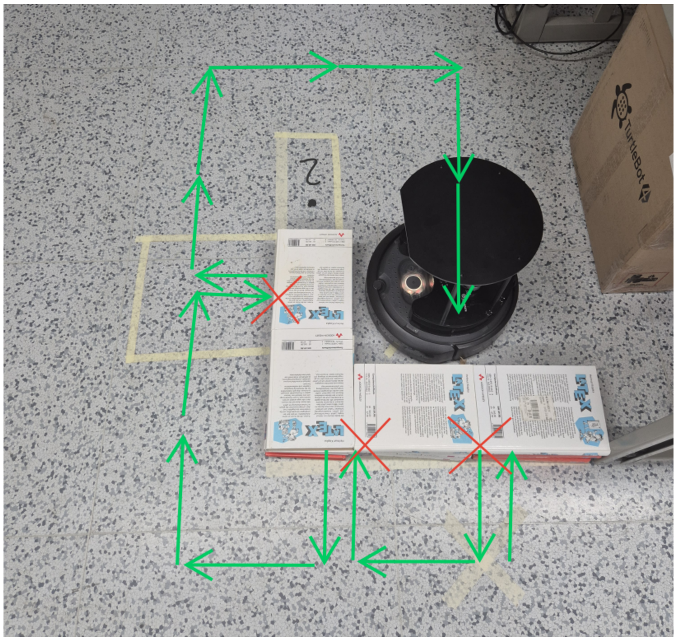

Fig. 3. Real world experiment: First Path from Start to goal, with strategy switch after 3 bumps

C. Cognitive Strategy Adaptation

The core contribution of this work lies in the integration of a cognitive learning and strategy adaptation mechanism into the navigation pipeline. Obstacle evaluation results are processed by the pyactr model and stored in its declarative memory. Each obstacle is associated with a specific map location and classified based on the outcome of the interaction from the robot as ”solid” or ”passable”.

Once an obstacle has been classified as ”solid”, the path-planning is triggered to recompute a new path that avoids the corresponding grid cell. This ensures that previously encountered obstacles are not repeatedly tested, reducing unnecessary collisions and improving navigation efficiency. Obstacles that the robot can pass without a collision are classified as fake obstacles and stored as ”passable” in the declarative memory. These obstacles are subsequently ignored during planning, allowing the robot to exploit shortcuts.

Beyond obstacle classification, the cognitive model also monitors the *number of collisions* of the robot with barriers in the environment. After multiple collisions, the model initiates a strategy switch. For demonstration purposes, the number of collisions after which the strategy switch should take place was set to three. Instead of planning the shortest path, the path-planning then enters a risk-aware mode in which all unknown obstacles are treated as ”solid”. This conservative strategy prioritizes safety and reliability over optimality, thereby preventing further potential collisions.

Once the robot reaches the goal location, it returns to the starting position using the fastest available path and initiates a new navigation attempt using the shortest-path strategy. This deliberate alternation between conservative and exploratory behavior enables the robot to refine its internal map and to identify potential shortcuts that were initially excluded. Through this process, the robot gradually improves its task performance with repeated executions by learning from the conditions of its environment.

IV. Results and Discussion

The experimental results demonstrate that the integration of a cognitive architecture into the ROS 2 navigation framework enables adaptive and experience-driven behavior in environments with uncertain obstacle properties. The initial path-planning is based on incomplete environmental knowledge. As a consequence, these paths frequently led the robot into contact with solid obstacles that had not yet been identified (see Fig. 3). This behavior reflects the expected exploratory phase, in which the robot prioritizes efficiency over safety due to the lack of prior information.

Through physical interaction with the environment, the robot was able to reliably classify obstacles based on collision feedback and remembered them. This memory-based update mechanism prevented repeated encounters with the same obstacles and significantly reduced collision events over time.

A key observation is the effect of the strategy switching mechanism on navigation performance. The switch led to no further collisions, confirming that the conservative strategy successfully increased navigation safety. However, this also led to longer paths and increased traversal time. On the way back to the start the robot was able to exploit previously identified fake obstacles and discovered shorter routes that had been inaccessible during an initial conservative phase.

The strategy adaptation mechanism illustrates how cognitive approaches can support interactive task learning by combining memory, feedback, and goal-directed behavior [9]. Rather than relying on static planning rules, the robot continuously adapts its navigation strategy based on accumulated experience, demonstrating a key advantage of cognitive integration over purely reactive navigation systems.

In addition, the experiments revealed a notable reality gap between the assumed discrete motion model and the robot’s actual physical behavior. The navigation logic relied on fixed translations of 30 cm and precise rotations of 90°or 180°. In practice, wheel slippage, odometry drift, and actuator inaccuracies led to deviations from these idealized motions, causing a gradual divergence between the robot’s true position and its internal grid-based representation. As a result, the robot occasionally collided with obstacles that should not have been encountered according to the symbolic map used with the cognitive model, shown in Fig.1. Consequently, some interactions were misinterpreted as obstacle properties rather than localization errors, as the current cognitive framework does not explicitly represent pose uncertainty.

V. Conclusion

This work outlines an approach for augmenting ROS 2-based navigation with a cognitive architecture in order to enable interactive task learning in TurtleBot4 robots. By integrating the pyactr cognitive framework with A* path planning, the system combines low-level robotic control with high-level reasoning about past interactions and environmental structure. The resulting architecture allows the robot to move beyond purely reactive navigation and to incorporate experience-based learning into its decision-making process.

The experimental results demonstrate that the robot can incrementally learn the properties of obstacles through physical interaction and store this knowledge in a persistent cognitive memory. This enables reliable classification of solid and fake obstacles and prevents repeated collisions. Furthermore, the implemented strategy adaptation mechanism allows the robot to dynamically balance efficiency and safety by switching between shortest-path and risk-aware navigation strategies. Through repeated task executions, the robot is able to exploit learned information to improve performance and discover more efficient routes.

The presented approach highlights the potential of cognitive architectures as a complementary layer to existing robotic middleware. Rather than replacing established planning algorithms, the cognitive model enhances them by providing memory, learning, and strategy control. Such integration is particularly valuable in unfamiliar or changing environments, where the properties of obstacles cannot be fully determined by perception alone and where interaction plays a central role in knowledge acquisition.

VI. Future Work

Future work should focus on extending the current approach to more complex and realistic real world scenarios. This includes scaling the environment representation from discrete grid maps to continuous spaces and integrating additional sensory modalities. Moreover, the reality gap should be addressed by tighter integration of localization uncertainty and adaptive revision mechanisms.

Furthermore, expanding the cognitive model to distinguish between static and dynamic obstacles, such as people or other robots, would be crucial for safe deployment in real-world environments with sometimes unpredictable characteristics. When expanding this scenario to a multi-agent system including additional robots, a method of communication between the various agents would be advantageous in order to classify obstacles to shared use.

References

[1] Anusha Alexander, Kalaichelvi Venkatesan, Jinane Mounsef, and Karthikeyan Ramanujam. A comprehensive survey of path planning algorithms for autonomous systems and mobile robots: Traditional and modern approaches. *IEEE Access*, 13:176287–176326, 2025. doi: 10.1109/ACCESS.2025.3618837.

[2] John R Anderson, Daniel Bothell, Michael D Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. An integrated theory of the mind. *Psychological review*, 111 (4):1036, 2004.

[3] Mubarak Badamasi Aremu, Ibrahim K. Kabir, Gamil Ahmed, and Sami El-Ferik. Autonomous mobile robot path planning techniques - a review: Classical and heuristic techniques. *IEEE Access*, 13:117999–118022, 2025. doi: 10.1109/ACCESS.2025.3579863.

[4] Adrian Brasoveanu and Jakub Dotlacil. *The ACT-R Cognitive Architecture and Its pyactr Implementation*, pages 7–37. 05 2020. ISBN 978-3-030-31844-4. doi: 10.1007/978-3-030-31846-8_2.

[5] Ade Candra, Mohammad Andri Budiman, and Kevin Hartanto. Dijkstra's and a-star in finding the shortest path: a tutorial. In *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*, pages 28–32, 2020. doi: 10.1109/DATABIA50434.2020.9190342.

[6] Seifalla Mohamed Elbouhy, Asan Adamanov, Philipp Maximilian Braun, and Hendrik Wilhelm Rose. Comparative analysis of local trajectory planning algorithms in ros2. In *2025 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pages 1–6, 2025. doi: 10.1109/SIMPAR62925.2025.10979015.

[7] Xing Fu, Zucheng Huang, Gongxue Zhang, Weijun Wang, and Jian Wang. Research on path planning of mobile robots based on improved a* algorithm. *PeerJ Computer Science*, 11:e2691, 02 2025. doi: 10.7717/peerj-cs.2691.

[8] Andrea Giuffrida, Nicola Basilico, and Francesco Amigoni. An empirical evaluation of learning-based multi-agent path finding algorithms in warehouse environments. *Robotics and Autonomous Systems*, 194: 105149, 2025. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2025.105149.

[9] Kevin A. Gluck, John Laird, and Julia Lupp. [front matter]. In *Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions*. The MIT Press, 09 2019. ISBN 9780262349420. doi: 10.7551/mitpress/11956.003.0029. URL https://doi.org/10.7551/mitpress/11956.003.0029.

[10] Miguel Á. González-Santamarta, Francisco J. Rodríguez-Lera, Ángel Manuel Guerrero-Higueras, and Vicente Matellán-Olivera. Integration of large language models within cognitive architectures for autonomous robots, 2024. URL https://arxiv.org/abs/2309.14945.

[11] Miguel Á. González-Santamarta, Francisco J. Rodríguez-Lera, and Vicente Matellán-Olivera. Cognitive architectures in autonomous robotics: A systematic review of behavior generation approaches and evaluation strategies. *IEEE Access*, 13:191619–191644, 2025. doi: 10.1109/ACCESS.2025.3630344.

[12] Miguel Á. González-Santamarta, Francisco J. Rodríguez-Lera, Vicente Matellán-Olivera, Virginia Riego del Castillo, and Lidia Sánchez-González. Sailor: perceptual anchoring for robotic cognitive architectures. *Scientific Reports*, 15, 01 2025. doi: 10.1038/s41598-024-84071-2.

[13] Ignazio Infantino, Agnese Augello, Umberto Maniscalto,

Giovanni Pilato, and Filippo Vella. A cognitive architecture for social robots. In *2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI)*, pages 1–5, 2018. doi: 10.1109/RTSI.2018.8548520.

[14] jakdot. pyactr. Technical report, 2026. URL https://github.com/jakdot/pyactr. GitHub repository.

[15] Kowsik K. The evolution of robot operating systems: A review of ros 2. 06 2025. doi: 10.5281/zenodo.15623121.

[16] Sebastian Kahl, Sebastian Wiese, Nele Russwinkel, and Stefan Kopp. Towards autonomous artificial agents with an active self: Modeling sense of control in situated action. *Cognitive Systems Research*, 72, 12 2021. doi: 10.1016/j.cogsys.2021.11.005.

[17] Unmesh Kurup and Christian Lebiere. What can cognitive architectures do for robotics? *Biologically Inspired Cognitive Architectures*, 2:88–99, 2012. ISSN 2212-683X. doi: https://doi.org/10.1016/j.bica.2012.07.004. URL https://www.sciencedirect.com/science/article/pii/S2212683X12000333.

[18] Mingxiu Lin, Kai Yuan, Chengzhi Shi, and Yutong Wang. Path planning of mobile robot based on improved a* algorithm. In *2017 29th Chinese Control And Decision Conference (CCDC)*, pages 3570–3576, 2017. doi: 10.1109/CCDC.2017.7979125.

[19] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66):eabm6074, 2022. doi: 10.1126/scirobotics.abm6074. URL https://www.science.org/doi/abs/10.1126/scirobotics.abm6074.

[20] Bastian Mannerow. Act-r multi-agent simulation framework (python / pyactr). Technical report, 2025. URL https://github.com/BastianMannerow/actr-multi-agent-simulation. GitHub repository.

[21] Morayo Ogunsina, Christianah Efunniyi, Olajide Osundare, Samuel Folorunsho, and Lucy Akwawa. Cognitive architectures for autonomous robots: Towards human-level autonomy and beyond. *International Journal of Frontline Research in Science and Technology*, 2:41–50, 09 2024. doi: 10.56355/ijfret.2024.2.1.0021.

[22] open robotics. Ros - robot operating system. Technical report, 2026. URL https://www.ros.org/.

[23] D. Ran, J. Ma, F. Peng, and H. Li. Summary of research on path planning based on a* algorithm. In *The 8th International Symposium on Test Automation & Instrumentation (ISTAI 2020)*, volume 2020, pages 84–89, 2020. doi: 10.1049/icp.2021.1291.

[24] Rockwell Automation, Inc. Turtlebot 4. Technical report, 2026. URL https://clearpathrobotics.com/turtlebot-4/.

[25] ROS 2 People. Ros 2. Technical report, 2026. URL https://github.com/ros2. GitHub repository.

[26] Stuart J. Russell and Peter Norvig. *Artificial intelligence: A Modern Approach*. Pearson, Boston, 2020.

[27] Vikram Shahapur, Blessing Dixon, and Urvishkumar Bharti. Robotic path planning implementation using search algorithms, 2024. URL https://arxiv.org/abs/2405.16445.

[28] Thomas Sievers and Nele Russwinkel. A path to gradual individual experience and recollection for social robots based on a cognitive architecture. In *Social Robotics + AI*, pages 641–647, Singapore, 2026. Springer Nature Singapore. ISBN 978-981-95-2379-5.

[29] Gustavo R. Silva, Leandro B. Becker, and Jomi F. Hübner. Embedded architecture composed of cognitive agents and ros for programming intelligent robots. *IFAC-PapersOnLine*, 53(2):10000–10005, 2020. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2020.12.2718. URL https://www.sciencedirect.com/science/article/pii/S2405896320334819. 21st IFAC World Congress.

[30] Paul Smart and Katia Sycara. Using a cognitive architecture to control the behaviour of virtual robots. In *Proceedings of EuroAsianPacific Joint Conference on Cognitive Science (EAPCogSci '15)*, pages 447–452, 09 2015.

[31] J. Gregory Trafton, Laura M. Hiatt, Anthony M. Harrison, Franklin P. Tamborello, Sangeet S. Khemlani, and Alan C. Schultz. Act-r/e: an embodied cognitive architecture for human-robot interaction. *J. Hum.-Robot Interact.*, 2(1):30–55, February 2013. doi: 10.5898/JHRI.2.1.Trafton. URL https://doi.org/10.5898/JHRI.2.1.Trafton.

[32] Sangeeth Venu and Muralimohan Gurusamy. A comprehensive review of path planning algorithms for autonomous navigation. *Results in Engineering*, 28:107750, 2025. ISSN 2590-1230. doi: https://doi.org/10.1016/j.rineng.2025.107750. URL https://www.sciencedirect.com/science/article/pii/S2590123025038034.

[33] Michael Wooldridge. *Intelligent agents*, pages 27–77. MIT Press, Cambridge, MA, USA, 1999. ISBN 0262232030.

[34] Siyu Wu, Rodrigo F. Souza, Frank E. Ritter, and Walter T. Lima Jr. Comparing llms for prompt-enhanced act-r and soar model development: A case study in cognitive simulation. *Proceedings of the AAAI Symposium Series*, 2(1):422–427, Jan. 2024. doi: 10.1609/aaaiss.v2i1.27710.

[35] Ziying Zhang, Nele Russwinkel, and Sabine Prezenski. Modeling individual strategies in dynamic decision-making with act-r: A task toward decision-making assistance in hci. *Procedia Computer Science*, 145:668–674, 2018. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2018.11.064. URL https://www.sciencedirect.com/science/article/pii/S1877050918323500. Postproceedings of the 9th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2018 (Ninth Annual Meeting of the BICA Society), held August 22-24, 2018 in Prague, Czech Republic.