  
UNIVERSITY OF HERTFORDSHIRE  
School of Physics, Engineering and Computer Science

MSc Artificial Intelligence and Robotics  
7COM1039-0509-2024 - Advanced Computer Science Masters Project  
July 14, 2025

LLM-Guided Multi-Agent-Robot Navigation Using A\* and Sensor-Based Control for Obstacle Avoidance in CoppeliaSim

Name: Mohamad Dirani  
Student ID: 23074344  
Supervisor: Grigorios Skaltsas

# Introduction and Overview

## Background and Motivation

Multi-robot autonomous systems are essential in modern industrial and service applications where tasks need to be distributed and executed efficiently. Key to the success of these systems is reliable navigation—finding and following safe paths to goals while avoiding collisions. Classical path planning algorithms such as A\* are widely used for global navigation (Candra et al., 2020), while real-time reactive methods handle local obstacle avoidance (Li et al., 2017).

Recent advances in Large Language Models (LLMs), such as OpenAI's GPT-4, offer new opportunities for human-robot interaction. These models can understand natural language instructions and convert them into structured commands, providing a flexible interface for dynamic goal assignment (OpenAI, 2023). This project leverages LLMs to introduce a human-in-the-loop control mechanism, integrating speech-based input with real-time autonomous planning and avoidance systems.

## Research Question

* Main Research Question:

How do dynamic obstacles and environmental complexity affect the robots' ability to navigate and complete tasks using the implemented avoidance logic? (Gourley and Trivedi, 1994)

* Sub-Research Questions:

1. How does coordination between multiple agents influence the overall task efficiency and response time in the system? (Längle and Wörn, 2001; Parisotto et al., 2017)
2. Does human intervention using LLM-based commands for one robot impact the task performance or progress of other robots operating simultaneously in a shared environment? (Längle and Wörn, 2001)
3. Does the robot perform differently when encountering static obstacles, dynamic obstacles, or other robots as obstructions? (Gourley and Trivedi, 1994)

## Literature Review

This literature review examines eight scholarly works central to LLM-driven, multi-agent robot navigation and control. Each article is contextualized, critiqued, and compared against the goals and implementation of the current project. The review addresses how each contributes to the four main research questions.

* Gourley and Trivedi (1994) – Sensor-Based Obstacle Avoidance and Mapping:

This early work uses ultrasonic sensors and reactive force fields for fast navigation. It supports sub-question 3 by showing the critical role of local perception in high-speed environments. My system draws from this principle by using directional vision sensors and recursive sidestepping to compare performance across static and dynamic obstacles, including robots.

* Candra et al. (2020) – Path Planning with A\*:

This comparative study of A\* and Dijkstra validates A\*’s advantage in planning speed and efficiency. Their results support my choice of A\* for goal assignment. I compare completion times and rerouting across agents, contributing directly to sub-question 1 on coordination and efficiency.

* Parisotto et al. (2017) – Neural SLAM:

Parisotto et al. propose a memory-augmented agent framework that excels at collaborative exploration. Though my system does not include learned SLAM, the scalability tests I conduct on multiple robots simulate their conditions, helping assess sub-question 1 regarding increased robot numbers and coordination.

* Längle and Wörn (2001) – Human–Robot Cooperation in Multi-Agent Systems:

This article introduces structured role negotiation between humans and robots. In my system, I issue commands mid-task to specific robots using LLMs. Observing how other robots react (e.g., pause, wait, reroute) directly informs sub-question 2 regarding interference from human intervention.

* Mu et al. (2023) – ChatGPT with Knowledge Graphs:

Mu et al. demonstrate improved task parsing with LLM-KG integration. While my project omits the KG, I evaluate GPT-4’s capacity to produce actionable JSON commands from speech. This benchmark addresses sub-question 2 and informs the system’s scalability.

* OpenAI (2023) – GPT-4 for Robotic Task Planning:

OpenAI’s work highlights GPT-4’s strength in transforming natural language into robotics plans. My project builds on this by evaluating GPT-4’s real-time responsiveness and task accuracy in simulation. This informs sub-question 2.

* Farley et al. (2022) – Comparing CoppeliaSim, Gazebo, MORSE, Webots:

This paper provides a detailed comparison of robot simulators. Their findings favor CoppeliaSim for accuracy and versatility, justifying my selection. It indirectly supports sub-question 1, as the simulator's precision and responsiveness affect coordination outcomes.

* KUKA Platform and Mecanum Design – Real-World Mecanum Wheel Dynamics (2020):

This paper provides a mathematical and simulation-based analysis of Mecanum wheels using the KUKA youBot as an example (Rus. J. Nonlin. Dyn., 2020). It accounts for real-world friction, wheel slip, and kinematic constraints in omnidirectional platforms. My simulation reflects these principles by translating theoretical wheel behavior into Python-based velocity mapping for precise robot control. This reinforces sub-question 3 by demonstrating how correct kinematics affect navigation reliability around static and dynamic obstacles.

This review supports the use of A\*, vision-based avoidance, and LLM-driven control in coordinated multi-robot systems. It also validates my choice of CoppeliaSim as a simulation platform. The reviewed works guide how I interpret obstacle types, robot interference, and communication timing across sub-tasks in my system.

## Methodology

## Project Artefact

The artefact is a modular software system developed in Python and integrated with CoppeliaSim via ZeroMQ. It simulates a team of omnidirectional robots equipped with front, back, left, and right sensors. Each robot:

* Receives human commands parsed by GPT-4 via OpenAI’s API.
* Uses an A\* algorithm to plan a path to the assigned goal.
* Applies sensor-based logic to avoid obstacles.

Modules developed include:

* main.py: simulation loop and orchestrator
* robot\_controller.py: control logic
* obstacle\_awareness.py: local avoidance
* astar\_env.py and map\_builder.py: path planning environment
* LLM.py: speech recognition and LLM integration

## Tools and Techniques

* Simulation: CoppeliaSim
* Programming: Python
* Control Architecture: A\* planner + LiDAR-based reactive logic
* LLM Interaction: OpenAI GPT-4 + speech\_recognition module (OpenAI, 2023)
* Communication: ZeroMQ

## Deliverables

* Fully functional multi-robot simulation
* Modular codebase
* LLM command interpreter
* Evaluation results (navigation success rate, command parsing accuracy, etc.)
* Final report and demonstration

## Ethical, Legal, Professional, and Social Issues

* Ethics Approval: Not required, as no real user data is collected
* Data Usage: All processed input is synthetic or developer-generated
* Professional Conduct: OpenAI API used in accordance with terms
* Social Implications: Promotes accessibility in robotics via natural interfaces

# Progress to Date

## Work Done

* Built full CoppeliaSim multi-robot environment with obstacle layout
* Implemented A\* with dynamic occupancy grid map updates
* Developed directional obstacle sensors and alignment logic
* Added recursive decision-making to navigate blocked paths
* Integrated GPT-4 via OpenAI API to convert spoken commands to robot goals

## Problems Encountered

* Incorrect motion due to mecanum wheel logic — fixed by adjusting velocity mapping
* Repetitive entry into failed movement loops — resolved via improved termination and fallback conditions
* Speech recognition errors — mitigated by repeating unclear prompts
* Initial difficulties connecting Python to CoppeliaSim via ZeroMQ — resolved through environment setup, version compatibility fixes, and timeout handling
* Adjusting sensor angles for accurate forward, side, and rear detection — required fine-tuning sensor orientation in the simulation
* Inconsistent sensor detection results — addressed by calibrating range thresholds and checking signal noise in CoppeliaSim
* Complexity in implementing avoidance logic — recursive sidestepping and axis-alignment logic had to be carefully coordinated with path planner and real-time sensor data

# Planned Work

## Remaining Tasks

* Implement inter-robot clearance behavior: when two goals are assigned, ensure that if one robot is blocking another’s path, it moves out of the way temporarily
* Expand the system to support more than two robots and assess performance impact and responsiveness as the number of agents increases
* Introduce new LLM command capabilities, such as stopping a specific robot or instructing it to return to its start position during an active simulation
* Collect performance data under varying command and obstacle scenarios
* Evaluate system responsiveness, goal success rate, and LLM accuracy
* Prepare final presentation and demo video

## Evaluation Methods

* Compare total task time and collision avoidance success across scenarios
* Assess LLM parsing precision by comparing expected vs. actual commands
* Document system behavior using logs and annotated simulation visuals

## Timeline

|  |  |
| --- | --- |
| **Task** | **Target Date** |
| Robot Coordination Upgrade | Aug 1 |
| Finish LLM Parser Logic | Aug 10 |
| Evaluation + Metrics Logging | Aug 20 |
| Final Report Writing | Aug 25–Sep 5 |
| Presentation + Submission | Sep 15 |

# Bibliography

* Candra, A., Budiman, M.A. and Hartanto, K. (2020) "Dijkstra's and A-Star in Finding the Shortest Path: A Tutorial." 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA).
* Gourley, J. and Trivedi, M.M. (1994) "Sensor-based obstacle avoidance and mapping for fast mobile robots." Proceedings of the IEEE Conference on Intelligent Robots and Systems, pp. 601–607.
* Farley, A., Wang, J., Marshall, J.A. (2022) "How to Pick a Mobile Robot Simulator: A Quantitative Comparison of CoppeliaSim, Gazebo, MORSE and Webots." Simulation Modelling Practice and Theory, 120, 102629.
* KUKA (2020) "Platform Taking into Account Real Design of Mecanum Wheels (as Exemplified by KUKA youBot)," Russian Journal of Nonlinear Dynamics, 16(2), pp. 291–307. Available at: https://www.mathnet.ru/eng/nd711
* Längle, T. and Wörn, H. (2001) "Human–Robot Cooperation Using Multi-Agent-Systems." Journal of Intelligent and Robotic Systems, 32, pp. 143–159.
* Mu, Q., Zhao, Y., Gao, Q., and Wang, X. (2023) "ChatGPT and Knowledge Graph for Human-Robot Interaction." arXiv preprint arXiv:2302.XXXX.
* OpenAI (2023) "ChatGPT for Robotics: A New Approach to Human-Robot Interaction and Task Planning."
* Parisotto, E., Song, S., Zhang, C., et al. (2017) "Neural SLAM: Learning to Explore with External Memory." ICLR.

# Appendices

* Code samples (Appendix A)
* Screenshots of robot execution (Appendix B)
* Sample GPT prompts and outputs (Appendix C)
* Sensors Execution (Appendix D)