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

1. Introduction and Overview

1.1 Background and Motivation  
Multi-robot systems play an essential role in logistics, service robotics, and intelligent automation. The navigation problem—how agents move from one point to another without conflict or collision—is foundational. Classical algorithms like A\* offer reliable planning over known maps (Candra et al., 2020), while real-time sensors ensure reactive navigation (Gourley & Trivedi, 1994).

Recently, Large Language Models (LLMs) such as GPT-4 have emerged as tools for enhancing human-robot interaction. These models can interpret natural language and generate structured outputs, creating novel interfaces for controlling multi-agent systems. However, integrating LLMs into robotics introduces questions of ambiguity handling, decision reliability, and responsiveness under uncertainty (Agashe et al., 2023).

This project explores the fusion of A\* path planning, directional sensor-based avoidance, and GPT-4-driven natural language commands for collaborative robot navigation.

1.2 Research Questions and Hypotheses

Main Research Hypothesis:  
Dynamic obstacles and ambiguous LLM inputs increase task time and coordination conflicts in multi-agent navigation systems.

Sub-Hypotheses:  
H1: Coordination conflicts increase with LLM-induced latency compared to deterministic GUI input.  
H2: LLM ambiguity results in lower goal assignment accuracy versus scripted instructions.  
H3: Reactive sidestepping strategies reduce path optimality under narrow obstacle conditions.

These hypotheses derive from ongoing debates in human-robot trust calibration (Rizk et al., 2023), multi-agent planning uncertainty (Chen et al., 2023), and LLM behavior evaluation under safety-critical systems (Li et al., 2025).

1.3 Literature Review

This section synthesizes prior work across four domains: obstacle avoidance, multi-agent coordination, LLM integration in robotics, and simulator benchmarking.

Obstacle Avoidance:  
Gourley and Trivedi (1994) presented sensor-based reactive behaviors for mobile robots, emphasizing local perception. My system extends this by using directional sensors in CoppeliaSim and dynamic avoidance loops.

Path Planning:  
Candra et al. (2020) compared A\* and Dijkstra, supporting A\*’s computational efficiency in structured grids. This reinforces its use for global path generation in our modular planner.

Multi-Agent Coordination:  
Längle and Wörn (2001) explored agent negotiation in human-robot teams, while Parisotto et al. (2017) introduced memory-augmented navigation through Neural SLAM. Neither explicitly addressed LLM-induced uncertainty, which my system tests through human-in-the-loop interruptions.

LLM Use in Robotics:  
Agashe et al. (2023) benchmarked GPT-4’s coordination ability, finding performance gaps in joint planning and ambiguity resolution. Mu et al. (2023) proposed Knowledge Graph-enhanced prompts, showing parsing gains. I build on these by analyzing GPT-4’s command latency and ambiguity effects.

Simulator Validation:  
Farley et al. (2022) compared simulators, validating CoppeliaSim’s responsiveness and modularity. My work extends its application to LLM-in-the-loop experiments.

Kinematics:  
The KUKA (2020) model of Mecanum wheels addresses kinematic friction and control accuracy, validating my implementation of grid-to-wheel velocity mapping.

Research Gap:  
Current literature lacks rigorous benchmarks for LLM-driven multi-agent coordination in physical or simulated spaces. No prior studies combine reactive obstacle logic, goal delegation, and ambiguity testing under varying obstacle density.

1.4 Methodology

Architecture Overview:  
The project uses a modular Python framework connected via ZeroMQ to CoppeliaSim. Robots use four directional SICK S300 sensors and execute A\*-derived paths with collision avoidance.

Evaluation Design:  
Variables:

* Independent: Command interface (LLM vs. scripted), obstacle density (none, static, dynamic), robot count.
* Dependent: Goal success rate, path optimality, response latency, collision count.

Control Groups:  
Baseline trials use static goal input via GUI. LLM-based trials introduce natural language at runtime. Each experiment is repeated with n=30 trials for statistical relevance.

Metrics:

* Task Completion Time (mean ± std. dev.)
* Goal Accuracy (Euclidean distance to target)
* LLM Response Latency (sec)
* Coordination Success Rate (shared path resolution)
* LLM Parsing Error Rate (% malformed commands)

1.5 Project Artefact

The artefact is a Python-based multi-robot simulation integrated with CoppeliaSim. Key modules include:

* main.py, run.py: Simulation loop and launcher
* LLM.py: GPT-4 input handler and goal parser
* astar.py, map\_builder.py, astar\_env.py: Path planning
* robot\_controller.py, robot\_motion.py: Execution logic
* obstacle\_awareness.py, sensor\_fetch.py: Real-time avoidance
* check\_nearest\_robot.py: Task allocation
* plotter.py, shared.py: Visualization and constants

1.6 Tools and Techniques

* Simulation: CoppeliaSim
* Language Models: GPT-4 (OpenAI API)
* Programming: Python (asyncio, speech\_recognition)
* Communication: ZeroMQ
* Planning: A\* with occupancy grid updates
* Sensor Logic: Directional collision detection via ray sensors

1.7 Deliverables

* Full codebase (public GitHub repository)
* Annotated simulation visuals
* Comparative logs (GUI vs. LLM assignment)
* Final report and demo video

1.8 Ethical, Legal, and Professional Issues

* No human data used; commands are synthetic
* GPT-4 used under OpenAI’s developer terms
* Sensor feedback used for deterministic decisions
* No real-time user tracking or recording
* Promotes inclusive robotics through natural language access

1. Progress to Date

2.1 Work Done

* Built multi-agent environment in CoppeliaSim
* Developed and debugged A\* + sensor-based avoidance
* Integrated GPT-4 with real-time goal assignment
* Calibrated Mecanum wheel dynamics for accurate velocity mapping
* Logged executed vs. planned paths in obstacle and obstacle-free settings

2.2 Problems Encountered

* LLM failures under ambiguous prompts → handled by timeout/retry
* Motion jitter from incorrect wheel mapping → fixed using KUKA (2020)
* Sensors misaligned → fixed via manual scene tuning
* Blocking robots → handled via dynamic sidestepping and path reallocation

1. Planned Work

* Introduce adversarial commands and ambiguous inputs for robustness testing
* Expand to 4 robots to study task conflict and coordination breakdown
* Introduce statistical reporting via t-tests and confidence intervals
* Measure effects of human interventions mid-task
* Develop fallback command strategy for LLM parsing failures

1. Evaluation Methods

Each scenario will be evaluated under the following conditions:

* LLM vs. GUI command interface
* Dynamic vs. static obstacle layouts
* 2 vs. 4 robots

Comparative Analysis:

* Goal Accuracy (distance error)
* Task Time (seconds)
* LLM Parsing Failures
* Blocking Frequency
* Interference Resolution Latency

Statistical Approach:

* ANOVA or Welch’s t-test between configurations
* Confidence intervals (95%) on success metrics
* Charts to visualize mean differences

1. Timeline

| **Task** | **Dates** |
| --- | --- |
| Setup + Planning | July 14–17 |
| Inter-Robot Clearance | July 18–22 |
| Multi-Robot Scaling | July 23–27 |
| LLM Command Expansion | July 28–Aug 1 |
| JSON + Goal Parser Final | Aug 2–Aug 5 |
| Accuracy Testing | Aug 6–Aug 9 |
| Real-Time Evaluation | Aug 10–Aug 13 |
| Metrics Logging | Aug 14–Aug 16 |
| Visual + Log Compilation | Aug 17–Aug 18 |
| Graph + Results Generation | Aug 19–Aug 21 |
| Final Report Writing | Aug 22–Sep 4 |
| Presentation Prep | Sep 5–Sep 12 |
| Final Submission | Sep 15 |

1. References

* Agashe, S., Fan, Y. & Wang, X. (2023). Evaluating Multi-Agent Coordination Abilities in Large Language Models. arXiv:2310.03903.
* Candra, A., Budiman, M.A. & Hartanto, K. (2020). Dijkstra’s and A-Star in Finding the Shortest Path: A Tutorial. IEEE DATABIA.
* Chen, J., et al. (2025). Multi-Agent Systems for Robotic Autonomy with LLMs. CVPR Workshop.
* Farley, A., Wang, J. & Marshall, J. (2022). A Quantitative Comparison of Mobile Robot Simulators. Simulation Modelling Practice and Theory, 120.
* Gourley, J. & Trivedi, M. (1994). Sensor-Based Obstacle Avoidance. IEEE IROS.
* KUKA (2020). Mecanum Platform Dynamics. Russian Journal of Nonlinear Dynamics, 16(2).
* Längle, T. & Wörn, H. (2001). Human–Robot Cooperation Using MAS. JIRS.
* Li, P., An, Z. & Zhou, L. (2025). Large Language Models in Multi-Robot Systems: A Survey. arXiv:2502.03814.
* Mu, Q., Zhao, Y., Gao, Q. & Wang, X. (2023). ChatGPT and Knowledge Graph for HRI. In: Intelligent Robotics and Applications, LNAI 14398.
* OpenAI (2023). ChatGPT for Robotics. OpenAI Tech Report.
* Parisotto, E. et al. (2017). Neural SLAM. ICLR.

Appendices

* A: Code Samples (GitHub repo)
* B: Annotated Screenshots (planned vs executed paths)
* C: GPT Failures and Success Logs
* D: Sensor Feedback Maps
* E: Gantt Chart and Schedule