**Group-9, CSE440.1**

Hafsa Mahbub 2021164042

Jahina Zaiyan Adiba 1931569642

Sarder Hadeeka Hea 1831384642

**Title: Development of an Autonomous Robot for Indoor Navigation Using A\* Pathfinding Algorithm**

**Introduction**

In today's fast-paced world, technology continues to advance at an unprecedented rate, significantly improving the quality of our lives. Among these advancements, autonomous robots stand out as transformative tools, enhancing workplace efficiency by automating repetitive tasks, ensuring safety, reducing operational costs, and boosting overall productivity. However, for these robots to function reliably, rigorous testing in diverse scenarios is essential. This project addresses such a need by developing a software-based simulation to model an autonomous robot navigating an indoor environment. The simulation is designed to demonstrate how the robot identifies and follows the shortest path to a specified goal while dynamically avoiding obstacles. This is achieved using the A\* search algorithm, a widely used heuristic-based approach for efficient pathfinding. By leveraging Python and Pygame, we have created an interactive simulation that illustrates the fundamental principles of pathfinding and environment-aware movement within a virtual setting. This approach allows us to test and refine the robot’s navigation algorithms in a controlled environment before real-world application.

**Methodology**

The primary objective of this project is to implement the A\* algorithm, a heuristic-based search method widely recognized for its efficiency in grid-based pathfinding tasks. To achieve this, we created a virtual grid environment where each cell represents a specific role, such as obstacles, the start point, or the goal point. This setup allows the autonomous robot to calculate an optimal path using the algorithm’s core principles, which combine movement costs (g-cost) and heuristic estimates (h-cost) to identify the shortest, most efficient path to the goal.

The virtual environment is developed using Pygame, a versatile Python library for graphical simulations. Within this environment, obstacles are strategically placed to mimic real-world barriers, ensuring the algorithm adapts to dynamic navigation challenges. A\* computes the f-cost for each cell by summing the g-cost and h-cost, guiding the robot through the path with the lowest total cost.

Key functions were designed to support this process, including those for calculating distances, reconstructing the path once the goal is reached, and evaluating neighboring cells to ensure only valid moves are considered. Movements and path adjustments are displayed in real-time, enabling users to observe and verify the algorithm’s accuracy and efficiency in navigating the grid.

**Tools Used**

The project is developed with Python, as an adaptable language its suited for both algorithmic programming and rapid prototyping. We used the following tools and libraries:

1. **Pygame:** A Python library that created the grid-based graphical interface, updated the visual imagery of movements and displayed obstacles in real-time.
2. **Python’s Standard Libraries:** These have heapq which managed the priority queue needed by A\*, allowing efficient handling of the cells based on their f-cost.
3. **Visual Studio Code:** A popular text editor and integrated development environment (IDE) for Python programming. This supports installation of virtual environments, debugging and real-time syntax feedback, which was used for developing the A\* algorithm.

**Challenges**

Throughout the project, we have faced several challenges that required careful problem-solving and adjustments. Implementing the A\* search algorithm within a 2D grid system proved to be a significant hurdle, particularly in estimating the heuristic values. Accurate heuristic estimation is critical, as overestimating these values caused unnecessary delays in the pathfinding process, while underestimating them prevented the robot from efficiently reaching its destination. Striking the right balance was essential to ensure smooth and optimal navigation.

Another major challenge was synchronizing the visual updates within Pygame to maintain real-time responsiveness. Any delays or inefficiencies in the simulation loop directly impacted the accuracy and fluidity of the robot’s path display, making it harder to observe and analyze the algorithm’s performance. Addressing these synchronization issues required careful coding and testing to ensure the simulation operated seamlessly without interruptions or lags.

**Progress**

At this stage, we have made significant progress in the development of the project, and the project is now complete. The core functionality of the A\* algorithm has been successfully implemented, enabling it to compute and display an optimal path from the start point to the goal point. The grid environment effectively identifies obstacles, allowing the robot to navigate around them and find the shortest possible path with a high degree of accuracy.

The simulation is further enhanced by the Pygame interface, which provides a clear visual representation of the grid, including the start and goal points, as well as the obstacles. Additionally, the dynamically computed path is highlighted in real-time, ensuring the simulation is both functional and visually engaging. These developments mark a strong foundation for the project, showcasing a working prototype of the intended functionality.

**Future Work Plan**

To further enhance the project, several key improvements are planned for the future. One of the primary goals is to optimize the A\* algorithm to handle dynamic obstacles more effectively. This would enable the robot to seamlessly adjust its path in real-time when new obstacles appear during navigation, ensuring smoother and more efficient movement.

Another planned improvement involves upgrading the graphical interface. By introducing clearer color distinctions, users can better understand the simulation elements at a glance. Additionally, implementing the project in 3D would add a layer of realism and expand its potential applications. To increase customization and interactivity, we aim to incorporate user controls for modifying the grid size and manually placing obstacles within the environment. These enhancements will not only make the project more flexible and user-friendly but also improve its visual appeal and overall functionality.

**Conclusion**

This project has offered meaningful insights into a contemporary and impactful field like robotics. By leveraging the power of the A\* search algorithm and combining it with the versatility of Pygame, we successfully developed a functional autonomous navigation system within a simulated environment. The use of widely accepted tools, such as Python's standard libraries and a reliable Integrated Development Environment (IDE) like Visual Studio Code, greatly streamlined the development process and enhanced the project’s efficiency.

The current model demonstrates basic navigation capabilities, effectively calculating and visualizing optimal paths while avoiding obstacles. However, this is just the beginning. Moving forward, we aim to refine the system by incorporating dynamic path adjustments and user interactivity to make the simulation even more realistic and engaging. This marks a significant step toward understanding and applying advanced robotics concepts in software-based environments.