

Interactive Indoor Map Making and Navigation

Prof. Ajay Keshav Talele¹, Prof. Madhuri Barhate², Abhishek Yadav³, Aditya Yadav⁴, Yash Dusankar⁵,
Yashraj Nalawade⁶ and Swarnim Yawale⁷

¹⁻⁷Department of Multidisciplinary Engineering (DOME), Vishwakarma Institute of Technology, Pune, 411037, Maharashtra, India

Abstract—Navigating indoor spaces present a myriad of challenges, from labyrinthine corridors to obscure signage, hindering the efficient exploration. To address the underlying issue, we have devised a project, names “Indoor map maker and navigator”, which is based on the concepts of Object Oriented programming. This innovative Java-based solution aims to simplify creation of indoor maps and streamline navigation within complex environment spaces. By harnessing the power of Java programming and its rich GUI features, our project promises to offer a user-friendly interface for constructing detailed indoor maps. With seamless integration of database, users can effortlessly navigate through intricate interiors, revolutionising the way we interact with indoor spaces and enhancing overall exploration efficiency.

Index Terms— Map Navigation, Graph Representation, MySQL, Edge creation, Java-based.

I. INTRODUCTION

Navigating round through surroundings can often be a confusing and ineffective task. In large shopping centres, office buildings, or complicated educational institutions, the lack of dependable navigation systems poses a significant challenge to people looking for convenient paths and clear directions. The lack of standardised indoor mapping solutions makes this problem even worse, forcing users to rely on outdated paper maps or laborious digital equivalents. Thus, there is a pressing need for a reliable, intuitive indoor mapping and navigation system that can easily and precisely lead people through intricate interiors.

In order to tackle this important but unnoticed problem, we're introducing the "Indoor Map Maker and Navigation" project, an innovative project that aims to transform the way humans interact with and navigate inside surroundings. Essentially, the project aims to close the gap between the demand for clear navigational aids and the complex indoor area layouts. We want to produce a sophisticated yet user-friendly solution that streamlines the process of creating indoor maps and improves navigation efficiency by utilising the power of the Java programming language and Object-Oriented Programming (OOP) ideas.

The "Indoor Map Maker and Navigation" project is an example of how cutting-edge technology and creative software engineering concepts may work together. With careful planning and execution, our group hopes to develop a flexible instrument that can produce intricate interior maps with little assistance from the user. By combining the scalability and flexibility of Java with the stability of OOP, we aim to build a platform that can be used in a variety of interior locations, such as shopping centres, theatres, and office buildings. The smooth integration of MySQL databases, which enables effective data management and storage, is essential to the project's success. We make sure that users can readily update and customise indoor maps to reflect changes in the environment in real time by storing map data in an accessible and structured way.

In addition to improving navigational information accuracy, this dynamic approach allows for ongoing

development and adaptability to changing interior environments.

Beyond simple practicality, the "Indoor Map Maker and Navigation" project has the ability to fundamentally alter our understanding of and interactions with interior environments. We want to improve overall exploration efficiency and reduce the difficulty of exploring complicated interiors by providing people with easy-to-use navigation tools and extensive indoor maps. Increasing accessibility for people with mobility impairments, streamlining emergency response protocols, and promoting a more seamless interior environment for all users are among our goals. In the end, we want to rethink interior navigation with this project and bring in a new era of lucidity.

II. LITERATURE REVIEW

In the realm of Geographical Information Systems (GIS) for transportation modeling, object-oriented (OO) approaches have garnered significant attention due to their potential in enhancing data modeling and handling [1]. These approaches offer a structured framework for representing complex transportation networks, facilitating the integration of various data sources and enabling more efficient analysis and simulation of transportation systems.

Java, as a programming language, has been instrumental in facilitating interactive learning in computer science education [2]. By providing a platform-independent environment and a rich set of libraries, Java allows educators to create engaging learning experiences, particularly in introductory courses where students explore fundamental concepts such as data structures and algorithms. The visualization of Java data structures as graphs, as proposed by Hamer (2004), not only enhances students' understanding of abstract data types but also fosters active exploration and experimentation, promoting deeper learning and problem-solving skills.

Java's versatility extends to its GUI desktop technologies, namely Swing and JavaFX, which can be integrated using design patterns like the Abstract Factory pattern [4]. This integration allows developers to leverage the strengths of each technology while abstracting away platform-specific details, thereby simplifying the development process and enhancing maintainability. Such patterns are particularly valuable in educational contexts, where they not only teach software engineering principles but also empower students to build robust and scalable applications.

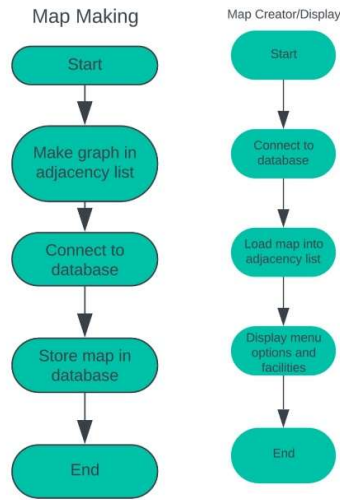
Efficient routing algorithms play a crucial role in transportation systems, particularly in solving the shortest path problem [5]. Rachmawati and Gustin's (2020) comparative analysis of Dijkstra's and A* algorithms provides valuable insights into their performance characteristics, guiding the selection of appropriate algorithms based on the scale and complexity of transportation networks. These algorithms form the backbone of route planning services in applications such as navigation systems and logistics management platforms, where real-time decision-making is critical for optimizing resource utilization and minimizing travel time.

Beyond transportation modeling, algorithmic optimization techniques have been explored in various domains [13]. From network optimization to pathfinding, researchers continue to investigate novel approaches for addressing complex computational challenges. Fundamental to many of these studies is the application of Object-Oriented Programming (OOP) principles [8]. OOP provides a modular and scalable framework for designing and implementing software systems, enabling developers to manage complexity and promote code reuse.

In summary, the reviewed literature underscores the significance of OO approaches in GIS and transportation modeling, the pedagogical benefits of visualizing Java data structures, and the application of efficient routing algorithms like Dijkstra's and A* [1][2][4][5][8][13]. These studies collectively contribute to advancing knowledge in GIS, computer science education, and algorithmic optimization for transportation systems, laying the foundation for future research and innovation in this field

III. METHODOLOGY

The two main components of the project's architecture, Map Creation and Map Navigation, both have complementary yet separate purposes. The main feature of Map Creation is its Graphical User Interface (GUI), which makes use of Java's Swing framework to give customers an easy-to-use platform for creating indoor maps. Users can easily add nodes and edges using this interface, which captures the spatial arrangement and interconnectedness of indoor spaces. By giving users the ability to precisely define places and relationships, this interactive method paves the way for precise navigation. Conversely, Map Navigation concentrates on assisting users in navigating the complexities of indoor environments. To achieve this, it employs advanced pathfinding algorithms to determine the best paths between designated source and destination nodes on the created maps.



A graph data structure is used in the project to describe the indoor map in an effective manner. This is a crucial element that allows for efficient navigation in complex interior situations. Nodes in this structure represent individual places, while edges represent the relationships between these sites. Together, these build a network that reflects the interior space's physical arrangement. To enable precise mapping and navigation, every node is created as an instance of the Node class, which contains crucial data including direction, distance, and vertex index. Adjacency lists, a data structure that facilitates the efficient storage and traversal of connections inside the map, are used to manage graph edges. This methodology guarantees peak efficiency in reaching and maneuvering across the networked areas, augmenting the general practicality and efficiency of the system.

The graph structure in the Java code is based on set theory principles. Vertices V are represented as integers from 0 to $n - 1$, where n is the total count of vertices. Edges E are tuples $(src, dest, distance, direction)$, where src and $dest$ are vertices, $distance$ is an integer, and $direction$ is a directional state. Adjacency lists $L(v)$ for each vertex v catalog edges incident upon it, facilitating graph traversal.

$$V = \{0, 1, 2, 3, \dots, n - 1\}$$

For each vertex v in V , an adjacency list $L(v)$ catalogs the incident edges. Each entry in $L(v)$ represents an edge, containing destination vertex, traversal distance, and directional orientation. This representation enables efficient graph traversal and manipulation.

Dijkstra's algorithm is a method used to find the shortest path from a source vertex to all other vertices in a weighted graph. Here's a mathematical explanation of Dijkstra's algorithm using set theory and the adjacency list representation of a graph:

Let $G = (V, E)$ be a weighted directed graph with vertices V and edges E . Each edge e has a weight (e) , representing the distance or cost associated with traversing that edge.

- S : The set of vertices for which the shortest path from the source vertex (initially empty) has already been found.
- Q : The set of vertices whose shortest distances from the source vertex are not yet finalized (initially contains all vertices).
- (v) : The current shortest distance from the source vertex to vertex v , initialized to infinity for all vertices except the source vertex.
- (v) : The predecessor vertex of v in the shortest path from the source vertex.

Dijkstra's algorithm iterates through the vertices, greedily selecting the vertex with the minimum shortest distance from S , until all vertices are included in S . At each step, the algorithm updates the shortest distances for vertices adjacent to the selected vertex.

Here's the step-by-step process:

1. **Initialization:**
 - $S = \emptyset$
 - $Q = V$
 - $d(v) = \infty$ for all $v \in V$, except $d(source) = 0$

- $\pi(v)$ =undefined for all $v \in V$
2. **Main Loop:**
 - While Q is not empty:
 - Select a vertex u from Q with the minimum (u) .
 - Remove u from Q and add it to S .
 - For each vertex v adjacent to u :
 - If v is in Q and $d(u)+w(uv)<d(v)$
 - Update $d(v)=d(u)+w(uv)$
 - Update $\pi(v)=u$
 3. **Termination:**
 - After the main loop, S contains all vertices, and $d(v)$ represents the shortest distance from the source vertex to vertex v for all $v \in V$
 4. **Shortest Path Reconstruction:**
 - Starting from the destination vertex, follow the predecessors defined by $\pi(v)$ until reaching the source vertex to reconstruct the shortest path.

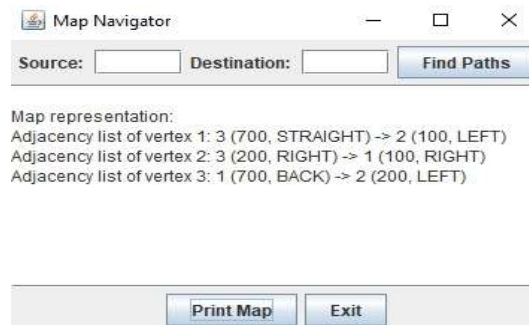
A. Map Creation Functionality

- The GUI interface provides users with intuitive tools for adding nodes and edges, facilitating the creation of detailed indoor maps.
- Users can specify edge properties such as distance and direction, enhancing the accuracy of the map representation.
- Map data is stored in a MySQL database, ensuring persistence and enabling seamless management and retrieval of map information.



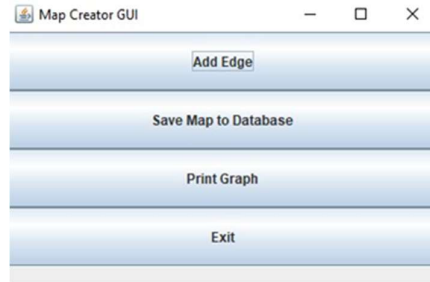
B. Map Navigation Functionality

- Pathfinding algorithms, including DFS and Dijkstra's Algorithm, are implemented to find optimal routes between specified source and destination nodes.
- DFS traversal explores paths recursively, while Dijkstra's Algorithm guarantees the shortest path, considering factors such as edge distances and directions.
- Users can interactively input source and destination nodes, triggering the navigation system to compute and display the optimal route.



C. User Interaction and Menu System

- The application provides a menu-driven interface for user interaction, offering options for map creation, navigation, and database management.
- Input prompts guide users to input source and destination nodes for navigation operations, ensuring ease of use and minimizing errors.
- Error handling mechanisms are implemented to handle invalid input and database connectivity issues gracefully, enhancing user experience.



Efficient storing and retrieval of map data is made possible via JDBC (Java Database Connectivity), which enables smooth interaction with MySQL databases. JDBC acts as a link between the Java programme and the database, enabling programmatic execution of SQL queries, data retrieval, and database activities. Performance, scalability, and dependability are all improved by this integration, which is essential for a strong interior navigation system.



IV. RESULTS AND DISCUSSIONS

An analysis of the indoor navigation and map generating system that has been put into place has produced some important findings. In terms of mapping, the system works by extracting map information from a database and letting users add edges that depict interior paths—along with their respective lengths and orientations. To evaluate the system's scalability and operational effectiveness, however, extensive testing is necessary, particularly when working with larger datasets. Furthermore, although the fundamental navigation features create paths between specified nodes well, real-time updates and dynamic route modifications should be made better. Enhancing the accuracy and usability of the system can be achieved through the integration of external APIs and GPS technology. In conclusion, these results highlight the system's potential utility for aiding in the building of interior maps and simple navigation tasks, while also offering directions.

V. FUTURE SCOPE

The project offers encouraging prospects for indoor navigation technology development in the future. Modern routing algorithms, multimodal transit compatibility, and enhanced editing capabilities are all integrated into the platform, which lays the groundwork for flexible and intuitive navigation experiences. Here is a list of a few of them:

Advanced Functionalities: To improve the accuracy and effectiveness of navigation, the project intends to integrate state-of-the-art routing algorithms. This incorporates multimodal and fluid mode switching capabilities, which enables users to switch between various modes of transportation with ease for a more thorough navigation experience.

Better Editing Tools: To make creating and customising maps easier, the project also adds better editing tools in addition to navigation improvements. To integrate other data sources and distribute maps between platforms, users can take advantage of the import and export features for map data. The project also provides annotable customisable visualisations that bring further context and insights into the indoor environment.

VI. CONCLUSION

To sum up, the project offers a reliable Java-based solution designed especially for creating and navigating indoor maps. The system utilises object-oriented programming, database integration, and sophisticated algorithmic techniques to provide a flexible solution that can be applied to a range of applications that need map-based features in indoor contexts. The project has the potential to be a useful asset in the transportation, urban planning, and logistics industries with room for growth and improvement. The project is well-positioned to tackle new issues and further the field of indoor navigation technology as it develops, which will ultimately improve productivity and usability in a variety of industries.

REFERENCES

- [1] Kwan, M., Gollidge, R. G., & Speigle, J. M. (1996). A Review of Object-Oriented Approaches in Geographical Information Systems for Transportation Modeling. UC Berkeley: University of California Transportation Center. Retrieved from <https://escholarship.org/uc/item/2qf6b23x>
- [2] Hamer, John. "Visualising java data structures as graphs." Proceedings of the Sixth Australasian Conference on Computing Education-Volume 30. 2004.
- [3] Koffman, Elliot, and Ursula Wolz. "A simple Java package for GUI-like interactivity." Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education. 2001.
- [4] Korunović, Ana, and Siniša Vlajić. "An Example of Integration of Java GUI Desktop Technologies Using the Abstract Factory Pattern for Education Purposes." ETF Journal of Electrical Engineering 29.1 (2023): 3-11.
- [5] Rachmawati, Dian, and Lysander Gustin. "Analysis of Dijkstra's algorithm and A* algorithm in shortest path problem." Journal of Physics: Conference Series. Vol. 1566. No. 1. IOP Publishing, 2020.
- [6] Mehta, Heeket, Pratik Kanani, and Priya Lande. "Google maps." International Journal of Computer Applications 178.8 (2019): 41-46.
- [7] Shrestha, Sratsav Ram. Graphical User Interface (GUI) Development for an Optical Communication Simulator. Minnesota State University, Mankato, 2017
- [8] Raut, Rushikesh S. "Research Paper on Object-Oriented Programming (OOP)." International Research Journal of Engineering and Technology (IRJET) 7.10 (2020): 1453.
- [9] Hamer, John. "Visualising java data structures as graphs." Proceedings of the Sixth Australasian Conference on Computing Education-Volume 30. 2004.
- [10] Stroustrup, Bjarne. "What is "Object-Oriented Programming"?(1991 revised version)." Proc. 1st European Software Festival. 1991.
- [11] BASIC CONCEPT OF OBJECT-ORIENTED PROGRAMMING International Research Journal of Modernization in Engineering Technology and Science Volume:05/Issue:09/September-2023
- [12] A Comparative Study and Analysis on the Performance of the Algorithms, IJCSMC, Vol. 5, Issue. 1, January 2016, pg.91 – 95.
- [13] Medak, Jogamohan. "Review and Analysis of Minimum Spanning Tree Using Prim's Algorithm." International Journal of Computer Science Trends and Technology (IJCTT)–Volume 6 (2018).
- [14] Zhou, Minhang, and Nina Gao. "Research on optimal path based on Dijkstra algorithms." 3rd International Conference on Mechatronics Engineering and Information Technology (ICMEIT 2019). Atlantis Press, 2019.
- [15] Arogundade, O. T., B. Sobowale, and A. T. Akinwale. "Prim algorithm approach to improving local access network in rural areas." International Journal of Computer Theory and Engineering 3.3 (2011): 413.