Optimizing Evacuation Routes: Rapid Pathfinding through Image-Based Grid Modeling and A* Algorithmic Route-Planning

Jordan Rowe
The Science Academy STEM Magnet

1 Abstract

When faced with an emergency situation, humans are largely unsuccessful in identifying the shortest paths. This can become increasingly difficult when there are many walls and obstacles present. The purpose of this experiment was to create a computational software that can expeditiously help the user navigate any given environment in the case of an emergency evacuation. Using pixel-level brightness analysis, an image-to-grid conversion pipeline was developed in order to distinguish walls and obstacles from walkable space. Building interiors were converted to discrete state spaces in order to apply them to real-life architecture. The A* algorithm was then implemented to find optimal routes under the detected constraints. The user is also able to refine the positions of walls, obstacles, exits, and their starting point. The resulting evacuation path is laid on top of the original floor plan for easy interpretation. After modeling a given environment as a grid, the A* algorithm was used to compute a cost function based on the nodes of each grid cell and the distances of these nodes from the start position and the exits. This computational software is able to find the most feasible paths without requiring comprehensive architectural models of the surrounding environment. This software bridges the gap between theoretical pathfinding algorithms and implementable safety protocols, with numerous applications including evacuation planning in schools, offices, and all other public facilities.

Keywords: evacuation routing, image-to-grid conversion, pixel brightness analysis, graph theory, A* algorithm, pathfinding, emergency safety planning.

2 Introduction

In emergency evacuations, every delay and routing error can significantly increase mortality rates, especially in large or complex buildings. In many public facilities, there exists an ongoing concern that in an actual emergency, occupants may not evacuate efficiently or have a clear direction of where to evade. Additionally, when occupants are under duress, cognitive abilities are momentarily diminished. Resultingly, they are not able to reliably interpret two-dimensional floor plans. Static maps and floor plans offer limited guidance in the case of an emergency. They do not adapt to the user's location nor do they aid in determining an evacuation route. The purpose of this project is to produce a software that utilizes image to grid modeling, graph theory, and A* algorithmic computation to yield optimal evacuation route plans.

Architectural interiors can be converted to graphs using grid-modeling. Discrete spatial regions are set to correspond to specific nodes on the grid. Based on adjacency relationships between walkable regions, a navigable network is formed. Using this strategy, topological features are preserved. Within architectural documentation, digital floor plans are readily available in image format. Using pixel-based analysis, walls and obstacles are easily distinguishable from navigable space. Once physical space is converted to computational form via grid modeling, A* can systematically evaluate any given environment to determine an optimal path. Even across varying environments, algorithmic routing is consistent and adaptable to user modifications after initial computation. Using this software, the way evacuation planning is approached can change entirely.

3 Methodology

3.1 Image-to-Grid Pipeline

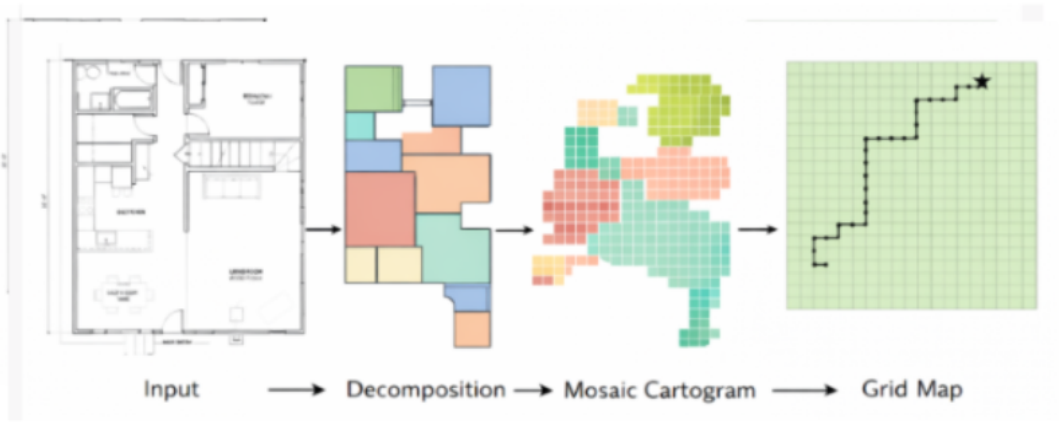

Figure 1: Image-to-grid pipeline: input floor plan, decomposition, mosaic cartogram, and resulting grid map.

Pixel-level brightness thresholds are applied to differentiate walls and obstacles. The image is then discretized into a grid where each cell is identified as either walkable space or restricted space. Spatial features including rooms and walls are preserved in grid representation.

3.2 Graph Theory

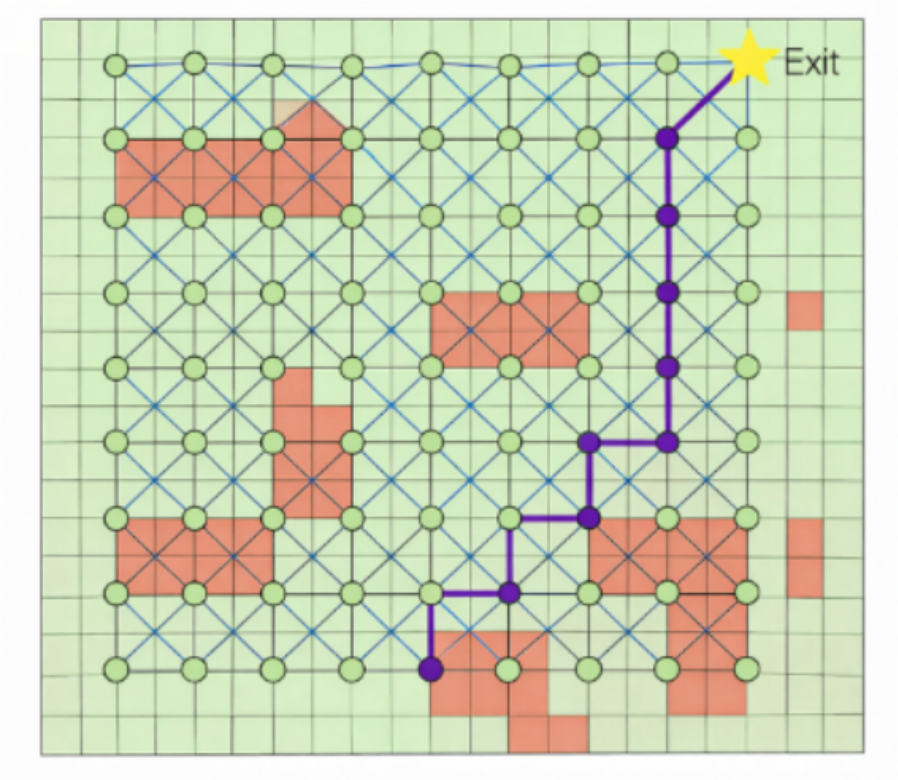

Figure 2: Graph-based representation of the grid map.

Using graph theory, the grid map was modeled as a graph where every walkable cell is represented as a node. Edges that connect each node are representative of valid movement between cells. Walls and obstacles are excluded from the web of nodes. Each edge is given a cost value that is dependent on the distance between nodes. The conversion of a grid map to node graph is necessary for a pathfinding algorithm to be applied effectively.

3.3 A* Algorithm

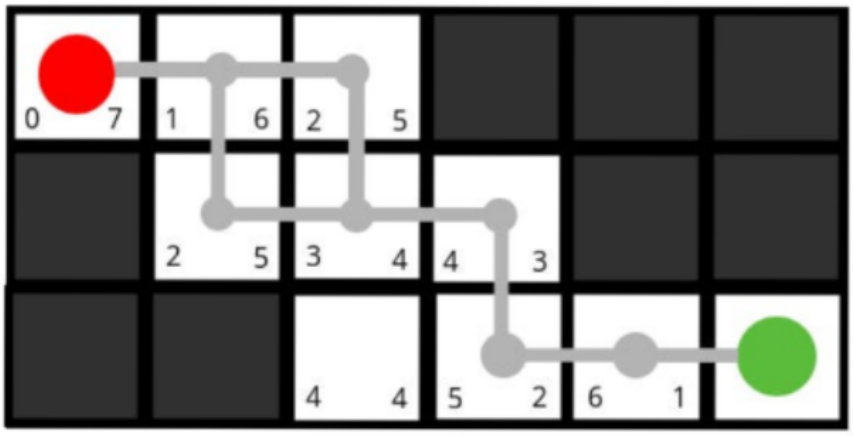

Figure 3: Example of A* pathfinding avoiding obstacles.

A* is a pathfinding algorithm that is used to compute the shortest route between two nodes via a cost function. Nodes are evaluated through the following function:

$$f(n) = g(n) + h(n)$$

g(n) denotes the distance traveled from the starting node to the current node. h(n) denotes the heuristic estimation of the remaining distance until an exit is reached. A* is able to evaluate a multitude of partial paths for efficiency while revising earlier decisions if a shorter path is found.

4 Research

In order to perform computational analysis on complex indoor environments, it is vital to first make these environments algorithmically interpretable. Exact geometric precision is not necessary in order to model evacuation routes. According to "Fire Safety Journal", "In evacuation modeling, overly precise geometric detail often increases computational cost without improving routing accuracy." Therefore, grid based discretization does not distort the validity of a floor plan. As long as walkable cells are linked and walls are blocked out, it is redundant to preserve exact spatial dimensions. Grid modeling is not only topologically accurate but also provides a computationally efficient representation of a wide range of architectural configurations.

A real-life building consists of continuous space, which makes it difficult for a computer to work with. In order to discretize this space, it needs to be divided into small, fixed-size grid cells. When a grid cell is deemed walkable, it is treated as a node in a graph. If the grid cell contains a wall or obstacle, it is excluded from the graph. The node will then store its position (row and column) and detect whether they are next to any neighboring nodes so they can determine which regions are walkable. When walkable regions are adjacent, this is registered in the graph as an edge. Using this format allows for the use of graph-search algorithms. Graph abstraction is highly effective in allowing navigation to be treated as a structured search problem because it systematically models spatial regions as nodes and feasible movement as edges. Additionally, realistic movement constraints are obstacles and walls are excluded from the graph. This method of graph based modeling provides a strong foundation for evacuation planning.

Modeling evacuation paths can be boiled down to a constrained navigation problem with spatial limitations. Therefore, it can be framed as an optimization problem. Through graph theory, all feasible paths can be determined. In order to determine the efficiency of these paths, their distance-based cost functions are necessary to produce a tangible criterion for route comparison. Using this approach the optimal evacuation route is essentially deemed as a path that reduces evacuation time and exposure to risk. In order to evaluate all paths consistently, under the same criterion it is necessary to implement optimization frameworks. Optimization is the bridge between the use of graph theory and search algorithms. It is a rigorous method for path planning that is considerably superior to intuitive decision-making.

A* Algorithm is the most applicable search algorithm for evacuation routes because of its goal-oriented nature. A* assigns a numerical cost to each feasible path using the function f(n)= g(n) + h(n). The function f estimates the cumulative cost of a path passing though nodes n. g(n) models the cost required to reach the closest node n from a given starting position. In a grid-based environment, this is easily quantifiable via monitoring valid movements between grid adjacent cells. h(n) models the cost required to reach an exit from node n. This function is computed using Manhattan distance in grid-based layouts.

Manhattan distance is defined as

$$h(n) = |x - x_{\text{goal}}| + |y - y_{\text{goal}}|$$

This formula validates the heuristic component of the function by ensuring that the true remaining cost is never overestimated. Resultingly, A* is mathematically guaranteed to compute the optimal path. It does this by maintaining a priority queue of ordered f(n) from lowest to highest. At each iteration, the node with the lowest f(n) will be temporarily selected. Then new f(n) values will be computed for neighboring nodes in order to discover the lowest-cost path. When a new node is selected, a new partial path is formed and f(n) values are updated correspondingly. There is essentially a competition between these partial paths until a web of nodes deemed most efficient are produced and are subsequently mapped to the shortest possible path.

Due to its unique ability to analyze a multitude of partial paths simultaneously, and therefore avoid redundant commitment to inefficient paths, the A* algorithm is a perfect fit for computing optimal evacuation routes. Another distinct capability of the algorithm is that it can revise earlier decisions if the f(n) values prove to be greater. A* is also able to account for spatial constraints by excluding invalid nodes from the graph representation. Additionally, path identification is remarkably fast with A*, as it terminates exploration of unnecessary paths immediately which avoids computational delay. A* is applicable to virtually all building layouts, since grid-based modeling allows for A* to operate on arbitrary structural configurations. Due to this flexibility A* can be utilized in a wide-range of public facilities. These distinct properties set A* apart from other search algorithms, proving it to be a practical tool for real life navigation.

5 Models

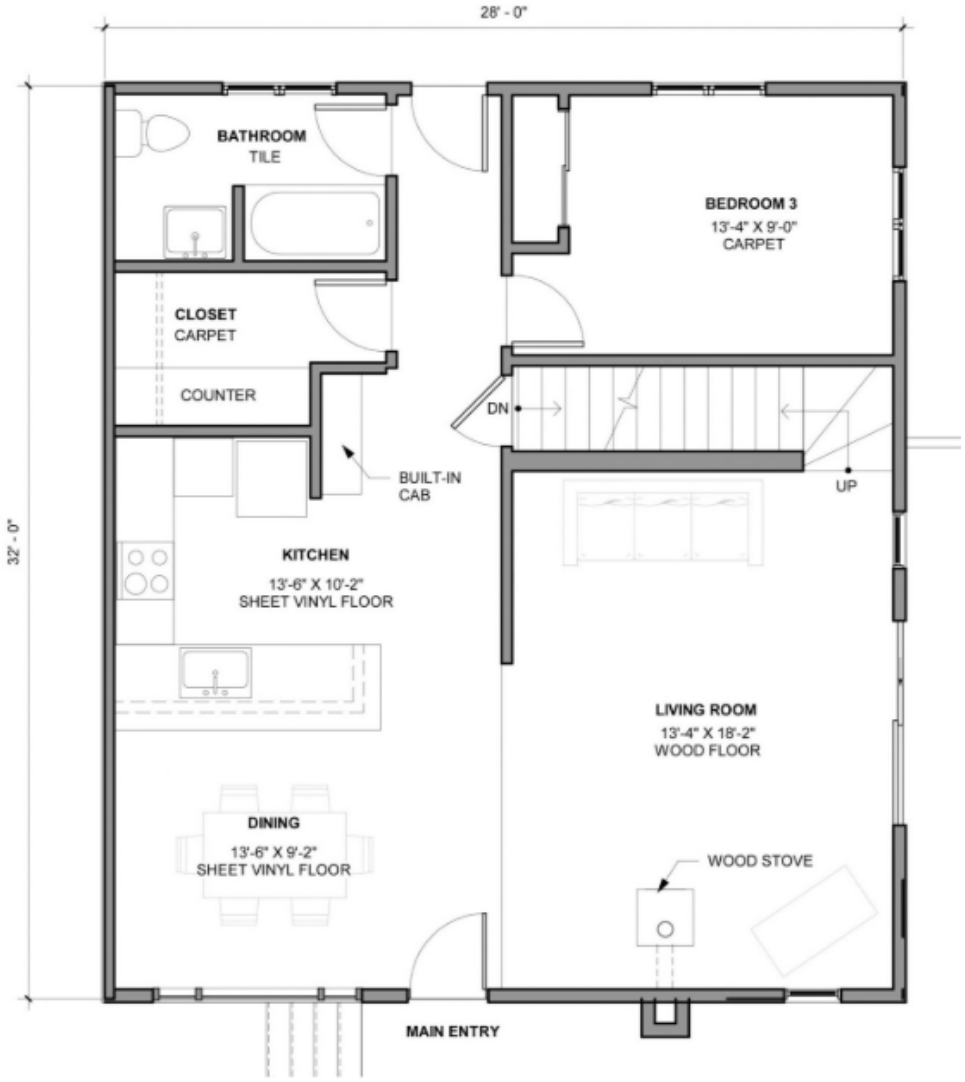

Figure 4: House Floor Plan

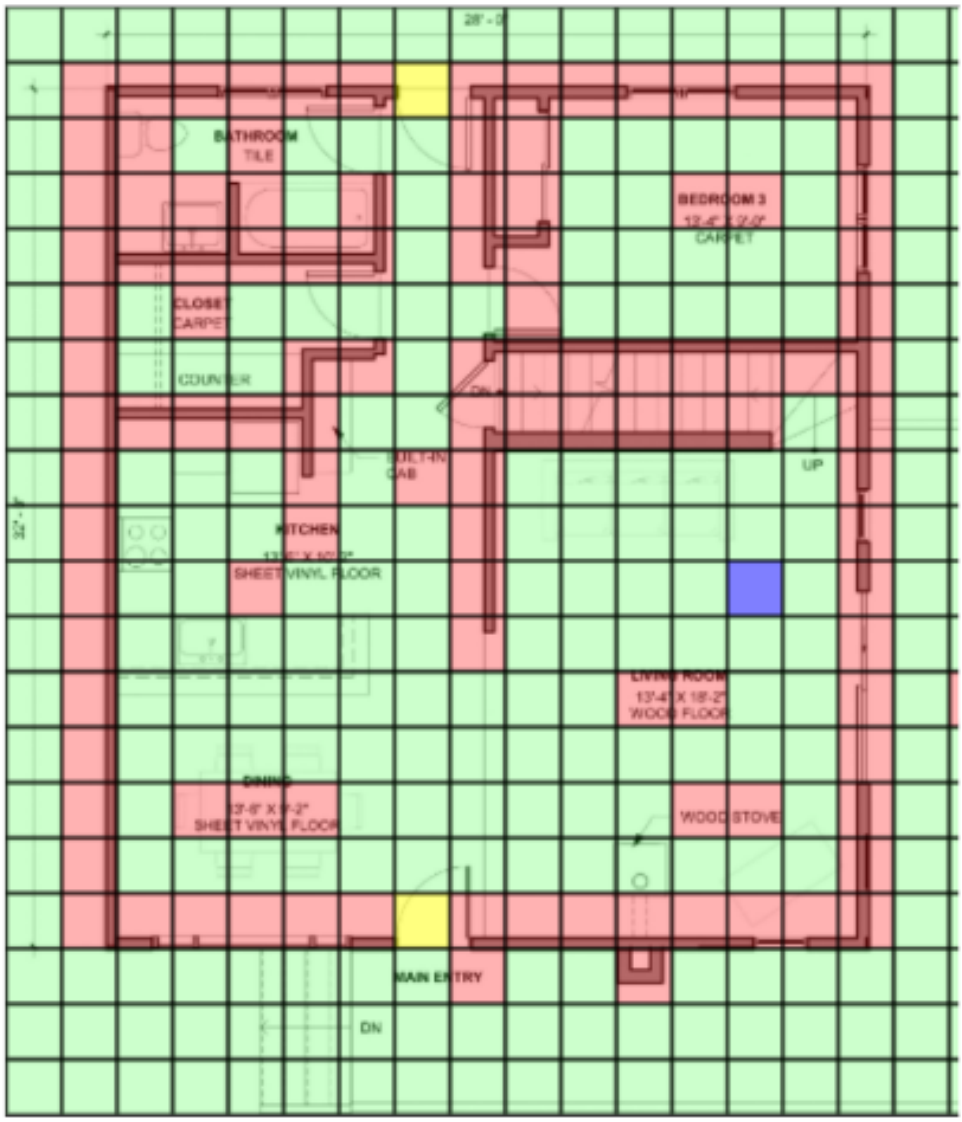

Figure 5: House Floor Plan with Exits (Yellow) and Start position (Purple)

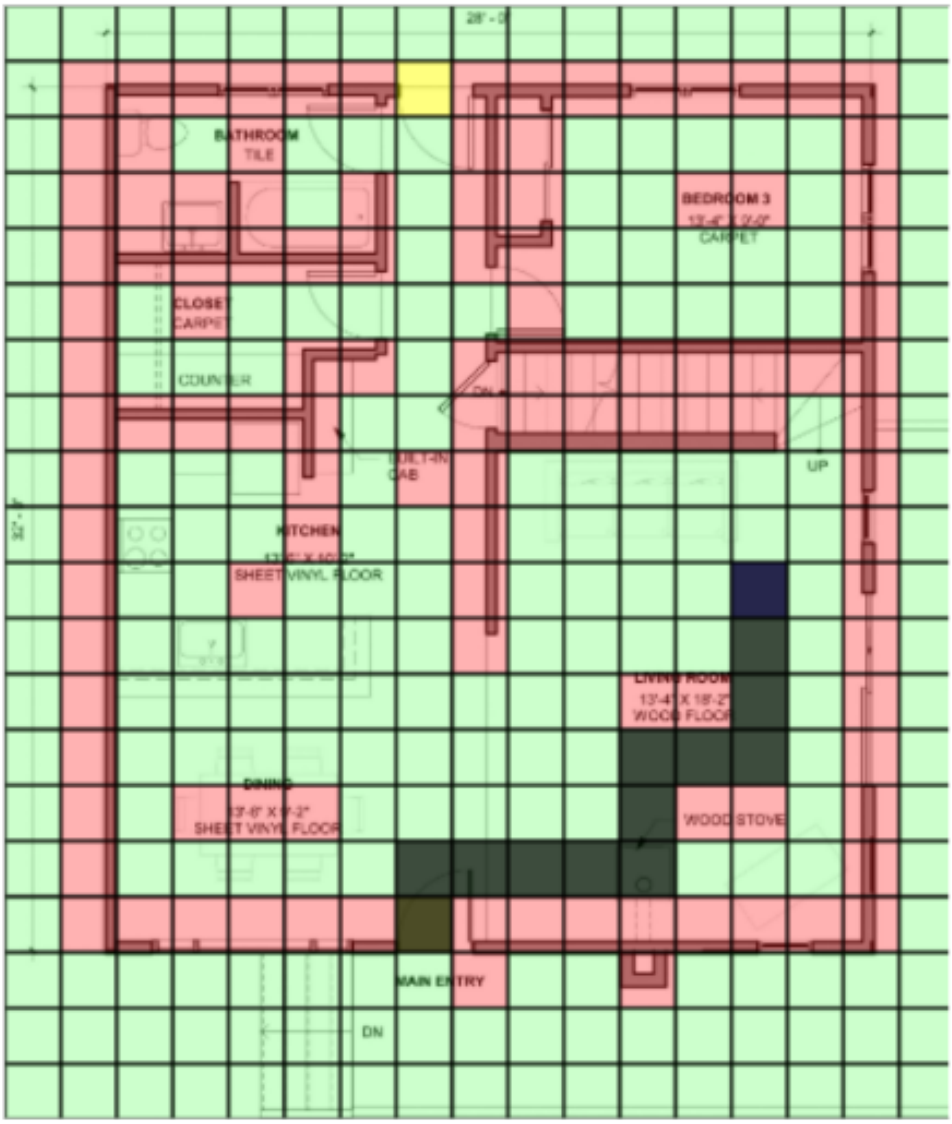

Figure 6: House Floor Plan After A* Algorithmic Computation

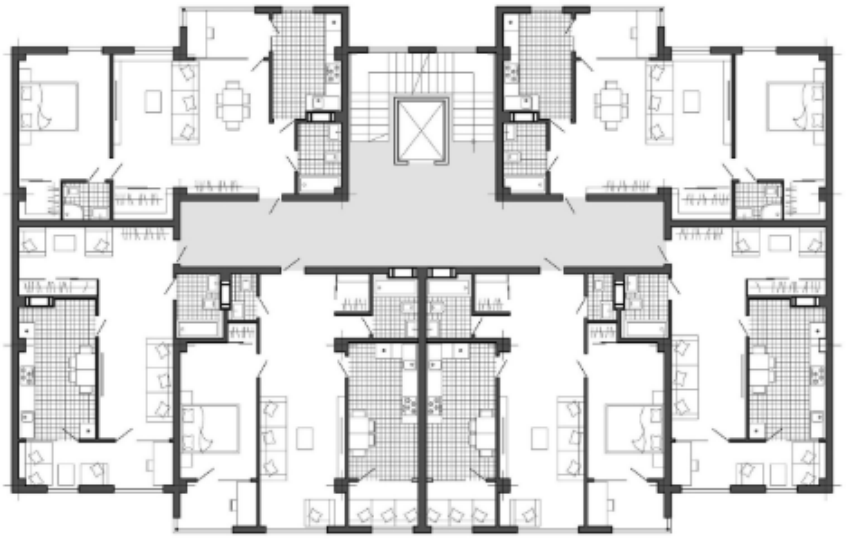

Figure 7: Complex Building Floor Plan

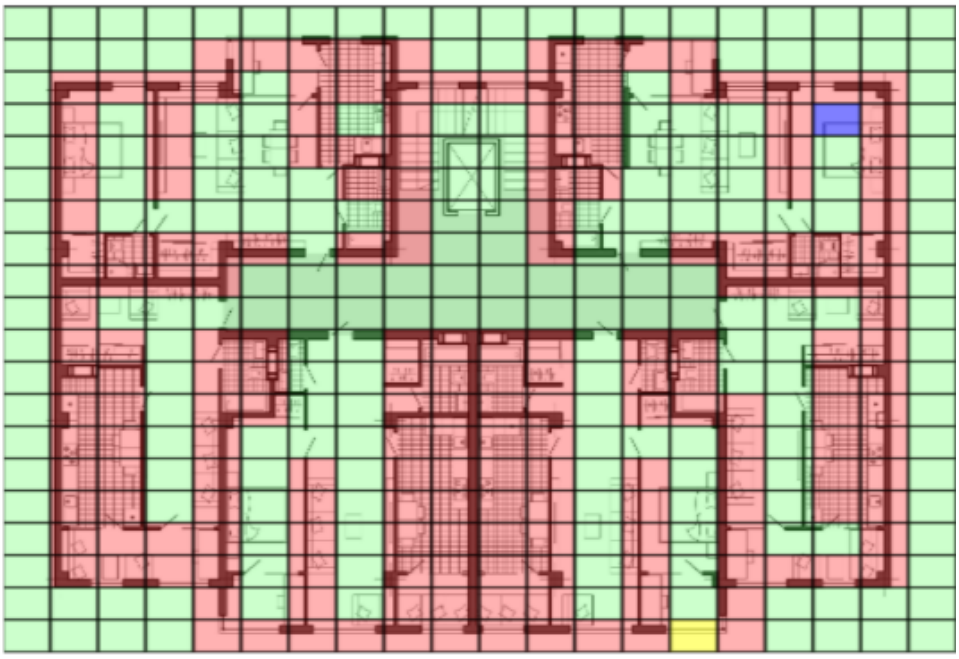

Figure 8: Complex Building Floor Plan Exit (Yellow) and Start position (Purple)

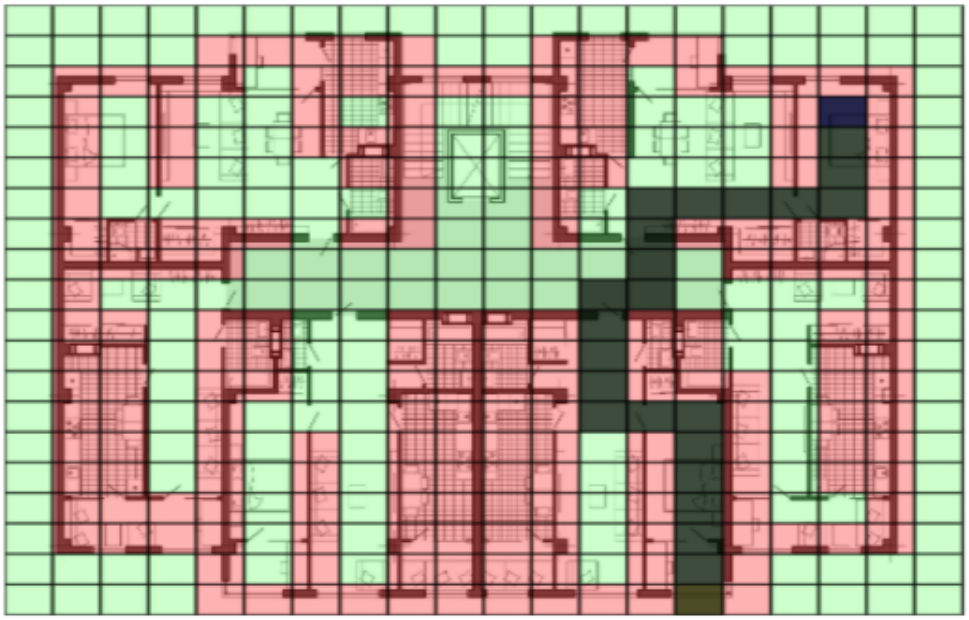

Figure 9: Complex Building Floor Plan After A* Algorithmic Computation

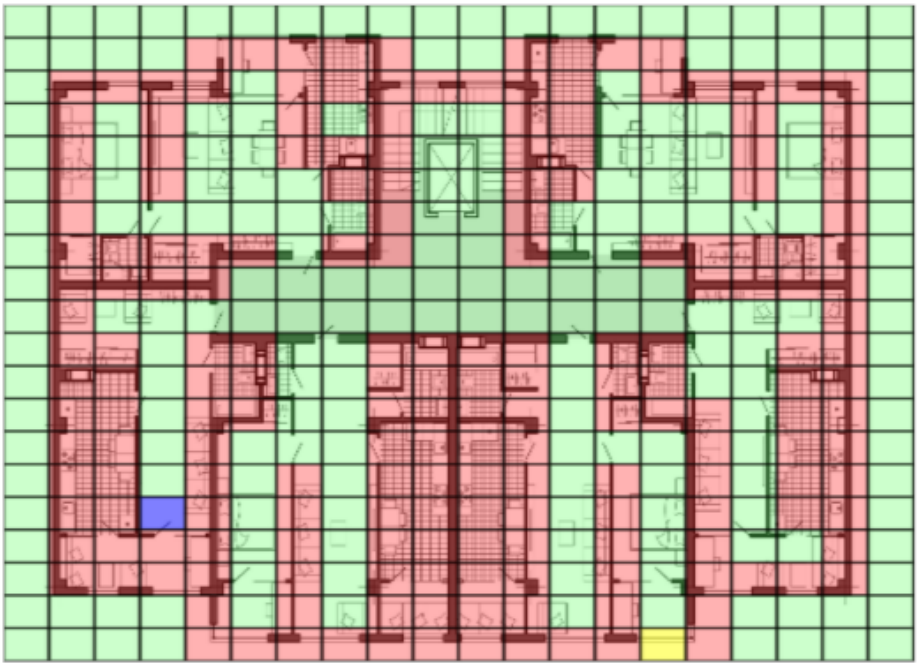

Figure 10: Complex Building Floor: Starting Position #2

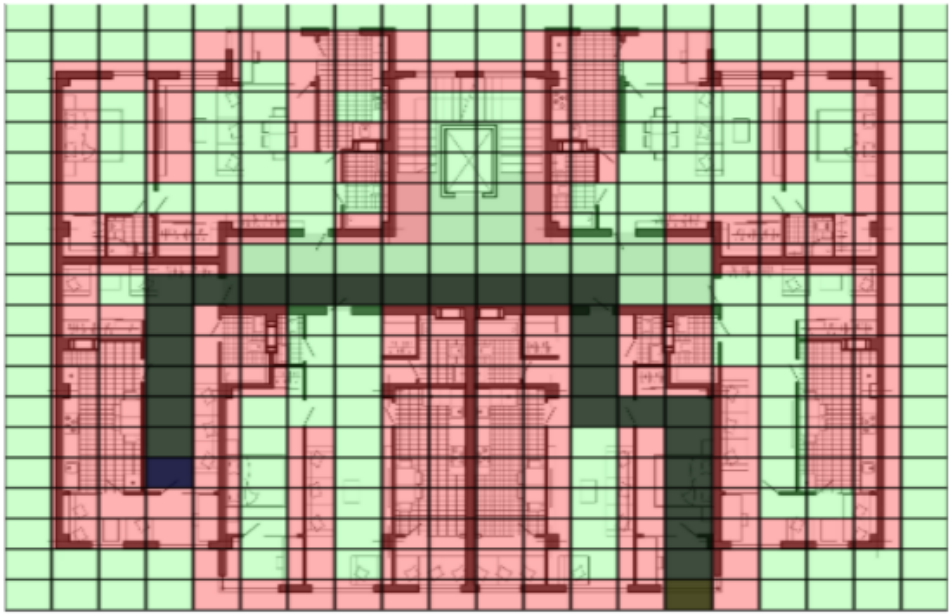

Figure 11: Complex Building Floor: Starting Position #2 After A* Algorithmic Computation

6 Discussion

The software was successfully able to output an optimal evacuation route given floor plan images. Using pixel-brightness and grid conversion was very effective in differentiating walls and obstacles from walkable space. After computation, the route is overlaid on the original floor plan, so it is easily interpretable. In the case of line thickness causing misclassified walls or exits, user modifications are available when selecting the positions of walls, exits, and starting position.

This project has a direct real-world application in safety-planning for public facilities. According to the US Fire Administration, over 108,500 evacuations take place each year in nonresidential buildings. This software functions rapidly enough for it to be used to determine optimal escape routes on the spot. The clear overlaid path is easy to follow which is important in emergency situations when users are under pressure and time is limited. Additionally, this software can be utilized for contingency planning by generating alternative routes ahead of time in the case that certain areas of the building become inaccessible. The system also addresses real-life variability. More specifically, evacuation drills assume ideal conditions, but in real evacuations, there is a factor of uncertainty caused by blocked exits and unprecedented blockages. However, the software can account for adverse conditions by eliminating routes that are no longer feasible. Although older buildings typically lack digital architectural models, this system is still able to be implemented, because it only requires a floor plan image. By prioritizing interoperability, accessibility, and adaptability, this software shows a strong potential for real world public safety planning.

7 References

1. Gwynne, S. M. V., Galea, E. R., Owen, M., Lawrence, P. J., & Filippidis, L. (1999). A review of the methodologies used in the computer simulation of evacuation from the built environment. *Fire Safety Journal*, 33(4), 303–327. https://doi.org/10.1016/S0379-7112(99)00013-2

2. Kuligowski, E. D., Peacock, R. D., & Hoskins, B. L. (2010). *A review of building evacuation models*. NIST Technical Note 1680. National Institute of Standards and Technology.

3. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107. https://doi.org/10.1109/TSSC.1968.300136

4. Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

5. Sime, J. D. (1983). Affiliative behavior during escape to building exits. *Journal of Environmental Psychology*, 3(1), 21–41.

6. Chen, X., Li, S., & Tang, Y. (2017). Automatic extraction of indoor navigation graphs from floor plan images. *ISPRS International Journal of Geo-Information*, 6(6), 180.