Path Planning for Autonomous Cinematic Drones

Brayden Tantisira

1. Abstract

This project explores the autonomous movement of drones in aerial cinematography. Specifically, based on certain locations in a scene, I examined algorithms that will allow a drone to autonomously find the smoothest path to reach each location while maintaining smooth camera quality. Drones play a significant role in modern cinematography, particularly in creating dynamic aerial shots and panoramic photographs. Given their importance, drones must operate smoothly and reliably. Implementing autonomous path planning will reduce common challenges, such as collision risks, while also limiting manual navigation, minimizing the need for human intervention while enabling safer, more efficient, and faster filming with high-definition (HD) video quality. I used a simulation system, Gazebo, and the ROS (Robot Operating System) interface as an open-source platform for coding, and together they simulated autonomous drone movements. By creating and using multiple path planning algorithms—A-star (A*), Dijkstra's, and Greedy Best First Search (GBFS)—I enabled the drone to autonomously move to a certain position while accounting for obstacles in a static environment. The drones were able to efficiently detect obstacles and calculate optimal routes, with A* demonstrating the highest average efficiency. In comparison to Dijkstra's algorithm, A* often reduced pathfinding time by around 70%. Additionally, its paths were generally more accurate than the GBFS algorithm in various situations, reducing travel time by around 30% on average when larger obstacles and environments were tested. The results of this study present A* as the optimal path planning algorithm that can be implemented to improve autonomous drone cinematography in future applications. Furthermore, another form of path planning, which uses roadmaps to autonomously move a drone, is also explored to help prevent concerns with restricted areas and cluttered environments, and to allow personalized cinematography on human-made paths.

2. Introduction

While Artificial Intelligence (AI) drones are being applied in various fields, their potential in cinematography offers unique challenges (such as ensuring reliable precision control) and exciting opportunities (like the use of multiple drones in cinematic scenes), especially for achieving complex camera movements autonomously. Currently, many drone companies are exploring ways to incorporate AI into aerial cinematography with heavy research in optimal shot prediction, complex maneuvers, object tracking, and motion recognition (Martijn, 2023). While the implementations of this technology are still relatively new due to the recency of major AI breakthroughs, these drones have evolved to navigate, make decisions without human control, create 3D maps, avoid obstacles, and more (Wayas, 2024). Integrating the use of new AI technology into cinematography will also give drones the ability to move on their own to locations that may be inaccessible or unviewable by humans. Traditionally drone cinematography required multiple highly trained professionals to both align the camera for smooth angles and to move the drone. Newly enhanced AI drones reduce the need for constant human intervention, making drone cinematography more accessible to less experienced filmmakers while maintaining professional-quality shots. At the same time, autonomous drones equipped with obstacle detection will allow for safer, quicker, and easier filming without compromising video quality. Still, the use of AI in drone cinematography is relatively new, and most drone companies continue to rely on manual control rather than AI-driven solutions. Thus, with continual research on incorporating AI and advanced path planning into drones, they will be able to seamlessly navigate complex environments, ensuring smooth, uninterrupted shots, enabling filmmakers to achieve a higher level of cinematic precision, and enhancing both the ease of operation and the overall user experience.

My project focuses on implementing various path planning algorithms to optimize drone navigation for cinematography. Path planning algorithms are methods used in autonomous robots in order to find the safest, most efficient, collision free, and least cost travel paths from an origin to a destination (Karur et al., 2021). They are most commonly known for their use in self-driving cars which allow for easy travel without the need for a human driver. However, their extensive use in drones is currently very challenging due to various factors, including gimbal rotation limits on cameras, battery constraints, light drone weights, cluttered environments, unpredictable weather, and other moving objects in the air (Gungan and Haque, 2023). While there are many different drone path planning algorithms which are chosen in order to reduce time, and maximize efficiency and accuracy based on the specific environment, the path planning algorithms used in this project ensure drones can reach virtually any location on a map while maintaining smooth camera transitions and optimal speed for cinematic purposes.

By constantly refining these path planning algorithms for specific professional applications, they will help naturally expand the use of drones to more fields, enhancing the future of human robot interactions. Although it's important to acknowledge the dangers of drones, especially with ethical and security concerns, heavily reducing such challenges was beyond the scope of this project which focuses on flight and cinematographic efficiency. By implementing strict guidelines and thoughtful design to mitigate current challenges, we can safely embrace a drone's capabilities and harness its full potential. In drone cinematography specifically, extending their recreational use while abiding by professional guidelines is essential to augment the aerial filmmaking experience.

3. Background

Despite being a relatively recent field, research on drone path planning algorithms for cinematography has already emerged. In their algorithm on drone trajectories, Sabetghadam et al. (2019) propose a method for autonomous flying with optimal trajectories and smooth cinematography to combat the trade offs of applying certain path planning algorithms to drone cinematography, as obstacles and gimbal rotation limits often result in poor video quality. Their algorithm optimizes autonomous movements in cinematography by finding the best angle in which to film while reducing camera jerk and producing visually appealing videos. However, the algorithm's effectiveness was limited by the drone's movement range and speed, which impacted its ability to maintain camera quality.

In contrast, Bonatti et al. (2018) produced an algorithm that delved more into the artistic part of drone cinematography and path planning. Their alternative algorithm which uses GPS systems for mapping identifies appropriate camera movements that can find collision-free, occlusion-free and smooth paths while reducing the cost. In addition, their algorithm is able to film dynamic shots, such as circling a person, following a vehicle, and soaring over a mountain. The algorithm resulted in a smooth system that can react based on motion, deal with noise, and move fast. Nevertheless, relying on GPS to map their environment meant that their algorithm was not able to respond solely based on camera input. Additionally, it was limited to only one drone and required user input to pick the shot that it takes, restricting autonomous capabilities.

While these other approaches create their own path planning algorithms, mine compares certain pre-existing and popular path planning algorithms used for autonomous movement of drones. Additionally, my approach explores the use of path planning in both known and unknown mapped areas in order to enhance the cinematic experience in various real world situations. By utilizing multiple different path planning algorithms, research explored in this paper delves deeper into methods that can be used in a multitude of autonomous situations, rather than focusing on algorithms used for a single cinematic shot.

4. Methodology/Models

Path planning for drone cinematography is typically applied in two distinct scenarios. The first, and most common for autonomous drones, involves flying in unmapped or unknown areas. The second pertains to situations where pre-existing maps of the area are available. In this project, while the latter is briefly modeled in sections dealing with roadmap-based path planning, the former is explored by testing multiple path planning algorithms to identify the most efficient option across various environments.

4.1 Coding

This project employed a life-like simulation software called Gazebo for drone visualization. By incorporating certain path planning algorithms and setting up an environment similar to that in the real world, Gazebo can demonstrate the path planning algorithms at work. Moreover, ROS, or Robot Operating System, is a framework system used in order to create software for robots. Combined, Gazebo and ROS worked seamlessly together to create realistic robot conditions: Gazebo created realistic 3D simulations of robotic environments while ROS's modular structure allowed integration of path planning algorithms with ease. By pairing both ROS and Gazebo together, as well as RVIZ—a tool to visualize the Gazebo navigation maps in 2D—I was able to create and visualize drone path planning algorithms for cinematography in real time, without the need for a physical drone. The three main path planning algorithms tested by these simulations tools and compared in this project—Dijkstra's, GBFS, and A*—were evaluated based on efficiency, storage costs, and accuracy. Furthermore, the roadmap-based path planning model for pre-mapped environments was also tested using these simulation tools and is proposed for autonomous travel in certain realistic drone situations.

4.2 Path Planning for Unknown Maps/Territories

In unknown maps and territories, Simultaneous Localization and Mapping (SLAM) is often used in order to create a map of the environment and determine the drone's location. In this project, the Cartographer system in ROS was used to provide SLAM, creating maps and localizing the drone in its environment. These maps allowed the path planning algorithms to then be used. Additionally, when using the path planning algorithms below, there were a few assumptions needed about the drone's environment and received data. Specifically, it is assumed that the environment is static, a map of the environment has already been created using the Cartographer, the aerial vehicle knows its own location within the map, and it can move freely throughout its environment in all directions, as long as the space is not an obstacle. In path planning algorithms, a graph is used to represent the map of the drone's environment. On the graph, there are points called nodes, and an edge is the path that connects the nodes. Each edge is assigned a weight, representing the cost (or travel distance) of moving between nodes. The total path cost is calculated by summing these weights. In the following three algorithms, the edge weight between nodes remains constant at one.

4.3 Dijkstra's Algorithm

Dijkstra's algorithm is one of the most popular path planning algorithms which is used to find the shortest paths between nodes in a weighted graph. It can be used in situations aside from robotics, such as network routing, GPS navigation, and telecommunications (Bhattacharyya, 2023). However, in this paper I will mostly discuss the use of path planning in autonomous drones. The travel cost in Dijkstra's algorithm is the cost to a node from the starting point. The total cost and weights were found using Manhattan distance. In Manhattan Distance, only the up, down, left, and right neighbors of the current node will be searched, while diagonal neighbors are excluded (Manhattan Distance is elaborated on more with the GBFS algorithm). In the case of my project, each weight (or node-to-node travel distance) is set to one. Below is a step-by-step explanation of how the algorithm works:

1. *Start at the origin node:* Begin at the selected origin node as the initial parent (current) node. Set the travel cost (often stored as g_cost) to zero.
2. *Consider the neighbors of the parent node:* Check adjacent nodes to the parent node to see if they are obstacles. If they are not, add them to the open list, which contains nodes which have not been processed as the parent node. This will be used to keep track of nodes which need to be searched as parent nodes.
3. *Update the travel costs:* Specify the travel distance from the parent node to each neighbor node. Update and store the travel cost (which is equal to the travel distance to the neighbor plus the travel cost of the parent node) and parent of each neighbor if they have not been neighbor nodes before or if the new travel cost is lower than previously recorded. Move the parent node into the closed list (which contains nodes that have already been explored), so it will not be searched again.
4. *Repeat the process*: Pick a new parent node from one of the neighbor nodes in the open list. Always choose the node with the lowest travel cost, but if multiple nodes have the same travel cost, any one of them can be chosen. Check the new parent node's neighbors and update travel costs, parent nodes, closed lists, and open lists. Note: The travel cost for a node would be the total distance from the origin node, not the travel distance from the current node to one of the neighbor nodes.
5. *Continue until reaching the destination:* Repeat the process in Step 4 until reaching the destination (goal) node, while continuously updating the travel cost, the open and closed lists, and the parent nodes. If the open list becomes empty and the destination has not been reached, this indicates that no path exists to the destination.
6. *Find the shortest path:* After reaching the goal node, trace backwards from the goal node to the origin node by using the parent nodes. This will reveal the shortest path, and the travel cost would be stored in the goal node.
7. *Move to the goal node:* Finally, move the robot along the identified shortest path to the goal node.

Dijkstra's algorithm is able to find the shortest direct path to any location on a map (as long as a path exists). However, it is an uninformed (or blind) search algorithm, meaning that it doesn't know which direction to go or if one node is closer to the goal than another. This causes the algorithm to search in a radial pattern from the start node, making it very CPU intensive, time consuming, and slow, especially with larger maps. Moreover, as the map size increases, the amount of storage required also greatly increases, as there will be an increasing amount of nodes in the open and closed lists. This is where the Greedy Best First Search algorithm comes in.

4.4 Greedy Best-First Search

The Greedy Best First Search (GBFS) algorithm is an informed search algorithm that uses a heuristic function to estimate the distance to the goal. This allows it to prioritize exploring nodes that are closer to the goal, meaning it is often more efficient than Dijkstra's algorithm as it generally doesn't have to search many irrelevant nodes to find the most efficient path to the destination. There are many different types of heuristic functions, however the most common ones are Euclidean distance and Manhattan distance. Euclidean distance is the straight line distance between the start node and the goal node (ignoring obstacles, just focusing on the displacement, the distance from the start to the end). Its function can be defined as $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ where d is the distance, x is the horizontal distance, and y is the vertical distance on a 2D map. While Euclidean distance is good for short, open maps, for larger maps it is often computationally expensive as it uses square root functions. Thus, Manhattan distance is often used for paths that are longer or have more obstacles. Since it doesn't require square root calculations, it is simpler and faster to calculate. Like in cities, it uses right angles and is defined by $d = |x_2 - x_1| + |y_2 - y_1|$. While there are many more different heuristics, for the GBFS algorithm I used Euclidean distance as the maps tested were relatively small. As previously stated, GBFS solely uses a heuristic to find the goal, meaning it searches and prioritizes nodes that have a closer estimated distance to the goal. Due to this, it can lead the drone to a path that looks promising but may not be due to large obstacles blocking the path. Although the path may not always be the shortest, GBFS is typically faster than Dijkstra's, often using over 90% less memory and reducing computation time by over 80% compared to Dijkstra's algorithm. However, since it expands to nodes that appear to be closer to the goal, it may plan a path going to nodes that are near the goal but are blocked off by obstacles, ultimately leading to the drone having to go backwards to get around such obstacles, causing an increased total travel time. In these situations, the h_cost (total heuristic cost) in GBFS is larger than the g_cost (total cost) in Dijkstra's algorithm since obstacles that are blocking that path cause the overall path to increasingly become longer. This means that it tends to be more accurate in open or less cluttered maps with fewer obstacles, but in very crowded or large maps with a lot of obstacles, it can take suboptimal routes and thus be longer.

4.5 A* Algorithm

A* ("A-star") is one of the most popular path planning algorithms which combines Dijkstra's and GBFS algorithms, making it more efficient than Dijkstra's but more accurate than the GBFS Algorithm. A* finds the distance from the goal node to the current node using a heuristic, like GBFS does, but also finds the distance from the start node to the current node, like Dijkstra's. The combined cost of the node in A* algorithm is equal to the travel cost from the start node to the current node and the heuristic cost from the current node to the goal node. For drones, I used this to find the travel cost from one point to another, with f_cost being the total cost, h_cost being the heuristic cost and g_cost being the travel cost from the start node to the current node. In other words, the f_cost was equal to the g_cost plus the h_cost. Whenever A* is evaluated, it looks for the node with the smallest f_cost. A* is an optimal algorithm meaning it will always find the shortest path and with relative speed, however downsides include not being able to account for dynamic obstacles causing constant recalculation and time consumption in dynamic situations, and with increasingly larger maps its optimal paths will require significantly more space and path searching time. These challenges to A*, however, were not explored in the scope of this project.

4.6 Roadmap-Based Path Planning for Pre-Existing Maps

Roadmap-based path planning uses road maps from OpenStreetMap (OSM), which maps out any location on Earth and provides up-to-date data that is free and open to any user. By adding a specific location from OSM and specifying nodes, ways, relations, and tags, we can create a roadmap with different determined lengths between nodes and create a drone that can fly above roads from one location to another while avoiding off-road areas and obstacles. In essence, this roadmap-based path planning can work on any mapped location on OSM. For this project, a map of the Berlin Zoo along with Dijkstra's algorithm was used for such path planning. Roadmap-based path planning using OSM was tested as a distinct method used for drones given a pre-existing map. While one of various mapping options for known maps, roadmap-based path planning was chosen due to its ability to be implemented in the real world, where large obstacles and restricted areas where drones cannot go are prominent. Since OSM contains constantly updated data and can be used with information such as no-fly zones, it can reduce concerns of autonomous drones going into private areas or restricted zones. Additionally, using a predefined map can reduce challenges with too many unknown obstacles existing in areas off known paths, especially in densely built environments such as cities. Furthermore, it allows for efficiency in filming as predefined maps already contain layouts of the environment, so it does not need to scan and create a map from scratch. It will also allow for cinematic shots along pre-mapped pathways for videos of many realistic experiences, such as shots following a person running. Thus, while it may not produce the most optimal route from one place to another (as it avoids off road paths, which are often faster), roadmap-based path planning is very useful in real life applications of drone cinematography.

5. Results and Discussion

5.1 Mapping Unknown Territory

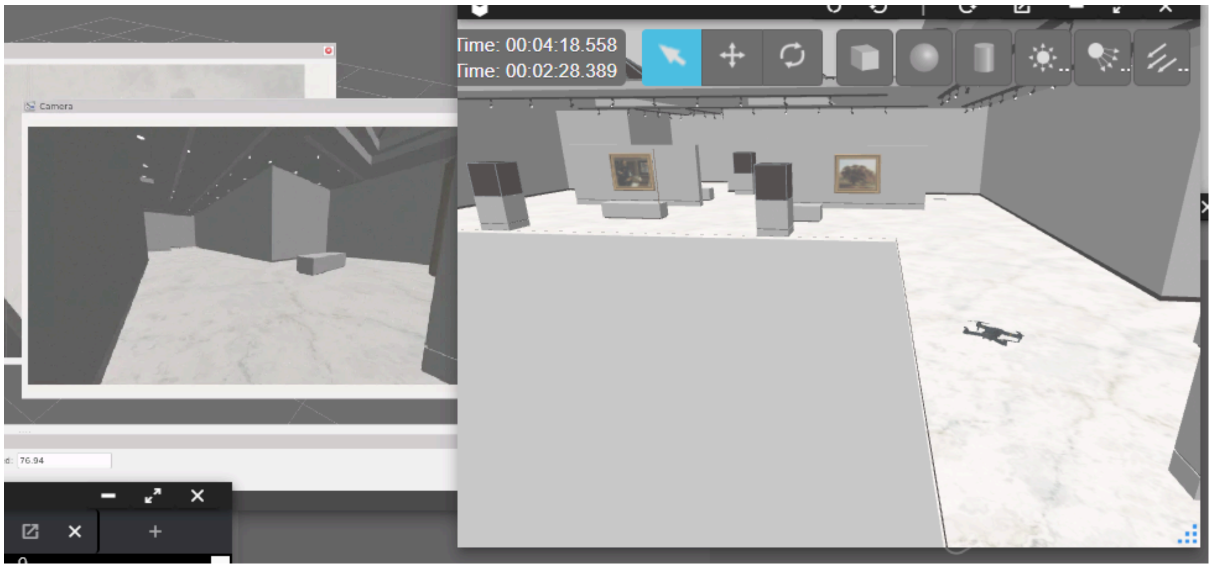
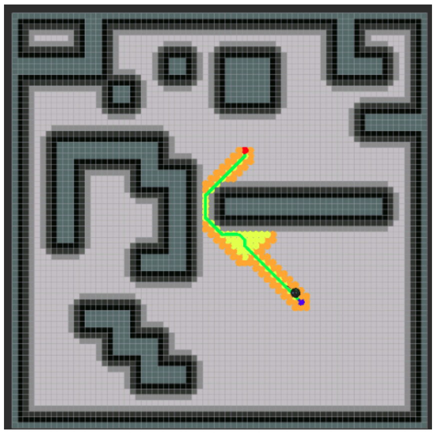


Figure 1: The figure above is a snapshot of the drone environment used for path planning. Even with higher speeds, the drone's camera was able to maintain film without too much noise or camera movement.

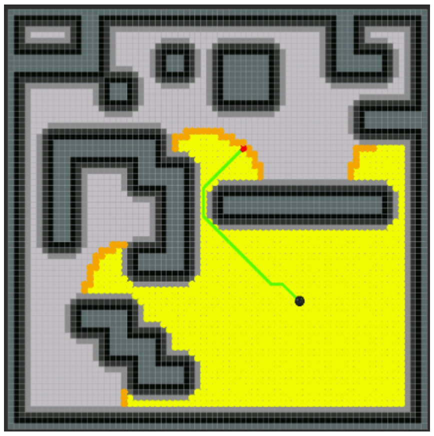
From my research, it was found that, for other than the simplest terrain in unmapped environments, A* is the most efficient algorithm used in various situations. The environment used for the project utilized a quadcopter drone and was a premade environment of an art gallery (**Figure 1**). In order to set up the path planning for the drone, this environment was first mapped. As previously stated, the localization and mapping process relied on the ROS Cartographer, which allows for map creation and localization in unmapped areas or environments. Since drones navigate based on their surroundings, the combination of LIDAR and Odometry data was used by the Cartographer in order to create a map of the environment with obstacles, setting up the drone for autonomous exploration. Next, by using drone localization through the AMCL (Adaptive Monte Carlo Localization) package, the drone's starting position was found on this newly created map and the drone environment was ready for navigation using path planning. Then the algorithms A*, Dijkstra's, and GBFS were tested. In a similar environment, the resulting figures and statistics were produced, showing the differences between the three algorithms (**Figure 2, Tables 1 and 2**).

Small Obstacle Map

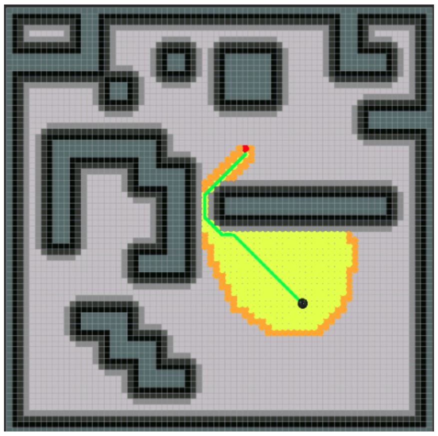
A. Greedy Best First Search



B. Dijkstra's

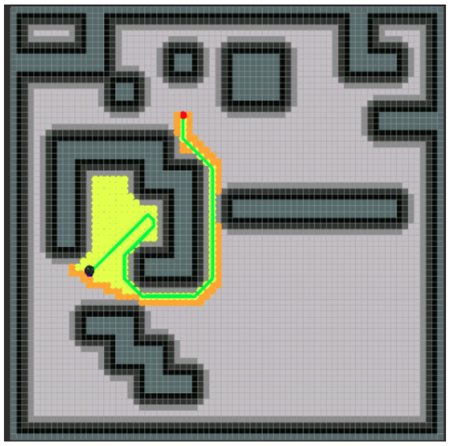


C. A-star

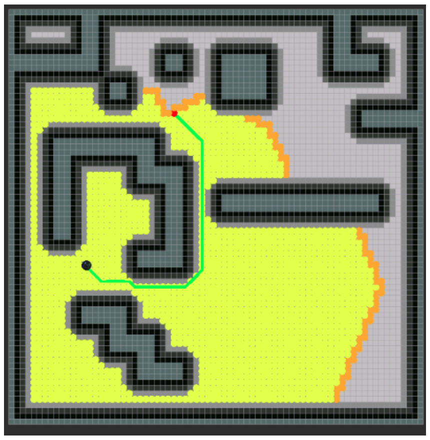


Large Concave Obstacle Map

D. Greedy Best First Search



E. Dijkstra's



F. A-star

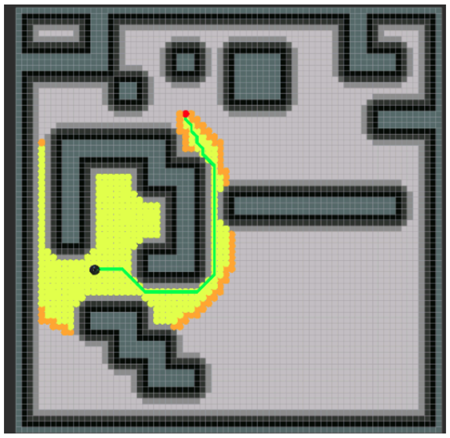


Figure 2: Small Obstacle and Large Concave Obstacle simulations. In the map visualizations above, the black dot is the current position of the drone and the red dot is the goal node. In addition, the green line is the path that the drone takes and the yellow dots are the nodes that were searched for. When comparing Greedy Best First Search (A, D) and Dijkstra's (B, E), while Greedy Best First Search searched for considerably less nodes, its route was not always the most optimal route in comparison to Dijkstra's (as shown in the concave obstacle map images). A-star (C, F) combines parts of both of these algorithms, finding the most optimal route with speed and preciseness, making it the best algorithm tested.

Table 1. Small Obstacle Map*

Algorithm	Nodes Searched	Search Time (Sec)	Travel Time (Sec)	Total Time (Sec)
Dijkstra's	1422	13.46	45.06	58.52
GBFS	49	0.02	45.40	45.42
A-star	347	4.11	45.04	49.15

*The table compares the three search algorithms based on the time searched for the path, the time it took for the drone to move, the total time, and the number of nodes that were searched with each path planning algorithm in the small obstacle map (see **Figure 2, A-C**).

Table 2. Large Concave Obstacle Map*

Algorithm	Nodes Searched	Search Time (Sec)	Travel Time (Sec)	Total Time (Sec)
Dijkstra's	1772	9.05	1:12.59	1:21.64
GBFS	199	1.94	1:42.84	1:44.78
A-star	397	2.78	1:10.51	1:13.30

*The table compares the three search algorithms based on the time searched for the path, the time it took for the drone to move, the total time, and the number of nodes that were searched with each path planning algorithm in the large concave obstacle map (see **Figure 2, D-F**).

As previously stated, it was found that, generally, A* is the most efficient out of the algorithms used, being not only fast but also accurate. A* resulted in around 70% less search time and 76% less nodes searched than with Dijkstra's algorithm, meaning that it conserved both time and storage space in comparison to Dijkstra's algorithm. In addition, with larger obstacles, A* had about 30% less travel time than GBFS, as it always found the most optimal route, in comparison to the GBFS algorithm which often resulted in longer and less optimal paths. While Dijkstra's algorithm can also find the most accurate path, without using the location of the goal node by using a heuristic, such a large search time and storage discrepancy can be seen. Furthermore, although GBFS can generally search and find a route faster than A*, and is often even better than A* in smaller, less cluttered environments, its sub-optimal route that is produced may often lead the drone to take an overall longer route in larger maps, explaining the difference in reliability between GBFS and A*. Thus, in environments tested, A* was both fast and accurate, constantly finding the correct path while allowing the drone the most time to film in an area and less time to get there. This is especially important in drone cinematography, since some acting or motion scenes cannot stop to wait for the drone to calculate its path and move inefficiently, and thus require the most rapid and accurate autonomous drone path planning.

Furthermore, as an outcome, the drone generally had smooth and steady camera visuals while flying throughout the environment due to minimal zig-zagged patterns in the route. Within environments which were not very large, A* often found the most optimal route rather quickly, often taking under 10 seconds to find the best path. However, in much larger and/or more cluttered environments, the path planning time took longer amounting to times which took over half a minute. If implemented in various real-world conditions where sometimes a large distance is needed, this may cause some drones to take even longer, amounting to a lot of computation being used and overall less drone battery life. This poor performance is likely due to the fact that A* always looks for the best path, meaning that it will recalculate every single path it takes and with larger maps, this will take a long time and a lot of memory. Other algorithms like D-star, try to fix these problems by using previous data instead of constant recalculation. Moreover, suboptimal but extremely fast algorithms like Rapidly Exploring Random Trees and Artificial Potential Fields sacrifice perfection for speed and memory conservation. Major details of such path planning algorithms in drones, however, were not tested in the project. Formal comparisons of these other approaches with mine are warranted. Realistically, however, cinematic drones will often use path planning in small to medium sized ranges where the user can see the drone. Thus, since many popular drones have limited controller range and in many cinematic cases do not need to travel far, A* will generally be one of the fastest and most useful algorithms that can be applied to path planning for drone cinematography.

In terms of quality, such drones were able to film without too much camera movement or noise, even when going at higher speeds and with larger ranges. However, this project was limited to a simulation and was not tested in real life. Therefore, it is important to research the camera movement in more realistic conditions with camera quality limits and varying weather. With such path planning algorithms, while camera quality and drone speed were not too much of an issue, the tradeoff between efficiency and range was very limited in this project, as the perfection of A* increases time substantially with larger maps. In addition, this project did not allow for the drone to autonomously choose which location to move to, but rather still required the user to input the goal and the start node. Thus, further research on autonomous decision making, especially with the use of computer vision, is necessary.

5.2 Mapping Known Territory

When using OSM for pre-mapped territory, the results showed an optimal path a drone can take along any road or path that was mapped in OSM (**Figure 3**). For situations where a map is known, roadmap-based path planning reduced the path finding time in comparison to the unknown mapped environments, as there were much less nodes searched so more distance was searched in less time. Furthermore, since there was no need for localization or mapping, the set up for the path planning was also much faster using OSM for roadmap-based path planning. However, the suboptimal routes of roadmap-based path planning increases the travel time so it may not always be the most effective in terms of efficiency, especially on maps that contain many turns and curves. Additionally, roadmap-based path planning is often ineffective in highly dynamic environments, as the maps on OSM do not update in real time and thus cannot detect constant changes in an environment. This challenge can be mitigated by using real-time sensors and dynamic path planning like D-star; however, these were not tested in the scope of this project. Still, roadmap-based path planning is often effective in reducing concerns with drones going into restricted and private areas. For example, in the case of the Berlin Zoo map tested below, it prevents concerns of disturbing animals in their habitats. Furthermore, it also helps prevent concerns with cluttered environments in areas off-mapped paths, while also allowing for personalization in drone cinematography. Often with drones, such path planning can allow for shooting videos along specific roads and pathways of things such as people and cars, especially if there are concerns with flying drones over certain unknown or dangerous areas, or when in cluttered environments like cities with very tall buildings. By using roadmaps, such drones cinematically can create appealing shots behind targets that may follow set paths, such as if someone were driving down a street. Thus, while roadmap-based path planning may not produce the most optimal path to get to the destination, it should be heavily considered for use in various drone cinematography and autonomous drone situations.

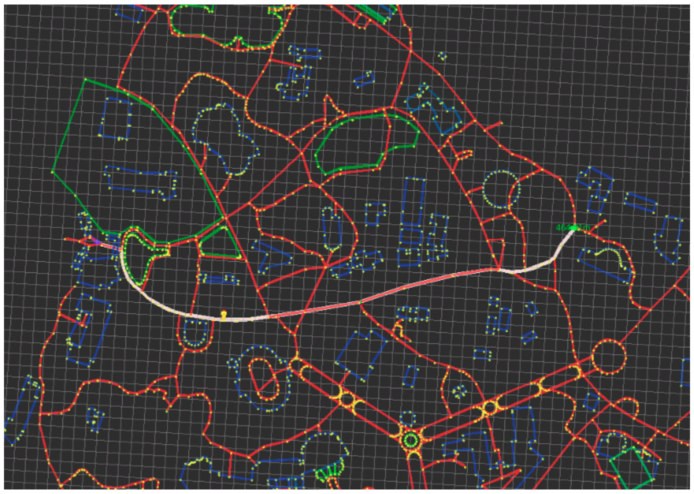


Figure 3: In the figure above, the white line represents the optimal path, the blue dot represents the start node, and the green dot represents the goal node. Next to the green dot is the total distance (in meters) of the path from the start node to the end node. The roadmap algorithm used was able to predict the most efficient path along a road where the drone can go to get from one point to another.

6. Conclusion

In conclusion, this cinematic drone project highlights the benefits of using various path planning algorithms in autonomous drones. The three main path planning algorithms, A*, GBFS, and Dijkstra's, all created a path from one location to another, and when tested with smaller maps, all took a reasonable amount of time, allowing the drone to autonomously move around the map. In larger maps, this path planning took much longer, but still all algorithms found a path to the location. However, in both cases, by combining a heuristic function, and the distance from the start node to the current node in order to find the lowest cost solution, A* was shown to outshine the other two algorithms, on average being more efficient than Dijkstra's and more accurate than the GBFS. In addition, the project also researched roadmap-based path planning which is greatly important when certain paths need to be followed in order for the drone to reach its destination. While much more research is necessary to refine and improve different path planning models, A* proves to be one of the most simple yet accurate and efficient models and has served as a common basis for many other search algorithms. By utilizing such path planning algorithms, drone flying in both recreational and professional filming will be easier and safer, allowing for an enhanced flying cinematic experience. Still, further research is necessary to incorporate path planning into all cinematic drones for various situations. First, it is important to continue researching the use of path planning for dynamic environments. In future works, I plan to integrate D* (D-Star), a search algorithm for dynamic environments, allowing drones to detect and adapt to new obstacles in real-time, ensuring more robust navigation in unpredictable settings. In drone cinematography, this will especially allow drones to account for the constantly changing sets in motion scenes, where there may be moving props or people. Furthermore, I also plan to try using multiple drones for path planning and autonomous movements to see how drones can be used together in cinematography without worrying about them colliding. This will also entail detecting the movement of dynamic objects (the other drones), where there is a constant need for path planning with very rapid response time and high accuracy. Lastly, further steps on using a physical drone with its path planning algorithms in real life will help validate the algorithms under real-world conditions. However, this may present new challenges, including hardware limitations that may reduce the available tools for path planning, as well as environmental conditions that could lead to signal interference. Still, the extensive research of AI and advanced path planning in drones will revolutionize the hobby and professional use of drones in cinematography, overall enhancing the field throughout the globe.

7. Acknowledgements

I am incredibly grateful to my mentor, Rima Rebei, for guiding me through the entirety of the project, editing this paper, and teaching me how to code using ROS. Furthermore, the Construct AI was also a great help in learning ROS as well creating the code and simulations for the drone movement. Lastly, thanks to Inspirit AI for giving resources to learn AI and create this project.

8. References

Bhattacharyya, S. (2023, September 11). *How Is Dijkstra's Algorithm Used In The Real World? | Analytics Steps*. https://www.analyticssteps.com/blogs/how-dijkstras-algorithm-used-real-world

Bonatti, R., Zhang, Y., Choudhury, S., Wang, W., & Scherer, S. (2018). *Autonomous drone cinematographer: Using artistic principles to create smooth, safe, occlusion-free trajectories for aerial filming*. https://arxiv.org/pdf/1808.09563

Daley, S. (2018). *Fighting fires and saving elephants: How 12 companies are using the AI drone to solve big problems.* Built In. https://builtin.com/artificial-intelligence/drones-ai-companies

Feiyu, Z., Dayan, L., Zhengxu, W., Jianlin, M., & Niya, W. (2024). Autonomous localized path planning algorithm for UAVs based on TD3 strategy. *Scientific Reports*, *14*(1), 763. https://doi.org/10.1038/s41598-024-51349-4

Gugan, G., & Haque, A. (2023). Path Planning for Autonomous Drones: Challenges and Future Directions. *Drones*, 7(3), 169. https://doi.org/10.3390/drones7030169

Karur, K., Sharma, N., Dharmatti, C., & Siegel, J. E. (2021). A Survey of Path Planning Algorithms for Mobile Robots. *Vehicles*, 3(3), 448–468. https://doi.org/10.3390/vehicles3030027

Kumar, N. (2023, September 3). Exploring Pros and Cons of Autonomous Drone Technology. Analytics Insight. https://www.analyticsinsight.net/drone/exploring-pros-and-cons-of-autonomous-drone-technology

Martijn. (2023, September 3). *Drones in Filmmaking: The New Age of Cinematography*. Drone Operator. https://www.drone-operator.com/drones-in-filmmaking-the-new-age-of-cinematography/

Nägeli, T., Meier, L., Domahidi, A., Alonso-Mora, J., & Hilliges, O. (2017). Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics*, 36(4), 1–10. https://doi.org/10.1145/3072959.3073712

Sabetghadam, B., Alcántara, A., Capitán, J., Cunha, R., Ollero, A., & Pascoal, A. (2019). *Optimal Trajectory Planning for Autonomous Drone Cinematography*. https://personal.us.es/jcapitan/preprint/sabetghadam_ecmr19_web.pdf

The Construct. (2024). Theconstruct.ai. https://app.theconstruct.ai/

Wayas, I. (2024, March 18). *Drones Are Becoming More Than Just Flying Cameras With AI | Cryptopolitan*. Cryptopolitan. https://www.cryptopolitan.com/drones-are-becoming-more-with-ai/