Optimized and Efficient Path Finding Algorithms: Bringing Robots To Restaurants

Akar (Ace) Kaung (kaung006@umn.edu)
Yanai Sun (sun00105@umn.edu)

Abstract

Both employees and customers have suffered from understaffing and overcrowding. Using robots to replace the shortage of workers can reduce the workload on the workers and satisfy customers' demands. However, to use a robot as a server, it needs to know a good path, and there are too many path-finding algorithms to implement. In this study, we assessed the performance of Dijkstra and A* path-finding algorithms in a simulated restaurant setting with all possible routes mapped by Probabilistic Roadmap (PRM) and Voronoi diagram algorithms. The effectiveness of each path-finding algorithm was then evaluated based on the time taken, the length of the path, and the number of collisions in the completed route. This paper concludes with the advantages and disadvantages of each path-finding and path-planning algorithm as if the algorithm was implemented in restaurant robots.

Introduction

The food service industry is the newest application of robots in the food industry. Applying robots in restaurants has the potential to reduce labor costs, improve service quality, and simplify operations (Seyitoğlu et al., 2021). All restaurants have well-defined operational procedures for storage, preparation, and serving customers. For example, when customers order food through the self-ordering kiosks, the order will notify the robots in the kitchen on what to prepare to cook; subsequently, the robot delivers the food to the customer. This repeated procedure with every customer makes it easy to create and program algorithms to make service automatic.

As a result of the coronavirus and associated social distancing rules, many restaurants experience a protracted labor shortage and raise awareness about safety and hygiene. Robots are thought of as one of the technologies to deliver physically remote service, making them the appropriate way to replicate contactless services in this scenario. For instance, there is a robot-served restaurant that was opened in Bengaluru, India, that mainly sells salads and risotto bowls. When customers walk in, they are welcomed by robots. Orders were also taken by robots, then they scooped out ingredients, poured them into pots, and cooked like a chef. Every process was completed by autonomous technology within three minutes (Cha, 2020). With robots replacing human workers, labor costs have been reduced and prices have decreased. As more restaurants see the benefits of using robots to serve their customers, there is a growing trend toward using robots in restaurants.

Overall, the evolution of national policy, economic benefits, social demands, and technological advancements may lead to the growth of restaurants where robots serve alongside humans. In this paper, we would like to find the best path-planning algorithm to serve food to customers in a collision-free and minimized traveled distance manner.

Problem Description

Customers have higher expectations of business services as technology advances, and this is true for the restaurant industry as well. Berezina and her team performed a study in 2019 and discovered consumers rated the highest convenience to see the robots as servers. They believe robots increase employee productivity and reduce the chance of error in a restaurant. To compete with other restaurants, owners have to fulfill customers' wants and needs. In other words, if restaurants want to use robots as their servers, their robots must match or exceed the capability of human servers, such as locating the desired spot without colliding with any objects and maintaining an optimized path to complete deliveries. In this paper, we are investigating existing research on path-planning and path-mapping algorithms of Dijkstra, A*, Probabilistic Roadmap (PRM), and a Voronoi diagram before implementing and evaluating our automated transportation.

Related Work

We want to identify the best path-planning algorithm that allows mobile robots to travel the shortest distance in a safe and collision-free path from the starting position to the target position. The Dijkstra algorithm is a classical shortest-path planning algorithm that considers the distance from the current node to the next node. Zhen Nie and Huailin Zhao researched robot path planning using a hybrid algorithm of Dijkstra and Ant Colony Optimization (ACO). They believe the greedy algorithm of Dijkstra will likely put the robot in a local optimum and make it prone to deadlock phenomena, so researchers combine Dijkstra and ant ACO algorithms. The Dijkstra algorithm was used for initial path planning and the ACO was used afterward for optimization. First, researchers built the environment model using visual graphs and used Dijkstra for initial path exploration. After initial path planning was completed, ACO was used to optimize the initial path by starting path searching and selecting the next node according to the current node information and the next node selection principle. The selection principle is that the probability of the next path selection is determined by pheromone concentration and heuristic information left on the path between ants (Dorigo, 2007). After the next node is selected, it updates the local pheromone on the path that the ant has traveled. The ant keeps searching for the optimal path and updates its global pheromone (Nie & Zhao, 2020). The hybrid algorithm was able to solve the fast search for the path of robots and get the shortest path. The research showed this hybrid algorithm can effectively avoid obstacles and achieve optimal path planning.

In 2019, Jungyun and Woojin experimented with an A* path planning method to minimize the travel costs for robots to complete pick and deliver job assignments. Their research emphasized resolving dispatching and routing problems involving multiple mobile robots with different loading conditions (Bae, 2014). It is difficult to dispatch multiple mobile robots to explore routes that are vertex-disjoint paths. It is expected that each target location should be visited at least once while the sum of the distances traveled by the robot is minimized. They believed that dissipation and routing were critical factors in producing a high-quality suboptimal solution for automated transportation. In the experiment, the researchers assumed each job had a distinctive set of robots to deal with routing problems. A path for each robot starts from its depot, handles a set of tasks in sequence, and terminates at the last delivery position (Bae & Chung, 2019). In their path-finding algorithm, all robots are initially set to zero traveling costs, and the edges are labeled with different costs. The heuristic function in A* sets a priority for the robots to choose a path that has a lower traveling cost. The traveling cost increased with every target

location the robot stopped by until it stopped at the termination. The result of their experiment showed A* is an effective algorithm to minimize robots' travel costs.

According to Bhattacharaya and Gavrilova's research in 2008, the Voronoi diagram is a well-known road map in the path-planning literature. Its edges offer a path with the greatest clearance between several discontinuous polygonal obstructions. Their research paper is about using the Voronoi Diagram for a clearance-based shortest path where there should be collision-free travel from the starting point to the destination. Even though the Voronoi diagram provides the greatest distance from the obstacle, it is far from optimal. It frequently contains unnecessary turns, and the path length may be excessively long at locations where the obstacles are far apart. Therefore, in another paper in 2007, they suggested removing unnecessary turns and reintroducing additional points to provide a more optimal and shorter path. In an attempt to change the path, the obstacle clearance should stay above a given value; otherwise, a collision can occur (Al-Dahhan, 2020). The minimum clearance of the obstacle was verified by Ben, Xue, Albert, and Frimpong in 2019 by using morphological dilation to inflate the obstacle region. In July 2020, Mohammed and Klaus tested out whether the method/algorithm computed via Al-Dahhan and Ben, Xue, Albert, and Frimpong's idea would compute the most clearance in a collision-free path as well as in the optimization area. It turns out that they compute a collision-free and more optimal path than the normal Voronoi algorithm.

In 1996, Kavraki and her research team invented a new motion planning algorithm which is known as a Probabilistic roadmap. This algorithm generates random nodes and constructs the graph using these generated nodes. The nodes correspond to collision-free configurations and their edges provide the optimal path between each node with collision-free configurations as well. However, in 2021, Khokhar created an environment where the obstacles are very close to each other and uses the PRM algorithm to generate the nodes. As a result, he found the algorithm to have some disadvantages. Since it generates the nodes randomly, the chances of generating one in between a small gap of two obstacles will be low. Thus, this led to a non-optimal graph.

After reviewing the existing research on Dijkstra, A*, PRM, and Voronoi graph algorithms and understanding their respective strengths and weaknesses, we will consider their approaches to innovate a new algorithm to find the best method to implement productive robots in restaurants.

Results and Insights

Results

In this study, we generated an environment or map of the virtual scene to simulate the actual layout of a real-life restaurant (See Fig 1.0). The green dot on the top left (x: 20, y: 180) represents the robot's starting position, and the green dot on the right bottom (x: 120, y: 30) represents the robot's goal position. The rectangles inside the environment represent the obstacles, such as tables, chairs, decorations, and more.

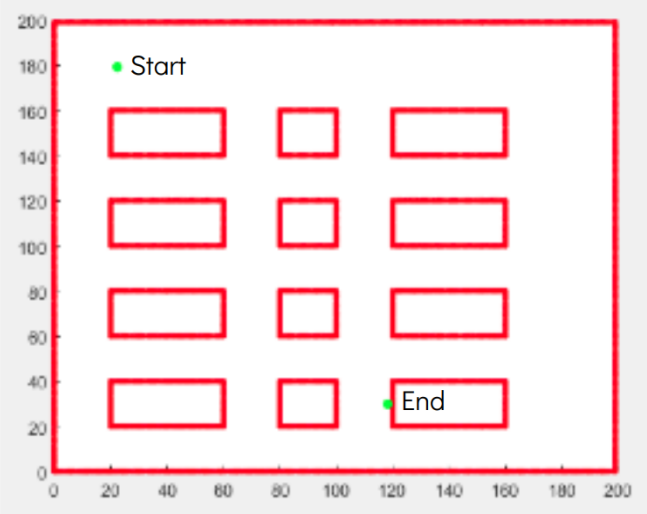


Fig 1.0 Virtual environment that simulates the actual layout of a real-life restaurant. The start and end points are labeled in green.

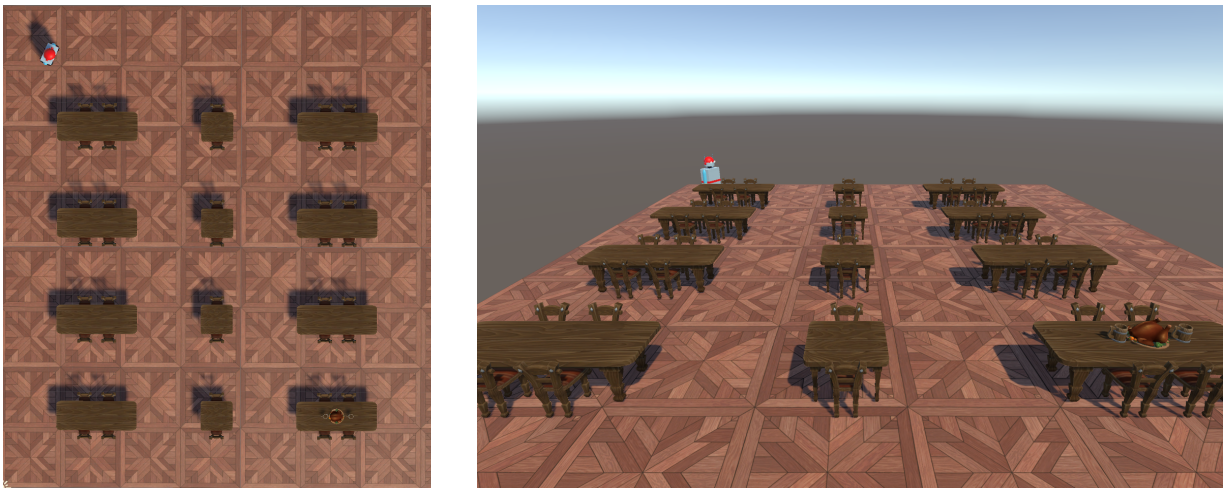
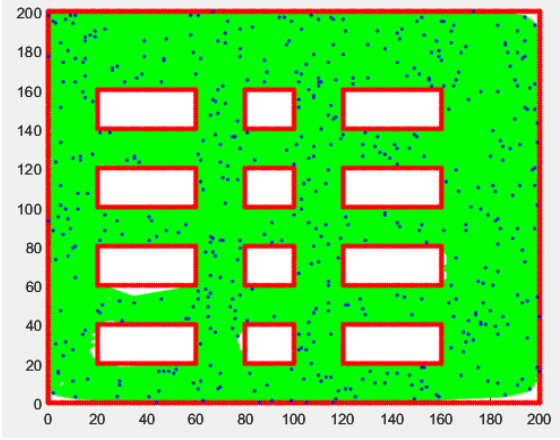


Fig 1.1 Animated and 3D environments simulate the actual layout of a real-life restaurant. Videos of the robot navigating from the start position to the destination using A* in the PRM and Voronoi graphs are shown in the supplementary materials.

PRM generated paths



Voronoi edges paths

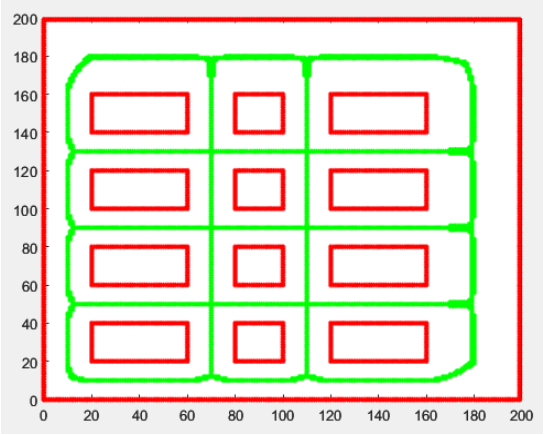
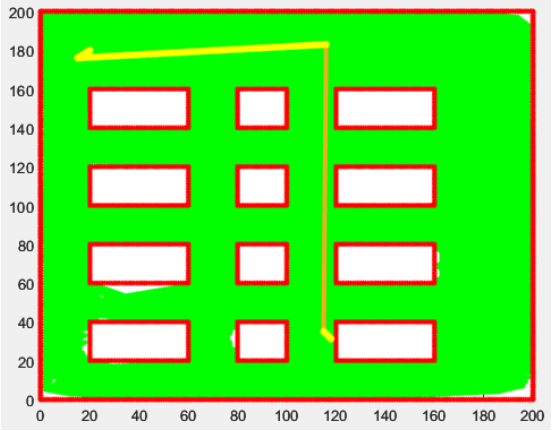


Fig 2.0 All possible routes generated by PRM And Voronoi

PRM generated paths + A*



A* using PRM paths

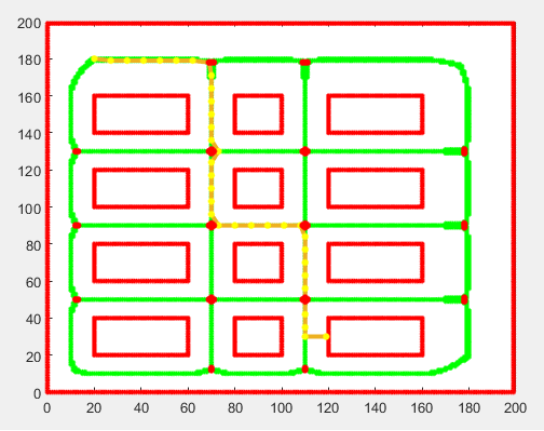
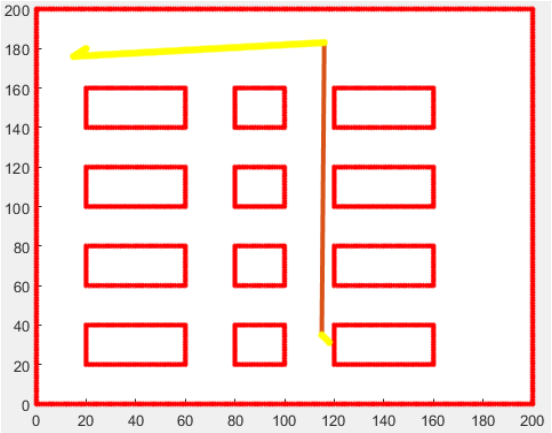


Fig 3.0 The shortest path selected from all possible routes in using A* on PRM and PRM

A* using PRM paths



A* using Voronoi edges paths

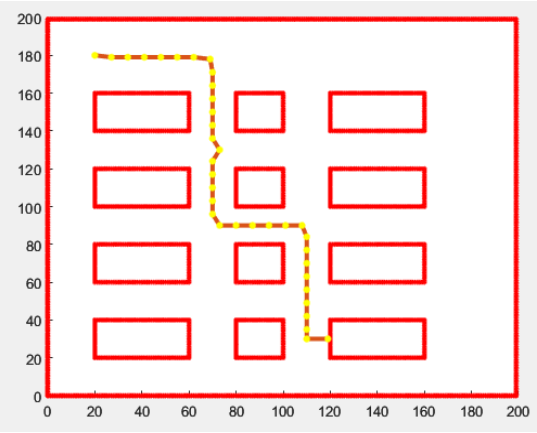


Fig 4.0 Clear view of A* path in PRM and Voronoi

By using the environment in Fig 1.0, we have generated all possible paths that are collision-free using PRM and Voronoi algorithms, shown in Figure 2.0. Then, we applied the A* path-finding algorithm to find the most optimal path to travel to the destination on generated routes, supported by extensive research comparing A* and Dijkstra (see the Insight section for more details) (Figure 3.0). Finally, we simplified the path further in Figure 4.0 to get a better view of the route discovered by A*.

Insight

The Voronoi algorithm is a path-finding algorithm that focuses on generating a path that can be traversed from one place to another. This algorithm cannot be compared with Dijkstra or A* since these two algorithms are used to compute the shortest path between two places. For our study, we will be using the Voronoi algorithm to generate a path, then using A* or Dijkstra to find the optimal path between two locations.

According to Dian and Lysander in 2019, both A* and Dijkstra algorithms have similar performance when using them to solve town or regional scale maps. However, A* is better when using the environment as a large-scale map. This claim is supported by the experiment conducted by Victor and Aron in 2016. Victor and Aron have measured the time taken, path length, and path nodes for both Dijkstra and A*algorithms with multiple maps. The experimental result showed that the A* algorithm computes the path by 1.8 times faster on average than the Dijkstra. The other performances such as the cost of the path are identical to each other. Thus, we can conclude that the A* algorithm computes better than the Dijkstra in terms of performance.

Our study result showed A* is the best path-finding algorithm. Hence, we have combined A* with the Voronoi and PRM algorithms. The Voronoi algorithm generates the edges that are the farthest away from the obstacles; the PRM generates the nodes randomly that are distant from impediments with a certain threshold. As shown in the first column of Figure 4.0, the Voronoi algorithm produced paths that were furthest away from the obstacles; the PRM algorithm generated routes that included both far and near the impediments. The second column shows the shortest path discovered using the A* algorithm. The A* used information about path cost and heuristics to find the minimized cost route. The final third column features the simplified path a robot should take according to the Voronoi or PRM algorithm.

Lesson Learnt

Personal Growth

Initially, we started with the desire to compare multiple algorithms; path-mapping (PRM, Voronoi, RRT, and Visibility Graphs); path-finding (A*, Dijkstra, DFS, and BFS). As time went by, we realized that we could not compare all of them as we don’t have sufficient time. So we had to reduce them one by one until it was down to two algorithms for each part. This alone has made us learn to see what could be done within the time frame that is given to us before proposing the idea. Despite working with limited algorithms, we have gained greater insights on each of them.

Technology

In this project, we introduce ourselves to the new platform and application which we have not used before. We learnt how to integrate our codings so that we can see it as the whole. We use C++ to program the path and produce the final path to take depending on the mapping algorithm as nodes. With the result nodes, we use Matlab to draw out the blueprint/layout along with the path (Fig 1.0, 2.0, 3.0, 4.0). With the same nodes, we pass it to Unity which uses C# to animate the scene. Learning to use the new platform was harder than expected, however, eventually we got a handle on it and because of that the result became better than before.

We use Matlab only to draw the lines from one node to another, thus it doesn't require much time to learn. However, learning Unity takes some time as it requires us to know some basic animation programming. With Unity, we learn how to animate the environment and make the entity (robot) move throughout the map. We animated the environment without actually drawing out the assets but by using free source assets which would fit with our use case. So finally, after combining everything that we learn, we finally achieve to animate our scene.

Algorithms

We have understood more in depth about the A* and Dijkstra algorithm after having extensive research for both of them. A* algorithm uses heuristic function to calculate the path additional to all of Dijkstra's algorithm. Thus, if the heuristic function of the A* is 0 for all the nodes, it will become the same as Dijkstra. Since A* uses heuristic function, it makes calculating the path faster. According to Robinson and Lukic, they stated that A* calculate 60 times faster in their computation case compared to Dijkstra. However, they also mentioned that the efficiency of A* algorithm is highly dependent on its heuristic function.

As for the Voronoi and PRM, both of them are different unlike A* and Dijkstra. PRM generates nodes randomly on the map while the Voronoi place the nodes furthest away from the obstacles. Since PRM generates the nodes randomly, it is not guaranteed to have the best available path for every attempt unless there are multiple hundreds of nodes depending on the map. However, if there are too many nodes, it makes the A* or Dijkstra longer to calculate the path. The disadvantage of Voronoi on the other hand is that it does provide the farthest node with the obstacle, but if the obstacles are too close to each other, it will still generate the path without actually considering the entity's diameter which actually might collide with the obstacle. From here, we know that Voronoi cannot be used if the obstacles were too close to each other.

Future Work

The research objective was to identify the best path-finding algorithm to implement in robots to serve alongside human workers in restaurants. We have developed a simplified scenario of robots delivering food from the starting position to the end position using the A* path-finding algorithm. A* is a global navigation system that analyzes the complete map of the environment and generates a path to reach the goal position. However, this would not apply to unstructured environments that change over time. In real-life situations, restaurants are often occupied by workers and customers walking around. We expect that when the robot encounters other moving objects, it must slow down, stop, or detour around them. Hence, the extension of this project would be implementing moving objects in the simulated environment that play the roles of

workers and customers and programming the robots to plan navigation in the presence of moving objects. Beyond using A* to find the shortest path, we need to implement reactive collision avoidance algorithms: the Artificial Potential Field (APF) and Dynamic Window Approach (DWA).

The APF algorithm calculates the distance between the starting and ending positions, as well as the locations of any obstacles in between, and then uses a position-related potential function to determine the robots' direction and speed to avoid obstacles (Hong & Arshad, 2015). APF, on the other hand, is prone to becoming trapped in local minima situations and passing through narrow passageways.Combining APF and DWA algorithms will provide real-time path planning to improve the robot's ability to avoid obstacles. DWA is a local navigation algorithm that ensures a collision-free path to the destination while taking velocity, a dynamic model of robots, optimized distances to obstacles, and the total cost of reaching the goal positions into account (Dobrevski & Skocaj, 2020). Both APF and DWA algorithms will help robots take action based on the sensor readings of the robot's surroundings instead of relying on the map of the environment. A similar study was conducted by Wenlin and his research team to evaluate the performance of amphibious robots navigating through an unknown environment with obstacles. Robot fish had to reach the destination with information from their surroundings; the result verified the efficiency, instantaneity, and reliability of the combined algorithms (Yang et al., 2021). Therefore, implementing APF and DWA would be a viable approach to improving the robots for our future work so they can recognize and avoid unexpected obstacles while finding the quickest and safest path to their destination.

Along with the plan to implement APF and DWA, we would like to extend our analysis of robots' performance through their computational time and speed of service. Implementing technological inventions in the service industry increases individuals' expectations for the quality of service and experience satisfaction. As of 2021, Vatan and Dogan discovered that implementing robots to work alongside human workers can increase the efficient distribution of tasks, reduce services, and enhance the positive experience for customers in restaurants. Therefore, we are hoping to achieve the efficiency of robots by evaluating the computation time to find the shortest path with all possible routes in a restaurant and the speed of service for the robots to reach from the start position to the end position.

Problem Encountered

Algorithms

There are multiple path mapping and planning algorithms available for us to use. However, we could not compare every single one of them as time did not permit us to do so. Thus, we had to decide which algorithm would be most suitable for our use case as well as the level of familiarity for us. Once we have decided on the most suitable algorithms, we have to implement them and this alone would take some time. So we approach in a way that if we already have the algorithm which was previously implemented and source code available to us, we take the source code and modify it so that it would fulfill our needs.

For the PRM and A*, we already have the source code which we had implemented before, so we decided to remodify a bit and use it as we need. However, the Voronoi source code isn’t available to us, so we had to look around the open source of it and found one available

source. But it was written in Spanish for most of the keywords in the code, so we were not able to fully utilize it until we translated it back into English.

Environments

To fully compare the algorithms for the use of restaurants and to arrive at the most optimized and efficient solution, we would need to create multiple virtual environments because not every restaurant's layout is the same. However, creating multiple environments would require multiple blueprints and layouts, and implementing a single environment alone takes a while. Due to the time constraints of balancing classes and working on this project, we decided to use the most common layout of restaurants that we could think of for our findings.

Conclusion

With the goal of improving the use of robots to reduce human workers' workload and increase customer satisfaction, we evaluated the performance of Dijkstra and A* path-finding algorithms in a simulated restaurant setting with all possible routes mapped by Probabilistic Roadmap (PRM) and Voronoi diagram algorithms. Through an evaluation of the time taken, the length of the path, and the number of collisions in the completed route, we discovered that A* with Voronoi is the best combination of path-finding and path-mapping algorithms for finding the best path for robots to travel in a restaurant. The limitation of our project is time. There are multiple path-finding algorithms and different restaurant layouts that we did not have time to explore and test our code on. We also neglected to include moving objects in our virtual restaurants to simulate the human workers and customers walking around in a real setting. Hence, the extension of work is to implement moving objects in the virtual environment and test the artificial potential field and dynamic window approach algorithms to program robots to avoid moving objects. In addition to our previous criteria for evaluating robots, we would also want to assess their computational time and speed of service. Overall, the current findings demonstrate that A* and Voronoi are effective methods for determining the shortest path for robots to take from their starting point to their destination.

Supplementary Materials

https://youtu.be/v7BP1RZ8xQs
https://youtu.be/86w2uoMARGI

References

Al-Dahhan, M. R. H., & Schmidt, K. W. (2020). Voronoi Boundary Visibility for Efficient Path Planning. *IEEE Access*, *8*, 134764–134781. https://doi.org/10.1109/ACCESS.2020.3010819

Ayawli, B. B. K., Mei, X., Shen, M., Appiah, A. Y., & Kyeremeh, F. (2019). Optimized RRT-A* Path Planning Method for Mobile Robots in Partially Known Environment. *Information Technology and Control*, *48*(2), 179–194. https://doi.org/10.5755/j01.itc.48.2.21390

Bae, J. (2014). *ALGORITHMS FOR MULTIPLE VEHICLE ROUTING PROBLEMS A Dissertation CORE View metadata, citation and similar papers at core.ac.uk provided by Texas A&M Repository*. https://core.ac.uk/download/pdf/79648558.pdf

Bae, J., & Chung, W. (2019). Efficient path planning for multiple transportation robots under various loading conditions. *International Journal of Advanced Robotic Systems*, *16*(2), 172988141983511. https://doi.org/10.1177/1729881419835110

Berezina, K., Ciftci, O. and Cobanoglu, C. (2019), "Robots, Artificial Intelligence, and Service Automation in Restaurants", Ivanov, S. and Webster, C. (Ed.) Robots, Artificial Intelligence, and Service Automation in Travel, Tourism and Hospitality, Emerald Publishing Limited, Bingley, pp. 185-219. https://doi.org/10.1108/978-1-78756-687-320191010

Bhattacharya, P., & Gavrilova, M. L. (2007). Geometric algorithms for clearance based optimal path computation. *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems - GIS '07*. https://doi.org/10.1145/1341012.1341064

Bhattacharya, P., & Gavrilova, M. L. (2008, June 10). *Roadmap-Based Path Planning - Using the Voronoi Diagram for a Clearance-Based Shortest Path*. IEEEXplore. https://ieeexplore-ieee-org.ezp2.lib.umn.edu/document/4539723

Blaer, P. (n.d.). *Robot Path Planning Using Generalized Voronoi Diagrams.* Www.cs.columbia.edu. Retrieved April 15, 2022, from https://www.cs.columbia.edu/~pblaer/projects/path_planner/

Cha, S. S. (2020). Customers' intention to use robot-serviced restaurants in Korea: relationship of coolness and MCI factors. International Journal of Contemporary Hospitality Management, 32(9), 2947–2968. https://doi.org/10.1108/ijchm-01-2020-0046

Dobrevski, M., & Skocaj, D. (2020). Adaptive Dynamic Window Approach for Local Navigation. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/iros45743.2020.9340927

Dorigo, M. (2007). Ant colony optimization. *Scholarpedia*, *2*(3), 1461. https://doi.org/10.4249/scholarpedia.1461

GitHub - akarhtutkaung/Voronoi-Path-Planning. (2022, November). GitHub. https://github.com/akarhtutkaung/Voronoi-Path-Planning

GitHub - akarhtutkaung/PRM-Path-Planning. (2022, November). GitHub. https://github.com/akarhtutkaung/PRM-Path-Planning

Hong, M. J., & Arshad, M. (2015). A Balance-Artificial Potential Field Method for Autonomous Surface Vessel Navigation in Unstructured Riverine Environment. Procedia Computer Science, 76, 198–202. https://doi.org/10.1016/j.procs.2015.12.341

Iswanto, I., Ma'arif, A., Wahyunggoro, O., & Imam, A. (2019). Artificial Potential Field Algorithm Implementation for Quadrotor Path Planning. International Journal of Advanced Computer Science and Applications, 10(8). https://doi.org/10.14569/ijacsa.2019.0100876

Kavraki, L. E., Svestka, P., Latombe, J.-C. ., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580. https://doi.org/10.1109/70.508439

Khokhar, A. (2021, February 12). *Probabilistic Roadmap (PRM) for Path Planning in Robotics*. ACM JUIT. https://medium.com/acm-juit/probabilistic-roadmap-prm-for-path-planning-in-robotics-d4f4b69475ea#:~:text=Disadvantages%20of%20PRM&text=Assume%20the%20gap%20between%20two

LaValle, S. M. (2006). *Planning Algorithms* (Illustrated). Cambridge University Press.

Martell, V. and Sandberg, A. (1970) *Performance evaluation of A* algorithms: Semantic scholar*, *Semantic Scholar*. Available at: https://www.semanticscholar.org/paper/Performance-Evaluation-of-A*-Algorithms-Martell-Sandberg/a229ff63a8fa8304e062fe08b0c65282d40d6894

Nie, Z., & Zhao, H. (2020). Research on Robot Path Planning Based on Dijkstra and Ant Colony Optimization. *Journal of Information and Communication Engineering*, *6*(1), 321–327. https://www.ascspublications.org/wp-content/uploads/woocommerce_uploads/2020/06/JICE_Vol.6_Zhen-Nie.pdf

Rachmawati, D., & Gustin, L. (2020). Analysis of Dijkstra's Algorithm and A* Algorithm in Shortest Path Problem. Journal of Physics: Conference Series, 1566(1), 012061. https://doi.org/10.1088/1742-6596/1566/1/012061

Robinson, S., & Lukic, M. (n.d.). Graphs in python - theory and implementation. Stack Abuse. Retrieved December 8, 2022, from https://stackabuse.com/courses/graphs-in-python-theory-and-implementation/lessons/dijkstras-algorithm-vs-a-algorithm/

Seyitoğlu, F., Ivanov, S., Atsız, O., & Çifçi, B. (2021). Robots as restaurant employees - A double-barrelled detective story. Technology in Society, 67, 101779. https://doi.org/10.1016/j.techsoc.2021.101779

Vatan, A., & Dogan, S. (2021). What do hotel employees think about service robots? A qualitative study in Turkey. Tourism Management Perspectives, 37, 100775. https://doi.org/10.1016/j.tmp.2020.100775

Yang, W.; Wu, P.; Zhou, X.; Lv, H.; Liu, X.; Zhang, G.; Hou, Z.; Wang, W. Improved Artificial Potential Field and Dynamic Window Method for Amphibious Robot Fish Path Planning. Appl. Sci. 2021, 11, 2114. https://doi.org/10.3390/app11052114