Aiding Emergency Evacuations Using Obstacle-Aware Path Clearing

Mollik Nayyar[1] and Alan R. Wagner[2]

Abstract— **We seek to develop robots capable of helping people evacuate. Some evacuation environments, however, may have obstacles blocking the person's path to the closest exit. This paper therefore explores the possibility of creating robots that detect and move obstacles in order to open evacuation pathways. Experiments were conducted using a simulated autonomous robot in photorealistic indoor environments. The system uses computer vision algorithms to gather information from the environment. We show that the gathered information can be used to decide whether an obstacle can or needs to be moved in order to open a new evacuation pathway. We then use simple push manipulations to successfully remove obstacles from a path. We show that the system decreases evacuation time for evacuees in simulations of indoor environments.**

I. INTRODUCTION

We envision autonomous robots employed as an active emergency response system, acting instantaneously in the event of an emergency to support guided evacuation, information gathering and giving valuable information to first responders without direct human operator oversight. However, several challenges need to be met before a robot can be effectively deployed for this purpose and many of these are active areas of research such as localization and navigation, manipulation, Human-Robot interaction (HRI) etc. We believe that a robot first responder must have three essential capabilities, it must be able to: 1) gather information from the environment to inform future plans and actions; 2) interact with the evacuees to influence their evacuation behavior and 3) interact with the environment to improve evacuation speed.

Prior research in robot-assisted emergency evacuations has focused on HRI related problems ranging from human compliance [1] to trust repair [2]. However, the evacuation environment may pose additional challenges to effective evacuations even if the evacuee complies with the robot's instructions. Some situations may require the robot to manipulate the environment to open pathways for efficient evacuation. As an example, consider a situation where the robot encounters something blocking the path to an exit. Evacuees in such a situation may be stuck behind an object that needs to be removed. A robot capable of autonomously navigating and moving obstacles could allow for quicker evacuations, shorter search operations and guide crowds from a busy exit. We hypothesize that outfitting the robot with the ability to move obstacles will save more lives and provide invaluable information to emergency response professionals saving time during relief operations.

To enable the robot to move obstacles, two skills are essential. First, the robot must be able to identify a target blockage and determine whether or not it is movable and second, it must be capable of moving the blockage to a more desirable location. Here the movability of an object is defined as a true or false value indicating whether or not the object can be displaced by the robot by force. In this study, we limit our actions to non-prehensile push actions and we assume that the robot is provided with some knowledge of the movability properties of common indoor objects such as tables, chairs etc. Non-prehensile push action is defined as the act of applying force on an object, away from the robot, without grasping the object.

This paper presents a preliminary study of how a robot can aid evacuation efforts by using a basic manipulation strategy. We introduce the concept of obstacle awareness where the robot is able to use sensors to identify objects and determine some of the key properties that influence movability. This paper therefore investigates if and how much a robot capable of moving evacuation blockages increases the number of people evacuated. We examine this problem using high-fidelity simulations of evacuations in different environments.

The following section II discusses some the related works and in section III we present the experimental setup. The methodology is then discussed in section IV and further details of the environments used are provided in section V. The experiments and results are discussed in the section VI. Finally we conclude in section VII with a brief outlook of the future directions of the work.

II. RELATED WORK

The related work focuses on three areas central to this research: emergency robotics, non-prehensile manipulation and navigation among movable objects.

A. Emergency Robotics

Research has shown that robots can play a significant role during emergencies and help guide evacuees to exits [3]. Shell and Matarić presented an audio beacon based evacuation algorithm in a pedestrian simulation [4]. They found that even a small number of audio beacons were effective in decreasing the pedestrian egress time. Nayyar and

*This material is based upon work supported by the National Science Foundation under Grant No. CNS-1830390. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation

[1]Mollik Nayyar is a Ph.D student in the Aerospace Engineering, The Pennsylvania State University, University Park, PA 16802, USA mxn244@psu.edu

[2]Alan R. Wagner is an Assistant Professor with the Department of Aerospace Engineering and a Rock Ethics Institute Researcher, The Pennsylvania State University, University Park, PA 16802, USA alan.r.wagner@psu.edu

Wagner studied the impact of robot guidance strategies on human compliance with a robot's instructions [5]. Simulated robot-assisted crowd evacuations have been used as a means of validating crowd simulation models [6], reducing crowd evacuation times [7] and providing the best exit option to pedestrians [8]. The role of trust in HRI during emergencies has also been extensively explored in [9], [10], [11]. Robinette et. al. implemented an information propagation model and found that if only 30% of the evacuees crowding an exit followed the robot survival rates could improve significantly [12]. Prior work has not, however, considered how a robot might manipulate the environment to improve evacuation times. We believe that a robot capable of pushing objects away from blocked paths could help reduce evacuation times. To the best of our knowledge, this is the first examination of how an autonomous robot could manipulate the environment to improve evacuation times during an emergency.

B. Non-Prehensile Manipulations

Object manipulation is challenging because the robot must manage the uncertainty caused by the manipulation and the dynamics of the environment [13]. Non-prehensile manipulations aim to affect a change in an object's world state without grasping the object of interest. Pushing is an essential action primitive in robotics [14] and even in the absence of a dedicated manipulator, simple pushing actions can support object delivery or path clearing operations [13]. Researchers have found that developing analytical models of object behavior while the object is being manipulated can be very cumbersome [15], [16]. As a result, researchers have focused on data-driven models which have been shown to outperform traditional analytical methods [17], [18]. The lack of available data, however, is a significant limiting factor for a variety of manipulation actions and their resultant environmental interactions. Hence, researchers have recently focused on online, goal-driven approaches that combine control inputs and manipulation actions in a feedback loop allowing future actions to be performed based on the real-time effect of previous actions [13] (see [14] for more information).

C. Navigation Among Movable Objects

Navigation among movable objects (NAMO) is defined as the problem of navigating a cluttered environment by manipulating and moving obstacles. Wilfong showed that the problem is NP-hard when the manipulated objects do not have a pre-defined final configuration (pose and orientation), as a part of the solution. However, multiple methods have been proposed that reduce the complexity by utilizing heuristics [19], artificial constraints [20] and probabilistic approaches [21], [22]. Others have also solved simplified versions of the NAMO problem with incomplete information [23], [24], [25]. More recently, an offline, recursive approach has been presented that significantly reduces the number of object movements by using concepts from computational geometry [26]. This paper considers a version of the NAMO problem, but does not focus on optimality, completeness or perception and control uncertainties. Our objective, rather, is to develop a system capable of aiding evacuations via push manipulations, therefore our proposed measure of success is the decrease in evacuation time or increase in the number of evacuees reaching safety.

III. Experimental Setup

A. Robot

For this research, a 4-wheeled mobile platform equipped with a forward facing camera, a forward facing Lidar and a front bumper for push actions, was utilized. The path planning, object detection and laser scanning were performed in real time to update an occupancy grid based map as a representation of the world. The ground truth poses of objects were assumed to be known after detection. The robot was also assumed to have a database of movable objects where a 'movable object' is defined as an object that the robot can successfully displace by applying a force. The robot was provided with a database that maps object labels to their movability value. Each grid point on the map was marked as occupied, free or occupied by a movable object. The robot was provided with a floor plan of the environment with information about the permanent elements of the environment such as walls, rooms or corridors. The combination of object detection and database of movable objects gives the robot the capability to determine which objects can or need to be moved. We do not consider the issues associated with localization or low-level control in this paper, the simulation environment provides localization data to the robot.

B. Environment

To demonstrate the feasibility of the algorithm in indoor environments, three different environments were designed using the Unity game engine. We created an office, a hospital and a classroom for empirical testing. Each environment had different configurations and layouts. The environments were populated with objects relevant to the environment. We did not assume that the objects in the environment had a particular shape or size. The experiments were designed to study the impact of the robot's ability to clear obstacles from evacuation pathways on the evacuation time of simulated human evacuees. Obstructions were placed at the entry or corridor intersections and the environment was designed such that unblocking the paths would result in a shorter path to exit.

IV. Methodology

The robot-aided path clearing problem was setup as a basic path planning problem with incomplete information. The robot gathered information from the environment and planned a path to the specified goal based on the updated information. The robot was free to move around the environment and could push an object as many times as the algorithm deems necessary. A global planner was responsible for keeping track of the main task (reaching the goal location) and while local re-planning was used to identify sub-tasks

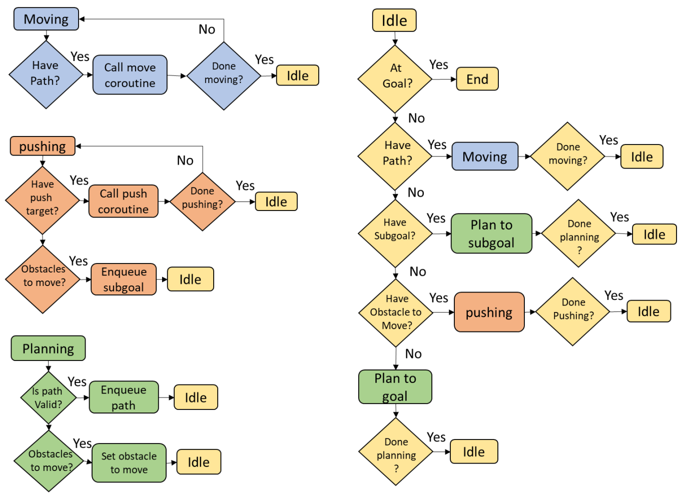

Fig. 1. The figure shows a high level description of the action states that the global planner executes. This flow chart details the events in each of the primitive states. The logic and transitions between the states is controlled by the global planner. The states that idle is switches to are shown on the left side. The Lidar scan runs concurrent to all other states and will allow a state change to 'Detect' in case of any event. We have omitted the 'Detect' state for simplicity .

such as approaching an identified movable object, determining push direction, starting and stopping push etc. The task planner delegated tasks to each of the modules based on the current state of the system and information received from the environment.

A. *Path Planner*

A grid search based path planning algorithm called A* with a diagonal distance heuristic was used to generate candidate paths for this paper [27]. Due to the map grid being finite, A* has the property of being complete i.e. it will return a solution if a solution exists [27], we exploit this property to determine whether a path to the goal exists. A* is solved on the initial map without any information about the obstacles in the environment. As the robot moves through the environment and more previously unseen obstacles are detected, the map is updated and the planner automatically takes into account the objects that need to avoided or moved for a cost effective path. A* allows paths to be planned 'through' the movable objects, which allows our system to calculate the displacement required to move an object to unblock a path.

B. *Obstacle Detection*

At the time of running these experiments, Unity support of object detection models was limited. Hence, we used a MobileNetSSD object detection model trained on the VOC0712 dataset for obstacle detection due to its fast inference speed at the cost of detection accuracy. The model is capable of detecting 20 basic classes of items available in the VOC0712 dataset. As the robot approaches an unknown object, the Lidar sensor triggers the camera to run the detection algorithm. The detection algorithm provides both a detection label and a bounding box of the object in the camera image. The robot is provided with a database that maps object labels to their movability value. This label is then used to determine movability and the object is updated on the map accordingly. If the object detector fails, the Lidar is used to mark that area on the map as immovable and the robot will not try to manipulate it for the remainder of the experiment.

C. *Mapping*

The robot generates an initial 2D representation of the environment based on knowledge of the floor plan. The robot's internal map used for path planning and recording the positions of obstacles is a discretized version of the floor plan. Each wall or obstacle fills the grid location it occupies. As the robot encounters obstacles in the world, its map is updated and a new global path is generated.

D. *State Machine*

The state machine breaks down robot behaviors into low-level action states. The global planner uses these states to describe high-level tasks for the robot. Here we define behavior as the immediate action being performed by the robot where as a task comprises of a set of actions combined together to achieve a goal. The primitive action states implemented in the state machine are planning, moving, pushing, detection and idle.

Each 'task' can be regarded as a sequence of primitive state operations performed in an order. As an example, 'moving' is a behavior during which the robot moves along a path, where as reaching a location is a task that requires planning from a start location to a goal location and then successfully navigating the environment while avoiding obstacles. The task of navigating to the goal is detailed in the figure 2 for clarity. As can be seen from the flow chart, the robot switches between different states such as planning, moving, idle and detection to perform a high-level task of 'navigate to goal'.

The global planner is responsible for keeping track of the main task (final goal location), the subtasks (generated on the fly) and controlling the transition between the states. A detailed description of the processes in each state are as follows:

1) Planning: The system starts the path planning routine by requesting a path from the current location to either the final goal or a subgoal. The planner performs a simple A* search on the map generating a path if it exists. Once the planner returns a path, the path is first checked for validity to determine if it has any intersecting objects. If the path is determined to be valid, it is added to a list of paths to follow and the state is switched to 'idle'. If the returned path is deemed invalid due to potential obstacle collisions, then the obstacles in the path are identified and added to the map and the state is switched to 'idle'.

2) Idle: The idle state defines the state between the transitions and controls the main flow of events. It is responsible for setting the next state based on the current information about the world or the previous task. The next state to be set is dependent on the availability of a path, subgoal,

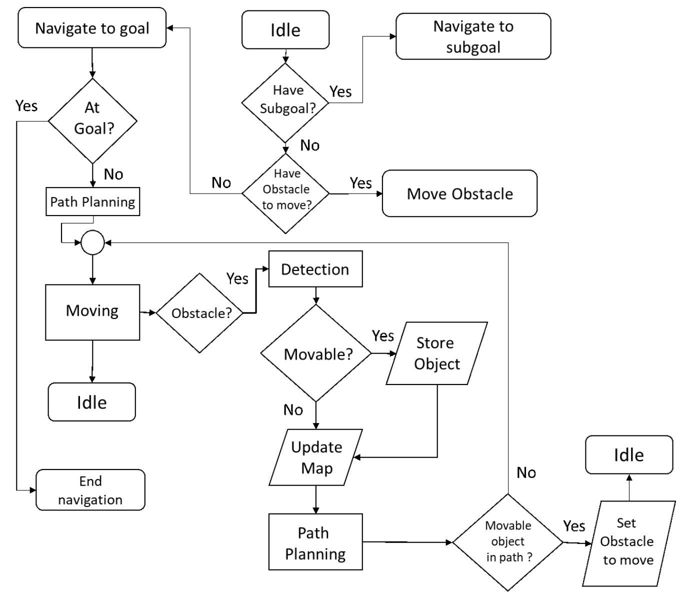

Fig. 2. The figure shows a high level description of the tasks that the global planner executes. As can be seen, the system switches between many states to complete a task. This flow chart details the events of 'navigate to goal' task. The logic and transitions between the states is controlled by the global planner. Note that 'navigate to goal', 'navigate to subgoal' and 'move obstacle' are tasks that comprise of their own flow of events.

obstacle to be pushed etc. The 'idle' state prioritizes available information. If an object has already been identified to be moved, the state is set to 'pushing', if a valid path is already available, the state is switched to moving or if a subgoal has been identified, the state is switched to planning to generate a path, and if none of the previous cases exist, the state is set to planning to find a path to the main goal.

3) Moving: The 'Moving' state is triggered by the 'Idle' transition state when a valid path is available to be traversed. The robot starts to move along the found path until either the final point on the path is achieved or an event is triggered by the front facing Lidar sensor, in which case the the state is set to 'Detect'.

4) Pushing: When the planner returns an invalid path, it checks to determine if the path is blocked by a movable object, in which case, the object is set as the object to be moved. Once the state switches to 'Pushing', first, a subgoal point near the obstacle is calculated based on the region around the obstacle on the map. The subgoal will require a path to be planned. If the returned path is valid, the robot will move to the location of the subgoal and initiate the push action on the obstacle. The distance to be pushed is tentatively calculated from the map. Once pushed, the internal map is updated and the state is set to 'idle' which will again request a path to the main target (assuming no other valid paths or subgoals exist).

5) Detect: The robot constantly checks if there is an 'unobserved' (not on its map) obstacle within its Lidar range. If an obstacle is in its path, a tentative 'unknown obstacle' with undetermined movability parameter is instantiated on the map, all other routines are stopped and the state is changed to 'Detect'. The system runs a object detection routine to determine the obstacle's movability. Figure 3 depicts an example of a successful detection. Note: not all movable objects become a target to be pushed. The planning state then checks if the a movable object is intersecting the current path before marking an object to be moved. The map is then updated according to the classification output from the object detector. If the object is classified as immovable, it is marked as immovable on the map and the next time the planning routine is run, the object is automatically avoided.

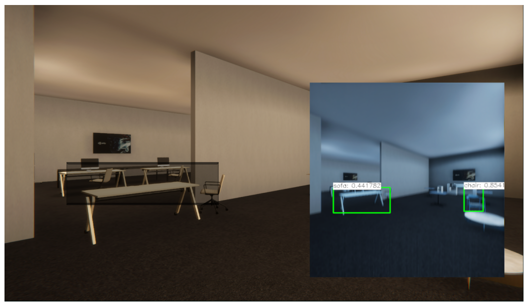

Fig. 3. The figure shows a first person view from the robot's camera perspective and the object classifications. As can be seen, the robot miss classifies a table as a sofa. The green box is the estimated bounding box of the object in pixel values.

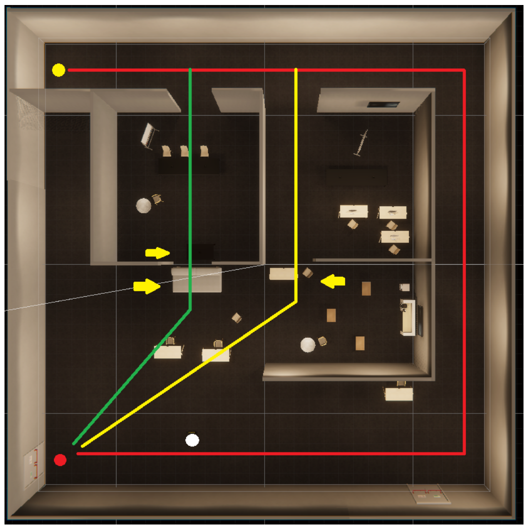

Fig. 4. The figure shows an overhead view of our office environment. The yellow dot indicates the human evacuees spawn point. The white dot indicates the robot start position. The red dot indicates the exit point. The red line is a longer path taken by the evacuees to the exit and was always unobstructed whereas the green line represents a shorter path taken by the evacuees once the blocking obstacle was removed. The yellow line is an intermediate path taken by the evacuees after the successful removal of the obstacle. The yellow arrow shows the obstacles to be moved.

V. Environments

The following three environments were developed for this research:

1) Office: An office environment was designed with two rooms, a long corridor, and a variety of obstacles. The rooms were obstructed which, when removed, could offer shorter paths to the exit as compared to the longer path along

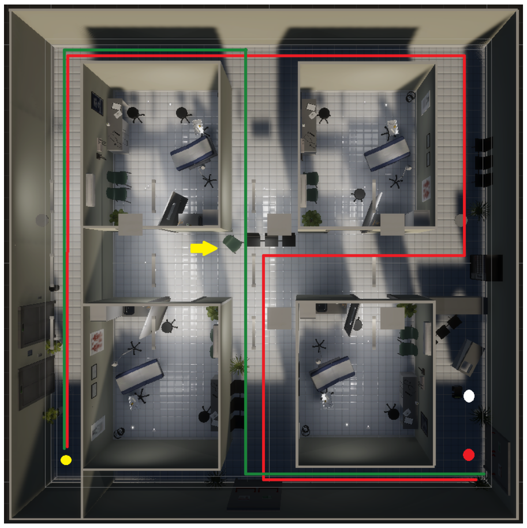

Fig. 5. The figure shows the overhead view of the hospital environment. The yellow dot indicates the evacuee's spawn point. The white dot indicates the robot start position. The red dot indicates the exit point. The red line is the longer path taken by the evacuees to the exit whereas the green line represents the shorter path taken by the evacuees after the robot removes the blocking obstacle. The yellow arrow shows the obstacles to be moved.

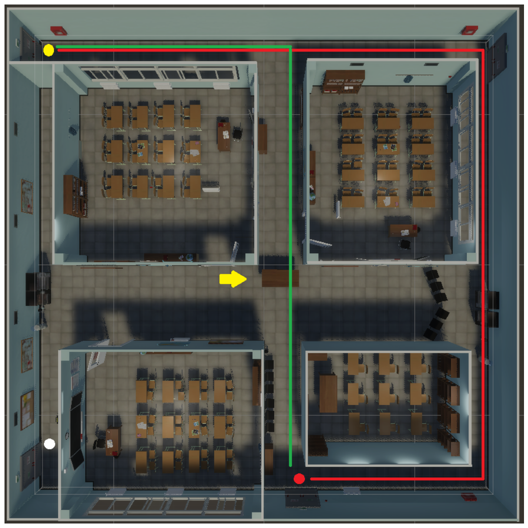

Fig. 6. The figure shows the overhead view of the school environment. The yellow dot indicates the human evacuees spawn point. The white dot indicates the robot start position. The red dot indicates the exit point. The red line is the longer path taken by the human evacuees to the exit whereas the green line represents the shorter path taken by the human evacuees after successful robot manipulation. The yellow arrow shows the obstacles to be moved.

the corridor (see figure 4). The environment was designed with only one exit, one unblocked path and two blocked paths, the intermediate path (shown in yellow) and then the shortest path (green line). The robot was tasked to remove the obstructions from both the blocked paths. This was done to observe the effect of each of the two paths on evacuation time. The evacuees spawn at the yellow dot at a frequency of one per second. They move towards the red dot taking shortest available path. By default, only the longer path (red line in figure 4) is open and the evacuees start moving towards the exit. As a shorter paths become available, the evacuees automatically change direction to move toward the exit via the shorter path. The robot starts at the location as shown by the white dot.

2) Hospital: The hospital environment contains four rooms with a long corridor around the rooms. The evacuees are spawned at the yellow dot as shown in figure 5 and move along the unblocked red (longer) path. The robot (shown as white dot) begins next to close to the exit point (shown as red dot). The environment has narrower corridors and unique obstacles and layout when compared to the office environment. The robot is tasked to remove the obstacle, marked by a yellow arrow in figure 5, from the the green path.

3) School: As was the case with hospital environment, the school environment also has narrow corridors and unique obstacles that block the characters paths. The exit points and spawn points are shown in figure 6. A table is used as the obstacle for blocking the shorter path.

VI. Experiments and Results

The experiments in this research were designed to study the impact of robot path clearing operations on emergency evacuations. The dependent variable for this work was evacuation time. We also wish to study the impact of grid resolution on the success of the planner, where unblocking the desired path such that it becomes available for evacuation was considered a successful trial. The evacuees speed ranged from 6 to 10 miles per hour. Evacuees move towards the exit as soon as they are spawned. The evacuation time is calculated from the moment the evacuees start to move towards the exit. The robot does not start moving towards the target immediately. The evacuees are allowed to take the longer exit for the first ten seconds to establish an initial baseline for the average evacuation time taken to exit via the longest path. After 10 seconds, the robot started to make its way to the target location. The target location was setup such that at least one path required an obstacle to be moved to reach the location. The results for each of the environments are presented below. Multiple trials were conducted per environment and at different resolutions of the map. A trial is considered a success if the robot unblocks the shorter path and evacuees manage to reduce their average evacuation time.

A. Office Environment

For this environment, the robot was tasked to remove the obstacles from two paths, the intermediate path first (yellow line in figure 4) and then from the shortest path (green line in figure 4). Five trials were run with the same environment and the same configurations using fine and coarse grid resolutions (figure 7 and 8). The upper dashed line, referred to as 'ideal longest time', represents average time taken by the evacuees along the longest path (red line in figure 4) and the lower dashed line, referred to as the 'ideal shortest time' represents the average time taken by the evacuees along the shortest path (green line in figure 4). These lines represent the ideal time taken by the evacuees if they start evacuating along the corresponding path without any deviations or obstructions.

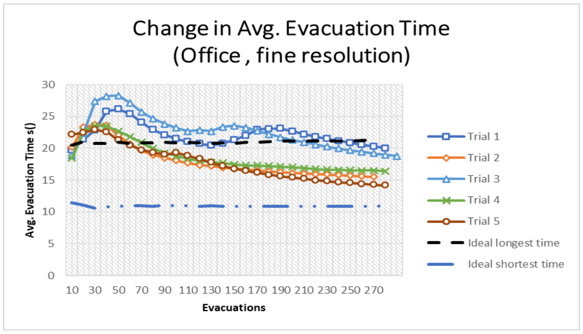

Fig. 7. The figure shows the results of the experiment conducted in the office environment and fine resolution. The robot removed a blockage paths from the intermediate path and shortest path respectively. The trend of decreasing evacuation times can be seen clearly.

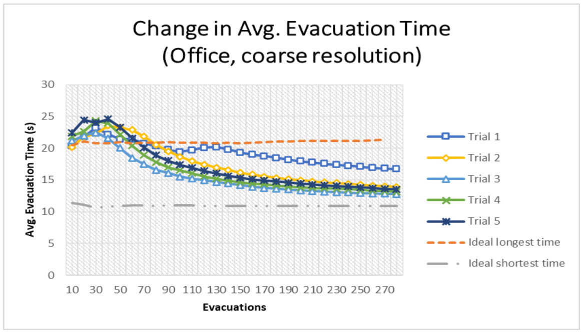

Fig. 8. The figure shows the results of the experiment conducted in the office environment and coarse resolution. The robot removed a blockage from the intermediate path and shortest path respectively. The trend of decreasing evacuation times can be seen clearly.

The graphs depict the trend of decreasing evacuation times (figure 8 and 7). This trend towards the lower boundary suggest that the robot's removal of the blockage leads to fast evacuation times. The lines that approach the 'ideal longest time' represent cases when the robot was either unable to remove the obstacle in time or completely failed. Figure 7 demonstrates some of these failures with trial 1 and trial 3. The initial peaks in the plots represent the evacuees wasting time running towards the path that the robot is actively trying to unblock. As the robot keeps trying to move the object out of the path, partial movements cause the path to be temporarily available to the evacuees. The evacuees then stop moving on their current path and try to take the shorter path. While the robot is continuing to perform the push actions, it blocks the shorter path. This causes the evacuees to stop moving towards the shorter path and reattempt the longer path. We observed some variability of the success rate of the robot on the map grid resolution. We hypothesized that allowing for finer grid resolutions would allow more successful runs at the cost of computational complexity as the robot relies on grid points around the object to start the push actions. However, in practice we observed that a finer grid resolution caused the robot to take longer to complete the task as seen in figure 7. Trial 1 and trial 3, while being successes, take much longer to complete. We believe that the additional time is due to the computational complexity of the problem. We provide a more detailed discussion about grid resolutions in section VI-D.

B. Hospital Environment

A similar experiment was conducted in the hospital environment. This environment was designed to be more challenging with narrower corridors and little space robot for the robot to maneuver. The robot removed one blockage from the shorter path (green line in figure 5). The experiment was conducted using fine and coarse grid resolutions (figure 10 and figure 9). The graphs depicts the results from the five trials of the experiment. Once again, 'ideal longest time' represents the average evacuation time if the longest path is taken (red line in figure 5) and the 'ideal shortest time' depicts the time if the shortest path is taken (green line in figure 5).

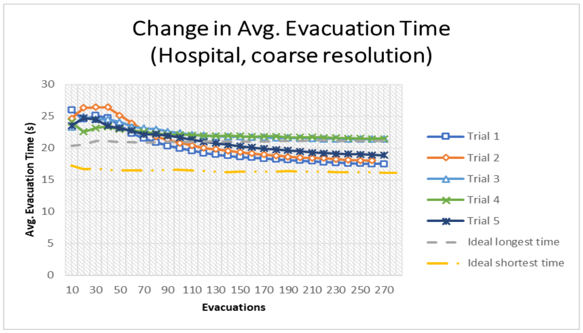

Fig. 9. The figure shows the results of the experiment conducted in the hospital environment and coarse grid resolution. The robot removed a blockage from the shortest path. The trend of decreasing evacuation times can be seen clearly.

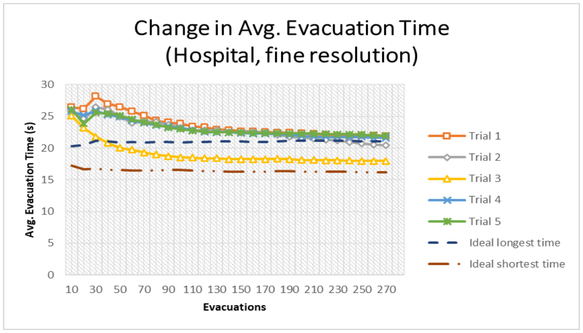

Fig. 10. The figure shows the results of the experiment conducted in the hospital environment and fine grid resolution. The robot removed a blockage from the shortest path. We observe that all but Trial 3 approach the upper bound due to robot's failure in removing the blockage from the path.

Once again a general trend of decreasing average evacuation times is observed. Similar to the case of the office environment, we observe initial peaks which are associated with erroneous evacuee behavior of them switching to a temporary shorter path before it is completely available. The dashed lines represent the average evacuation times via the longest and shortest path (red and green lines in figure 5) respectively. Lines approaching the 'ideal shortest time' suggest that robot unblocked the path successfully. As can be seen in figure 9, trial 3 and trial 4 suggest robot failures as average evacuation time approaches the upper dashed line. Surprisingly, a higher number of failures was observed in the experiment with fine resolution (figure 10). Decreasing evacuation times can be seen for successful trials.

C. School Environment

The school environment contains wider corridors compared to the hospital environment, but narrower compared to the office environment. Five trials of the experiment was conducted using fine and coarse grid resolutions (figure 12 and 11) with dashed lines suggesting the longest and shortest ideal evacuation times via the longest and the shortest path (red line and green line in figure 6) respectively.

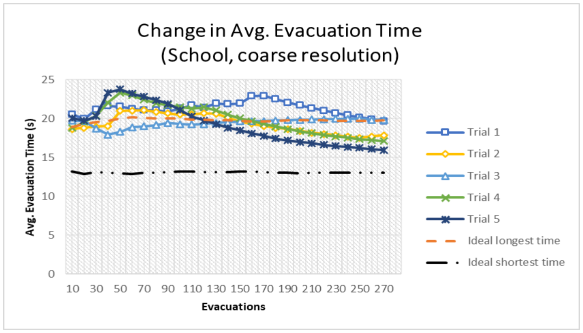

Fig. 11. The figure shows the results of the experiment conducted in the school environment. This experiment was performed with a resolution of 0.15. The robot was tasked to remove blocked paths from the shortest path. The trend of decreasing evacuation times can be seen clearly.

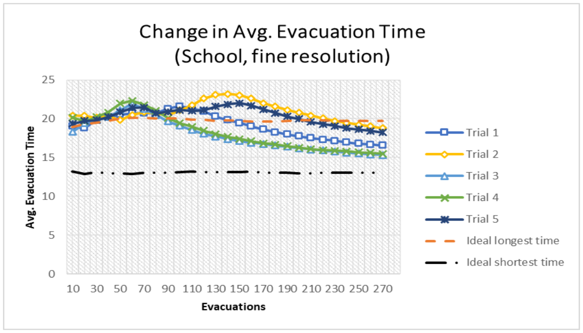

Fig. 12. The figure shows the results of the experiment conducted in the school environment. This experiment was performed with a resolution of 0.1. The robot was tasked to remove blocked paths from intermediate path and shortest path respectively. The trend of decreasing evacuation times can be seen clearly.

The average evacuation times decrease, suggesting that the robot managed to remove the blockage from the paths in both the graphs. Large peaks were observed in figure 12 but all of the lines tend to the lower dashed line as more evacuations are recorded. This implies that while the robot was able to unblock the path, it was inefficient and took a long time to unblock the paths. As before, the larger peaks suggest new paths momentarily being open, causing the evacuees to turn back from their current path. Lower resolution of the grid resulted in less failed trials as evidenced by lines approaching the lower boundary.

D. Grid Resolution

It was observed that the resolution of the grid map had an impact on the ability of the algorithm to unblock the path. The resolution units are in meters. To further investigate this, the office environment was run multiple times with different resolution settings to measure the number of successes of the algorithm. A trial is considered a success if the robot unblocks the shorter path and evacuees manage to reduce their average evacuation time. It was noted that as we decrease the resolution of the grid, i.e. finer grid resolution, the percentage of successes the algorithm increased (figure 13). The graph shows the percentage of successful trials for office environment per grid resolution. However, it was observed that the a resolution of 0.15 outperformed a resolution of 0.10 suggesting that there might be a trade-off with higher grid resolutions. Finer grid resolution, increases the number of nodes in the graph used by the A* planner causing the planner to take longer as the time complexity of the problem scales with the number of nodes on the grid.

However this was not always true as in the cases previously presented in figure 10, we observed that the robot found it self stuck in areas it could not move out as it kept pushing against immovable objects. We believe that success rates also depends on the design of the environment. For a narrower corridor, some of grid points on coarse resolution may be too close to an immovable object making them invalid while a finer grid resolution ensures more points are available on the map per unit distance on the map. However, we observed that due to this there were more opportunities for the robot to find itself stuck in spaces as it was unable of move out of them.

Additionally, finer grid resolutions caused the algorithm to run much slower as can be seen in figure 7 and 12 where some of the trials take much longer to start decreasing even though they are successful trials. Each operation on the map data is computationally more taxing. Further investigation will be necessary to understand these results.

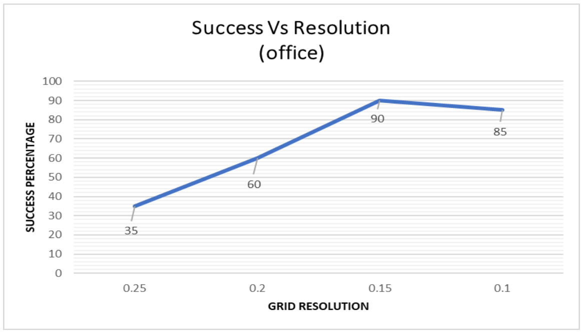

Fig. 13. The figure shows the impact of grid resolution on the performance of the algorithm. 20 trials were conducted per grid resolution.

VII. Conclusion

This paper presents a preliminary study exploring robot initiated evacuation path clearing. We present a method allowing the robot to use and create plans for obstacle detection and path clearing. Our experiments demonstrate that our approach is generalizable to multiple environments. The results show that the robot's efforts are able to remove blockages and improve average evacuation time.

We observed a number of cases where the robot failed to finish the task. The failures can be attributed to two main issues. First is the lack of a fine tuned object detection model and the second is the lack of additional ways to move an obstacle which would allow the robot to tackle a wider variety of tasks. It is possible that repeated push manipulations cause the obstacle to attain an orientation that cannot be resolved with just a push action. It was also found that the object sometimes gets caught at the edge of another object and hence gets stuck. A better method for detecting if a path is open would allow the robot to perform even partial manipulations which would be sufficient for evacuation operations. Object detection based issues can be mitigated by fine tuning a dedicated object detection model trained on the data based on movability information. This would yield significantly better results with detection tasks. Another non-prehensile action such as 'pulling' can improve the system's performance.

In the future, we propose to use a fine tuned model for object detection and data driven model to generate movability estimates of objects. Additional manipulation capability can be incorporated into the system to compare the impact of manipulations on task success. Different planning methods, such as RRT may introduce some randomness in the path generation algorithm. A well defined human behavior model can be used to influence the evacuation behavior of the humans. We believe that a capable system equipped with the right tools can make a significant impact in saving human lives during emergency operations.

References

[1] M. Nayyar, Z. Zoloty, C. McFarland, and A. R. Wagner, "Exploring the effect of explanations during robot-guided emergency evacuation," in *Social Robotics*. Springer International Publishing, 2020, pp. 13–22.

[2] M. Nayyar and A. R. Wagner, "When should a robot apologize? understanding how timing affects human-robot trust repair," in *International conference on social robotics*. Springer, 2018, pp. 265–274.

[3] I. Sakour and H. Hu, "Robot-assisted crowd evacuation under emergency situations: A survey," *Robotics*, vol. 6, no. 2, p. 8, 2017.

[4] D. A. Shell and M. J. Matarić, "Insights toward robot-assisted evacuation," *Advanced Robotics*, vol. 19, no. 8, pp. 797–818, 2005.

[5] M. Nayyar and A. R. Wagner, "Effective robot evacuation strategies in emergencies," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2019, pp. 1–6.

[6] E. Boukas, I. Kostavelis, A. Gasteratos, and G. C. Sirakoulis, "Robot guided crowd evacuation," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 739–751, 2014.

[7] I. Sakour and H. Hu, "Robot assisted evacuation simulation," in *2016 8th Computer Science and Electronic Engineering (CEEC)*, 2016, pp. 112–117.

[8] B. Tang, C. Jiang, H. He, and Y. Guo, "Human mobility modeling for robot-assisted evacuation in complex indoor environments," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 5, pp. 694–707, 2016.

[9] P. Robinette, A. R. Wagner, and A. M. Howard, "Investigating human-robot trust in emergency scenarios: methodological lessons learned," in *Robust Intelligence and Trust in Autonomous Systems*. Springer, 2016, pp. 143–166.

[10] D. J. Atkinson and M. H. Clark, "Methodology for study of human-robot social interaction in dangerous situations," in *Proceedings of the second international conference on Human-agent interaction*, 2014, pp. 371–376.

[11] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human factors*, vol. 46, no. 1, pp. 50–80, 2004.

[12] P. Robinette, P. A. Vela, and A. M. Howard, "Information propagation applied to robot-assisted evacuation," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 856–861.

[13] S. Krivic and J. Piater, "Pushing corridors for delivering unknown objects with a mobile robot," *Autonomous Robots*, vol. 43, no. 6, pp. 1435–1452, 2019.

[14] J. Stüber, C. Zito, and R. Stolkin, "Let's push things forward: A survey on robot pushing," *Frontiers in Robotics and AI*, vol. 7, p. 8, 2020.

[15] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.

[16] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2016, pp. 30–37.

[17] M. Bauza and A. Rodriguez, "A probabilistic data-driven model for planar pushing," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3008–3015.

[18] P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," *arXiv preprint arXiv:1606.07419*, 2016.

[19] P. C. Chen and Y. K. Hwang, "Practical path planning among movable obstacles," Sandia National Labs., Albuquerque, NM (USA), Tech. Rep., 1990.

[20] M. Stilman and J. Kuffner, "Planning among movable obstacles with artificial constraints," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1295–1307, 2008.

[21] D. Nieuwenhuisen, A. F. van der Stappen, and M. H. Overmars, "An effective framework for path planning amidst movable obstacles," in *Algorithmic Foundation of Robotics VII*. Springer, 2008, pp. 87–102.

[22] J. Van Den Berg, M. Stilman, J. Kuffner, M. Lin, and D. Manocha, "Path planning among movable obstacles: a probabilistically complete approach," in *Algorithmic Foundation of Robotics VIII*. Springer, 2009, pp. 599–614.

[23] H.-n. Wu, M. Levihn, and M. Stilman, "Navigation among movable obstacles in unknown environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 1433–1438.

[24] M. Levihn, "Navigation among movable obstacles in unknown environments," Ph.D. dissertation, Georgia Institute of Technology, 2011.

[25] Y. Kakiuchi, R. Ueda, K. Kobayashi, K. Okada, and M. Inaba, "Working with movable obstacles using on-line environment perception reconstruction using active sensing and color range sensor," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 1696–1701.

[26] S. K. Moghaddam and E. Masehian, "Planning robot navigation among movable obstacles (namo) through a recursive approach," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 3-4, pp. 603–634, 2016.

[27] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.