2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Path Re-Planning with Stochastic Obstacle Modeling: A Monte Carlo Tree Search Approach

Francesco Trotti[1], Alessandro Farinelli[2] and Riccardo Muradore[1]

***Abstract*— Path re-planning and repairing are key topics for robust planning and navigation in open dynamic environments, finding applications in various domains such as fleet control of Unmanned Ground Vehicles (UGVs) in warehouses. The use of UGVs in open and dynamic environments requires flexible cooperation between human operators and the UGV fleet within a shared environment. In this paper, we propose a local strategy to re-plan the path of robots encountering unexpected and dynamic obstacles. Specifically, starting from a given Multi-Agent path, we model the re-planning problem as a Markov Decision Process (MDP) considering a stochastic obstacle lifespan, and we propose two local approaches based on Monte-Carlo Tree Search to re-plan the path of the robots that encounter obstacles. We compare these approaches with traditional Multi-Agent Path Finding (MAPF) algorithms to obtain new collision-free paths when an obstacle is detected. The evaluation is performed in simulation using benchmarking instances of warehouses and experimentally in a research facility with a scaled-down Industry 4.0 production line.**

I. INTRODUCTION

The problems of re-planning and repairing Multi-Agent path are key topics for controlling a fleet of UGVs in dynamic environments where the probability of encounter some obstacles is very high, such as in the context of Industry 4.0 and large warehouse scenarios.

The re-planning problem usually assumes that the robot fleet already has a plan to reach the goal by solving a Multi-Agent Path Finding (MAPF) problem. There is a wide selection of possible approaches in the literature to address MAPF, including consistent prioritization [1], [2], conflict-based approaches [3], or large-neighborhood search [4]. MAPF algorithms have found applications in various domains ranging from large-scale warehousing [5], [6] to aircraft coordination [7].

Most MAPF solution approaches consider a known static map, where the only moving objects are the robots that avoid conflicts thanks to the coordination approach. However, in practical applications, it is essential to consider possible changes to the map that happen while the robots are executing their paths. In these situations, robots must re-plan or repair their paths to manage such changes. Some approaches based on MAPF algorithms to re-plan the paths have been proposed in the literature; for instance, the authors in [8] introduced an algorithm based on large neighborhood search to repair the paths of colliding robots, while in [9], the authors proposed a prioritized planning approach for re-planning paths. In [10], a re-planning strategy based on globally guided reinforcement learning reduces the computational cost. Other MAPF approaches consider uncertainty/with delay probabilities such as [11] and [12]. Other strategies based on Partially Observable Markov Decision Processes (POMDPs) have been analyzed in the literature, such as [13]. The authors proposed a dynamic re-planning approach based on belief space, integrating online and offline POMDP techniques to maneuver robots in a continuously changing environment. However, this approach only scales to systems with a few agents due to the computational cost of POMDP.

In this paper, we formalize the re-planning problem as a Markov Decision Process (MDP), considering a stochastic obstacle lifespan model in the transition function. In particular, we propose a local strategy based on the concept of re-planning so as not to "disturb" the paths of other robots and avoid general or partial re-planning of the fleet. Given a path calculated by a MAPF algorithm (we use Conflict-Based Search (CBS) [14] in this paper), our approach re-plans only the path of the robot that encounters an obstacle. Specifically, our method is based on Monte Carlo Tree Search (MCTS) [15][16] to solve the MDP. More in detail, we provide two different MDP-based algorithms for re-planning: MCTS Planner and MCTS Heuristic. Both planners use a stochastic description of the obstacle lifespan to calculate a new path. In the first algorithm, the MDP selects, at each iteration, the action that minimizes the distance from the target and samples a believed obstacle lifespan. The second algorithm is formalized as Bandit MDP [17], where only the initial movement is chosen, and from the new position, Space-Time A^* [18] is executed. At each MDP simulation, we sample from the probability distribution of the obstacle lifespan, and the algorithm tries to minimize the travel distance considering this information. Finally, we also provide a standard MAPF re-plan strategy as a benchmarking baseline. Such a strategy is based on the re-invocation of the CBS algorithm at each collision. In the result, we compare the three approaches to analyze their performance. The simulated (using standard warehouse scenario [19]) and real experiments confirm that by exploiting the stochastic model for the obstacle lifespan, we can make better local decisions, thus improving the resulting quality of the solution (i.e., minimizing the total travel time and the makespan).

The main contributions of this paper are:

[1] Francesco Trotti and Riccardo Muradore are with the Department of Engineering for Innovation Medicine, University of Verona, 37134, Italy francesco.trotti@univr.it
riccardo.muradore@univr.it

[2] Alessandro Farinelli is with the Department of Computer Science, University of Verona, 37134, Italy alessandro.farinelli@univr.it

- The formalization of the re-planning problem as an MDP exploiting a stochastic obstacle lifespan model
- A local MDP-based strategy to solve the re-planning problem
- A comparison between the three algorithms with quantitative and qualitative experiments

II. BACKGROUND

A. MDP and MCTS

Markow Decision Processes (MDPs) framework allows the generation of a policy for an agent in a fully observable environment. A MDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, where:

- $\mathcal{S}$ is the state space with $s \in \mathcal{S}$
- $\mathcal{A}$ is the actions space with $a \in \mathcal{A}$
- $\mathcal{T}$ is the transition probability $\mathcal{T}^a_{s,s'} = \mathbb{P}(s_{t+1} = s' | s_t = s, a_t = a)$ that defines the probability to evolve from the current state s at time t to the future state s' at time $t+1$ by taking the action a
- $\mathcal{R}(s, a)$ is the reward function from the current state $s \in \mathcal{S}$ taking the action $a \in \mathcal{A}$
- γ is the discount factor ($\gamma \in [0, \dots, 1]$)

Consequently, the MDP policy maps from states into actions, determining which action will be chosen from each state.

The online solvers for the MDP are widely used for their lower computational time and memory usage compared to offline solvers. A popular online MDP solver is Monte Carlo Tree Search (MCTS), which alternates between planning and execution phases to provide online actions for the agents. MCTS operates through four phases:

1) Selection: The algorithm selects the node to reach the leaf of the tree using Upper Confidence bounds applied to Trees (UCT) [17].
2) Expansion: New nodes are added to the leaf node
3) Simulation: A simulator denoted by $\mathcal{G}$ is used to emulate the possible agent behaviors in the future. Given a state and an action, the simulator provides the state at time $t+1$ (s_{t+1}) and a reward value r_{t+1}. The simulator can be defined as follows: $(s_{t+1}, r_{t+1}) \sim \mathcal{G}$.
4) Back-propagation: The score obtained during the simulation is back-propagated to the root.

B. Multi Agent Path Finding

Multi-Agent Path Finding (MAPF) addresses the problem of finding a collision-free path for multi-agent systems optimizing a function (e.g., the flowtime or the makespan). The MAPF problem is formalized as follows:

- M is the number of the agents, and m_i represents the i-th agent with $i \in M$.
- $G = (V, E)$ is an undirected graph, where V is the set of the nodes and E is the set of the edges.

Each agent can wait at its current node or move to adjacent nodes. The combination of these actions generates a path $\pi_i = \{\pi_1(t), \dots, \pi_1(t+1)\}$, with t the discrete time. A plan is defined as a set of paths for each agent $\pi = \{\pi_1, \dots, \pi_M\}$ on which collisions are checked. The collision can be:

- Vertex collision: It occurs when two agents occupy the same node at time t
- Edge collision: It occurs when two agents traverse the same edge in opposite directions at the same time t

A MAPF solution is a collision-free path that minimizes the number of steps for each agent. Conflict-Based Search (CBS) is an optimal and complete MAPF algorithm. This algorithm is based on the definition of a set of constraints defined as a tuple (a_i, v, t) where a_i is the i-th agent and v is the prohibited vertex at time step t. When the algorithm independently solves the planning problem for each agent complying with the constraints, a collision check between paths is done. If there is a collision, a new constraint is added. This approach iterates until a collision-free plan is generated.

III. PROBLEM STATEMENT

Our local re-planning approach is tailored to scenarios like warehouses, where we define various obstacles: static (e.g., walls or stocked shelves), dynamic, and temporal. Dynamic obstacles include time information regarding their path (e.g., other UGVs following a known trajectory), while temporal obstacles are static obstacles with stochastic lifespans (e.g., a human operator performing a task on a machine). Specifically, given a scenario (a grid or a graph), a set of agents (M), and an optimal collision-free plan for all agents (π), we introduce temporal obstacles at crucial points on the map to prompt the re-planning of specific robot paths. We utilize a Conflict-Based search (CBS) algorithm to find an initial optimal plan for all robots, given a set of initial and goal positions and a random assignment. The re-planning strategy hinges on formalizing a Markov Decision Process (MDP) that considers the stochastic lifespan of encountered obstacles in the transition function. Therefore, the computation of the new path depends on the paths of other robots (considered dynamic obstacles with defined trajectories) and the probability distribution of the lifespan of temporal obstacles. The optimal new path might involve waiting until the obstacle disappears, moving away to clear passage for other robots, or calculating a new collision-free path. The following assumptions commonly used in the literature are needed:

Assumption 1: When a robot re-plans, it shares its path with other robots

Assumption 2: Time is fully discretized, and the space is discretized as a grid

A. Stochastic Lifespan Obstacle Model

We model the lifespan of any obstacle using the probability density function (PDF) of the gamma distribution

$$\mathbb{P}(X) = f(X, \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} X^{\alpha-1} e^{-\beta X} \tag{1}$$

where X represents the believed obstacle lifespan, α is the shape parameter of the gamma distribution, β is the rate parameter of the gamma distribution, and $\Gamma(\alpha) = (n-1)!$. It is important to note that the gamma distribution is defined for

$X \geq 0$, which makes sense as we are modeling the lifespan of an obstacle. The relations between the parameters α and β and the mean (μ) and variance (σ^2) are

$$\alpha = (\frac{\mu}{\sigma})^2 \quad \mu = \frac{\alpha}{\beta}$$
$$\beta = \frac{\mu}{\sigma^2} \quad \sigma^2 = \frac{\alpha}{\beta^2}$$

1) Update rules for the gamma distribution: By leveraging Bayesian inference, it is possible to update the prior distribution $\mathbb{P}(X)$, which follows a gamma distribution with parameters α and β, using the likelihood of observed data to obtain a posterior distribution. We propose two approaches to update the gamma distribution (deterministic and stochastic).

Deterministic update. The deterministic update is used when the robot can directly observe the obstacle at time t. In this case, the update is straightforward and is based on updating the α and β values considering the time t of the observation. Therefore, the new values are

$$\begin{cases} \alpha^{'} = \alpha \\ \beta^{'} = \beta + \frac{t}{X} \end{cases} \tag{2}$$

The shape of the gamma distribution (α) remains the same as before, while the rate parameter (β) is increased by the ratio between the observation time t and the believed obstacle lifespan X.

Stochastic update. The stochastic update is based on Bayes' theorem: given a prior distribution and a probability distribution to observe an event, their combination generates a posterior distribution. We use the gamma function as the prior distribution, and we model the probability that the obstacle is still present at time t, even if the robot does not directly observe it, as an exponential distribution

$$\mathbb{P}(t|X) = \frac{1}{X} e^{-\frac{1}{X}t} \tag{3}$$

where t is the observed time. In this way, a smaller obstacle lifespan (X) results in higher event rates, while larger obstacle lifetimes lead to lower event rates. Applying Bayes theorem, we end up with

$$\begin{aligned} \mathbb{P}(X|t) &\propto \mathbb{P}(X)\mathbb{P}(t|X) \\ &\propto \left(\frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}\right)(\lambda e^{-\lambda t}) \\ &\propto \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{1}{X} x^{\alpha-1} e^{-\beta x - \frac{t}{X}} \end{aligned} \tag{4}$$

To obtain the updated values of α and β, we normalize by integrating over the entire range. This allows us to reshape the gamma distribution based on the previous observations

$$\begin{cases} \alpha^{'} = \alpha + 1 \\ \beta^{'} = \beta + \frac{t}{X} \end{cases} \tag{5}$$

The shape parameter (α) is incremented by one to take into account the new observed event, while the rate parameter (β) is incremented by the ratio between the observation time t and the believed obstacle lifespan X.

IV. Re-Planning Algorithms

In this section, we describe the three proposed algorithms. Firstly, we describe our benchmarking baseline, which is a re-planning algorithm based on CBS, where the obstacle lifespan is sampled from the gamma distribution each time an obstacle is encountered. Secondary, the two MDP-based algorithms are formalized. The first provides an action to the robot at each iteration, while the second generates a collision-free path.

A. CBS Re-plan

The CBS Re-plan algorithm is a popular re-planning approach used as a baseline. This planner invokes the CBS algorithm each time an obstacle is encountered, estimating its lifespan by sampling the gamma distribution. However, it becomes computationally inefficient with many agents, requiring recalculation of the paths of all robots for each encountered obstacle. Additionally, if the estimated obstacle lifespan matches the real lifespan, the algorithm provides an optimal solution, and the results are complete as CBS. Therefore, the accuracy of this algorithm is closely related to the obstacle lifespan estimation. Finally, the algorithm updates the gamma distribution using the deterministic method each time a robot encounters an obstacle.

B. MCTS Planner

This algorithm is based on the formalization of the problem as an MDP, which is solved using MCTS. The policy generated by the MDP provides an action to move the robot to a new collision-free position. The MDP is formalized as follows.

States: The state consists of the robot position (p), the obstacle position (o), and the plans of other robots (π)

$$s = [p, o, \pi] \tag{6}$$

Actions: The actions are the possible movements of the robot in a Cartesian-like grid

$$a = [\text{up},\ \text{down},\ \text{left},\ \text{right}] \tag{7}$$

Transition model: The transition model is deterministic for the robot movements and stochastic for the obstacle lifespan. Given an action and a current state, the robot is certain to move in the next state, while the state of the obstacle is sampled from the gamma distribution illustrated in Subsection III-A.

Reward: The reward function is designed to minimize the distance to the goal while penalizing incorrect movements that result in collisions with the obstacles. The reward function is defined as

$$\begin{aligned} R &= ||p - p_g|| + k \\ k &= \begin{cases} 0, & \text{if } p \neq o \\ \infty, & \text{if } p = o \end{cases} \end{aligned} \tag{8}$$

where p and p_g represent the robot actual position and the goal position, respectively, and o represents the position of static, dynamic, or temporal obstacles.

Sampling from the gamma distribution at each MCTS simulation allows the algorithm to evaluate different actions, considering specific obstacle lifespans in each simulation. This strategy simulates and evaluates numerous obstacle lifespan configurations to determine the best local action. After the robot movement in the environment, the gamma distribution is updated in a stochastic manner since the robot is not always close to the obstacle. Therefore, we consider the probability that the obstacle at time t is in place to update the gamma distribution. This process continues by calling MCTS again until the robot reaches the goal.

C. MCTS Heuristic

This algorithm is based on the Bandit MDP, where the simulation phase is limited to one iteration. In this case, the policy generated by the MDP is a completely new collision-free path. The MDP is formalized as follows.

States and *Actions* are defined as in Sections IV-B.

Transition model: The transition model is deterministic for the robot movements and stochastic for the obstacle lifespan. Specifically, given an action and a current state, the robot is certain to move to the next state, and from that, it invokes Space-time A^* to find a path. The state of the obstacle is sampled from the gamma distribution to define the believed lifespan used in Space-time A^* for the planning.

Reward: The reward function is designed to minimize the travel time

$$R = \begin{cases} \sum_{i=1}^{n-1} w_i, & \text{if } n \neq 0 \\ \infty, & \text{if } n = 0 \end{cases} \tag{9}$$

where w_i represents the i-th waypoint, w is the list of waypoints, and n is the total number of waypoints. Also, in this case, the algorithm samples an obstacle lifespan during each simulation and executes the simulation with this fixed value. Differently from the previous algorithm, the policy is the entire optimal path, not just a local action to move the robot. This approach evaluates the paths generated by Space-time A^* from different starting points and considers different possible obstacle lifespans. If the robot reencounters the obstacle during the path, a deterministic update of the gamma distribution is made, and the algorithm restarts.

D. Discussion

The proposed strategies operate locally, and Assumption 1 is necessary to ensure the correctness and the computation of a new collision-free path. However, this assumption can be easily relaxed in the context of a warehouse or, more broadly, in an Industry 4.0 production line, where network and information sharing between machines is a common feature. Considering a standard publisher/subscriber network infrastructure, the network traffic is minimal, given the capabilities of state-of-the-art networks. In practical terms, assuming a simple message composed of the robot ID (4 bytes) and the new paths ($4n$ bytes, with n the length of the path), the maximum traffic on the network caused by one re-planning robot is determined by the product of the bytes of the package and the number of agents. It is important to note that the broadcasting of the message occurs only when a robot re-plans its path, resulting in very low frequency under normal conditions. Assumption 2 is also relaxed in our context; the algorithms operate on fully discretized time and space since additional information is unnecessary for planning. A local planner with a low-level controller [20] manages the robot motion through the waypoints generated by the proposed algorithms.

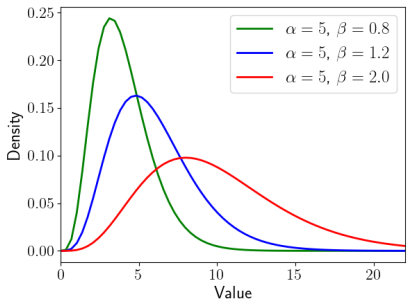


Fig. 1. Gamma distribution for α and β values

V. RESULT

The proposed approach has been tested and validated in two experiments: 1) in a simulated warehouse scenario (a 14×14 grid with 4 agents) [19], and 2) in a research facility with an Industry 4.0 production line using two RB-Kairos mobile robots. The metrics used to evaluate our approach are the total travel time (the sum of the arrival times of all agents at their goals) and the total makespan (the maximum arrival time of all agents at their goals), which are standard metrics in the literature for evaluating planning algorithms.

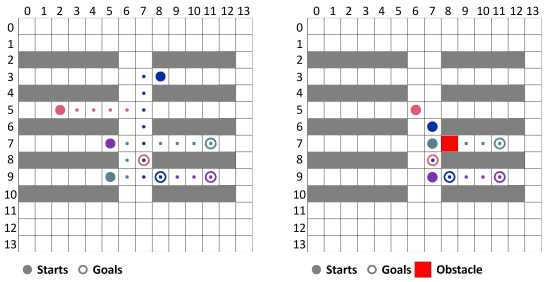
A. Scenario 14×14 with 4 agents

Obstacles lifespan are computed with a constant shape ($\alpha = 5$ is a reasonable value for the experiments) and with different scale values (β), as shown in Figure 1. Each simulated experiment is executed 100 times on a 14×14 grid scenario with 4 agents, and the MCTS simulations are fixed at 300 iterations. The real obstacle lifespan remained constant for all runs, while the believed lifespan was sampled from the gamma distribution at each run. The results are obtained using a computer equipped with standard hardware[1]. In this scenario, an obstacle is placed to block a robot path that consequently obstructs the paths of the other robots.

Figure 2(a) shows the scenario without obstacles and the paths generated by the first CBS run. In this case, the total travel time is 36, and the makespan is 9. A snapshot of the scenario when the light blue agent encounters the obstacle (at time 5) is shown in Figure 2(b). The obstacle obstructs the light blue agent path, which consequently blocks the path of the blue and red agents.

Table I lists the re-planned paths using the three different algorithms discussed in Section IV. In this case, we use $\alpha = 5$ and $\beta = 1.2$, with the real obstacle lifespan set to 2 (the obstacle disappears at time 2, so the cell becomes free

[1]Intel Core i7-9750H processor and 32GB of RAM.



(a) Scenario without obstacle with the CBS paths.

(b) Scenario with the obstacle at time 5.

Fig. 2. 14×14 scenario with 4 agents and 1 obstacle

	CBS Re-plan
●	$[(6,7)(7,7)(8,7)(9,7)(9,8)]$
●	$[(7,7)(7,6)(6,6)(5,6)(5,7)(5,8)(5,9)(5,10)$ $(5,11)(5,12)(5,13)(6,13)(7,13)(7,12)(7,11)]$
●	$[(5,6)(5,7)(6,7)(7,7)(8,7)]$
●	$[(9,7)(9,8)(9,9)(9,10)(9,11)]$
	MCTS Planner
●	$[(6,7)(7,7)(8,7)(9,7)(9,8)]$
●	$[(7,7)(7,6),(7,7),(7,8),(7,9),(7,10),(7,11)]$
●	$[(5,6)(5,7)(6,7)(7,7)(8,7)]$
●	$[(9,7)(9,8)(9,9)(9,10)(9,11)]$
	MCTS Heuristic
●	$[(6,7)(7,7)(8,7)(9,7)(9,8)]$
●	$[(7,7)(7,6)(7,7)(7,8)(7,9)(7,10)(7,11)]$
●	$[(5,6)(5,7)(6,7)(7,7)(8,7)]$
●	$[(9,7)(9,8)(9,9)(9,10)(9,11)]$

TABLE I

RE-PLANNED PATHS IN THE 14×14 SCENARIO

at time 3). The paths are shown starting from the snapshot at time 5 (since the previous parts of the paths are identical for all algorithms). The CBS Re-plan algorithm completely re-planned the agent paths to avoid the obstacle but increased the total travel time (44) and the makespan (17). The other two algorithms, based on the MDP and incorporating the stochastic lifespan model, estimate the optimal re-planning strategy more effectively. In this case, only the path of the light blue agent is re-planned because the MDP is designed to replan only the agent that encounters the obstacle. These two algorithms provide the same paths. Precisely, the light blue agent moves back to allow the passage of the other robots and then returns to its previous position towards the goal. This behavior is generated by leveraging multiple simulations with different samples from the gamma distribution, enabling the algorithms to provide the best new paths. These two algorithms plan paths with a total travel time of 38 and a makespan of 11. This case serves as an example to emphasize the differences between the algorithms; in this case, CBS makes the worst choice as its paths are too long compared to the other algorithms, likely due to a large value sampled from the gamma distribution.

Table II displays the quantitative results of the three algorithms evaluated on three configurations of the gamma function across 100 runs. Each set of results aligns with the trend observed in the previous experiment: the two MDP-based algorithms exhibit similar performance, leveraging the

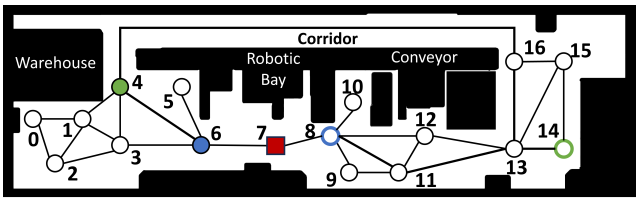


Fig. 3. Topological map of our laboratory at the collision time. The filled circles represent the robot positions, the contoured circles the goal positions, and the red square the obstacle

probabilistic model for the obstacle lifespan. The CBS Re-plan algorithm demonstrates higher mean makespan and mean total travel time compared to the other algorithms. This is because to achieve an optimal makespan or travel time, the gamma distribution's sampled value must be aligned with the real obstacle lifespan. In runs where $\beta = 1.2$, the algorithm shows slightly better performance, likely due to the probability being more centered on the real obstacle lifespan. The MCTS Planner consistently delivers superior performance across all scenarios, thanks to its stochastic updating of the gamma distribution. In each simulation, the algorithm samples and simulates different values from the gamma distribution, updating the distribution after moving the robot to a new position through a stochastic procedure. The MCTS Heuristic provides solutions of comparable quality by leveraging internal simulations and testing various configurations of the believed obstacle lifespan. However, it works slightly worse than the MCTS Planner, as the update of the belief of the obstacle lifespan occurs only when the robot perceives the obstacle. Across all three gamma distributions, results for both MDP-based algorithms (MCTS Planner and MCTS Heuristic) remain consistent. This can be attributed to the explicit utilization of the probabilistic model in the re-planning procedure, enabling the approaches to consider the expected value of the lifespan (as in the CBS re-planning approach) and the distribution shape.

B. Validation on real robotic platforms

In the second experiment, we implement the algorithms in a realistic environment involving two mobile robots (RB-Kairos). Each robot knows the topological map and the position of the other robot. Figure 3 illustrates the topological map at the moment of collision: the green robot starts its path from node 0, the blue robot in node 1, and the snapshot is captured at time step 3. For this experiment, we introduce a real obstacle situated at node 7 with a lifespan of 2. We set $\alpha = 5$ and $\beta = 1.2$; a collision is detected if the subsequent node is obstructed. In Table III, the paths generated by the three algorithms are reported with the total travel time and makespan. All three algorithms yield identical results due to the confined environment. The blue robot (which must estimate the obstacle lifespan) has to calculate a new path, and in this case, the total travel time of the recalculated path is slightly worse than the original one. Achieving a better total travel time is only possible when the believed lifespan

	CBS Re-plan			MCTS Planner			MCTS Heuristic		
	$\alpha = 5$ $\beta = 0.8$	$\alpha = 5$ $\beta = 1.2$	$\alpha = 5$ $\beta = 2.0$	$\alpha = 5$ $\beta = 0.8$	$\alpha = 5$ $\beta = 1.2$	$\alpha = 5$ $\beta = 2.0$	$\alpha = 5$ $\beta = 0.8$	$\alpha = 5$ $\beta = 1.2$	$\alpha = 5$ $\beta = 2.0$
Avg. Total Travel T.	44	42	45	35	35	35	36	36	37
Std. Total Travel T.	3.42	5.33	6.19	2.81	3.22	3.00	1.02	1.23	1.34
Avg. Makespan	16	15	17	11	12	12	10	10	11
Std. Makespan	3.74	3.70	3.77	1.02	1.16	1.12	2.81	3.22	3.00
Avg. Exec. time	6.42s	6.28s	7.01s	0.36s	0.47s	0.56s	1.55s	1.57s	1.59s

TABLE II

RESULTS OBTAINED AFTER 100 RUNS IN THE 14×14 SCENARIO WITH 4 AGENTS USING DIFFERENT GAMMA DISTRIBUTION CONFIGURATIONS.

	Original path	Total Travel T.	Makespan
●	[1, 3, 6, 7, 8]	11	6
●	[0, 1, 4, 16, 13, 14]		

	CBS Re-plan MCTS Planner MCTS Heuristic	Total Travel T.	Makespan
●	[1, 3, 6, 4, 16, 13, 11, 8]	14	8
●	[0, 1, 4, 16, 13, 14]		

TABLE III

ORIGINAL AND RE-PLANNED PATHS STARTING FROM THE INITIAL NODE ON THE REAL SCENARIO (IN RED THE NODES AT THE COLLISION TIME)

matches the real lifespan, but this occurrence has a low probability. Therefore, all the proposed algorithms provide an equal solution. These findings are further supported by a video demonstrating the execution of the algorithms.

VI. CONCLUSIONS

In this paper, we employ a stochastic model for obstacle lifespan and formalize the re-planning problem as a Markov Decision Process (MDP). This approach generates alternative paths that minimize travel time by utilizing the stochastic lifetime model of the obstacle during the re-planning phase. Specifically, we utilize an MCTS-based solver for the MDP and two variants of this approach. We compare these variants with a benchmarking baseline that re-plans paths using the CBS MAPF solver. Quantitative evaluations are conducted in simulation and a research facility with an Industry 4.0 production line. Future research includes scalability testing on large-scale environments.

REFERENCES

[1] H. Ma, D. Harabor, P. J. Stuckey, J. Li, and S. Koenig, "Searching with consistent prioritization for multi-agent path finding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 7643–7650.

[2] S. Zhang, J. Li, T. Huang, S. Koenig, and B. Dilkina, "Learning a priority ordering for prioritized planning in multi-agent path finding," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 15, no. 1, 2022, pp. 208–216.

[3] J. Kottinger, S. Almagor, and M. Lahijanian, "Conflict-based search for explainable multi-agent path finding," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 32, 2022, pp. 692–700.

[4] J. Li, Z. Chen, D. Harabor, P. Stuckey, and S. Koenig, "Anytime multi-agent path finding via large neighborhood search," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[5] J. Li, A. Tinka, S. Kiesel, J. W. Durham, T. S. Kumar, and S. Koenig, "Lifelong multi-agent path finding in large-scale warehouses," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, 2021, pp. 11 272–11 281.

[6] Z. Liu, H. Wang, H. Wei, M. Liu, and Y.-H. Liu, "Prediction, planning, and coordination of thousand-warehousing-robot networks with motion and communication uncertainties," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 1705–1717, 2020.

[7] R. Morris, C. S. Pasareanu, K. S. Luckow, W. Malik, H. Ma, T. S. Kumar, and S. Koenig, "Planning, scheduling and monitoring for airport surface operations." in *AAAI Workshop: Planning for Hybrid Systems*, 2016, pp. 608–614.

[8] J. Li, Z. Chen, D. Harabor, P. J. Stuckey, and S. Koenig, "Mapf-lns2: fast repairing for multi-agent path finding via large neighborhood search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 9, 2022, pp. 10 256–10 265.

[9] J. Li, Z. Chen, Y. Zheng, S.-H. Chan, D. Harabor, P. J. Stuckey, H. Ma, and S. Koenig, "Scalable rail planning and replanning: Winning the 2020 flatland challenge," in *Proceedings of the international conference on automated planning and scheduling*, vol. 31, 2021, pp. 477–485.

[10] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6932–6939, 2020.

[11] H. Ma, T. S. Kumar, and S. Koenig, "Multi-agent path finding with delay probabilities," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[12] G. Wagner and H. Choset, "Path planning for multiple agents under uncertainty," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 27, 2017, pp. 577–585.

[13] A.-a. Agha-mohammadi, S. Agarwal, S.-K. Kim, S. Chakravorty, and N. M. Amato, "Slap: Simultaneous localization and planning under uncertainty via dynamic replanning in belief space," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1195–1214, 2018.

[14] J. Li, W. Ruml, and S. Koenig, "Eecbs: A bounded-suboptimal search for multi-agent path finding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 14, 2021, pp. 12 353–12 362.

[15] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[16] T. Dam, G. Chalvatzaki, J. Peters, and J. Pajarinen, "Monte-carlo robot path planning," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 213–11 220, 2022.

[17] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, pp. 235–256, 2002.

[18] D. Silver, "Cooperative pathfinding," in *Proceedings of the aaai conference on artificial intelligence and interactive digital entertainment*, vol. 1, no. 1, 2005, pp. 117–122.

[19] R. Stern, N. R. Sturtevant, A. Felner, S. Koenig, H. Ma, T. T. Walker, J. Li, D. Atzmon, L. Cohen, T. K. S. Kumar, E. Boyarski, and R. Bartak, "Multi-agent pathfinding: Definitions, variants, and benchmarks," *Symposium on Combinatorial Search (SoCS)*, pp. 151–158, 2019.

[20] F. Trotti, A. Farinelli, and R. Muradore, "An online path planner based on pomdp for uavs," in *2023 European Control Conference (ECC)*. IEEE, 2023, pp. 1–6.