A Nature-Inspired Algorithm to Adaptively Safe Navigation of a Covid-19 Disinfection Robot

Tingjun Lei[1], Timothy Sellers[1], Shahram Rahimi[2], Shi Cheng[3], and Chaomin Luo[1](✉)

[1] Department of Electrical and Computer Engineering, Mississippi State University,
Mississippi State, MS 39762, USA
Chaomin.Luo@ece.msstate.edu

[2] Department of Computer Science and Engineering, Mississippi State University,
Mississippi State, MS 39762, USA

[3] School of Computer Science, Shaanxi Normal University, Xi'an 710119, China

Abstract. Autonomous mobile robots have been extensively used in medical services. During the Covid-19 pandemic, ultraviolet type-C irradiation (UV-C) disinfection robots and spray disinfection robots have been deployed in hospitals and other public open spaces. However, adaptively safe navigation of disinfection robots and spray disinfection robots have not been adequately studied. In this paper, an adaptively safe navigation model of Covid-19 disinfection robots is proposed using a nature-inspired method, cuckoo search algorithm (CSA). A Covid-19 disinfection robot is adaptively navigated to decelerate in the vicinity of objects and obstacles thus it can sufficiently spray and illuminate around objects, which assures objects to be fully disinfected against SARS-CoV-2. In addition, the path smoothing scheme based on the $\mathcal{B}$-spline curve is integrated with adaptive-speed navigation to generate a safer and smoother trajectory at a reasonable distance from the obstacle. Simulation and comparative studies prove the effectiveness of the proposed model, which can plan a reasonable and short trajectory with obstacle avoidance, and show better performance than other meta-heuristic optimization techniques.

Keywords: Covid-19 disinfection robot · Adaptive speed navigation · Path planning · Cuckoo search algorithm · Speed modulation

1 Introduction

Nowadays, mobile robots have been broadly used in many fields, such as medical services, material transportation, household services and exploration. Ultraviolet type-C irradiation (UV-C) disinfection robots and spray disinfection robots have been commonly deployed in hospitals and other public areas [1,2]. Robot path planning and navigation problems can be addressed by defining a path with obstacle avoidance from the robot's starting position to its target position according to certain evaluation criteria in the environment with obstacles.

X.-J. Liu et al. (Eds.): ICIRA 2021, LNAI 13015, pp. 123–134, 2021.
https://doi.org/10.1007/978-3-030-89134-3_12

To maximize the effect of UV-C irradiation disinfection and spray disinfection, Covid-19 disinfection robots are expected to autonomously adjust their speed to decelerate in the vicinity of obstacles, avoid collisions with obstacles, and accelerate in open areas. Many well-established methods have been proposed to resolve robot the navigation issue, such as sampling-based algorithm [3,12], graph-based method [4], learning-based model [5,12], neural networks [6,7], firework algorithm [8], ant colony optimization [9,14], particle swarm optimization [10], genetic algorithms [11], etc.

Chintam *et al.* [3] proposed a sampling-based algorithm framework that extends the rapidly exploring random tree (RRT) algorithm to plan trajectories with obstacle avoidance for mobile robots. A graph theory-based model has been proposed in [4], which decomposes the robot workspace into maps with multiple morphological layer sets so that the self-reconfigurable robot can be successfully navigated to perform a complete coverage task. Moussa *et al.* [5] proposed a learning-based real-time path planning model with virtual magnetic fields. Based on the bio-inspired neural network approach, Yang and Luo [6] proposed an efficient real-time robot coverage path planning model, which enables autonomous robots to avoid obstacles in dynamic environments. Then, Luo *et al.* [7] evolved the neural network approach to real-time multiple mobile intelligent agents formation and navigation.

Based on the optimization and search capabilities of evolutionary algorithms, researchers have recently explored many evolutionary computation approaches to solve robot path planning and navigation problems. For instance, a hybrid fireworks algorithm based on LIDAR-based local navigation algorithm was proposed in [8], which is able to plan reasonable and short trajectories in unstructured environments with obstacle avoidance. Lei *et al.* [9] proposed an ant colony optimization (ACO) combined with a graph representation model to navigate the robot under kinematics constraints. In [10], a couple of improved particle swarm optimization (PSO) algorithms are proposed to resolve local optima issues in basic PSO, which can successfully navigate the autonomous mobile robots in complex environments. Sarkar *et al.* [11] proposed a domain knowledge-based genetic algorithm integrated four operators based on the domain knowledge to search a trajectory with obstacle avoidance from the starting point to single or multiple goals. Some researchers integrate two or three algorithms together for the algorithm of robot navigation. Wang *et al.* [12] developed a novel learning enabled path planning approach Neural RRT*, which combines the convolutional neural network with the sampling-based algorithm RRT*. In [13], a two-layer algorithm based on ACO and tabu search is proposed, which uses a hierarchical and partitioned navigation method for coverage path planning. Chen *et al.* [14] suggested a hybrid structure, ACO-APF, which integrates the ant colony optimization (ACO) mechanism with the artificial potential field (APF) algorithm for unmanned surface vehicles path planning. Luo *et al.* proposed a hybrid approach that integrates the bio-inspired neural network model and a heuristic algorithm for intelligent mobile robot motion planning in an unknown environment.

There are merely a few studies on UV-C disinfection robot navigation. Onsite disinfection robot evaluation has been performed to measure doses of UV-C radiation. Although genetic algorithms (GA) and an adjustable artificial potential field (APF) as path planners were able to maximize the delivered UV-C dose. They can only traverse at a constant speed without considering the vicinity of objects [1]. Conroy *et al.* [2] proposed a waypoint-based Dijkstra path planner for ultraviolet light irradiation Covid-19 disinfection robot using traveling salesman problem (TSP). However, this navigation method fails to effectively cover entire field with a variety of paces. Overall, the above methods consider no robot speed in path planning. In real-world applications, intelligent mobile robots assume to move at variable speeds based on environmental information. The Covid-19 disinfection robot traverses at low speed in the vicinity of obstacles to adequately spray and illuminate around objects, while for efficiency, it should move at high speed in open areas.

An adaptive speed navigation approach of an autonomous Covid-19 disinfection robot in adaptive environment scenarios are developed in cooperation with a cuckoo search algorithm (CSA) in this paper. The smooth and safe trajectory is planned with more reasonable distance away from the obstacles by the proposed segmented cubic $\mathcal{B}$-spline curve-based smoothing paradigm. This paper is organized as follows. Adaptive speed navigation based on cuckoo search algorithms is proposed in Sect. 2. In Sect. 3, a smoothing scheme based on segmented cubic $\mathcal{B}$-spline curve is considered to generate safe and smooth trajectories away from obstacles. An adaptive speed approach is applied to decrease the odometry error and safely navigate the robots in the turning. Afterward, the simulation and comparative studies of cuckoo search algorithm path planning of the Covid-19 disinfection robot in various environments are described in Sect. 4. Finally, important properties of the proposed model are concluded and future work is directed in Sect. 5.

2 Developed Path Planning Algorithm for Disinfection

An improved cuckoo search method is integrated with a local search approach to perform disinfection robot navigation. Utilized the grid representation of the map, the weight of the adjacency matrix between each grid is updated according to environmental information to generate the shortest path.

2.1 Environment Modeling

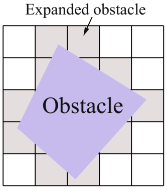
To achieve a high degree of robustness and autonomy in mobile Covid-19 disinfection robot navigation, environment modeling or map construction enables autonomous robots to generate trajectories with obstacle avoidance. In this paper, we consider 2D navigation in an environment with various obstacles. The grid map is composed of equal sized grid cells. The path planning becomes more accurate as the number of grids in the map increases; nevertheless, as the number of grids increases, the path planning takes more computational efforts. Thus, the grid of the map required for optimal path planning is defined by actual requirements. The environment is modeled as a matrix where element 0 represents open areas and element 1 represents obstacles. Each obstacle or dangerous region is represented as one or more grids, which is expanded based on the grid map as illustrated in Fig. 1. The trajectory is defined as the initial point $\mathcal{S}$, target point $\mathcal{T}$ and n waypoints between them.

$$\mathcal{P} = [\mathcal{S}, wp_1, wp_2, \ldots, wp_n, \mathcal{T}] \tag{1}$$

Each point is defined by its coordinates (x, y) of the grid, and the center of the grid pixel is regarded as the specific point. The sum of the Euclidean distance between two adjacent points on the trajectory is the length of the path:

$$L(\mathcal{P}) = \sum_{i=0}^{n} \sqrt{\left(x_{wp_{i+1}} - x_{wp_i}\right)^2 + \left(y_{wp_{i+1}} - y_{wp_i}\right)^2} \tag{2}$$

where x_{wp_0} denotes the initial point, and $x_{wp_{n+1}}$ depicts the goal.



$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Fig. 1. Grid-based map comprising of grids, each with free or occupied states

2.2 Cuckoo Search Algorithm

The Cuckoo Search algorithm (CSA) is a meta-heuristic swarm-based search algorithm inspired by the breeding behavior of the cuckoos in combination with the Lévy flight phenomenon in some birds and fruit flies [16]. The common cuckoo is an obligate brood parasite; it does not build its own nest, but lays its eggs in the nest of other species, leaving the host bird to take care of its eggs. For simplicity in describing the breeding behavior of cuckoo species, it could be conceptualized as some rules (please refer to [16]).

The Lévy flight pattern can be observed when some animals move and forage normally, accompanied by short-term movements in random directions. The Lévy flight process is essentially a model of random walk that has a power-law step length distribution with a heavy tail [16]. Let N be the population size, which represents the number of search agents in the workspace. Let D be the dimension of the variable, which denotes the number of desired waypoints to be generated in the workspace. A Lévy flight is achieved in Eq. (3) to create a new solution $Q^i(\tau + 1)$.

$$Q^i(\tau + 1) = Q^i(\tau) + \alpha \oplus \text{Lévy}(\mu) \tag{3}$$

where $Q^i(\tau)$ represents the position of the i-th nest at the τ-th iteration, and $Q^i(\tau + 1)$ is the new nest generated by Lévy flight. The $\oplus$ denotes dot product. α is a parameter defined in Eq. (4):

$$\alpha = \alpha_0 \times \left(Q^i(\tau) - Q_{\text{best}}\right) \tag{4}$$

where Q_{best} denotes the global best solution. α_0 is the scale factor, generally taking $\alpha_0 = 0.01$. Lévy denotes the process of the Lévy flight, which could be defined as

$$\text{Lévy}(\mu) = \frac{\eta}{|v|^{\frac{1}{\mu}}} \tag{5}$$

where η and v follow the normal distribution.

$$\eta \sim N\left(0, \sigma_\eta^2\right), \quad v \sim N\left(0, \sigma_v^2\right), \quad \sigma_v = 1$$

$$\sigma_\eta = \left(\frac{\varGamma(1+\mu) \times \sin\left(\frac{\pi \times \mu}{2}\right)}{\varGamma\left(\frac{1+\mu}{2}\right) \times \mu \times 2^{\frac{\mu-1}{2}}}\right)^{\frac{1}{\mu}} \tag{6}$$

where $\mu = 1.5$, $\varGamma$ denotes the Gamma function. In summary, we can combine the above equations to obtain the final expression of Lévy flight exploitation random walk as follows:

$$Q^i(\tau+1) = Q^i(\tau) + \alpha_0 \frac{\eta}{|v|^{\frac{1}{\mu}}} \left(Q^i(\tau) - Q_{\text{best}}\right) \tag{7}$$

In the exploration process, there should be a certain rate of random generation of new solutions, which ensures that the system will not fall into a local optimal state and provide fine diversity and exploratory properties in the entire search space. The probability p_a is introduced to abandon the worst nests and construct the new ones at new positions in light of Eq. (8).

$$Q^i(\tau+1) = \begin{cases} Q^i(\tau) + r\left(Q^j(\tau) - Q^k(\tau)\right), & p < p_a \\ Q^i(\tau), & p \geq p_a \end{cases} \tag{8}$$

where r and p are random variables uniformly distributed over [0,1]; p_a is a parameter that tunes the exploration and the exploitation of CSA ($0 \leq p_a \leq 1$); τ is current iteration number; $Q^j(\tau)$ and $Q^k(\tau)$ are the two randomly selected nest locations in the τ-th iteration.

2.3 Cuckoo Search Algorithm for Robot Path Planning

A trajectory to avoid obstacles while seeking the shortest distance to reduce energy consumption and improve efficiency should be formed. In this paper, we take advantage of the fast convergence characteristics of the CSA, combined with the local search process to rapidly search the shortest trajectory in the simplified grid map. Among them, the first step to find the optimal trajectory is to eliminate the infeasible trajectory with obstacle collision. In order to improve the performance of finding the best trajectory through the CSA algorithm, we first gradually construct a trajectory based on the random points generated by the algorithm (all in free space, outside the obstacle).

Dijkstra's algorithm is utilized to find the shortest trajectory in the graph. The trajectory is established from the starting point $\mathcal{S}$, and the next path point is selected from the points randomly generated by N. For selected points, the same process will be performed until the final goal is reached. When the connecting line passes through obstacles, we set it as an infeasible solution. By setting the distance between impassable nodes to infinite, the feasible solution is obtained, thereby generating the shortest trajectory. The advantage of the local search process used is that it can filter out the infeasibility between nodes, and the optimized path points obtained by the CSA algorithm can quickly generate the final trajectory. Through the algorithm of CSA, we obtain the waypoints in the map. Set the number D as the number of the generated waypoints, which is also the number of dimensions in the CSA algorithm. However, at the same time, it will consume more running time. Take advantage of local search based on Dijkstra's algorithm, our improved CSA algorithm can obtain a collision-free path accurately and efficiently. The algorithm of the CSA is identified in Algorithm 1.

Algorithm 1: Pseudo-code of Cuckoo Search Algorithm (CSA)

```
Parameter Initialization
Initialize the size of population $N$, the probability of replacing $p_a$, maximum
iteration time $T_{max}$
Population Initialization
Provide $N$ initial solutions in the workspace
Initialize the solution generated by Dijkstra Algorithm in the workspace
Evaluate the fitness value for all initial nodes
for $\tau = 1 : T_{max}$ do
    for $i = 1 : N$ do
        Generate new solution $Q^i(\tau+1)$ via Lévy flight based on Eq. (7);
        Evaluate its fitness value $f(P^i(t+1))$;
        if $f(Q^i(\tau+1)) < f(Q^i(\tau))$ then
            Replace $Q^i(\tau)$ by the new solution $Q^i(\tau+1)$;
        end
    end
    Replace the worst nodes with a probability $p_a$ with new ones in light of Eq.
    (8);
    Maintain the best solutions;
    Rank the solutions and find the current best;
end
Return the best solution and optimal fitness value;
```

3 Path Smoother with Adaptive Speed Robot Navigation

3.1 Segmented Cubic $\mathcal{B}$-spline Path Smoother

The smoothing scheme is utilized to smooth the trajectory in the vicinity of the turning point near the obstacles [17]. Unlike the traditional $\mathcal{B}$-spline curve, the segmented $\mathcal{B}$-spline curve merely smooths the trajectory at each corner separately. At the same time, the smoothed trajectory achieves continuous curvature to ensure that the disinfection robot reaches a precise trajectory and a smooth steering command for the robots. Therefore, a segmented cubic $\mathcal{B}$-spline curve is proposed to smooth the path.

A $\mathcal{B}$-spline curve can be represented by fundamental functions $\mathcal{N}_{i,k}(u)$, control points $\mathcal{P}_i$ and degree $(k-1)$, the equation is defined as

$$C(u) = \sum_{i=1}^{n+1} \mathcal{N}_{i,k}(u)\mathcal{P}_i \tag{9}$$

where $\mathcal{P}_i = [\mathcal{P}_{ix}, \mathcal{P}_{iy}]$ are the (n + 1) control points and a knot vector u. $\mathcal{N}_{i,k}(u)$ are the fundamental functions defined recursively:

$$\mathcal{N}_{i,k}(u) = \frac{(u - x_i)}{x_{i+k-1} - x_i}\mathcal{N}_{i,k-1}(u) + \frac{(x_{i+k} - u)}{x_{i+k} - x_{i+1}}\mathcal{N}_{i+1,k-1}(u) \tag{10}$$

$$\mathcal{N}_{i,k}(u) = \begin{cases} 1, \; u_i \leq u \leq u_{i+1} \\ 0, \quad \text{otherwise} \end{cases} ; u \in [0, 1] \tag{11}$$

where with the limitation condition of the $0/0 = 0$ for $k = 1$. The illustration of the G^2 $\mathcal{B}$-spline curve is shown in Fig. 2.

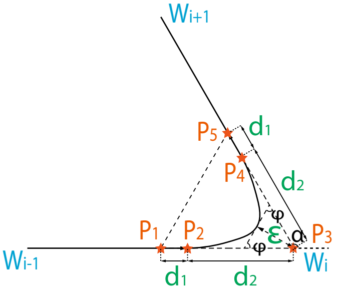


Fig. 2. Illustration of the G^2 $\mathcal{B}$-spline curve

To adapt to the parameterization difference, in the curve smoothing process, we use geometric continuity to evaluate the trajectory smoothness. G^2 continuity is a curve smoothing criteria based on the disinfection robot's kinematic. It can make an identical curvature vector and tangent unit at the intersection of two consecutive segments, avoiding the discontinuity of normal acceleration and result in a safe trajectory. Higher levels of continuity require more calculation resources, and the cubic $\mathcal{B}$-spline curve we utilized in this paper is due to the lowest degree of G^2 continuity. To make our smoothing method not affect the overall path trajectory, we insert a $\mathcal{B}$-spline curve into the existing straight trajectory and realize the continuity of G^2. In the smoothing process, the control points $\mathcal{P}_i$ of of $\mathcal{B}$-spline curve relative to the path point W_i is defined as

$$\begin{aligned}
\mathcal{P}_1 &= W_i - (1+c)d_2 u_{i-1} \\
\mathcal{P}_2 &= W_i - d_2 u_{i-1} \\
\mathcal{P}_3 &= W_i \\
\mathcal{P}_4 &= W_i + d_2 u_i \\
\mathcal{P}_5 &= W_i + (1+c)d_2 u_i
\end{aligned} \tag{12}$$

where c is smoothing length ratio $c = d_1/d_2$, u_{i-1} is the unit vector of line $W_{i-1}W_i$ and v_i is the unit vector of line W_iW_{i+1}. The sum of d_1 and d_2 is the smoothing length. $\phi = \alpha/2$ is half of the corner angle. If knot vector $[0,0,0,0,0.5,1,1,1,1]$ is defined, the smoothing error distance ε and the maximum curvature K_{max} of smooth trajectory may be obtained analytically as

$$\varepsilon = \frac{d_2 \sin(\varphi)}{2} \tag{13}$$

$$K_{\max} = \frac{4\sin(\varphi)}{3d_2\cos^2(\varphi)} \tag{14}$$

From (13) and (14), the smoothing error distance ε is defined by the existing maximum curvature K_{max} provided by the disinfection robots:

$$\varepsilon = \frac{d_2 \sin(\varphi)}{2} \tag{15}$$

3.2 Adaptive Speed Navigation

Adaptive speed navigation is designed to drive the disinfection robot at variable speeds to adapt to the surrounding environments. When the robot is approaching obstacles, especially when making a turn, it is better for the robot to slow down for precise navigation to avoid an accumulation of odometry errors due to instrument errors. In our model, the disinfection robot moves at a faster speed in an open area and slowly moves near an obstacle. Once the obstacle area (such

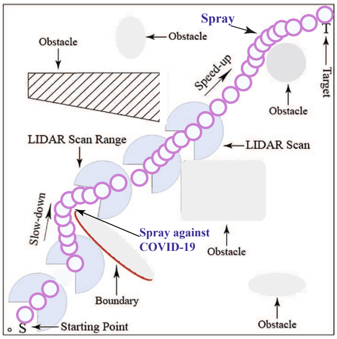


Fig. 3. Illustration of adaptive speed robot navigation

as the red border of the obstacle) is sensed by on-board LIDAR, the disinfection robot will decrement its speed near the obstacle (see Fig. 3). In the global path planning, when the robot enters the turning path generated by the path smoother, it starts to decelerate. The deceleration point adjusted during the turn is $\mathcal{P}_1$ in Fig. 2, and it continues to return to the original speed at $\mathcal{P}_5$.

4 Simulation and Comparative Studies

In order to validate our developed real-time adaptive speed-based CSA model, we compare our CSA method with other typical algorithms, multiple maps are used to carry out the simulation and comparative studies of the Covid-19 disinfection robot navigation.

Table 1. Comparison of trajectory length and number of turns

Methods	Minimum length	Increase of length (%)	Number of turns	Difference of turns (%)
Proposed CSA model	27.6709	———	5	———
SA-PSO [10]	28.7831	3.86	9	44.44
NLI-PSO [10]	30.3623	8.86	10	50.00
Basic PSO [10]	32.0153	13.57	13	61.53

4.1 Comparison of Our Adaptive Speed Algorithm with PSO Model

The developed cuckoo search algorithm associated with adaptive speed path smoother is used to compare with PSO algorithm, nonlinear inertia weight PSO (NLI-PSO) and simulated annealing PSO (SA-PSO) approaches, respectively. As we are aware, Nie *et al.* [10] suggested a hybrid PSO method to solve the issue of robot motion planning. However, their model has not yet considered the tuning curve and speed modulation in need of intelligent robot navigation systems. In this section, a comparative study is conducted to validate the effectiveness of the developed algorithm. The minimum trajectory length and number of turns of our model and others are compared as follows. The test scenario is based on Fig. 6 of [10], which is shown in Fig. 4 in this paper. The size of the workspace with a grid map is 20×20. The trajectory length generated by the proposed CSA model is 27.6709. The computed trajectory lengths for the trajectories in [10] and the produced values were 32.0153, 30.3623 and 28.7831 by PSO, nonlinear inertia weight PSO and simulated annealing PSO algorithms, respectively. The developed CSA algorithm found 13.57% less than PSO, 8.86% less than NLI-PSO and 3.86% less than SA-PSO, respectively. The number of turns of the developed CSA algorithm is 61.53% better than basic PSO, 50.00% less than NLI-PSO and 44.44% less than SA-PSO, respectively (Table 1). Then the segmented cubic $\mathcal{B}$-spline path smoother is used to the proposed CSA model. The different smooth trajectory obtained based on the constraints of the robot is illustrated in Fig. 5. The adaptive speed navigation is performed to slow down the robot in the dark red area as shown in Fig. 5.

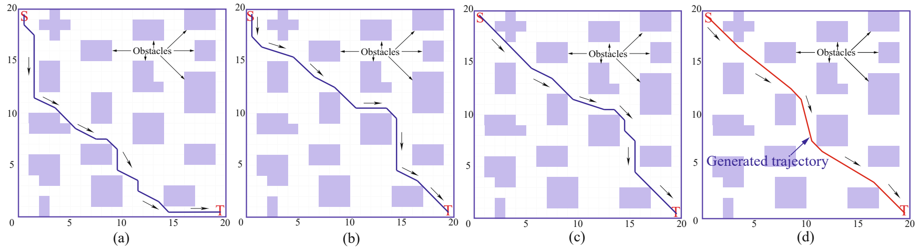


Fig. 4. Illustration of a variety of methods of navigation. (a) Basic PSO model [10]; (b) Nonlinear inertia weight PSO model (NLI-PSO) [10]; (c) Simulated annealing PSO model (SA-PSO) [10]; (d) The proposed CSA model.

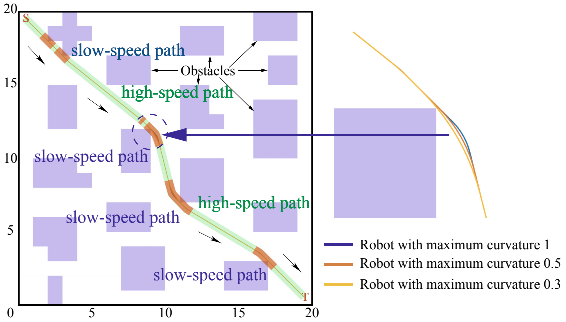


Fig. 5. Simulation results of the adaptive speed navigation with different robot tuning curvature constraint based on segmented cubic $\mathcal{B}$-spline path smoother. The light green represents the high speed areas whereas dark red represents the slow down areas. (Color figure online)

4.2 Comparison of the Proposed Adaptive Speed Model with Various Algorithms

We then apply the proposed model to room-like test scenarios. The shorter trajectory is generated by our CSA model in comparison with other commonly used path planning algorithms, Probabilistic Road Map (PRM), Rapidly exploring Random Tree* (RRT*) and Q-learning approaches as shown in Fig. 6(a). The path distance generated by our CSA model is 4.04% less than the PRM model, 5.97% less than the RRT* and 8.42% less than the Q-learning method, respectively. The number of turns of developed CSA approach is 15.38% better than the PRM, 76.08% less than RRT* and 8.33% less than Q-learning method, respectively. In Table 2, we observe that our model outperforms others in terms of path length and number of turns. We have the significant feature of adaptive speed navigation. In the vicinity of obstacles, the disinfection robots operate at a slow speed to adapt environment with the placement of obstacles, while the robot operates at high speed in free space as shown in Fig. 6(b).

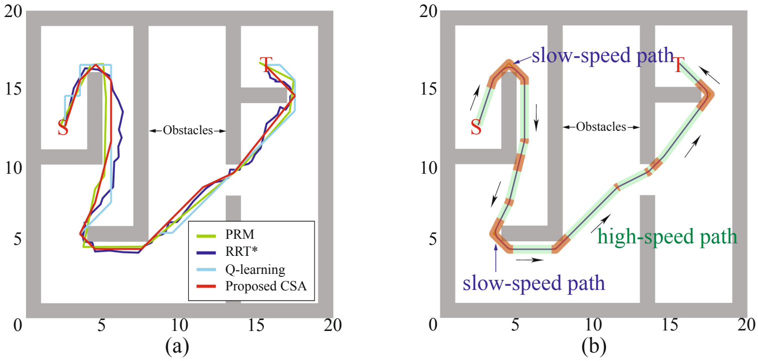


Fig. 6. (a) Comparison of disinfection robot navigation with various models. (b) Result of the adaptive speed navigation based on segmented cubic $\mathcal{B}$-spline path smoother.

Table 2. Comparison of trajectory length and number of turns

Methods	Minimum length	Increase of length	Number of turns	Difference of turns
Proposed CSA model	37.8997	———	11	———
PRM	39.4963	4.04%	13	15.38%
RRT*	40.3049	5.97%	46	76.08%
Q-learning	41.3848	8.42%	12	8.33%

5 Conclusion

In this paper, we have developed an efficient adaptive speed CSA algorithm. The proposed method combines a cuckoo search algorithm with a local search method based on a 2D grid-based map. For the process of approaching obstacles and turning, the smoothing scheme in light of segmented cubic $\mathcal{B}$-spline curve integrated with the method of adaptive speed navigation, so that the Covid-19 disinfection robot can plan a safer and smoother trajectory at a reasonable distance from objects and obstacles while safely and adaptively navigating the autonomous Covid-19 disinfection robot. Simulation and comparative studies have proved the efficiency and robustness of the developed CSA method.

References

1. Tiseni, L., Chiaradia, D., Gabardi, M., Solazzi, M., Leonardis, D., Frisoli, A.: UV-C mobile robots with optimized path planning: algorithm design and on-field measurements to improve surface disinfection against SARS-CoV-2. IEEE Robot. Autom. Mag. **28**(1), 59–70 (2021)
2. Conroy, J., et al.: Robot development and path planning for indoor ultraviolet light disinfection. arXiv preprint arXiv:2104.02913 (2021)
3. Chintam, P., Luo, C., Liu, L.: Advised RRT*: an optimal sampling space enabled bi-directional RRT*. Technical report, IEEE Robotics and Automation Letters (RA-L) (submitted)

4. Cheng, K.P., Mohan, R.E., Nhan, N.H.K., Le, A.V.: Graph theory-based approach to accomplish complete coverage path planning tasks for reconfigurable robots. IEEE Access **7**, 94642–94657 (2019)
5. Moussa, M., Beltrame, G.: Real-time path planning with virtual magnetic fields. IEEE Robot. Autom. Lett. **6**(2), 3279–3286 (2021)
6. Yang, S.X., Luo, C.: A neural network approach to complete coverage path planning. IEEE Trans. Syst. Man Cybern. Part B **34**(1), 718–725 (2004)
7. Luo, C., Yang, S.X., Li, X., Meng, M.Q.-H.: Neural dynamics driven complete area coverage navigation through cooperation of multiple mobile robots. IEEE Trans. Industr. Electron. **64**(1), 750–760 (2017)
8. Lei, T., Luo, C., Ball, J.E., Bi, Z.: A hybrid fireworks algorithm to navigation and mapping. In: Handbook of Research on Fireworks Algorithms and Swarm Intelligence, pp. 213–232. IGI Global (2019)
9. Lei, T., Luo, C., Ball, J.E., Rahimi, S.: A graph-based ant-like approach to optimal path planning. In: 2020 IEEE Congress on Evolutionary Computation, vol. 1, no. 6 (2020)
10. Nie, Z., Yang, X., Gao, S., Zheng, Y., Wang, J., Wang, Z.: Research on autonomous moving robot path planning based on improved particle swarm optimization. In: 2016 IEEE Congress on Evolutionary Computation, CEC, pp. 2532–2536 (2016)
11. Sarkar, R., Barman, D., Chowdhury, N.: Domain knowledge based genetic algorithms for mobile robot path planning having single and multiple targets. J. King Saud Univ. Comput. Inf. Sci. (2020)
12. Wang, J., Chi, W., Li, C., Wang, C., Meng, M.Q.-H.: Neural RRT*: learning-based optimal path planning. IEEE Trans. Autom. Sci. Eng. **17**(4), 1748–1758 (2020)
13. Wang, J., Chen, J., Cheng, S., Xie, Y.: Double heuristic optimization based on hierarchical partitioning for coverage path planning of robot mowers. In: 12th International Conference on Computational Intelligence and Security, pp. 186–189 (2016)
14. Chen, Y., Bai, G., Zhan, Y., Hu, X., Liu, J.: Path planning and obstacle avoiding of the USV based on improved ACO-APF hybrid algorithm with adaptive early-warning. IEEE Access **9**, 40728–40742 (2021)
15. Luo, C., Yang, S.X., Krishnan, M., Paulik, M.: An effective vector-driven biologically motivated neural network algorithm to real-time autonomous robot navigation. In: IEEE International Conference on Robotics and Automation, pp. 4094–4099 (2014)
16. Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. In: 2009 World Congress on Nature and Biologically Inspired Computing (NaBIC), pp. 210–214 (2009)
17. Farin, G., Rein, G., Sapidis, N., Worsey, A.J.: Fairing cubic B-spline curves. Comput. Aided Geom. Des. **4**(1–2), 91–103 (1987)
18. Huang, Y., Wang, P., Yuan, M., Jiang, M.: Path planning of mobile robots based on logarithmic function adaptive artificial fish swarm algorithm. In: 2017 36th Chinese Control Conference (CCC), pp. 4819–4823 (2017)