APEM journal
Advances in Production Engineering & Management
Volume 16 | Number 4 | December 2021 | pp 405–417
https://doi.org/10.14743/apem2021.4.409
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Optimal path planning of a disinfection mobile robot against COVID-19 in a ROS-based research platform

Banjanovic-Mehmedovic, L.[a], **Karabegovic, I.**[b,*], **Jahic, J.**[c], **Omercic, M.**[d]

[a]University of Tuzla, Faculty of Electrical Engineering, Tuzla, Bosnia and Herzegovina
[b]Academy of Sciences and Arts of Bosnia and Herzegovina, Sarajevo, Bosnia and Herzegovina
[c]University of Cambridge, Cambridge, United Kingdom
[d]iLogs Gmbh, Klagenfurt, Austria

A B S T R A C T

Due to COVID-19 pandemic, there is an increasing demand for mobile robots to substitute human in disinfection tasks. New generations of disinfection robots could be developed to navigate in high-risk, high-touch areas. Public spaces, such as airports, schools, malls, hospitals, workplaces and factories could benefit from robotic disinfection in terms of task accuracy, cost, and execution time. The aim of this work is to integrate and analyse the performance of Particle Swarm Optimization (PSO) algorithm, as global path planner, coupled with Dynamic Window Approach (DWA) for reactive collision avoidance using a ROS-based software prototyping tool. This paper introduces our solution – a SLAM (Simultaneous Localization and Mapping) and optimal path planning-based approach for performing autonomous indoor disinfection work. This ROS-based solution could be easily transferred to different hardware platforms to substitute human to conduct disinfection work in different real contaminated environments.

A R T I C L E   I N F O

Keywords:
Disinfection mobile robot;
COVID-19;
Optimal path planning;
Particle Swarm Optimization (PSO);
Simultaneous localization and mapping (SLAM);
Dynamic Window Approach (DWA);
Robot Operating System (ROS);
ROS-based platform

**Corresponding author:*
isak1910@hotmail.com
(Karabegovic, I.)

Article history:
Received 19 November 2021
Revised 27 November 2021
Accepted 28 November 2021

1. Introduction

Service mobile robots provide task services at different public places, e.g., in factories, airports, malls or hospitals. Because of similarities between operational environments in hospitals and manufacturing facilities, many service tasks are adapted in both environments [1]. We are currently witnessing a pandemic caused by the COVID-19 virus, so disinfection is a necessary task in many institutions. Using service disinfection robots is very attractive and advantageous for highly effective treatments, highly effective pathogen elimination, whole area treatment, disinfection of all surfaces in the area, higher productivity, etc.

Numerous design solutions for disinfection robots have been developed. Shen *et al.* [2], during the COVID-19 pandemic, compared over 200 robotic systems that could assist with Covid-related issues. The study concludes that robotics systems are overall suitable for dealing with COVID-19 related challenges, including disinfection. Robots for disinfection are mainly categorized [3] into UV light based and liquid agent spray solutions. Considering their mobility, they are often categorized [4] as mobile based, drones or semi-autonomous. After comparing several robots for disinfection, the survey concludes that liquid agent spray robots have a great advantage of being able to function 24/7, performing disinfection of target areas following a specific path. However, the survey notices that these robots may not be useful in areas with gaps or holes where is hard to reach by the liquid. According to the previously mentioned categorization, our solution fits into a category of mobile robots based on UV light disinfection. Even before the COVID 19 pandemic started, research conducted in robotics investigated questions that robotics solutions need to address to "effectively deploy robots in infectious diseases outbreaks" [5]. One of the three main research questions, relevant to our work, focuses on the interaction between robots and patients.

Recent research in the field of mobile robotics focusing on disinfection tackles problem related to planning and execution of navigation path, especially on optimization of the path considering navigation and disinfection target constraints. Shi *et al.* [6] approach the challenge of making mistakes in the path planning. They transform the collaborative scheduling problem of intelligent disinfection use to the distributed constraint optimization problem.

Tan *et al.* [7] introduced Aimi-Robot UVC robot that uses a graph-optimized SLAM algorithm to enable localization and map creation in the unknown environments. Aimi-Robot uses lidar sensors, gyroscopes, and odometers to obtain environmental information, and with the aforementioned SLAM algorithm it is able to increase real-time localization accuracy. Conte, Leamy, and Furukawa [8] presented an unmanned ground vehicle robot with map-based teleoperation that disinfects in complex indoor environments. The main contribution of this work is the two-stage mapping technique enabling teleoperation and allowing a human to operate the robot beyond the line-of-sight. The map dynamically constructs the map of 3D surfaces. To get a detailed map, Sayed *et al.* [9] presented a robot controlled by a joystick, which uses the Microsoft Kinect Xbox360 sensor. The navigation challenge is approached by using ROS navigation algorithms, SLAM, and machine vision. Marques *et al.* [10] suggest how to optimize a trajectory of a robot by building a probabilistic roadmap in the robot's configuration space. To deal with the fact that it is an NP-hard Mixed-Integer Linear Programming, they employ a two-stage solver that combines a Linear Program (LP) and a Travelling Salesman Problem (TSP). Conroy *et al.* [11] present a low-cost robotic platform for ultraviolet light disinfection, with off-the-shelf components and ROS navigation stack (standard mapping, control, and odometer packages), and use simulation to verify their concept. Ruan *et al.* [12] introduced a low-cost robot that combines the Hydrogen Peroxide Vaporous and SLAM for automated disinfection operation in the complex indoor environment. The robot builds a map of the environment using SLAM, powered by the ROS navigation stack, and then it follows a prescribed path. UltraBot robot [13] created for UV disinfection combines SLAM, Monte Carlo localization, and autonomous navigation in known environments for localization and path planning. This platform is a partially 3D printed and uses standard the ROS packages. These factors combined enable it to achieve high accuracy in localization.

Compared to the discussed robots, our solution combines Particle Swarm Optimization (PSO) algorithm with Dynamic Window Approach (DWA) for optimal path planning and obstacle avoidance. In addition, the suggested solution relies on ROS components. Instead of presenting a concrete solution, we present a concept that is easy to transfer to different hardware platform as a non-expensive and practical solution, and which is adjustable for specific applications in this domain.

The rest of this paper is organized as follows: Section 2 describes current planning methods used in the autonomous vehicles research field, with focus on the PSO-based optimization method; Section 3 introduces the ROS-based research platform and its modules related to the navigation; Section 4 introduces our solution- SLAM and path planning based approach for performing disinfection, which are implemented in ROS, while Section 5 describes the simulation setup and discussion. Finally, the last section contains the conclusions and future work.

2. Path planning methods

Service robots are supposed to navigate in crowded environments in a way that requires methods for motion planning and obstacle avoidance. The main problem in robot path planning is the capability of creating collision-free waypoints in the path to reach the destination point regarding to some optimization criteria such as the shortest distance or minimum time. Usually, the path planning module is divided into the global planner, which uses a priori information of the environment to create the best possible path and the local planner, which recalculates the initial plan to avoid possible dynamic obstacles [14].

Path planning can be classified into initial practical planners, grid-based algorithms, sampling-based planning (SBP) approaches, algorithms based on optimization and machine learning based algorithms [15].

Initial complete practical planners such as Road Map (RM), Potential Fields, and Cell Decomposition (CD) techniques are unable to deal with dynamic and complex high dimension problems [16].

Grid-based algorithms transform the environment in a grid-mesh. Dijkstra's algorithm is always able to find the shortest path between two points, but it has very high computational complexity. Many algorithms have been created which are able to find the shortest path with the lowest computational cost: A*, Bi A*, Breadth-first, Best-First path planning [17]. The A* uses heuristics to be faster. The Bidirectional A* algorithm is a graph search algorithm that finds the shortest path from an initial vertex to a goal vertex in a directed graph running two simultaneous searches. The Breadth-first Search is a classic graph search algorithm, which works by expanding and systematically exploring a given node and progressively redoing the same procedure for all its neighbours. The Best-First algorithm is one of the most popular in the literature. The Best-First algorithm uses the given heuristic function, which is applied equally through the search space, to quantify the value of each candidate exploited during the process and, thus, continues the exploration until reaches the point of interest. These algorithms, however, become extremely costly when applied to large environments or environments with dynamic objects [17].

Sampling Based Planning (SBP) approaches are the most influential approaches in path planning. SBPs are probabilistically complete, i.e., it finds a solution, if one exists, provided with infinite run time. The most popular SBP algorithms are Probabilistic Roadmap (PRM), Rapidly-exploring Random Tree (RRT) and Rapidly-exploring Random Tree Star (RRT*). PRM based methods are mostly used in highly structured static environments such as factory floors. RRT and RRT* based approaches extend non-holonomic constraints and support dynamic environment as well. The major advantages of SBP algorithms are low computational cost, applicability to high dimensional and complex problems [18].

Evolutionary Algorithms (EA) are based on the principle of natural evolution, where randomly select a candidate set of solutions and apply the quality function as an abstract fitness measure. The evolutionary algorithms tend to find an optimal solution by converging from the initial state to the global optimal using a fitness function. A few of optimization techniques based on nature-inspired optimization heuristics are Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization Algorithm (ACO), Artificial Bee Colony (ABC), Cuccko Search (CS). The Genetic algorithm is based on a direct analogy to Darwinian natural selection and mutations in biological reproduction. ACO is inspired by the behaviour of real ant colonies and PSO is used for optimizing continuous nonlinear functions and performs a parallel search in a space of solutions. The PSO has many advantages compared to other evolutionary methods: it has few parameters, small population and fast convergence [19].

Recently, many researchers have tried to solve path planning problems using machine learning [15]. The classical Q-Learning algorithm is improved using deep learning and this combination gives very good results.

PSO algorithm

The PSO is inspired by the social behaviour of a flock of migrating birds trying to reach an unknown destination [20]. In the PSO algorithm, a candidate solution is presented as a particle. The algorithm utilizes a collection of flying particles in a search space (current and possible solutions) and moves towards a promising area to get to a global optimum [21]. A given number of state space vectors (particles)are initialized randomly. Particles fly through the search space at predefined velocities, which are dynamically adjusted according to its own previous best value pbest$_i$and its previous best group's *gbest* [22-25]. The performance of each particle is measured according to a known fitness function, which is related to the problem to be solved.

The best solutions are evolved through several generations. Each particle updates its position and velocity as follows:

$$X_i^t = X_i^t + V_i^{t+1} \tag{1}$$

$$V_i^{t+1} = \omega V_i^{t+1} + c_1 r_1 \left(pbest_i - X_i^t\right) + c_2 r_2 \left(gbest - X_i^t\right) \tag{2}$$

where V_i^{t+1} is the velocity of the i-th particle at time t + 1, X_i^t is the position of particle at time t; c_1 and c_2 are accelaration coefficients, which denote the cognitive and social parameters, respectivelyand are usually set to a value of 2, although good results have been also produced with $c1$ = $c2$ = 4; r_1 and r_2 are two random functions in the range [0,1]. The inertial weight ω is used for regulating the global exploration and local exploration abilities. It presents the global search behaviour, set to a large value in the beginning of the searching process and dynamically reduced during the optimization (which emulates a more local search behaviour). Its range is suggested to be $0.2 \leq w \leq 0.4$. All particles move to the global best solution to finish the search process [24].

3. ROS-based research platform

3.1 General information about ROS

Robot Operating System (ROS) [26] is a framework that facilitates development of robotic applications. It is organised around the idea of common data structures for standard robotic operations, and it comprises a large database of libraries and tools. In its current version ROS2, it supports a significant number of robots through abstractions usually expected from an operating system (e.g., support with abstraction of hardware to software). However, ROS is a meta operating system as it runs on top of a real operating system (Linux, macOS, or Windows 10). ROS relies on distributed communication between nodes and is hence suitable for controlling larger robotics environments, involving other robots and large number of sensors on-robot and off-robot sensors. Projects of ROS are organised into stacks, where the ROS navigation stack [33] is one of the most prominent projects.

ROS2, officially released in 2017, in addition to features supported in ROS1, also supports C++17 standard and works with Python version 3.5 (where ROS1 supports C++03 and C++11 and works with Python 2). In addition to these, ROS2 provides various Quality of Service policies which improve communication over different networks.

One of the most used simulators with ROS is the Gazebo simulator [27, 34]. The focus of Gazebo is to simulate physics of robotics, support rendering, offer communication interfaces and a user interface. The simulator offers possibilities for both indoor and outdoor navigation modelling and supports robot models with different level of details (from simple platforms to robots with full sensor suite). Some of the dynamic operations that the simulator supports include motion, constraints, collision, and contact friction. It also has support for various sensors, including cameras, Kinect, lasers, RFID, etc. The Gazebo ROS package provides interface between the Gazebo and ROS framework.

Another simulator often used with ROS is SVL Simulator [35]. It is a simulation platform used for autonomous vehicles. Applications of the SVL simulator include warehouse robotics, autonomous racing, sensor/sensor systems development and marketing, real-time embedded systems for automotive, etc. Commonly supported scenarios are complex traffic scenarios, testing localization module in new Digital Twin environments, testing autonomous vehicle stack in real, etc. SVL supports running ROS2 simulations with its ROS navigation stack.

Rviz [36] is a 3D visualization tool commonly used in ROS. It enables visualisation of data, and is most commonly used to show sensor data and dynamic states of different processes. Typical examples of data visualisation using Rviz include data from laser sensors, camera, odometry, etc. It has several built-in data types including camera, image, map, point, path, etc.

3.2 ROS navigation stack and Nav2

ROS navigation [37] stack performs 2D navigation operations. As input, it takes odometry, sensor streams, and a goal pose. It outputs velocity commands that are sent to a mobile base.

Nav2 project [33] is a successor of the ROS navigation stack to be used in ROS2. Nav2, among others, contains tools to load, serve, and store maps (Map Server), localize the robot on the map (AMCL), plan a path from A to B around obstacles (Nav2 Planner), control the robot as it follows the path (Nav2 Controller), convert sensor data into a Costmap representation of the world (Nav2 Costmap 2D), compute recovery behaviours in case of failure (Nav2 Recoveries), follow sequential waypoints (Nav2 Waypoint Follower), and plugins to enable custom algorithms and behaviours (Nav2 Core), Fig. 1.

4. SLAM and path planning in ROS

The disinfection mobile robot first builds the map of the designated environment based on Simultaneous localization and mapping (SLAM). Then, it accomplishes the disinfection mission of the environment according to the selected path avoiding obstacles during this process. Our solution combines Particle Swarm Optimization (PSO) algorithm with Dynamic Window Approach (DWA) for global path planning and obstacle avoidance, respectively. The suggested concept of our approach is presented in Fig. 2.

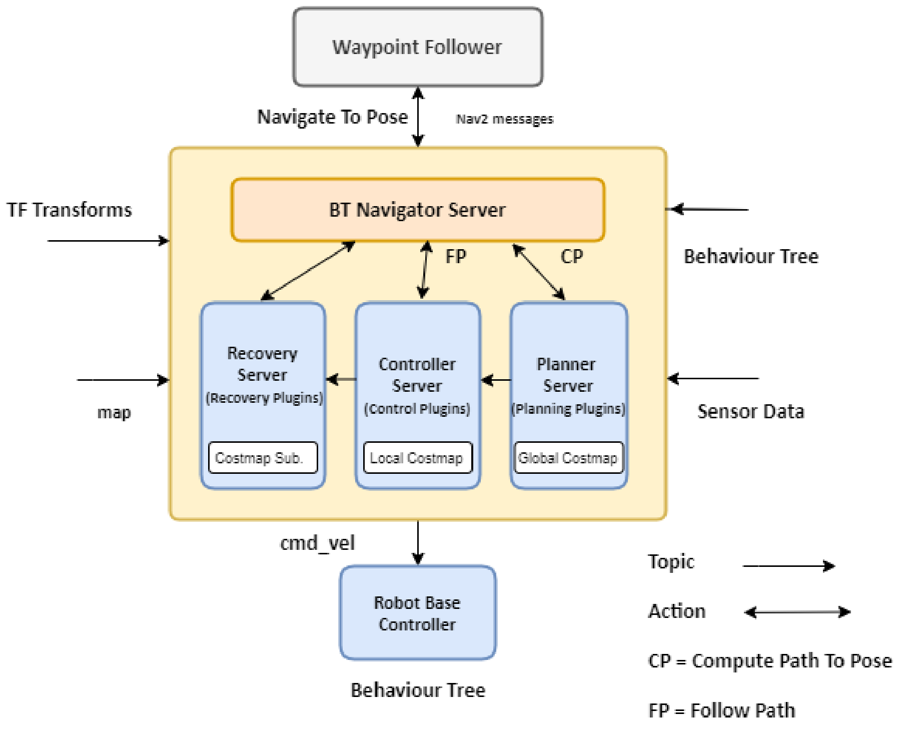


Fig. 1 ROS Nav2 stack

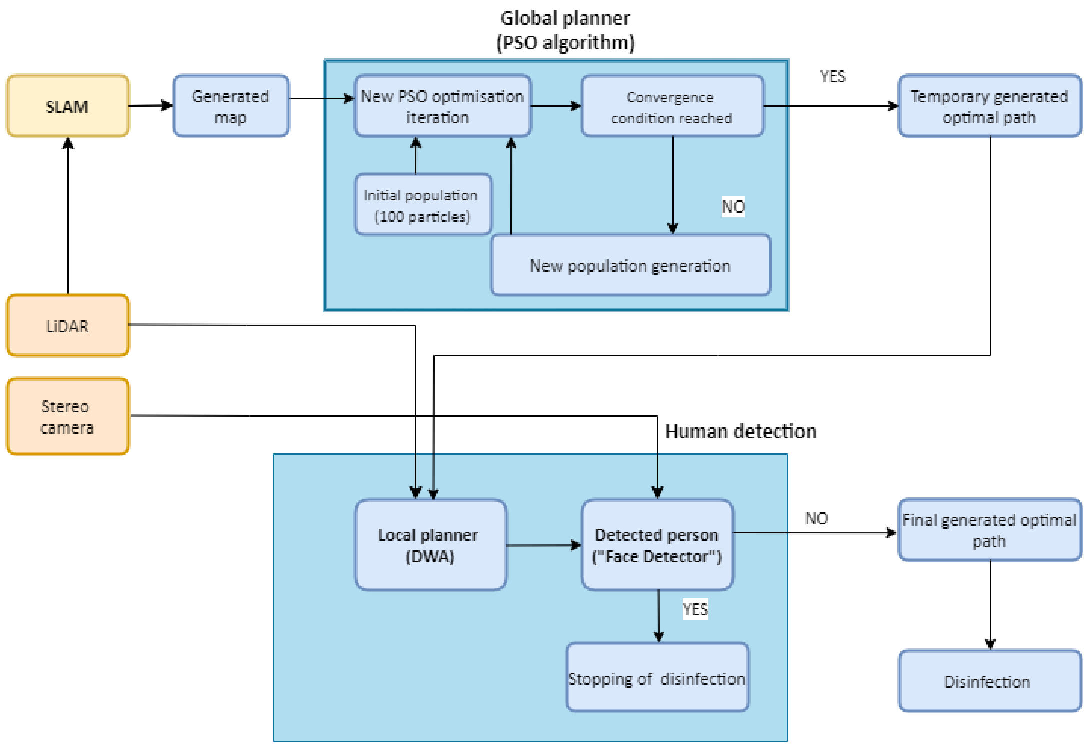

Global planner
(PSO algorithm)
SLAM
Generated map
New PSO optimisation iteration
Convergence condition reached
YES
Temporary generated optimal path
Initial population (100 particles)
NO
New population generation
LiDAR
Stereo camera
Human detection
Local planner (DWA)
Detected person ("Face Detector")
NO
Final generated optimal path
YES
Stopping of disinfection
Disinfection

Fig. 2 The suggested concept of the disinfection mobile robot relies on ROS components

4.1 SLAM implementation in ROS

For the proposed concept of Optimal path planning of the disinfection mobile robot, we used the Gmapping SLAM algorithm due to its robustness in indoor environments and limited computational burden.

Simultaneous localization and mapping (SLAM) is a concept concerned with the problem of building a map of an unknown environment by a mobile robot while at the same time enables localization and navigation through the environment using the map [28]. The main function of SLAM is to analyze the input data to determine the pose of the robot and build an environment map in order for the robot to move autonomously.

The SLAM problem could be solved using filtering or smoothing approaches. The main filtering SLAM techniques are Kalman filters (EKF, UKF) and particle filters, and they are designed as on-line approaches. The smoothing approaches like the graph SLAM, estimate the full robot trajectory by processing the full set of the sensor measurements and they are classified as the full SLAM problem.

The SLAM could also be classified into 2D and 3D SLAM. The most common 2D-lidar SLAM algorithms in ROS include Gmapping, Cartographer, Hector, Core, Lago. Gmapping is a SLAM algorithm based on 2D-lidar using Rao-Blackwellized Particle Filters (RBPF) to build the 2D grid map. It decomposes the SLAM problem into two parts: localization and mapping through the conditional joint distribution [29].

In recent years, the machine learning and deep learning techniques have been involved in research regarding SLAM to effectively reduce the positioning error, achieve high-precision tracking detection, and improve the accuracy of robot target detection [28, 30].

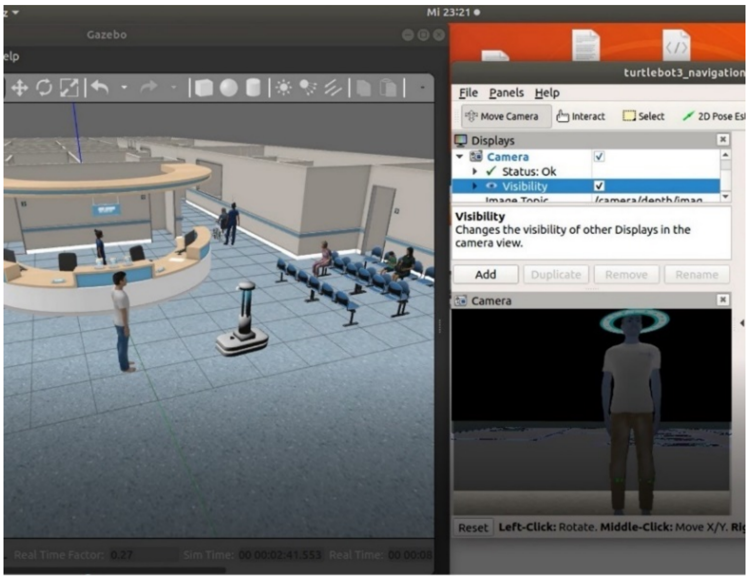
4.2 PSO based global path planner

After building the map, it is necessary to solve the path planning problem to enable the robot to perform multi-point disinfection at a specified location in the generated 2D map. The path planner could be classified into two categories: the global path planner and the local planner. We selected the PSO algorithm as the global path planner in our approach.

The PSO algorithm searches for the best values by performing a simulation for each particle in the population at the given environmental data. The number of particles is generated around the robot's initial position and within its sensing range. Each particle takes a new velocity and position based on the constantly updated PSO equations. A candidate for the robot's next position is determined by the position of the best particle, i.e., the one nearest to the goal. When the criteria for the PSO exiting are satisfied, a raw trajectory is obtained and postprocessed to generate the optimized trajectory.

In case of obstacles, which are suddenly appearing on the pre-planned path, the local path planner performs partial secondary path planning to avoid obstacles and to come to the target position and the disinfection process is done, Fig. 2. In case of people detection, where the ROS package for face recognition "Face Detector" from a stereo camera data is used, the robot must stop its disinfection process, Fig. 3. The face detector employs the OpenCV face detector, based on a cascade of Haar-like features to obtain an initial set of detections [38].

To disinfect the space as efficiently as possible, it is necessary to select points in the space from which the largest possible area of the room could be covered, Fig. 4.



Visibility
Changes the visibility of other Displays in the camera view.

Left-Click: Rotate. Middle-Click: Move X/Y.

Fig. 3 ROS package "Face Detector" for people detection

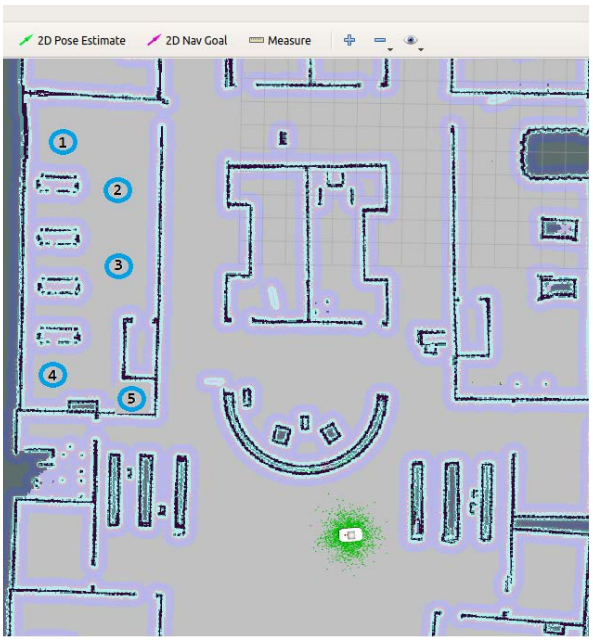


Fig. 4 Selected points in the mapped robot environment

Most path planners aim to generate an optimal path considering a single objective like path length or path travel time. However, in practice, path planning in dynamic environments emphasizes the necessity of considering multiple objects in path planning [21]. The used fitness function is generated as the weighted sum of the three objectives: the path shortness fitness function F_{sh}, the path smoothness fitness function F_{sm}. The shortest path is defined as the Euclidean distance between each newly generated particle and the goal in each iteration, and the smoothest path is defined as the robot moving angle in two successive iterations.

$$F = \alpha_1 \cdot F_{sh} + \alpha_2 \cdot F_{sm} \tag{3}$$

$$F_{sh} = \sqrt{\left(px_i^t - x_G\right)^2 + \left(py_i^t - y_G\right)^2} \tag{4}$$

$$F_{sh} = \cos^{-1} \frac{\left(\left(px_i^t - x_G\right) \cdot (x_B - x_G)\right) + \left(\left(py_i^t - y_G\right) \cdot (y_B - y_G)\right)}{\sqrt{\left(\left(px_i^t - x_G\right)^2 + \left(py_i^t - y_G\right)^2\right)} \cdot \sqrt{((x_B - x_G)^2 + (y_B - y_G)^2)}} \tag{5}$$

where (px_i^t, py_i^t) is the position of i-th particle in iteration t, (x_G, y_G) is the goal position and (x_B, y_B) is the best position for the swarm. The weights of the shortest and smoothest fitness functions, α_1 and α_2 respectively, are in interval $0 \leq \alpha_1 + \alpha_2 \leq 1$.

The suitable path is obtained by minimizing this function F according to the assigned weights of each criterion.

4.3 DWA local planner

Dynamic Window Approach (DWA) to reactive collision avoidance which exists as ROS robot local navigation packet employed as the local planner in our research. Using a map, the planner creates a kinematic trajectory for the robot to get from a start to a goal location. The advantage of this approach is that reduces the search space to the dynamic window, which consists of the velocities reachable within a short time interval and which yield a trajectory on which the robot is able to stop safely [31]. Each simulated trajectory can be evaluated using an objective function that incorporates characteristics such as: proximity to obstacles, proximity to the goal, proximity to the global path:

$$F = \alpha_1 \cdot F_{ep} + \alpha_2 \cdot F_{lg} + \alpha_3 \cdot F_{oc} \tag{6}$$

where, F_{ep} denotes the distance to the path from the endpoint of the trajectory in meters, F_{lg} denotes the distance to the local goal from the endpoint of the trajectory in meters, and F_{oc} denotes the maximum obstacle cost along the trajectory [32]. α_i are the weighted coefficients of the related terms.

This ROS controller serves to connect the path planner to the robot and send the velocity value gained by maximizing an objective function. DWA is used here because its calculation time is short, the local path can be updated in real time and there is no dead angle obstacle or local optimization in the disinfection scenes [12].

5. Simulation and results

To analyse the function of the proposed PSO path planner algorithm, we ran several simulations in the ROS environment. The number of iterations, the particle number and the particle velocity are important factors influencing the performance of the PSO algorithm. The approach used in this simulation is varies those parameters and compares mobile robot trajectories and fitness function values. The same start and goal coordinates are used for all parameter values and the generated trajectories. Based on the results from these simulations, we tuned the algorithm parameters to optimal values.

The weights α_1 and α_2 are tuned through extensive simulation and try and errors, with the best-found values α_1 = 0.2 and α_2= 0.8. As a competing method for comparing the PSO based global path planner using different weights α_1 and α_2, the efficient standard Dijkstra's and Astar algorithms were selected, Fig. 5. and Fig. 6, respectively.

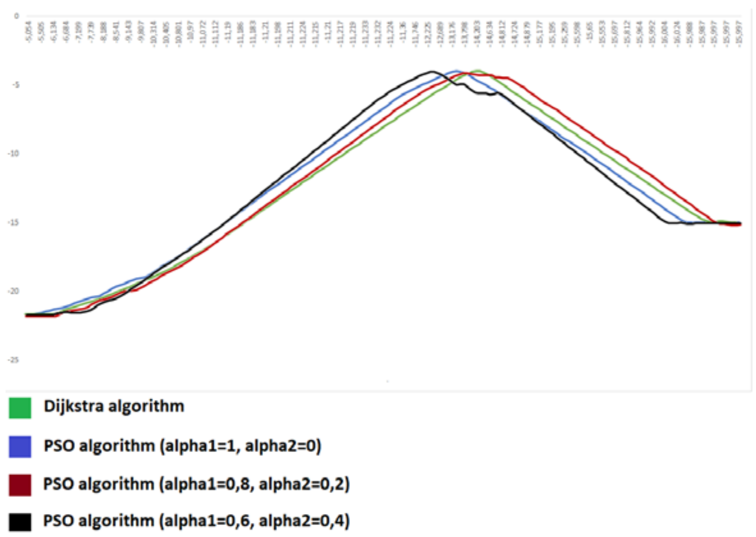


Fig. 5 Comparison of PSO global path planner of the mobile robot using different weights α_1 and α_2, with the efficient standard Dijkstra's algorithm

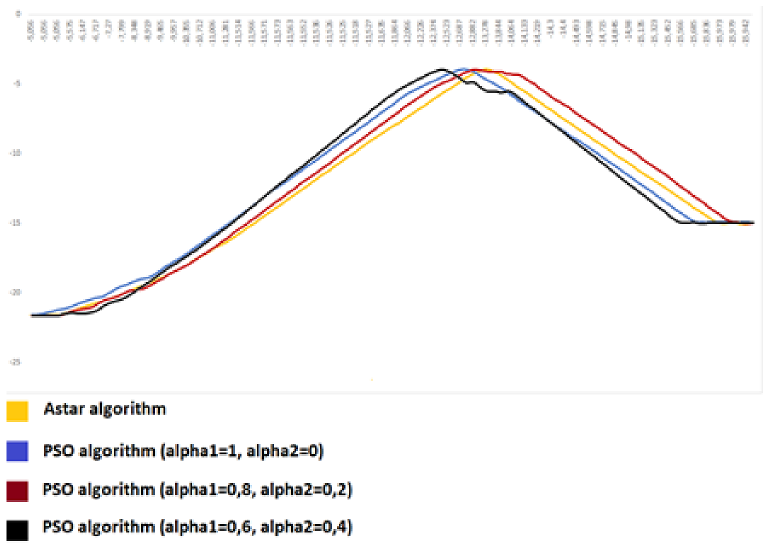
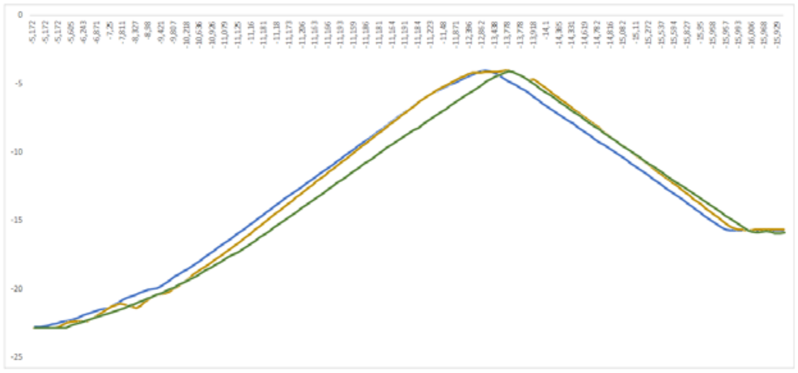


Fig. 6 Comparison of PSO global path planner of the mobile robot using different weights α_1 and α_2, with the efficient standard Astar method

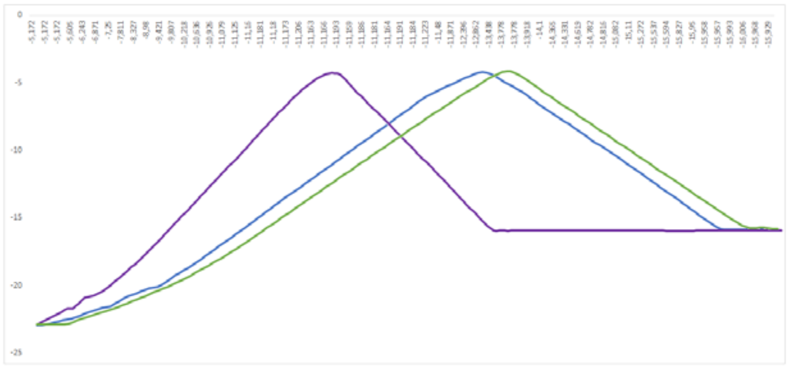
The results show that Dijkstra's and Astar algorithms generate almost identical paths. The PSO based path planner with weights $\alpha_1 = 0.2$ and $\alpha_2 = 0.8$ is the closest to the path generated by the Dijkstra's or the Astar algorithm. This is the path that is closest to the ideal path of the robot from the start to the end point, so this choice of coefficients is the best in those experiments. Variations can be made by reducing/increasing the number of iterations per calculation. In this experiment, the number of iterations was initially set to 25. Since the global planner performs a new plan calculation every 2 seconds, the 25-iteration PSO algorithm could not give up-to-date results and forward to the global planner, so it often happened that the robot deviated from the calculated path. Increasing the number of iterations results in greater accuracy in calculating the ideal path but requires more computing resources at the same time and results in a slower calculation. Fig. 7 presents the robot path with 2 iterations per calculation and the path with 5 iterations per calculation, compared to the path generated with the Astar algorithm. Both PSO paths have an initial variation in motion, because the robot loses the path for a short period of time until the calculation is performed. The deviation is larger when moving with 5 iterations per calculation because more time is required for the calculation. Therefore, the number of iterations was reduced to 2 iterations. The Fig. 8 presents comparison of path planning results, where the number of particles for the PSO algorithm was changed from 100 particles to 50 particles.


🟦 PSO algorithm (2 iterations per calculation)

🟫 PSO algorithm (5 iterations per calculation)

🟩 AStar algorithm

Fig. 7 Comparison of PSO path planning from the start to the end point for different values of the number of iterations with the Astar method


🟦 PSO algorithm (particle count = 100)

🟪 PSO algorithm (particle count = 50)

🟩 AStar algorithm

Fig. 8 Comparison of path planning produced by PSO algorithm for different values of the number of particles and the Astar method

A few graphical results of running DWA as the local path planner and PSO as the global path planner on problems with and without people as obstacles are illustrated consecutively in Figs. 9 and 10. Implementation of the PSO path planner with disinfection in the selected environment is presented in Fig. 9. A scenario where the mobile robot enters the room but recognizes people in the room and interrupts the disinfection process is presented in Fig. 10.

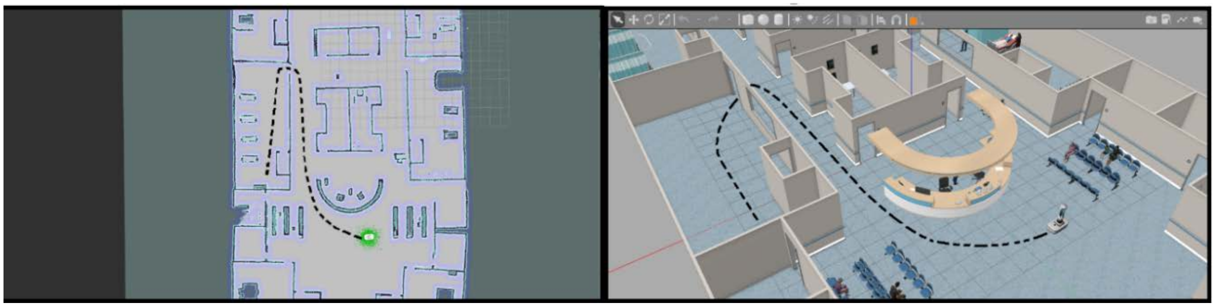


Fig. 9 Optimal path planning based on PSO with the finished disinfection task.

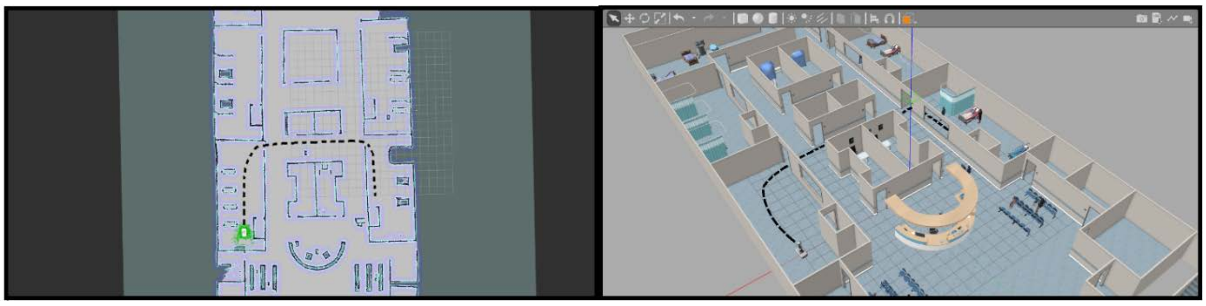


Fig. 10 Optimal path planning based on PSO with the cancelled disinfection task

6. Conclusion

In this paper, we present the DWA and PSO based path planning of an autonomous mobile disinfection robot based on Gmapping SLAM. The optimal path planning of the disinfection robot system was tested in the ROS indoor environment. By comparing PSO with the Dijkstra's and the Astar algorithms, it can be concluded that it is possible to easily improve path planning performance using PSO by adjusting parameters such as particle number, number of iterations, or fitness function coefficients. The simulation results show that the proposed method is effective as optimal path planner in combating COVID-19. In the future, more detailed work needs to be done in multi-criteria optimization of path planning combined with a disinfection model and people detection using deep learning. We are also planning to design and implement a disinfection mobile robot structure based on the discussed concepts.

Acknowledgement

This research is supported in scope of „Project Investigation of the optimal path of a mobile robot for Corona virus space disinfection", funded by Ministry of Education, Science and Youth of Sarajevo Canton, Bosnia and Herzegovina, 2020-2021.

References

[1] Ozkil, A.G., Fan, Z., Dawids, S., Aanes, H., Kristensen, J.K., Christensen, K.H. (2009). Service robots for hospitals: A case study of transportation tasks in a hospital, In: *Proceedings of 2009 IEEE International Conference on Automation and Logistics,* Shenyang, China, 289-294, doi: 10.1109/ICAL.2009.5262912.

[2] Shen, Y., Guo, D., Long, F., Mateos, L.A., Ding, H., Xiu, Z., Hellman, R.B., King, A., Chen, S., Zhang, C., Tan, H. (2021). Robots under COVID-19 pandemic: A comprehensive survey, *IEEE Access,* Vol. 9, 1590-1615, doi: 10.1109/ ACCESS.2020.3045792.

[3] Marsh, A. (2020). We've been killing deadly germs with UV light for more than a century, *IEEE Spectrum*, from *https://spectrum.ieee.org/tech-history/dawn-of-electronics/weve-been-killing-deadly-germs-with-uv-light-for-more-than-a-century*, accessed November 11, 2021.

[4] Moore, K.S. (2020). Flight of the GermFalcon: How a potential coronavirus-killing airplane sterilizer was born, *IEEE Spectrum*, from *https://spectrum.ieee.org/germfalcon-coronavirus-airplane-ultraviolet-sterilizer-news*, accessed November 11, 2021.

[5] Kraft, K. (2016). Robots against infectious diseases, In: *Proceedings of 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI),* Christchurch, New Zealand, 627-628, doi: 10.1109/HRI.2016.7451889.

[6] Shi, M., Yang, H., Liao, X., Chen, Y., Xiao, S., Wu, J. (2021). Research on strategy of intelligent disinfection robot based on distributed constraint optimization, In: *Proceedings of 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA),* Dalian, China, 82-85, doi: 10.1109/ICAICA52286.2021.9497927.

[7] Tan, X., Zhang, H., Zhou, X., Zhong, H., Liu, L. (2021). Research on graph-based SLAM for UVC disinfection robot, In: *Proceedings of 2021 IEEE International Conference on Real-time Computing and Robotics (RCAR),* Xining, China, 1064-1069, doi: 10.1109/RCAR52367.2021.9517506.

[8] Conte, D., Leamy, S., Furukawa, T. (2020). Design and map-based teleoperation of a robot for disinfection of COVID-19 in complex indoor environments, In: *Proceedings of 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR),* Virtual event, 276-282, doi: 10.1109/SSRR50563.2020.9292625.

[9] Sayed, A.S., Ammar, H.H., Shalaby, R. (2020). Centralized multi-agent mobile robots SLAM and navigation for COVID-19 field hospitals, In: *Proceedings of 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES),* Giza, Egypt, 444-449, doi: 10.1109/NILES50944.2020.9257919.

[10] Correia Marques, J.M., Ramalingam, R., Pan, Z., Hauser, K. (2021). Optimized coverage planning for UV surface disinfection, In: *Proceedings of 2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, 9731-9737, doi: 10.1109/ICRA48506.2021.9561032.

[11] Conroy, J., Thierauf, C., Rule, P., Krause, E., Akitaya, H., Gonczi, A., Korman, M., Scheutz, M. (2021). Robot development and path planning for indoor ultraviolet light disinfection, In: *Proceedings of 2021 IEEE International Conference on Robotics and Automation (ICRA),* Xi'an, China, 7795-7801, doi: 10.1109/ICRA48506.2021.9561405.

[12] Ruan, K., Wu, Z., Chio, I., Zhang, Y., Xu, Q. (2021). Design and development of a new autonomous disinfection robot combating COVID-19 pandemic, In: *Proceedings of 2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM),* Chongqing, China, 803-808, doi: 10.1109/ICARM52023.2021.9536167.

[13] Perminov, S., Mikhailovskiy, N., Sedunin, A., Okunevich, I, Kalinov, I., Kurenkov, M., Tsetserukou, D. (2021). UltraBot: Autonomous mobile robot for indoor UV-C disinfection, In: *Proceedings of 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE),* Lyon, France, 2147-2152, doi: 10.1109/CASE49439.2021.9551413.

[14] Marin-Plaza, P., Hussein, A., Martin, D., De la Escalera, A. (2018). Global and local path planning study in a ROS-based research platform for autonomous vehicles, *Journal of Advanced Transportation,* Vol. 2018, Article ID 6392697, doi: 10.1155/2018/6392697.

[15] Gao, P., Liu, Z., Wu, Z., Wang, D. (2019). A global path planning algorithm for robots using reinforcement learning, In: *Proceeding of 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO),* Dali, China, 1693-1698, doi: 10.1109/ROBIO49542.2019.8961753.

[16] Khan, A., Noreen, I., Habib, Z. (2017). On complete coverage path planning algorithms for non-holonomic mobile robots: Survey and challenges, *Journal of Information Science and Engineering*, Vol. 33, 101-121, doi: 10.6688/JISE.2017.33.1.7.

[17] Ferreira, J., Júnior, A.A.F., Galvão, Y.M., Barros, P., Fernandes, S.M.M., Fernandes, B.J.T. (2020). Performance improvement of path planning algorithms with deep learning encoder model, In: *Proceedings of* 2020 *Joint IEEE 10th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, Valparaiso, Chile, 1-6, doi: 10.1109/ICDL-EpiRob48136.2020.9278050.

[18] Noreen, I., Khan, A., Habib, Z. (2017). Optimal path planning using RRT* based approaches: A survey and future directions, *International Journal of Advanced Computer Science and Applications,* Vol. 7, No. 11, 97-107, doi: 10.14569/IJACSA.2016.071114.

[19] Robinson, J., Sinton, S., Rahmat-Samii, Y. (2002). Particle swarm, genetic algorithm, and their hybrids: Optimization of a profiled corrugated horn antenna, In: *Proceedings of IEEE Antennas and Propagation Society International Symposium (IEEE Cat. No.02CH37313),* San Antonio, USA, 314-317.

[20] Sulistijono, I.A., Kubota, N. (2006). A comparison of particle swarm optimization and genetic algorithm for human head tracking, In: *Proceedings of Joint 3rd International Conference on Soft Computing and Intelligent Systems and 7th International Symposium on advanced Intelligent Systems*, 2204-2209. doi: 10.14864/softscis.2006.0.2204.0.

[21] Masehian, E., Sedighizadeh, D. (2010). A multi-objective PSO-based algorithm for robot path planning, In: *Proceedings of 2010 IEEE International Conference on Industrial Technology,* Via del Mar, Chile, 465-470, doi: 10.1109/ICIT.2010.5472755.

[22] Xu, W., Yin, Y. (2018). Functional objectives decision-making of discrete manufacturing system based on integrated ant colony optimization and particle swarm optimization approach, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 389-404, doi: 10.14743/apem2018.4.298.

[23] Banjanovic-Mehmedovic, L., Baluković, A. (2020). PSO optimized fuzzy controller for mobile robot path tracking, In: Karabegović, I. (ed.), *New technologies, development and application III, NT 2020, Lecture notes in networks and systems*, Springer, Switzerland AG, Vol. 128, 413-421, doi: 10.1007/978-3-030-46817-0_47.

[24] Yu, M.R., Yang, B., Chen, Y. (2018). Dynamic integration of process planning and scheduling using a discrete particle swarm optimization algorithm, *Advances in Production Engineering & Management*, Vol. 13, No. 3, 279-296, doi: 10.14743/apem2018.3.290.

[25] Wang, J.F., Kang, W.L., Zhao, J.L., Chu, K.Y. (2016). A simulation approach to the process planning problem using a modified particle swarm optimization, *Advances in Production Engineering & Management*, Vol. 11, No. 2, 77-92, doi: 10.14743/apem2016.2.211.

[26] Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J. Wheeler, R., Ng, A. (2009). ROS: An open-source robot operating system, *Proceedings of ICRA Workshop on Open Source Software,* Kobe, Japan.

[27] Koenig, N., Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator, In: *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566),* Vol. 3, Sendai, Japan, 2149-2154, doi: 10.1109/IROS.2004.1389727.

[28] Alsadik, B., Karam, S. (2021). The simultaneous localization and mapping (SLAM) – An overview, *Surveying and Geospatial Engineering Journal*, Vol. 2, No. 1, 1-12, doi: 10.38094/sgej1027.

[29] Grisetti, G., Stachniss, C., Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized particle filters, *IEEE Transactions on Robotics,* Vol. 23, No. 1, 34-46, doi: 10.1109/TRO.2006.889486.

[30] Wang, D., Tan, K., Dong, Y., Yuan, G., Du, X. (2020). Estimating the position and orientation of a mobile robot using neural network framework based on combined square-root cubature Kalman filter and simultaneous localization and mapping, *Advances in Production Engineering & Management*, Vol. 15, No. 1, 31-43, doi: 10.14743/apem2020.1.347.

[31] Fox, D., Burgard, W., Thrun, S. (1997). The dynamic window approach to collision avoidance, *IEEE Robotics & Automation Magazine,* Vol. 4, No. 1, 23-33, doi: 10.1109/100.580977.
[32] Chen, D., Lin, H. Zhao, C., Lei, J., Zou, J., Huang, L. (2021). Train carriage disinfection robot based on visual SLAM, *Journal of Computers*, Vol. 32, No. 3, 210-221, doi: 10.3966/199115992021063203015.
[33] The Nav2 project, from *https://navigation.ros.org*, accessed November 11, 2021.
[34] Gazebo robot simulator, from *http://gazebosim.org/*, accessed November 11, 2021.
[35] SVL Simulator, from *https://www.svlsimulator.com/docs/tutorials/robotics-ros2/*, accessed November 11, 2021.
[36] 3D visualization tool for ROS, from *http://wiki.ros.org/rviz*, accessed November 11, 2021.
[37] ROS Navigation Stack, from *https://github.com/ros-planning/navigation*, accessed November 11, 2021.
[38] ROS face detector, from *http://wiki.ros.org/face_detector*, accessed November 11, 2021.