Expert Systems With Applications 213 (2023) 119049

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Autonomous path planning with obstacle avoidance for smart assistive systems

Charis Ntakolia [a,b,*,1], Serafeim Moustakidis [c,2], Athanasios Siouras [c,3]

[a] Laboratory for Maritime Transport, National Technical University of Athens, 15780 Zografou, Greece
[b] Department of Aeronautical Studies, Sector of Materials Engineering, Machining Technology and Production Management, Hellenic Air Force Academy, Dekeleia Base, 13672 Acharnes, Greece
[c] AiDEAS OU, Narva mnt 5, Tallinn, Harju maakond 10117, Estonia

ARTICLE INFO

Keywords:
Mixed Integer Programming
Metaheuristic Algorithm
Swarm Intelligence
Computer-Assisted Path Planning
Obstacle Avoidance

ABSTRACT

Given the increased interest in smart assistive technologies and autonomous robot vehicles, path planning has emerged as one of the most researched and challenging topics in navigation. Moving to partially known or unknown environment, an assistive navigation system should be able to extract spatiotemporal information and dynamically identify objects and adjust the route. Current approaches typically rely on external services to perform high demanding computations and employ a plethora of overlapping sensors to accurately scan the surrounding environment. This increases their energy demands, size and weight, while incommodes their use in real time applications making their application to wearable assistive systems, such as smart glasses, a challenge. Aiming to provide a comfortable and computationally efficient wearable solution that can be used by human or robotic assistive systems, in this study we propose a novel two-level hierarchical architecture combining global and local path planning. The macroscale navigation involves the construction of the initial global path while the microscale navigation includes the local path planning with obstacle detection and avoidance. The methodology consists of: (i) a novel chaotic ant colony optimization algorithm with fuzzy logic (CACOF) for path construction; (ii) powerful and light weight deep convolutional neural networks for obstacle detection; and (iii) a Bug-like algorithm enhanced with fuzzy rules for obstacle avoidance in case of static objects. A vast experimental evaluation was conducted to test the proposed methodologies in a simulation environment based on the topology of real area. The results proved the computational efficiency and ability of the proposed path planning algorithms to address effectively multi-objective global and path planning problems which make them suitable for real time applications.

1. Introduction

Designing an efficient navigation strategy for autonomous navigation assistive systems or mobile robots and ensuring their securities are the most important issues in autonomous movement. Similarly to unmanned vehicles, visually impaired individuals (VIIs) need to walk in complex environments where the conditions change unpredictable, such as in the case of terrain conditions with moving obstacles among others. Based on recent studies, on time alerts and accurate directions are important operations that VIIs expect from an autonomous assistive navigation system (Iakovidis, Diamantis, Dimas, Ntakolia, & Spyrou, 2020; Ntakolia, Dimas, & Iakovidis, 2020). To this end, autonomous path planning becomes a necessity in order to reassure the efficiency and the safeness of the selected path during the movement of the user without the need of human interference (Fernandes, De Oliveira, & Neto, 2018; Patle, Pandey, Parhi, Jagadeesh, & others, 2019).

Global path planning (GPP), or off-line path planning, is defined as the procedure that finds a high-level low-resolution path for a robotic vehicle based on known environmental map. On the other hand, local path planning (LPP), or on-line path planning focuses on dynamic collision detection, joint limit detection and collision avoidance. It does not need a-priori knowledge of the working environment. It provides a

* Corresponding author.
E-mail addresses: cntakolia@naval.ntua.gr (C. Ntakolia), s.moustakidis@aideas.eu (S. Moustakidis).
[1] 0000-0001-9780-9815.
[2] 0000-0002-1090-2177.
[3] 0000-0002-7654-0543.

https://doi.org/10.1016/j.eswa.2022.119049
Received 29 April 2021; Received in revised form 13 October 2022; Accepted 13 October 2022
Available online 19 October 2022

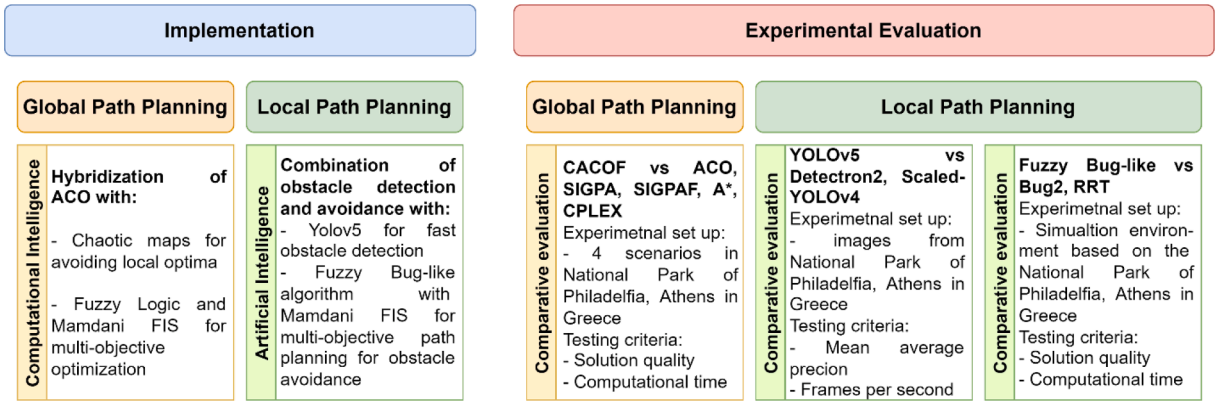


Fig. 1. An overview of the structure and the methodologies adopted in this study.

low-level high-resolution path over the path constructed by the global path planning with the environmental information acquired by the sensors (Mac, Copot, Tran, & De Keyser, 2016; Raja & Pugazhenthi, 2012). GPP works well in known environments but it is proven inadequate to address path planning problems with unknown or dynamic obstacles in the working environment. LPP works efficiently in dynamic environments but when the target is far from the current location the LPP fails to provide an optimal path. Hence, a combination of both methods is preferred to enhance the advantages and limit the weaknesses of both path planning strategies (Imran & Kunwar, 2016; Zhang, Butzke, & Likhachev, 2012). The main navigation principles consist of: (i) localization; (ii) path planning; (iii) path execution; and (iv) obstacle avoidance.

A powerful assistive navigation system incorporates both macroscale and micro-scale navigation for reaching one or more remote destinations and sensing the immediate environment, respectively. The macroscale navigation provides the user with a global route and orientation, while the micro-scale navigation covers sensorimotor behaviors and context awareness based on visual object localization. The above hybrid path planning problem can be formulated as an optimization problem where the optimal trajectory between two (starting and ending position) or more (targets to be visited) points should be found by avoiding the collision with static and dynamic obstacles. The detection / recognition of obstacles can be seen as a classification problem where deep learning is employed (Fernandes, Costa, Filipe, Paredes, & Barroso, 2019).

In this study a novel end-to-end path planning methodology is proposed that incorporates a global and local path planner in a hierarchical architecture. Fig. 1 shows the methodologies adopted to implement the global and local path planning along with the evaluation methodologies used to test the effectiveness of the proposed hybrid scheme. The presented study contributes to the existing literature in the domain of smart assistive systems with the followings:

- A novel assistive system with low number of sensors (need for an RGB-D camera for smart glasses)
- A novel computationally efficient multi-objective path planning algorithm based on hybrid Ant Colony Optimization with Fuzzy Logic and Chaotic maps (CACOF) for global path planning
- A lightweight deep learning-based obstacle detection framework (YOLOv5)
- A novel computationally efficient obstacle avoidance algorithm with fuzzy logic for multi-objective local path planning in case of static objects (Fuzzy Bug-like)
- Application to sensitive social group of individuals with disabilities, such as visual impaired individuals (VIIs)
- Application to simulation environment based on the topology of real environment, such as a park, namely the National Park of Nea Philadelphia, Athens in Greece.

The rest of this study is structured as follows: In Section 2 the related work regarding the path planning algorithms and the obstacle avoidance methodologies are presented. Section 3 presents the proposed framework, covering the system's architecture, the global and the local path planning methods adopted in this study. Section 4 includes the evaluation methodology used for examining the efficiency of the proposed methods, while conclusions and future work are provided in Section 5.

2. Related work

2.1. Path planning algorithms

Path planning algorithms have been widely used for outdoor and indoor path planning. In the context of pathfinding algorithms for smart assistive systems, classical methods and heuristic methods are two of the common categories of navigation and motion planning approaches. Most of the classical approaches involve graph-based algorithms for indoor applications. These algorithms use graphs to represent the environment. The most popular ones include the Dijkstra algorithm, A* algorithm, probabilistic roadmap (PRM), Rapidly exploring Random Tree algorithm (RRT) and Orthogonal Jump point search algorithm (OJPS) (H. Fernandes et al., 2019; Mahida, Shahrestani, & Cheung, 2018).

A variant of Dijkstra algorithm was proposed in (Fadzli, Abdulkadir, Makhtar, & Jamal, 2015) for finding the optimal path by using a multilayer dictionary storage structure. Another variant of Dijkstra based on greedy search, OPTIPATH, was developed for solving the path planning problem (Nandini & Seeja, 2019). For urban environment, a path in reference Dijkstra's algorithm variation was developed with binary heap as a priority queue (Balata, Mikovec, & Slavik, 2018). A* is another traditional but also popular algorithm involved in smart assistive systems to support visually impaired individuals with navigation (Joseph et al., 2015; Xiao et al., 2015) and especially for indoor environments (Zhang et al., 2016). For indoor navigation, a greedy algorithm was used, based on a semantic path planning and a time-stamped map Kalman filter algorithm for obstacle avoidance (Li et al., 2018); turn-by-turn navigation approaches were adopted (Ahmetovic, Oh, Mascetti, & Asakawa, 2018). These algorithms translate the environment into a graph with straight paths and turning points. An adaptive artificial potential field (AAPF) path planning algorithm was presented in (Zhang, Yao, Zhu, & Hu, 2019) that considers both efficiency and safety. A combination of various graph-based algorithms, such as A*, Dijkstra and RRT*, was presented in (Bevilacqua et al., 2016) for generating an initial non-smooth path planning for assistive robots and

nonlinear programming (NLP) to minimize the human discomfort.

Metaheuristic algorithms create a sub-population of possible maneuvers in each iteration compared to the presented classical methods where a global path is created based on the prior knowledge, such as a map of the environment (Chen, Zhou, Lv, Deveerasetty, & Dike, 2018; Jain & Malhotra, 2020). Such approaches mostly include swarm intelligence-based algorithms. However, the application of metaheuristics to path planning for human assistive navigation systems is limited while the majority of these studies are focusing on robots (Bevilacqua et al., 2016; Liu, Xu, Lu, & Zhang, 2018). Grey wolf and whale algorithms were employed for path planning and obstacle avoidance in the context of mobility-assisted localization (Alomari, Phillips, Aslam, & Comeau, 2017). An improved bee colony algorithm was applied to path planning for assisting a crowd evacuation (Liu et al., 2018) by employing the strategies of grouping and exit selection.

The great majority of the aforementioned approaches are limited to indoor known environments. They are using graph-based algorithms that may decrease the computational performance especially in case of complicated environments that lead to big graphs. Moreover, shortest distance is commonly adopted as a primary criterion for optimality (Mahida, Shahrestani, & Cheung, 2017; Ntakolia & Iakovidis, 2021a). In addition, few studies use NLP solvers, such as IPOPT, which are considered as computation demanding approaches, especially when they are applied to large or complex maps. In case of unknown indoor environments, most of the approaches employ the Simultaneous Localization and Mapping (SLAM) which lead to excessive computational effort and problematic applications especially when dealing with large maps where various sensors are needed to accurately map the area (Saeedi, Trentini, Seto, & Li, 2016). Depending on the adopted SLAM method, limitations can be presented regarding the terrestrial scan mechanism, sensor resolution, range limit and the density of produced point cloud (Chen et al., 2018). The aforementioned classical methods are considered to be less capable in handling unknown, partially known, or dynamic environments in their basic form and are known to be computationally intensive. Moreover, graph-based approaches demand complete prior knowledge of the environment under examination to create a feasible path between a starting and a destination point. To overcome the limitations of graph-based approaches, metaheuristic algorithms in path planning have attracted the attention of researchers due to their simplicity, computational efficiency and effectiveness in solving real path planning problems in unknown or partially known environments. However, most of the existing pathfinding approaches generate paths suitable for robots or robotic assistive systems, characterizing by sharp turn-by-turn paths and the necessity of an adequate number of peripheral sensors.

Therefore, a more human-friendly path planning approach should be adopted in case of individuals with disabilities respecting their special needs and spatial perception, as well as, decreasing significantly the number of sensors for a comfortable wearable smart assistive navigation system (Iakovidis et al., 2020; Ntakolia et al., 2020). Hence, this highlights the need for a further study and development of new approaches for smart assistive navigation systems. In our study, a novel global path planning algorithm is proposed based on Ant Colony Optimization (ACO) metaheuristic algorithm by (i) incorporating chaotic maps during the pheromone update process in a probabilistic way; and (ii) employing Fuzzy Logic for evaluating the generated paths with respect to the objective criteria and fuzzy rules. The proposed Chaotic Ant Colony Optimization with Fuzzy Logic (CACOF) algorithm is capable of solving multiobjective path planning problems. In the context of developing a smart assistive navigation system for visually impaired individuals, CACOF algorithm is used to find the path with the maximum quality. The quality of the path is based on the travelled distance, the number of obstacles in the path, the number of sharp turns and the multiple crossovers from the same point of interest.

2.2. Obstacle avoidance

Obstacle avoidance has been a challenging research area that has attracted the interest of the scientific community. Most of the reported studies rely on navigation strategies that are capable to operate in either known (Pratama, Nguyen, Kim, Kim, & Kim, 2016) or unknown (Kamil, Tang, Khaksar, Zulkifli, & Ahmad, 2015) environments making use of a variety of sensors. The collected data facilitates the development of advanced algorithms that monitor and model the dynamics of the surrounding environment by detecting objects of interest (that could be considered as obstacles) and then determining avoidance routes via well-known heuristic approaches such as BUG-based algorithms relying on Simultaneous Localization And Mapping (SLAM) or Visual Odometry (VO) methods (McGuire, Croon, & Tuyls, 2019; Wolf & Sukhatme, 2005). Multiple applications of such intelligent obstacle avoidance systems have been proposed including (semi-)autonomous robots (Han & Liu, 2019; Kim, Chen, Kim, & Cho, 2018; Kudriashov, Buratowski, Giergiel, & Malka, 2020; Lentzas & Vrakas, 2020) and assistive technologies for people with visual or kinetic disabilities (Li, Dai, Shi, Zhao, & Ding, 2019; Manjari, Verma, & Singal, 2020; Minetto, Kozievitch, Silva, Almeida, & Santi, 2016; Tapu, Mocanu, & Zaharia, 2018).

Current obstacle avoidance can be categorized as robotic travel aids or electronic travel aids. The first category uses mobile robots for active walk guidance and mainly relies on sensors that acquire range data about the distances to obstacles (Akiyoshi, Chugo, Muramatsu, Yokota, & Hashimoto, 2020; Campbell et al., 2020; Luckcuck, Farrell, Dennis, Dixon, & Fisher, 2019; Raja & Pugazhenthi, 2012). However, the human body is subjected to a variety of micro-movements while walking or moving and consequently this data variability adds complexity and uncertainties to the problem. Moreover, these robotic solutions strongly depend on the power and mobility of robots that in most of the cases are capable to walk on relatively flat and smooth surfaces. Electronic travel aids are portable systems that employ vision sensors capturing forward facing images. Appropriate image processing algorithms and advanced deep learning analytics are then applied to detect obstacles or find pathways. A novel backbone network architecture, CBNET, was proposed in (Liu et al., 2020) for obstacle detection and avoidance by assembling multiple identical backbones to form a more powerful backbone. A Single Shot MultiBox Detector (SSD) algorithm was proposed in (Liu, Gao, Chi, & Fan, 2021) for pedestrian detection by using a depthwise Separable Convolution and an hierarchical structure fusion to enhance features. MobileNetV2 (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018) was used for feature extraction, whereas the final layers were fine-tuned to detect typical road obstacles such as, cars, motorcycles and pedestrians, among others. Wearables have also been utilized to assist the data collection process. In (Vaishnav, Rao, & Bade, 2021) a device, based on a portable Raspberry PI Zero W platform, was attached to a human's hat performing object detection with the use of the well-known You Only Look Once (YOLO) detector (Farhadi & Redmon, 2018). One of the novelties of the proposed system was that it was capable to perform detection in offline mode facilitating its deployment in outdoor environments where the network availability is sometimes problematic.

While these systems are capable of detecting objects and obstacles at high framerates, they typically rely on voice commands to inform the user about the presence and the type of the obstacle along with the location of it with respect to the input sensors. This can be confusing as the user may have to perform multiple direction changes until the obstacle is lost from their field of view. To deal with this problem, in this paper we propose a novel bug-like human centric obstacle avoidance and navigation framework. More specifically the proposed framework relies on the latest advances in objection detection utilizing state-of-the-art (SoA) deep learning algorithms such as scaled-YOLOv4, YOLOv5 and Region-based CNNs via Detectron2. A comparison analysis was performed to identify the most effective network performing the obstacle detection and recognition, whereas a variation of BUG algorithm

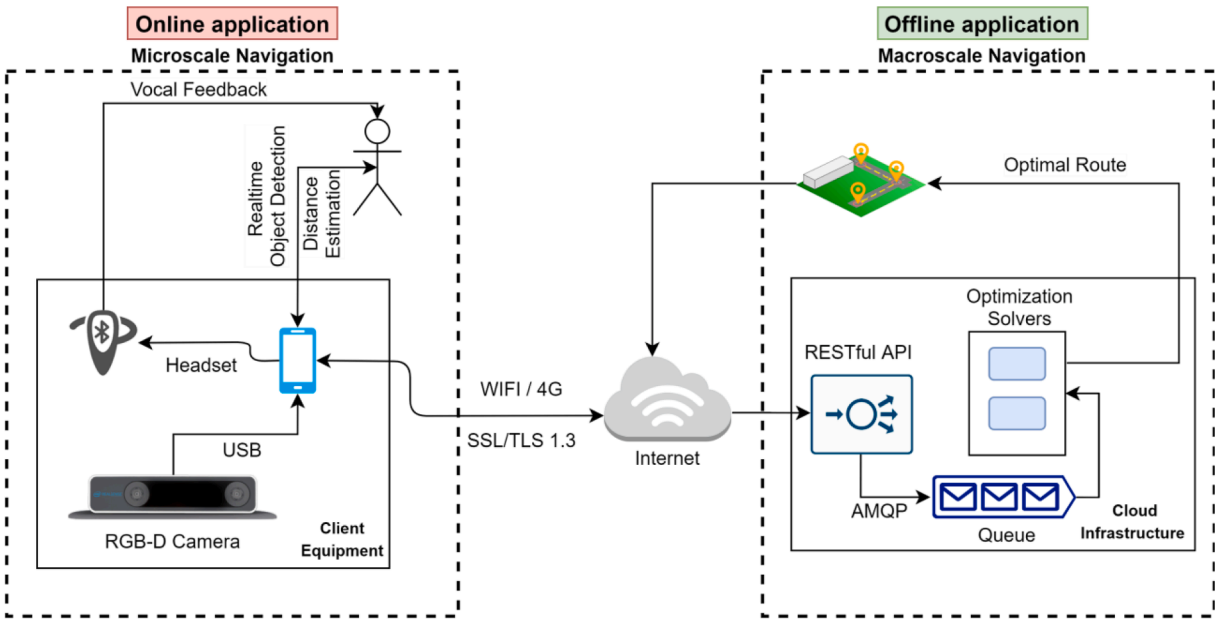


Fig. 2. Two-level hierarchical architecture for optimal path planning and real time safe navigation.

considers the detection obstacles and proposes the optimal avoidance path based on the distance and angle of the individual. To avoid multiple notifications for the same obstacle in short timespan, the detected objects are tracked following the tracking-by detection (Bochinski, Eiselein, & Sikora, 2017) approach combined with features obtained from the selected backbone feature extractor. The performance of the framework was evaluated both in terms of obstacle detection and recognition and in terms of navigation in a simulated environment obtaining promising results.

A novel local path planning algorithm for obstacle avoidance is also proposed based on principles of bug algorithms and fuzzy rules. The algorithm provides local geospatial directions to the user for avoiding a detected obstacle within the path.

3. Proposed framework

3.1. System architecture

A two-level hierarchical architecture is presented for addressing the problem of safe navigation in outdoor cultural spaces. The macroscale level represents the macroscale navigation level where the optimal tour plan is generated based on user's preferences and spatial information. The macroscale navigation level is implemented as a binary integer programming problem. The target is to minimize: (i) the traveled distance; (ii) the collision occurrences; (iii) the multiple crossovers from the same paths; and (iv) the brut and continuous deviations in the path trajectory. The optimization of the aforementioned terms guarantees a shortest collision-free and smooth path. The formulated global path planning problem is solved by a novel metaheuristic Chaotic Ant Colony Optimization with Fuzzy Logic (CACOF) algorithm. The algorithm is based on Ant Colony Optimization for finding the optimal path in a network, chaotic maps for a probabilistic pheromone update process and fuzzy logic for evaluating the solutions.

The microscale lever represents the local path planner where static obstacles are detected in real time. It gets as input the optimal global path developed by the macroscale level and the information from RGB-D sensor of smart glasses. The microscale navigation level is implemented with a Deep Neural Network (DNN) detector, such as YOLOv55, and a bug-like algorithm for obstacle avoidance.

The macroscale level takes as input the user preferences such as the selected POIs to visit and the energy state. The output of this level is the generated optimal global path plan that will be used as input for the lower level. The lower level will detect in real time possible collision with obstacles and will provide a local path planning for obstacle avoidance (Fig. 2).

3.2. Macroscale navigation (Global path planning)

3.2.1. Chaotic ant colony optimization with fuzzy logic algorithm (CACOF)

Ant Colony Optimization (ACO) Algorithm mimics the process of ants to trace a food source and map the path from their nest to the food source. To achieve this, they exude a chemical substance called pheromone, as a trail. This trail guides other ants during the search for food. As time passes, the trail is evaporated, however, a shorter path corresponds to a shorter time of volatilization of the pheromone whereas a longer path corresponds to a longer time. Hence, a shorter path becomes more traversed due to slower pheromone evaporation and larger quantity of pheromone. To this end, the ants will choose the shorter path for food. ACO algorithm, by adopting this behavior, finds the shortest path in a graph among a population of ants (Mellal & Williams, 2018). Bellow, the main steps of ACO are briefly presented:

1st step: State transition rules.

The probability of an ant k to traverse an edge (i,j) at iteration t is calculated by:

$$P_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{j\in allowed_k}\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}} \quad (1)$$

where $\tau_{ij}(t)$ is the amount of pheromone trail on point on edge (i,j) at iteration t, $j \in allowed_k$ represents the nodes j that the ant k is permitted to travel from node i, $\alpha \geq 0$ and $\beta \geq 1$ are the parameters to control the influence of $\tau_{ij}(t)$ and η_{ij}, respectively. A higher value of α indicates a stronger preference by ants for the pheromone concentration between connected nodes when they choose the path, whereas a higher value of β indicates a stronger preference by ants for the distance between the current node i and next node j. We should notice that when $\alpha \gg \beta$, search stagnation is likely to happen, causing the result to trap into local optimal solutions. Also, η_{ij} is the heuristic desirability. Usually, $\eta_{ij} = \frac{1}{d_{ij}}$ where d_{ij} is commonly the Euclidean distance between the two points (i, j) or other distance metric (Lee, 2017; Mellal & Williams, 2018).

2nd step: Pheromone Update.

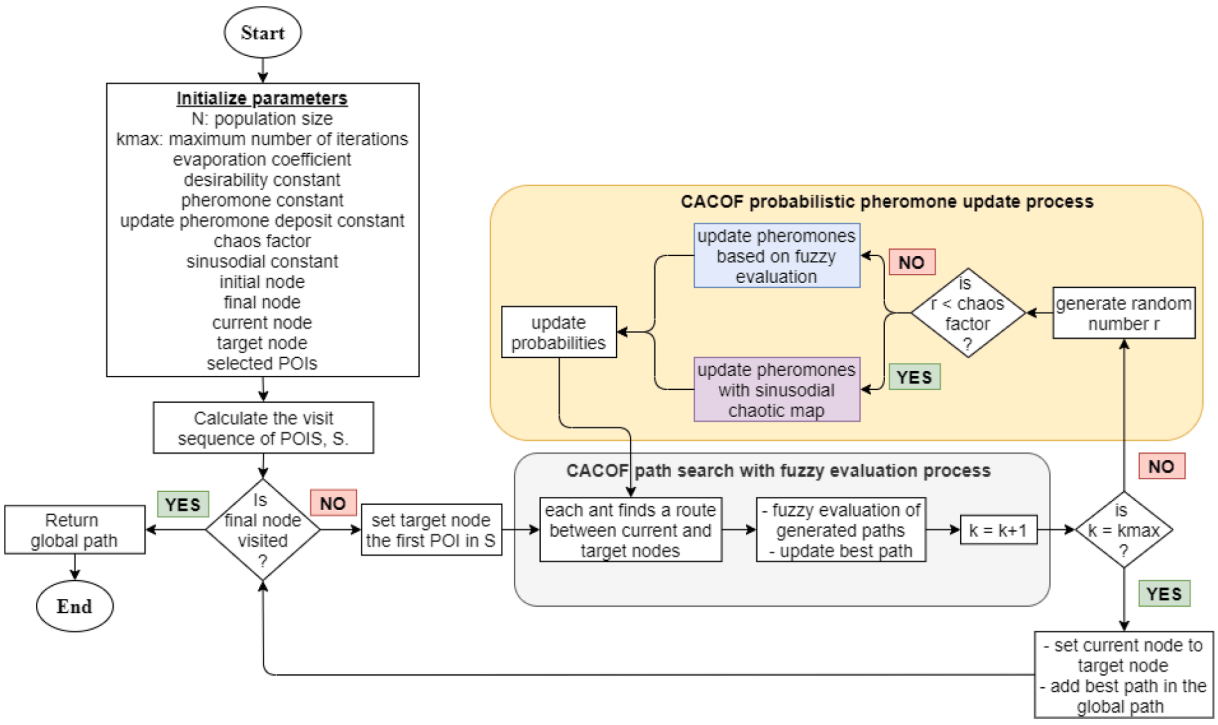
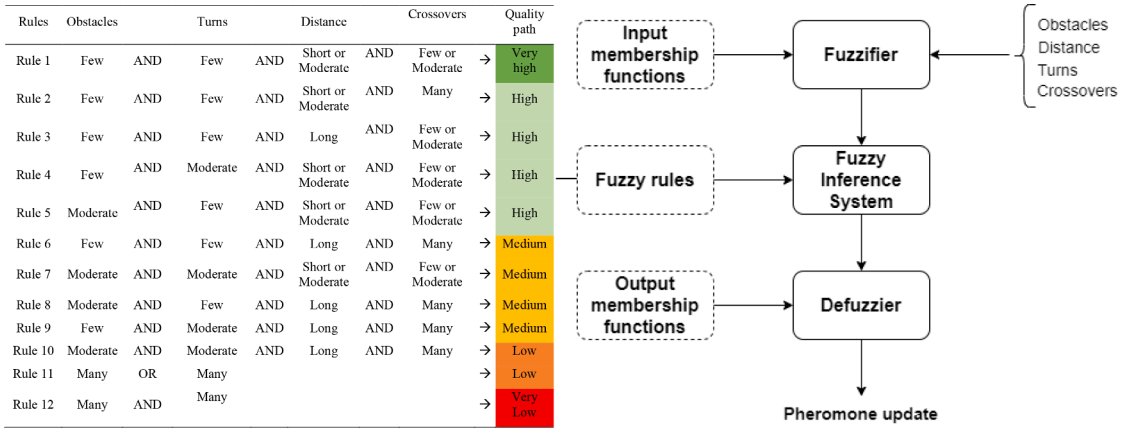


Fig. 3. CACOF flowchart.



Rules	Obstacles		Turns		Distance		Crossovers		Quality path
Rule 1	Few	AND	Few	AND	Short or Moderate	AND	Few or Moderate	→	Very high
Rule 2	Few	AND	Few	AND	Short or Moderate	AND	Many	→	High
Rule 3	Few	AND	Few	AND	Long	AND	Few or Moderate	→	High
Rule 4	Few	AND	Moderate	AND	Short or Moderate	AND	Few or Moderate	→	High
Rule 5	Moderate	AND	Few	AND	Short or Moderate	AND	Few or Moderate	→	High
Rule 6	Few	AND	Few	AND	Long	AND	Many	→	Medium
Rule 7	Moderate	AND	Moderate	AND	Short or Moderate	AND	Few or Moderate	→	Medium
Rule 8	Moderate	AND	Few	AND	Long	AND	Many	→	Medium
Rule 9	Few	AND	Moderate	AND	Long	AND	Many	→	Medium
Rule 10	Moderate	AND	Moderate	AND	Long	AND	Many	→	Low
Rule 11	Many	OR	Many					→	Low
Rule 12	Many	AND	Many					→	Very Low

Fig. 4. Fuzzy Logic control system of CACOF.

In each iteration the ant population travels a path. Each time, after finishing an iteration, the selected paths by the ants are compared and the shortest is chosen. Then, it is compared to the previous shortest path to determine the current shortest path. Based on the current path, the pheromone is updated by:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho \sum_{k \in K} \Delta\tau_{ij}^{k}(t) \tag{2}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k}, if edge(i,j) is selected by ant k \\ 0, otherwise \end{cases} \tag{3}$$

where $0 \leq \rho \leq 1$ represents the evaporation coefficient of the pheromone, K is the set of the ant population, $\Delta\tau_{ij}^{k}(t)$ is the amount of the remaining pheromone deposited by the ant k in this search, L_k is the cost of the path that the ant k traveled and Q is a constant associated with the amount of remaining pheromone.

The ACO has the following defects. First, the ACO is a heuristic bio-inspired optimization method. The random initialization, the large search space that leads make it inevitable to ants to produce a large number of local cross paths, circular paths and sawtooth paths, and the risk that many ants may lose their bearings and cannot find the full paths, can lead to local traps and failure to find the global optimum solution (J. Liu, Yang, Liu, Tian, & Gao, 2017). To cope with the aforementioned issues, in this study we propose an improved ACO that incorporates chaotic maps in the state transition and fuzzy logic for the pheromone update. Specifically, to update the pheromones of ACO we adopt a probabilistic approach based on chaotic maps and fuzzy logic. Let $c \in [0,1]$ be the chaos factor and $r \in (0,1)$ a randomly generated probability. If $r < c$ then the pheromone update process is implemented by using the sinusoidal map otherwise by using the fuzzy evaluation (Fig. 3).

Only few studies have been conducted that use ACO and fuzzy logic for path planning. These approaches use fuzzy logic: (i) to minimize the iterative learning error of ACO in the context of obstacle avoidance for

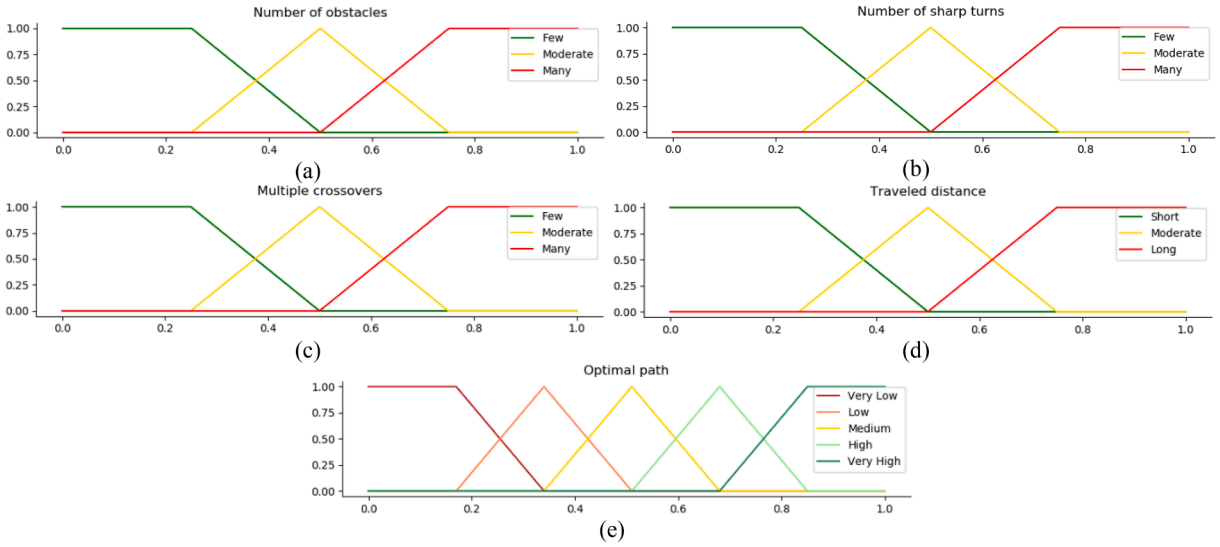


Fig. 5. Membership functions of universal variables.

robots (Yen & Cheng, 2018); (ii) to evaluate the paths generated from ACO algorithm for autonomous mobile robot navigation (Garcia, Montiel, Castillo, Sepulveda, & Melin, 2009), and unmanned surface vehicles (USVs) (Ntakolia & Lyridis, 2022); and (iii) to estimate the collision free area for obstacle avoidance (Sangeetha et al., 2021). Also, ACO is used to optimize the parameters of the fuzzy rules that are used to on-line path planning in case of mobile robots path planning in dynamic environments (Purian & Sadeghian, 2013). To the best of our knowledge, no previous study has incorporated fuzzy logic and chaotic maps to ACO algorithm for path planning in case of smart assistive systems for individuals with disabilities.

Chaotic Maps

Chaotic maps are incorporated in the pheromone update process due to their ability to improve the search ability and the convergence speed of evolutionary algorithms (EAs) (Lu, Wang, Fei, & Qiu, 2014). However, different chaotic maps in different phases have different effects on EAs. In our study, we adopt a one-dimensional chaotic map, named sinusoidal iterator to re-initialize, in a probabilistic approach, the probabilities in the 1st step (eq. 1):

$$x_{t+1} = \alpha x_t^2 \sin(\pi x_t), x_t \in (0,1) \tag{4}$$

where α is the control parameter. In this study, the simplified and more popular equation of this map is used by using $a = 2.3$ and $x_0 = 0.7$ which can be written as (Arora & Anand, 2019):

$$x_{t+1} = \sin(\pi x_t), x_t \in (0,1) \tag{5}$$

Sinusoidal map was chosen due to its value field $(0,1)$ that fits to the permitted values of a probability.

Fuzzy Evaluation

The Fuzzy Logic (FL) control (Fig. 4) is implemented based on Mamdani FL control system (Mamdani, 1974) by using fuzzification, rule evaluation and defuzzification processes. To evaluate the path that is selected by an ant in each iteration, fuzzy logic is applied, and fuzzy rules are imposed. The fuzzy evaluation shows the quality of a generated path with respect to:

(i) the number of known obstacles in the generated path:

$$z_1 = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{N}}\sum_{\substack{j\in N:\\(i,j)\in A}}\left(o_{(i,j)}X^t_{(i,j)}\right) \tag{6}$$

where $X^t_{(i,j)} = \begin{cases} 1, \text{ if the user is travelling } arc(i,j) \text{ at time period } t \\ 0, \text{ otherwise} \end{cases}, \forall (i,j) \in \mathcal{A}, t \in \mathcal{T}$, $o_{(i,j)}$ is the number of obstacles in the $arc(i,j)$, $\mathcal{T} = \{1, \cdots, t_{max}\}$ the set of time periods (time frame) in which the tour should finish, $\mathcal{N}$ the set of nodes in the network, and $\mathcal{A} = \{(i,j) : i,j \in \mathcal{N}\}$ the set of permitted arcs in the network (paths, pavements, roads, etc.).

(ii) the number of turns that the user should perform in order to implement the path under evaluation:

$$z_2 = \sum_{t\in\mathcal{T}}\sum_{j\in N}\left(\sum_{\substack{i\in N:\\(i,j)\in A}} X^{t-1}_{(i,j)}\left(\sum_{\substack{k\in N:\\(j,k)\in A}} \alpha_{(i,j,k)}X^t_{(j,k)}\right)\right) \tag{7}$$

where $a_{(i,j,k)}$ is the angle that is formed from the $arc(i,j)$ and the $arc(j,k)$

(iii) the number of multiple crossovers from the same POI:

$$z_3 = \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{N}}\sum_{\substack{j\in N:\\(i,j)\in A}}\left(d_{(i,j)}X^t_{(i,j)}\right) \tag{8}$$

where $d_{(i,j)}$ is the length of the $arc(i,j)$ (in the desirable distance unit).

(iv) and the travelled distance from the beginning point to the ending point:

$$z_4 = \sum_{t\in\mathcal{T}}\sum_{m\in\mathcal{M}}(Z_m) - \sum_{m\in\mathcal{M}}(Z_m) \tag{9}$$

where $Z_m = \begin{cases} 1, \text{ if the user has visited POI } m \\ 0, \text{ otherwise} \end{cases}, \forall m \in \mathcal{M}$ and $\mathcal{M} = \{i : i \in \mathcal{N}\}$ is the set of POI that the user wants to visit.

The criteria (i)-(iv) constitute the inputs of the FL control system, such as the number of obstacles, the traveled distance, the number of turns and the multiple crossovers from POIs. Three fuzzy subsets were designed for the input variables, while five were used for the output variable. Given each crisp value, the uncertainty can be modeled by fuzzy sets:

$$\tilde{z}_{1,v} = \left\{\left\langle x, \mu_{z_{1,v}}(x)\right\rangle | x \in U_{z_1}\right\}, v = \{few, moderate, many\} \tag{10}$$

$$\tilde{z}_{2,v} = \left\{\left\langle x, \mu_{z_{2,v}}(x)\right\rangle | x \in U_{z_2}\right\}, v = \{few, moderate, many\} \tag{11}$$

$$\tilde{z}_{3,v} = \left\{\left\langle x, \mu_{z_{3,v}}(x)\right\rangle | x \in U_{z_3}\right\}, v = \{few, moderate, many\} \tag{12}$$

$$\widetilde{z}_{4,v} = \left\{ \left\langle x, \mu_{z_{4,v}}(x) \right\rangle | x \in U_{z_4} \right\}, v = \{short, moderate, long\} \quad (13)$$

$$\widetilde{q}_v = \left\{ \left\langle x, \mu_{q_v}(x) \right\rangle | x \in U_q \right\}, v = \{verylow, low, medium, high, veryhigh\} \quad (14)$$

representing overlapping value intervals that can be expressed linguistically, e.g. “few”, “moderate”, “many” in the case of the number of obstacles (z_1), number of sharp turns (z_2) and multiple crossovers (z_3), and “short”, “moderate” and “long” in case of distance (z_4). For this, four fuzzy universes are defined: U_{z_1}, U_{z_2} U_{z_3} and U_{z_4} representing the universe of discourse for the number of obstacles, turns, crossovers and distance, respectively. Fig. 5 shows the membership functions of the universal variables (a)-(d) and the output variable (e), e.g. the path quality. The values are normalized based on the results of the ant population in each iteration. Therefore, in eq. (3) instead of the cost $\frac{1}{L_k}$ the defuzzification value that corresponds to the quality of the path is used.

3.3. Microscale navigation (Dynamic local path planning and object detection and avoidance)

The local path planning framework can be broken down into the following two components:

- The detection, the recognition and localization of the obstacle dynamically with respect to the individual
- The navigation of the individual around static obstacles

The first component uses an RGB-D sensor (Intel RealSense D435i) mounted on the head of the individual in a form of smart glasses and provides high resolution RGB images along with depth maps of the surroundings in 30fps. The RGB-D sensor is connected to a common, Android mobile device using USB3.0 interface and acts as the processing unit of the local path planning subsystem. The RGB images are used as an input to an object detection network that enables obstacle detection and recognition at a real-time frame rate. Three recent and powerful deep learning networks were investigated for their suitability to implement the object detection task:

- *Detectron2*: Detectron 2
- (Muhawenayo & Gkioxari, 2021) is Facebook’s AI Research’s framework for building state of the art object detection and image segmentation models. It is based on its previous version, (Detectron) that originates from maskrcnn-benchmark. Detectron2 is powered by the PyTorch deep learning framework including high-quality implementations of state-of-the-art object detection algorithms, including DensePose, panoptic feature pyramid networks, and numerous variants of the pioneering Mask R-CNN model family. Within its important novelties, the training process of Detectron2 can be easily distributed over multiple GPU servers, whereas the entire training pipeline can be transferred to GPU making Detectron2 faster than the original Detectron.
- *Scaled-YOLOv4*: Scaled YOLO v4 is a series of neural networks built on top of the improved and scaled YOLOv4 network. Being published in December 2020
- (Wang, Bochkovskiy, & Liao, 2020), scaled YOLO v4 is at the moment the most accurate neural network (55.8% AP in Microsoft’s COCO dataset) among all competing object detection networks. It has been proved as the best in terms of both absolute accuracy and accuracy-to-speed ratio in the entire range of accuracy and speed from 15 FPS to 1774 FPS. It uses optimal network scaling techniques that are applicable to small and large networks while maintaining optimal speed (up to 50% for the Darknet backbone in terms of number of FLOPs) and accuracy. Compared to YOLOv4, in scaled YOLOv4 a separate neural network is trained for each resolution of the network, whereas changed activations are applied for width and height allowing faster network training.
- *YOLOv5*: It was released in 2020 as an additional implementation of YOLOv4 (with improved performance) that was completely implemented with PyTorch[4]. It supports conversion to ONNX and CoreML, which is convenient for users to deploy on mobile terminals and it is nearly 90 percent smaller than YOLOv4 facilitating its deployment in embedded devices much more easily. YOLOv5 trains faster on sample tasks, and the batch inference can produce real-time results with an FPS up to 140 compared to YOLOv4 that typically achieves up to 50 FPS.

The most effective among the three aforementioned deep learning networks, YOLOv5, was finally trained to detect 8 different obstacle classes (trees, staircases, benches, wall, lamps, persons, holes, cars), which are commonly found in unknown urban environments.

The second component of the local path planning framework uses the detected obstacles from the first component and performs the avoidance if needed. To navigate the individual around an obstacle the algorithm takes into consideration the following criteria:

- The distance from the obstacle
- The location of the obstacle with respect to the individual
- The location of the obstacle with respect to other detected obstacles
- The angle required to by-pass the obstacle

The pseudocode of the algorithm is shown in Algorithm 1. More specifically, the component using the depth map from the RGB-D camera, uses the bounding boxes of the detected obstacles to calculate the distance from the individual. Distance is measured by averaging the depth map of the bounding box. In some cases, noisy depth points might exist in the depth map, that can affect the measurement performance. To compensate for this, before the distance calculation, the standard deviation of the depth map is computed, and outliers are removed. Thus, the distance s_o from an obstacle o enclosed by a bounding box B of size $W_B \times H_B$ where w is the width and h the height of, can be expressed as:

$$s_o = \frac{\left(\sum_{i=1}^{w} \sum_{j=1}^{h} \begin{cases} \mu, |B_{ij} - \mu| \rangle \sigma \\ B_{i,j}, |B_{ij} - \mu| \leq \sigma \end{cases} \right)}{W_B H_B} + c \quad (15)$$

where μ is the mean depth of the elements of B, σ the standard deviation, and c the collision risk area that is added to the detected obstacle for safety passage.

We found that accounting for outliers minimizes the average distance error from ± 0.7 m down to ± 0.2 m.

To calculate the angle in which the individual can avoid the obstacle, the algorithm finds the angle from the left and right side of the collision avoidance area (predefined number of pixels) that is added to the initial bounding box of the detected object with respect to the individual. As the horizontal and vertical field of view (FoV) of the camera is known, the horizontal angle φ_h and vertical angle φ_v of a pixel p are calculated based on:

$$\varphi_h^p = \left(\frac{\left(p_x - \frac{H_I}{2} \right) FoV_h}{W_B} \right) \quad (16)$$

$$\varphi_v^p = \left(\frac{\left(p_y - \frac{H_I}{2} \right) FoV_v}{W_I} \right) \quad (17)$$

where W,H denote the width and height of an image, respectively, and p_x, p_y the location of the pixel p in the image.

[4] https://github.com/ultralytics/yolov5.

Fig. 6. Examples of obstacle avoidance: (a) user moving toward an obstacle; (b) obstacle detection with respect to the field of view adding a collision avoidance area (red circle) round the detected objects based on eq(15); (c) obstacle avoidance based on the generated safe area based on eq(16)-(18). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

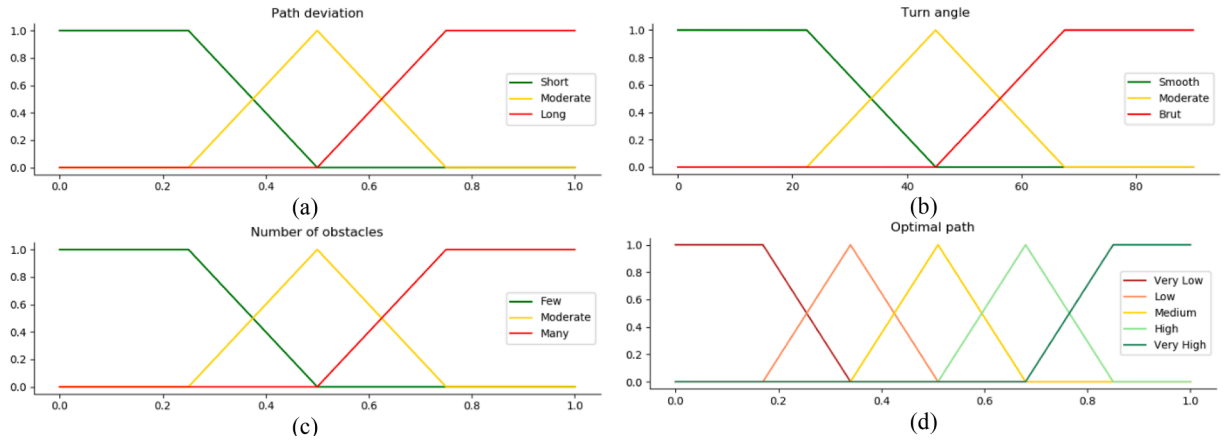


Fig. 7. Membership functions of universal variables.

Knowing the horizontal and vertical angle of a pixel we can calculate the angle of the pixel φ_p according to (18).

$$\varphi_p = \frac{\left((\varphi_h^p)^2 + (\varphi_v^p)^2\right)}{2} \tag{18}$$

Using the outer most pixel of each side of bounding box, the algorithm uses (18) to calculate the angle of the left and right side of the obstacle. Similarly, the location of the obstacle with respect to target GPS coordinates, is calculated by translating the GPS coordinates of the individual using the angle and distance of the previous step. This process is repeated for all detected obstacles that are within a specific distance c. This threshold is a hyper-parameter of the algorithm and is selected based on the application needs. For instance, in the case of visually impaired individuals, based on (Ntakolia et al., 2020), this is defined as any object within the distance of 2.0 m. In case of two or more detected obstacles, if their collision avoidance areas are intersected then the objects are considered as one and the same process that is presented above is followed.

In contrast to robots, humans usually interpret the surrounding environment in verbal, vague terms; that is, instead of expressing turns using degrees, they use verbal terms, such as "small", "large", "medium" turn. For these reasons, the Bug-like obstacle avoidance algorithm uses fuzzy logic to determine the optimal path of navigating the individual around the obstacle. This process involves the fuzzification of the crisp values, number of obstacles n in the altered path, distance of path deviation s from the initial global path and the turn angle a that the individual needs to perform in order to avoid safely the obstacle, into the fuzzy domain. Given each crisp value, the uncertainty can be modeled by fuzzy sets:

$$\widetilde{y}_{1,v} = \left\{ \left\langle x, \mu_{y_{1,v}}(x) \right\rangle | x \in U_{y_1} \right\},\ v = \{short, moderate, long\} \tag{19}$$

$$\widetilde{y}_{2,v} = \left\{ \left\langle x, \mu_{y_{2,v}}(x) \right\rangle | x \in U_{y_2} \right\},\ v = \{smooth, moderate, brut\} \tag{20}$$

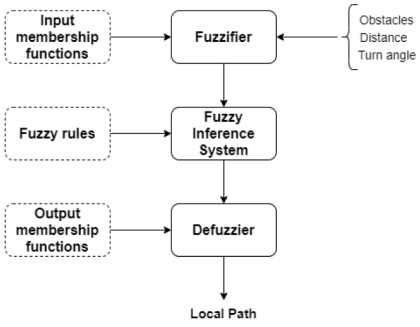
$$\widetilde{y}_{3,v} = \left\{ \left\langle x, \mu_{y_{3,v}}(x) \right\rangle | x \in U_{y_3} \right\},\ v = \{few, moderate, many\} \tag{21}$$

$$\widetilde{q}_v = \left\{ \left\langle x, \mu_{q_v}(x) \right\rangle | x \in U_q \right\}, v = \{verylow, low, medium, high, veryhigh\} \tag{22}$$

representing overlapping value intervals that can be expressed linguistically, e.g. "smooth", "moderate", "brut" in the case of the angle, "short", "moderate" and "long" in case of path deviation from the original global path due to obstacle avoidance and "few", "moderate", "many" in the case of number of obstacles that are detected in the generated local path during an obstacle avoidance.

Rules	Obstacles		Turn		Distance		Quality path
Rule 1	Few	AND	Small	AND	Short	→	Very high
Rule 2	Few	AND	Medium	AND	Short	→	High
Rule 3	Few	AND	Medium	AND	Moderate	→	High
Rule 4	Few	AND	Medium	AND	Long	→	Medium
Rule 5	Moderate	AND	Small or Medium	AND	Moderate	→	Medium
Rule 6	Moderate	AND	Large	AND	Short	→	Medium
Rule 7	Many	AND	Small	AND	Short	→	Medium
Rule 8	Moderate	AND	Large	AND	Moderate or Long	→	Low
Rule 9	Many	AND	Medium or Large	AND	Moderate	→	Low
Rule 10	Many	AND	Medium or Large	AND	Long	→	Very Low

(a)



(b)

Fig. 8. Visual representation of (a) fuzzy rules of the (b) fuzzy control system.

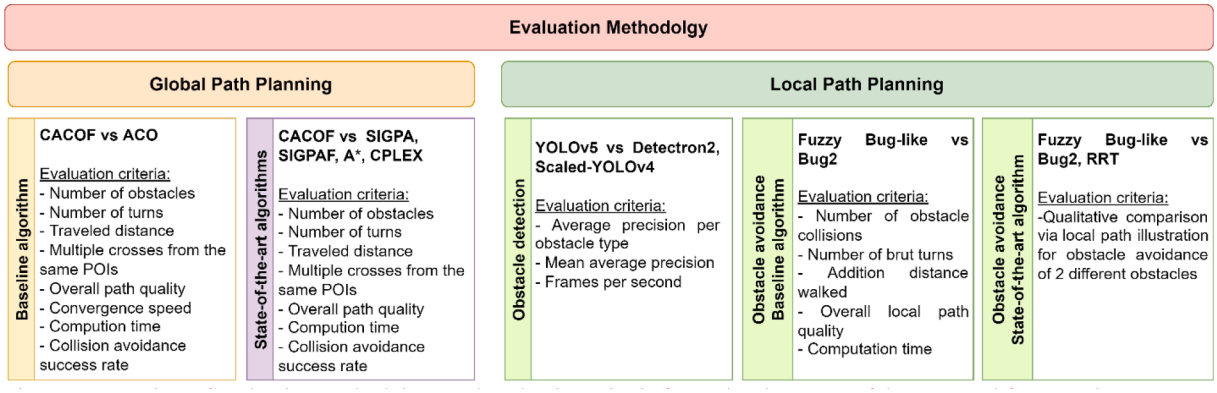


Fig. 9. Overview of evaluation methodology and evaluation criteria for each sub-system of the proposed framework.

For this, three fuzzy universes are defined: U_{y_1}, U_{y_2} and U_3 representing the universe of discourse for the path deviation, turn angle and number of obstacles, respectively. The universe U_q corresponds to the quality of a path p. The membership functions were selected based on the possible values of each variable and are illustrated in Fig. 6. To determine the best path of avoiding the obstacle, for each side, the algorithm translates the crisp numerical values of number of obstacles, distance of the individual and the turn angle that the individual has to perform, into the fuzzy domain.

The quality of the path is determined using the fuzzy rules defined in Fig. 7 and the Mamdani inference methodology. The quality of each path p is then defuzzyfied into a crisp value by calculating the Center of Gravity (CoG) eq.(23).

$$q_{t_{path}} = \frac{\sum_{i=1}^{k} q_{t_i} \bullet \mu_{q_v}(q_{t_i})}{\sum_{i=1}^{k} (\mu_{q_v}(q_{t_i}))} \tag{23}$$

where t represents the side (left or right), k the number of subareas obtained by the rule inference and q_{t_i} the value of q_t in the center of the area i.

Algorithm 1. Fuzzy Bug-like pseudocode.

Obstacle avoidance pseudocode of Fuzzy Bug-like.
At each time step t,
Initialization.
- Access all detected obstacles as they have been identified by the object detection network at the specific time step
- Group/merge obstacles if their collision avoidance areas are intersected
- **Output**: the set of available non-intersecting obstacles $G_o^t = \{o_1, \cdots, o_i, \cdots, o_K\}$

Paths generation.
- For each detected obstacle o_i,
- The distance s_o between the individual and the obstacle o_i is calculated using (15)
- IF $s_o < c$, where c is a pre-determined distance parameter ($c = 2m$, in our study),

(continued on next column)

(continued)

- THEN the angle φ_p required to by-pass the obstacle is calculated by (16)-(18)
- ELSE continue to the next obstacle
- **Output:** the set of available paths

Fuzzification and path selection.
- For each available path,
- the quality of the path $q_{t_{path}}$ is determined using the fuzzy rules defined in Fig. 7 and the Mamdani inference methodology using (23)
- The path $q^*_{t_{path}}$ with the highest quality is finally selected

The path with the highest quality is then selected. To navigate the individual to the new location, eq.(16) and (18) are calculated again taking into consideration the minimum distance c from which the individual can bypass the obstacle safely. The new location is then translated into verbal instructions, giving the angle and distance to the individual. When the obstacle disappears form the image, the algorithm re-calculates the path according to the GPS coordinates of the individual and the target location, informing the individual the angle change. This process is repeated until target location is reached.

Apart from the safety path planning, the microscale navigation aligned with the requirements of visually impaired individuals for accurate and not repeated receive of information concerning the surrounding environment (Ntakolia et al., 2020), the algorithm adopts the following tracking-by detection approach (Bochinski et al., 2017) and in cooperation with the DNN detector, tracks the detected obstacles and notifies the user only in the first frame where the obstacle is presented. To this end, the user will not be notified for the same detected obstacle in the consecutive frames.

Fig. 10. The area used for the experimental evaluation taken by Google Earth. The yellow road corresponds to the path that are used in the designed scenarios. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Case studies

4.1. Evaluation methodology

For testing the feasibility and the computing efficiency of the proposed methodology, the upper and lower level of the architecture was tested separately in simulated and real data, respectively. Fig. 9 depicts the evaluation steps and criteria adopted for each methodology proposed in this study. The operation area used in the evaluation process was based on the based on the topology of National Park of Philadelphia, Athens in Greece, where the real case studies will be implemented (Fig. 10). In the simulation environment, the width of the road is three times the width of the avatar, represented the VII, so that the VII will have 3 options to traverse the path, *i.e.* left side, right side and middle. The obstacles are placed on the left, right side or in the middle of the road. In the operational area (Fig. 10) there in total 870 obstacles were placed. The goal in the global path planning is to find a global path that will pass from all the desired POIs by walking the minimum distance, altering the minimum from the initial direction and minimizing the collision risk by choosing paths with fewer obstacles.

Regarding the collision avoidance ability, we define the collision avoidance success rate calculated based on the ratio between the number of obstacles among the path (O_p) and the total number of obstacles in the operation area (O_a) illustrated in Fig. 9. Specifically, the collision avoidance success rate indicates the ability of the algorithm to find a global path by avoiding as many as possible obstacles that can be found in the area. It is calculated by the following formula:

$$SR_{gca} = \frac{O_a - O_p}{O_a}\% \tag{24}$$

Similarly, for the local path planning the collision avoidance success rate is calculated based on the ratio between the number of obstacles that the VII must avoid due to collision (O_c) and the total number of obstacles that can be found in the selected global path (O_p) derived from CACOF (macroscale level):

$$SR_{lca} = \frac{O_p - O_c}{O_p}\% \tag{25}$$

The goal in local path planning is to accurately identify the obstacles within the path, which of them block the trajectory of the VII and then to find a local path for obstacle avoidance by minimizing the additional traveled distance, the brut changes with respect to the initial direction and to minimize the collision risk with more obstacles due to this change in the initial path.

4.1.1. Evaluation of global path planning

The upper level, for global path planning, was tested by conducting 4 scenarios in an area (Fig. 8). In the scenario 1 (Sc1) the user visits 2 POIs, in the Sc2 the user visits 4 POIs, in Sc3 the user visits 7 POIs and in the Sc4 the user visits all the 11 POIs. Initially, a comparison with the baseline algorithm (ACO) was conducted for proving the improvement of the ACO via the hybridized scheme, where each scenario ran 20 times. In this step of evaluation, the evaluation criteria between the comparative algorithms were:

(i) the objective criteria:
   a. the number of obstacles that are present among the derived global path;
   b. the number of turns that should be performed among the path;
   c. the distance of the path that the user must travel; and
   d. the number of multiple crosses from the same POIs
(ii) the overall path quality calculated by the aggregation of the above objective terms by using the Mamdani FIS (defuzzification value)
(iii) the computation time
(iv) the convergence speed based on the number of iterations needed to find an optimal solution
(v) the collision avoidance success rate (SR_{gca}) in (24).

The distance is calculated in kilometers, the convergence speed in number of iterations, and the computation time in milliseconds.

Table 1
Mean path quality value and standard deviation of the 20 runs for each scenario solved by CACOF and ACO. The best values are shown in bold.

	Path quality (defuzzification value)	
Scenarios/Algorithms	CACOF	ACO
Scenario 1	**0.91** ± 0.02	0.82 ± 0.08
Scenario 2	**0.82** ± 0.04	0.71 ± 0.06
Scenario 3	**0.75** ± 0.08	0.55 ± 0.10
Scenario 4	**0.78** ± 0.04	0.43 ± 0.17

Table 2
Mean computation time in milliseconds (ms) and standard deviation of the 20 runs for each scenario solved by CACOF and ACO. The best values are shown in bold.

	Compution time (ms)	
Scenarios/Algorithms	CACOF	ACO
Scenario 1	**12.342** ± 0.545	15.427 ± 1.050
Scenario 2	**24.318** ±1. 053	30.802 ± 1.243
Scenario 3	**45.482** ±1. 103	70.921 ± 2.008
Scenario 4	**70.481** ±2. 125	155.713 ± 4.324

A second loop of experiments was then conducted by comparing CACOF with state-of-the-art algorithms for solving multi-objective path planning problems. To this end, SIGPA algorithm was used in the experiments (Ntakolia & Iakovidis, 2021b). SIGPA algorithm is a novel metaheuristic algorithm that proposes an alternative solution to multi-objective path planning compared to fuzzy logic. Similar to SIGPA, SIGPAF algorithm was also used in the experimental evaluation which is a fuzzy approach of SIGPA algorithm enhanced with Mamdani FIS used to solve multi-objective USV path planning problems (Ntakolia & Lyridis, 2021). Also, a commonly used graph-based algorithm to solve path planning problems, A* algorithm, and the popular solver for global optimization IBM ILOG CPLEX were used to test the quality of the path and the computational effort. Specifically, in this step of evaluation, the evaluation criteria between the comparative algorithms were the (i)-(iii) and (v) of the baseline comparison.

The models and the graph-based area were implemented in Python programming language. The computer used for the tests has 64-bit windows 10 Pro environment, with AMD Ryzen 7 3800X 8-Core Processor 3.89 GHz and 32 GB RAM.

4.1.2. Evaluation of local path planning

For the lower level, the obstacle detection and the obstacle avoidance methodologies are tested separately as it is illustrated in Fig. 10.

Evaluation methodology of obstacle detection

For the evaluation of the obstacle detection, an annotated dataset was used with photos taken by a walk in the National Park of Nea Philadelphia, Athens in Greece, where the real cases will be implemented in the imminent future. The dataset was composed of 764 images and 8 classes were selected for the annotation such as trees, staircases, benches, wall, lamps, persons, holes and cars. In total 2720 obstacles were included as objects in our dataset including 686 trees, 163 staircases, 152 benches, 230 walls, 621 lamps, 615 people, 155 holes and 98 cars.

A hold out mechanism was employed to evaluate the classification performance of the proposed object detection methodologies. Specifically, the dataset was partitioned into three stratified subsets (training, validation and testing) with 70%, 10% and 20% for each, respectively. The training subset was used for training the detection networks, the validation subset was utilized for optimising the networks and the final performance was calculated on the testing subset.

For each class, the following metrics are then calculated:

True Positive (TP(c)): a proposal was made for class c and there actually was an object of class c;

False Positive (FP(c)): a proposal was made for class c, but there is no object of class c;

The average precision (AP) for class c can be calculated by $AP(c) = TP(c)/(TP(c) + FP(c))$ whereas the mean average precision (mAP) is defined as given below:

$$mAP = \frac{1}{|Classes|}\sum_{c \in Classes} AP(c) \tag{26}$$

Mean average precision (mAP) was used as the performance metric to compare the performance of the different object detection approaches. The computational performance of the models was evaluated by calculating the frame per second (FPS) capacity.

Evaluation methodology of obstacle avoidance

Regarding, the comparison of the proposed obstacle avoidance algorithm, initially the baseline Bug2 algorithm was compared to the proposed Fuzzy Bug-like algorithm for proving the improvement of the algorithm by integrating fuzzy logic. In this step of evaluation, the following criteria were adopted:

(i) the objective criteria:
   a. the number of obstacle collisions;
   b. the number of brut turns that should be performed to avoid the detected obstacles; and
   c. the additional distance (compared to the original path) that the user should walk to avoid the detected obstacles
(ii) the overall local path quality calculated by the aggregation of the above objective terms by using the Mamdani FIS (defuzzification value)
(iii) the computation time
(iv) the obstacle avoidance success rate (SR_{lca}) in (25).

The distance is calculated in kilometers, the convergence speed in number of iterations, and the computation time in milliseconds.

Then, for a qualitative comparison of the developed local paths for obstacle avoidance, the popular Bug2 and RRT algorithms were used for a comparative experimental evaluation. Two experiments were performed within a selected path where 1 static obstacle was placed in the user's trajectory.

4.2. Results

4.2.1. Macroscale navigation

Initially a comparative evaluation between baseline algorithm ACO and CACOF was performed. Table 1 shows the mean path quality value, derived from the defuzzification value of Mamdani FIS, and standard deviation of the 20 runs for each scenario solved by the comparative algorithms, while Table 2 presents the mean computation time and standard deviation for these scenarios and runs. We should remind that the algorithms that achieve high path quality score (close to one) generate better routes in terms of the objective terms (criteria). Fig. 11 illustrates the evaluation of the algorithms with respect to the objectives criteria under examination for the generated paths for each scenario, while Fig. 12 depicts the convergence speed of ACO and CACOF in each scenario.

In general, CACOF outperforms the baseline algorithm ACO in all cases (Table 1) achieving better paths in terms of the criteria under examination (Fig. 11), with better speed convergence (Table 1, Fig. 12) and lower computational effort (Table 2). We observe that the paths generated by ACO algorithm traverse from more POIs compared to CACOF. However, this occurs due to multiple crossovers from the same POIs. Only in the first two scenarios (Scenario 1 and Scenario 2), that correspond to smaller graphs, the ACO algorithm manages to generate a path that passes from the same number of POIs as CACOF. Specifically, the up-left subfigure of Fig. 11 shows that CACOF generates paths that have less obstacles than ACO. A fact that contributes to the energy

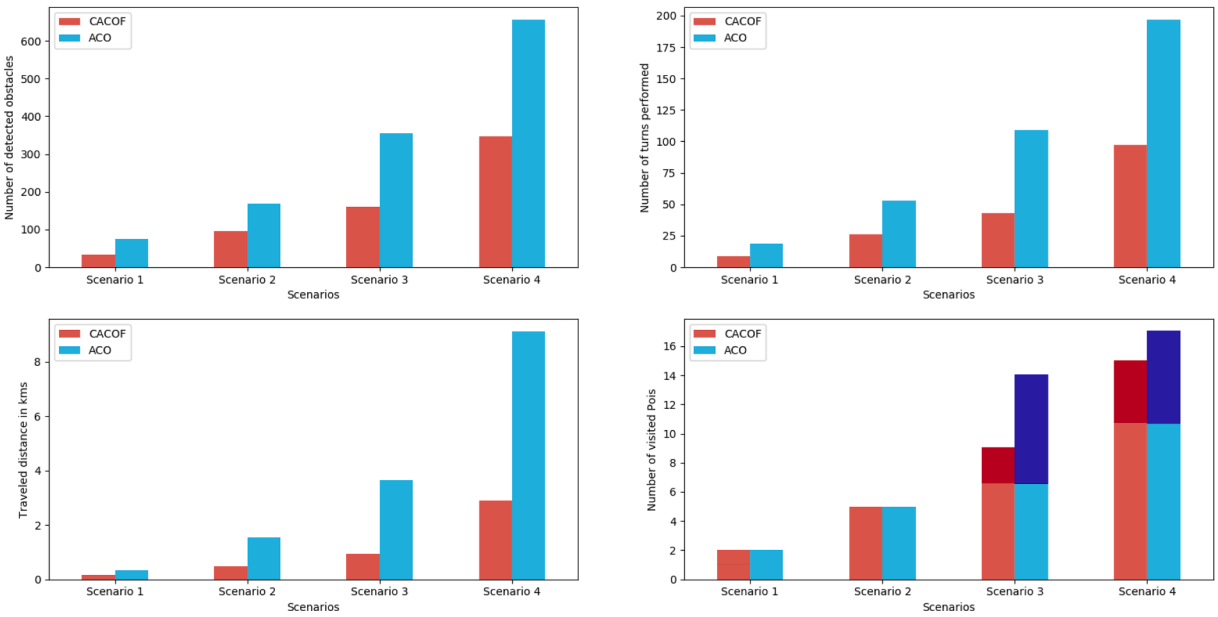


Fig. 11. The results from the generated paths for each scenario from ACO and CACOF. In the down and right plot the multiple crossovers are denoted with bold color.

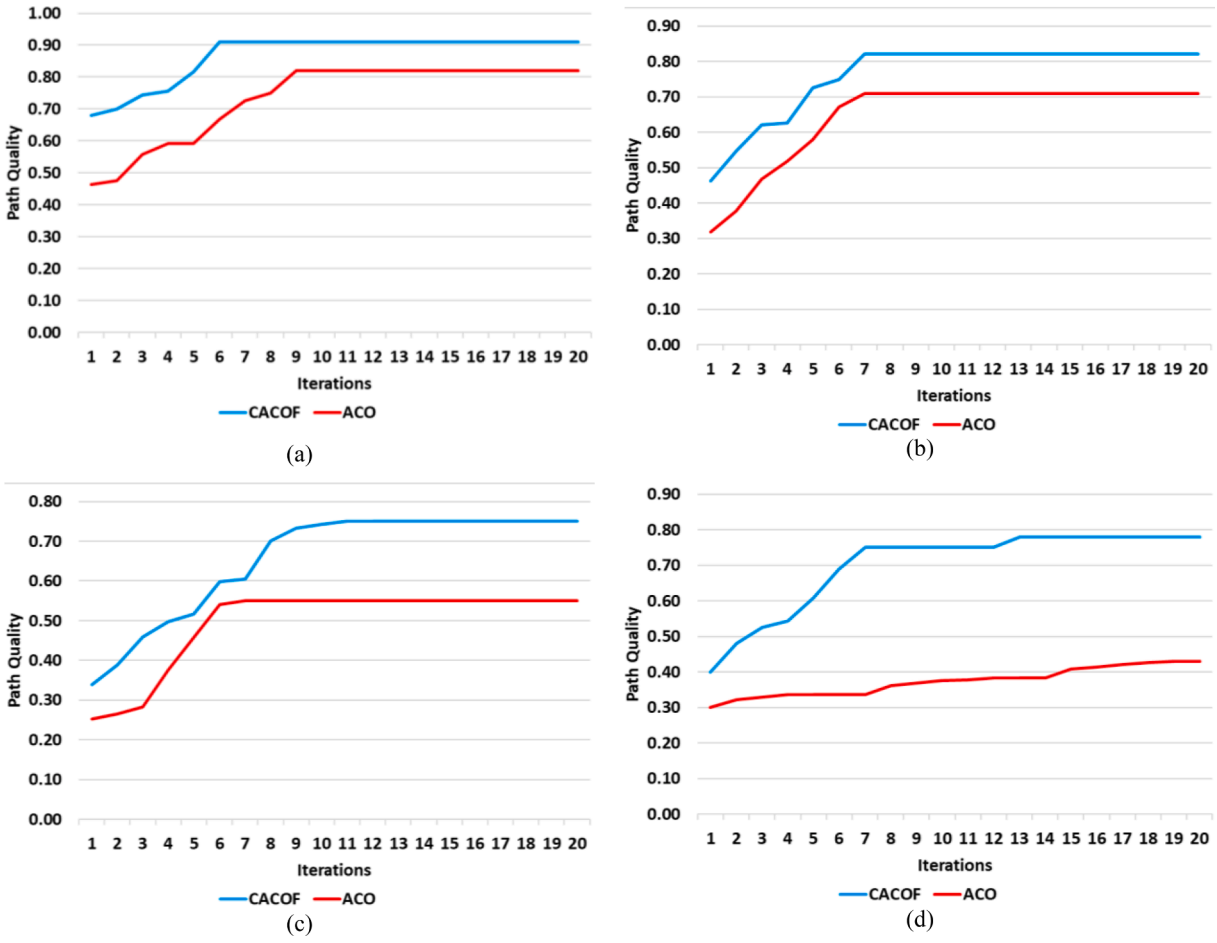


Fig. 12. Convergence of mean results from 20 runs of (a) Scenario 1; (b) Scenario 2; (c) Scenario 3; and (d) Scenario 4.

efficiency of system since there is not much need for obstacle detection and local path planning intervention for obstacle avoidance. This also decrease the users' effort for traversing an area. CACOF has proven effective in generating paths by avoiding sharp changes in the direction of the users (Fig. 11, up-right corner). This ability enhances the confidence of users to maintain their orientation. Last, CACOF is capable of generating shorter paths compared to ACO (Fig. 11, down-right corner). All the above combined, make CACOF a promising approach for dealing

Table 3
Mean values of CACOF, SIGPA, SIGPAF, A* and CPLEX algorithms.

Algorithm	Measures	Scenario 1	Scenario 2	Scenario 3	Scenario 4
CACOF	Compution time (ms)	**12.09**	**24.654**	**46.065**	**69.105**
	Path quality	**0.91**	**0.83**	**0.78**	**0.77**
	Number of obstacles	**47**	**96**	**238**	**392**
	Number of turns	**8**	**26**	**60**	**113**
	Traveled distance (kms)	**0.48**	**1.07**	**1.49**	**3.1**
	Number of multiple crosses from POIs	**0**	**0**	**1**	**4**
SIGPA	Compution time (ms)	12.121	29.353	57.534	78.012
	Path quality	0.91	0.81	0.75	0.71
	Number of obstacles	47	98	245	403
	Number of turns	8	27	64	124
	Traveled distance	0.48	1.16	1.57	3.9
	Number of multiple crosses from POIs	0	1	4	7
SIGPAF	Compution time (ms)	12.203	29.472	59.562	82.135
	Path quality	0.91	0.82	0.77	0.74
	Number of obstacles	47	96	241	398
	Number of turns	8	27	62	119
	Traveled distance	0.48	1.09	1.51	3.5
	Number of multiple crosses from POIs	0	0	2	6
A*	Compution time (ms)	13.428	29.774	63.044	88.178
	Path quality	0.89	0.80	0.72	0.70
	Number of obstacles	49	98	247	405
	Number of turns	9	29	66	122
	Traveled distance	0.52	1.21	1.59	4.01
	Number of multiple crosses from POIs	0	3	3	7
CPLEX	Compution time (ms)	96.872	240.655	560.123	1000
	Path quality	0.87	0.77	0.69	0.48
	Number of obstacles	49	101	249	425
	Number of turns	9	31	64	143
	Traveled distance	0.55	1.24	1.60	4.78
	Number of multiple crosses from POIs	1	5	5	12

with multi-objective graph-based problems for generating solutions that balance among the optimization criteria. Fig. 12 illustrates the mean convergence from the 20 runs with respect to the 20 iterations for each run. In all cases, CACOF presented a higher convergence speed reaching a better score (path quality) faster than ACO.

Table 3 summarizes the computation time and the path quality score of the competitive algorithms for each scenario based on the defuzzification value of Mamdani FIS. Also, the scores with respect to the objective criteria are also given. Based on the evaluation criteria of the objective function, CACOF manages not only to satisfy each one separately, but also to generates global paths of higher quality (better balance among the criteria) compared to the other SoA algorithms. Furthermore, as the dimensionality of the problem increases, CACOF algorithm proves its efficiency to generate routes of higher quality compared to the competitive algorithms. However, in case of global optimization solver CPLEX, a threshold in the computing time was set to 1 s that thought acceptable for a near real-time navigation. Response-time requirements for typical real-time application such as control systems processes and automation is set to $100\mu s - 100ms$ (Buttazzo, 2011). Therefore, the increase in the dimensionality of the path planning problems significantly decreases the accuracy of the solution achieving a relative tolerance MIP gap higher than 10%. The relative tolerance MIP gap calculates the difference between the best integer objective and the objective of the best node remaining and it is given by the expression: $\frac{|bestbound-bestinteger|}{1e-10+|bestinteger|}$ (Ntakolia & Iakovidis, 2021b). Indeed, CPLEX is used for global optimization and multi-objective path planning where all the objective terms can be expressed in the same unit or have the same magnitude. Hence, CPLEX solver did not succeed in finding the optimal path in order to balance among the optimization criteria. Also, apart from the least demanding scenario, Scenario 1, CPLEX could not find an optimal solution in the rest scenarios within the acceptable time requirements for real time applications $(100\mu s - 100ms)$.

Regarding the ability of all comparative algorithms, namely CACOF, ACO, SIGPA, SIGPAF, A* and CPLEX, Table 4 presents the collision avoidance success rate of all algorithms for the 4 scenarios. The best path derived by each algorithm for each scenario was selected for comparison. The results indicate the ability of CACOF algorithm to find paths with fewer obstacles and therefore lower risk for the VII to collide with obstacles or to perform additional actions to avoid an obstacle.

4.2.2. Microscale navigation

Evaluation of obstacle detection

To evaluate the performance of the proposed object detectors in the problem of obstacle detection, we initially collected 764 high resolution images of size 1280x720 pixels from the National Park of Philadelphia, Athens in Greece. The images were then manually annotated, collecting obstacles of different sizes from 8 different classes. The dataset images and their corresponding annotations were then used to train the object detectors in the problem of obstacle detection. Comparative results from between the two obstacle detectors are included in Table 5, where YOLOv5 achieves higher object detection accuracy (mAP of 71.93), maintaining the highest frame rate per second (FPS of 13.69). The experiments were conducted on an Android mobile device with 8 GB RAM and Exynos Octa core 9820 processor with max CPU frequency 2,73Ghz, using TensorFlow Lite (Abadi et al., 2015) mobile ML framework. Based on the reported results, YOLOv5 was selected as the obstacle detector in our study to be adopted in the proposed system. Sample images of the dataset along the associated object detection results are depicted in Fig. 13.

Evaluation of obstacle avoidance

Table 4
The results of collision avoidance success rate for the 4 scenarios of CACOF, ACO, SIGPA, SIGPAF, A* and CPLEX. The higher scores are denoted with bold.

		Comparative algorithms					
	Scenarios	CACOF	ACO	SIGPA	SIGPAF	A*	CPLEX
SR_{gca}(%)	Scenario 1	**94.60**	94.48	94.60	94.60	94.37	94.37
	Scenario 2	**88.97**	88.78	88.74	88.97	88.74	88.39
	Scenario 3	**72.64**	71.39	71.84	72.30	71.61	71.38
	Scenario 4	**54.94**	43.68	53.68	54.25	53.45	51.15

Table 5
Obstacle detection performance evaluation of the competing object detectors.

Network	AP								mAP	FPS
	trees	staircase	bench	wall	lamps	persons	holes	cars		
Detectron2	24.19	26.38	33.37	20.74	20.08	34.6	7.7	40.79	25.98	6.71
Scaled-YOLOv4	54.7	64.4	50.6	57.2	70.7	60.3	74.5	60.1	61.56	12.52
YOLOV5	58.9	71	85.2	71.5	67.9	59.2	74.5	87.2	**71.93**	**13.69**

Fig. 13. Indicative recognized objects from the created dataset of National Park of Nea Philadelphia, Greece.

Fig. 14. Sample images from the local path planning simulation.

To evaluate the performance of the proposed local path planning algorithm, we conducted qualitative experiments in a controlled simulated environment (Fig. 14), based on the same 4 scenarios used in the global path planning. Regarding the comparison of the proposed local path planner with popular mobile robot path planning Bug2 algorithm, Table 7 summarizes the computation time. Bug2 algorithm is popular for robotic motion path planning and obstacle avoidance along the pathway. When an obstacle is detected by the robot, the robot follows the counter of the obstacle until it finds again the initial trajectory. The developer chooses whether the robot will walk on the right or left counter side of the obstacle. In the experiments, the right side was chosen (Yufka & Parlaktuna, 2009).

The results show that Fuzzy Bug-like algorithm due to its simple philosophy and low complexity demands lower computational effort than the baseline Bug2 algorithm. This is more evident especially as the complexity of the scenarios and the size of the graphs increase. Moreover, Table 6 illustrates the qualitative evaluation based on the objective criteria: (i) number of obstacles needed to be avoided; (ii) number of brut angles that needed to be performed in order to avoid obstacles; and (iii) the increase of the length of the path compared to the predefined on from global path planner. The qualitative evaluation of the two algorithms showed that the proposed Bug-like algorithm enhanced with Fuzzy Logic, contributes to the development of local paths of better quality in case of obstacle avoidance with respect to the optimization criteria. Specifically, Fuzzy Bug-like algorithm guided the user to avoid less obstacles compared to Bug2. This is because Fuzzy Bug-like algorithm takes into account the obstacles in the local area with respect to a safe area and calculates a collision free path. Also, the smoothness factor in the objective helps improve the brut turns make the trajectory similar to human motion. Last, Fuzzy Bug-like algorithm aiming to minimize the

Table 6
The results from the generated local paths for obstacle avoidance for each scenario from Fuzzy Bug-like and Bug2.

Algorithm	Measures	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Fuzzy Bug-like	Number of obstacles in the path	**22**	**45**	**61**	**152**
	Number of brut turns	**48**	**53**	**89**	**198**
	Additional distance walked (km)	**2.12**	**3.96**	**7.84**	**7.82**
Bug2	Number of obstacles in the path	45	76	117	273
	Number of brut turns	81	79	150	286
	Additional distance walked (km)	3.89	5.91	12.70	12.78

Table 7
Mean computation time in milliseconds (ms) and standard deviation of the 20 runs for each scenario solved by Fuzzy Bug-like and Bug2. The best values are shown in bold.

	Computation time (ms)	
Scenarios/Algorithms	Fuzzy Bug-like	Bug2
Scenario 1	**8.341** ± 0.034	9.231 ± 0.081
Scenario 2	**12.343** ±0. 058	19.732 ± 0.105
Scenario 3	**23.836** ±0. 090	36.765 ± 0.613
Scenario 4	**31.044** ±1. 012	58.197 ± 0.524

Table 8
The results of local collision avoidance success rate for the 4 scenarios of Fuzzy Bug-like and Bug2. The higher scores are denoted with bold.

		Comparative algorithms	
	Scenarios	Fuzzy Bug-like	Bug2
SR_{lca}(%)	Scenario 1	**53.19**	4.26
	Scenario 2	**53.13**	20.83
	Scenario 3	**74.37**	50.84
	Scenario 4	**61.22**	30.36

deviations from the predefined global path by avoiding the obstacles with the optimal safe distance, managed to minimize the additional distance that the user should walk in order to avoid obstacles in the path in all cases compared to Bug2. Overall, the proposed Fuzzy Bug-like algorithm presented a stable and robust performance in all scenarios compared to Bug2 algorithm, proving that it can be considered as a potential approach to multi-objective local path planning. Regarding the local collision avoidance success rate, Table 8 presents the results for Fuzzy Bug-like and Bug2. The CACOF best path for each scenario was taken as the global path where the number of total obstacles within the path is indicated in Table 3. The target is to collide with and, consecutively, need to avoid as fewer obstacles as possible. The results proved that the proposed Fuzzy Bug-like algorithm has the ability not only to avoid obstacles with smoother trajectories of shortest additional distance but also during this process, it takes into account the position of detected obstacles within a specified range so that the updated local path will not collide with these obstacles.

To make a qualitative comparison of the proposed obstacle avoidance algrithm with popular Bug2 and RRT algorithms, two examples were implemented to present the constructed obstacle avoidance paths. Fig. 15 and Fig. 16 illustrate two examples of obstacle avoidance in case of static obstacles, such as a lamp and a bench, respectively, where a global path is already obtained. Fig. 15a and Fig. 16a show the initial global path; Figs. 15-16b, Figs. 15-16c and Figs. 15-16d present the local path developed by Bug2, RRT and Fuzzy Bug-like algorithms, respectively, for the two examples of obstacle avoidance. In both examples, we can see that Bug2 perform more turns and brut changes to avoid the obstacle following the contour of the obstacle until the user finds the initial path again. RRT performs significantly better than Bug2 however, the user needs to walk longer distance with less smoothness in the path deviation in order to avoid the obstacle and reach the next node in the graph. On the other hand, Fuzzy Bug-like algorithm's path takes into account the initial node, the obstacle position and the target node to alter locally the initial global path in order to avoid the obstacle achieving the best trade-off between the objective terms (objective criteria).

5. Conclusions and Future work

In this study a novel autonomous path planning methodology was proposed for smart assistive systems. The path planner is composed by two hierarchical levels: (i) the global path planner for the macroscale navigation; and (ii) the local path planner for microscale navigation. Given a semi-unknown environment, the global path planner uses a metaheuristic approach to find a global path in case of multiple POIs. For this, a novel hybrid metaheuristic algorithm, CACOF, based on Ant Colony Optimization that incorporates Fuzzy Logic and Chaotic maps is developed. CACOF is capable of developing paths with multi-objectives. The local path planner consists of a DNN detector, (YOLOv5) to detect in real time obstacles among the global path and a bug-like algorithm with Fuzzy Logic for safe obstacle avoidance in case of static obstacles. The aim of this study is to develop a holistic approach towards path planning for smart assistive systems for individuals with disabilities. However, the presented system can also be adopted to autonomous vehicles for a low-cost employment of a path planner system.

Tests were conducted under simulation environment based on the topology of the National Park of Nea Filadelfia in Athens, Greece. The results showed that the proposed CACOF algorithm outperformed in all cases the baseline ACO algorithm, by developing better paths with respect to the objectives. Also, CACOF proved to be computationally more efficient compared to popular metaheuristic and global optimization algorithms, a fact that constitutes it suitable for real time applications. Based on the global paths, similar tests were conducted in the same simulation environment for local path planning and obstacle avoidance. The presented bug-like algorithm with Fuzzy Logic was compared to the baseline Bug2 algorithm. The results showed that the proposed algorithm needed less computational effort to develop local paths of better quality with respect to the objectives. Last, an experimental evaluation of various DNN detectors was conducted with YOLOv5 achieving the best detection performance in the developed dataset of 8 classes: trees, staircases, benches, walls, lamps, people, holes and cars.

Due to COVID-19 limitations, the present experimental evaluation was conducted on simulation environment. Future work includes the implementation of real experiments in the National Park of Nea Filadelfia in Athens, Greece. For this reason, a prototype of the smart wearable assistive system (SWAS) is under implementation. The design is realized based on the user-centered designed approach and the relevant requirements derived from a similar study (Ntakolia, Dimas, & Iakovidis, 2020). An initial attempt to design the SWAS prototype is illustrated in Fig. 15. Future work includes the development of a simulation area where various avatars will walk simultaneously with different local path planning algorithms for a 3D comparative evaluation with visualization aid, such as colored path and vectors. Moreover, to extend our work of the simulation environment, a virtual reality training application will be implemented to support the individuals with impairment to get familiar with such smart assistive systems in a safe environment, such as their homes, before using it in real world. Furthermore, the proposed methodology of local path planning will be

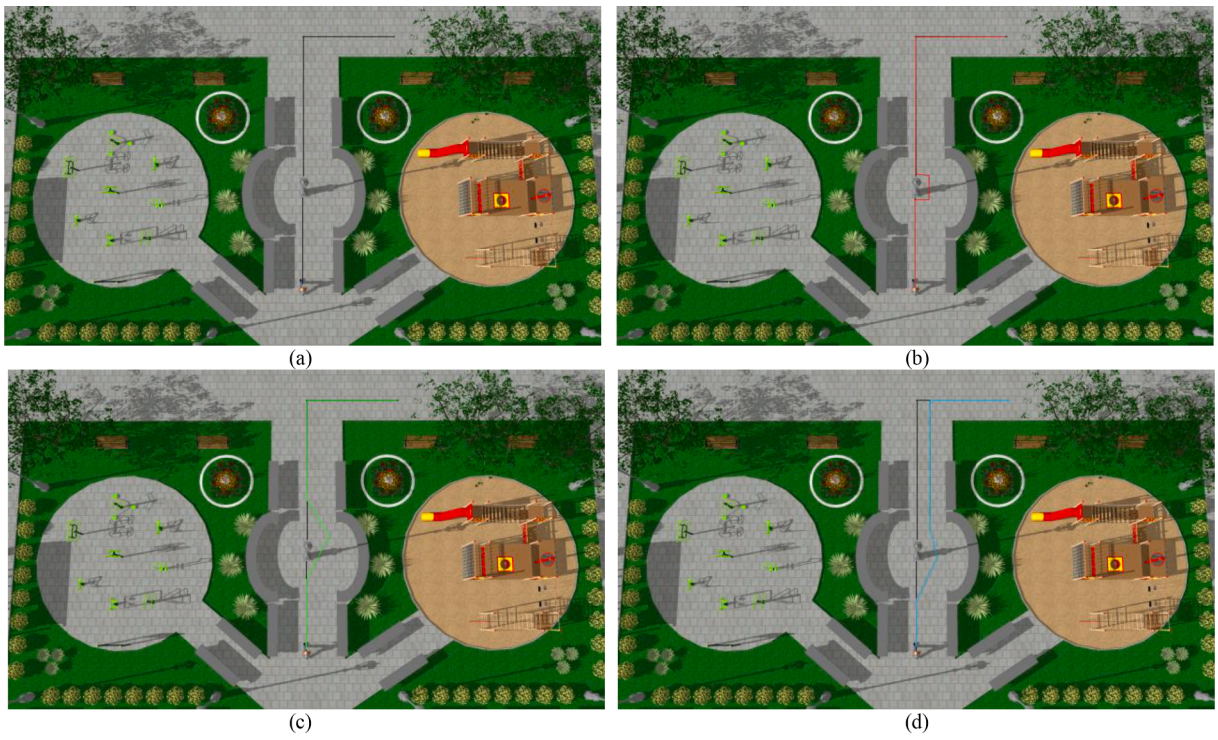


Fig. 15. Example of obstacle (lamp) avoidance of: (b) Bug2 algorithm; (c) RRT algorithm and (d) Fuzzy Bug in case of the initial global path (a).

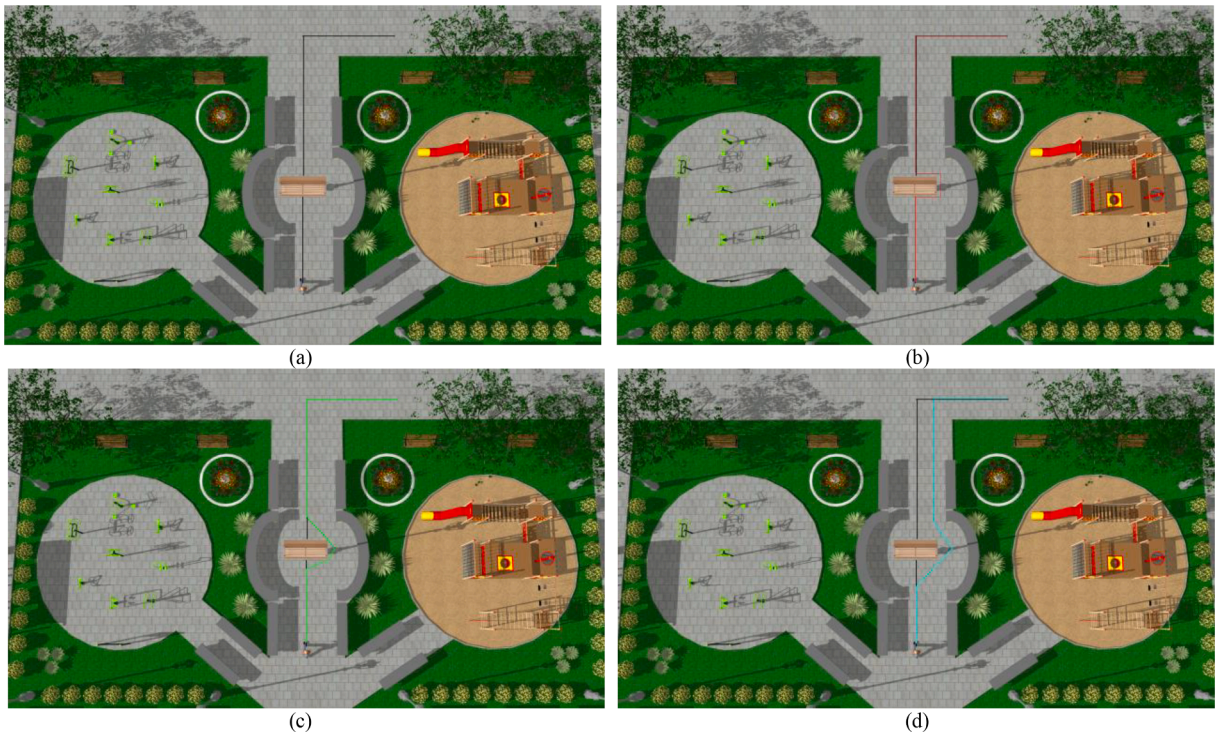


Fig. 16. Example of obstacle (bench) avoidance of: (b) Bug2 algorithm; (c) RRT algorithm and (d) Fuzzy Bug in case of the initial global path (a).

extended to moving obstacles in real conditions. Future work also includes the comparative evaluation of CACOF with state-of-the-art ACO algorithms that have been proposed in the literature for solving multi-objective problems, adapting them for the specific application.

CRediT authorship contribution statement

Charis Ntakolia: Conceptualization, Methodology, Software, Writing – original draft, Visualization, Validation, Formal analysis, Supervision. **Serafeim Moustakidis:** Methodology, Software, Validation, Writing – original draft, Visualization, Writing – review & editing. **Athanasios Siouras:** Software, Validation, Writing – original draft.

Fig. 17. A 3D representation of an initial design of the SWAS prototype.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., et al. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Retrieved from https://www.tensorflow.org/.

Ahmetovic, D., Oh, U., Mascetti, S., & Asakawa, C. (2018). Turn right: Analysis of rotation errors in turn-by-turn navigation for individuals with visual impairments. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (pp. 333–339).

Akiyoshi, K., Chugo, D., Muramatsu, S., Yokota, S., & Hashimoto, H. (2020). Autonomous mobile robot navigation considering the pedestrian flow intersections. In *2020 IEEE/SICE International Symposium on System Integration (SII)* (pp. 428–433). IEEE.

Alomari, A., Phillips, W., Aslam, N., & Comeau, F. (2017). Swarm intelligence optimization techniques for obstacle-avoidance mobility-assisted localization in wireless sensor networks. *IEEE Access, 6*, 22368–22385.

Arora, S., & Anand, P. (2019). Chaotic grasshopper optimization algorithm for global optimization. *Neural Computing and Applications, 31*(8), 4385–4405.

Balata, J., Mikovec, Z., & Slavik, P. (2018). Landmark-enhanced route itineraries for navigation of blind pedestrians in urban environment. *Journal on Multimodal User Interfaces, 12*(3), 181–198.

Bevilacqua, P., Frego, M., Bertolazzi, E., Fontanelli, D., Palopoli, L., & Biral, F. (2016). Path planning maximising human comfort for assistive robots. In *2016 IEEE Conference on Control Applications (CCA)* (pp. 1421–1427). IEEE.

Bochinski, E., Eiselein, V., & Sikora, T. (2017). In *High-speed tracking-by-detection without using image information* (pp. 1–6). IEEE.

Buttazzo, G. C. (2011). *Hard real-time computing systems: predictable scheduling algorithms and applications* (Vol. 24). Springer Science & Business Media.

Campbell, S., O'Mahony, N., Carvalho, A., Krpalkova, L., Riordan, D., & Walsh, J. (2020). Path planning techniques for mobile robots a review. In *2020 6th International Conference on Mechatronics and Robotics Engineering (ICMRE)* (pp. 12–16). IEEE.

Chen, J., Zhou, Y., Lv, Q., Deveerasetty, K. K., & Dike, H. U. (2018). A review of autonomous obstacle avoidance technology for multi-rotor UAVs. In *2018 IEEE International Conference on Information and Automation (ICIA)* (pp. 244–249). IEEE.

Chen, Y., Tang, J., Jiang, C., Zhu, L., Lehtomäki, M., Kaartinen, H., et al. (2018). The accuracy comparison of three simultaneous localization and mapping (SLAM)-based indoor mapping technologies. *Sensors, 18*(10), 3228.

Fadzli, S. A., Abdulkadir, S. I., Makhtar, M., & Jamal, A. A. (2015). Robotic indoor path planning using dijkstra's algorithm with multi-layer dictionaries. In *2015 2nd International Conference on Information Science and Security (ICISS)* (pp. 1–4). IEEE.

Farhadi, A., & Redmon, J. (2018). Yolov3: An incremental improvement. *Computer Vision and Pattern Recognition*. cite as.

Fernandes, H., Costa, P., Filipe, V., Paredes, H., & Barroso, J. (2019). A review of assistive spatial orientation and navigation technologies for the visually impaired. *Universal Access in the Information Society, 18*(1), 155–168.

Fernandes, P. B., De Oliveira, R. C. L., & Neto, J. V. F. (2018). A modified QPSO for robotic vehicle path planning 2018. In *2018 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1–7). IEEE.

Garcia, M. P., Montiel, O., Castillo, O., Sepulveda, R., & Melin, P. (2009). Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. *Applied Soft Computing, 9*(3), 1102–1110.

Han, X., & Liu, C. (2019). Design of Autonomous Navigation Algorithm For Security Inspection Robot. 2019 International Conference on Computer, Network, Communication and Information Systems (CNCI 2019) (pp. 1–9). Atlantis Press.

Iakovidis, D. K., Diamantis, D., Dimas, G., Ntakolia, C., & Spyrou, E. (2020). *Digital enhancement of cultural experience and accessibility for the visually impaired. Technological Trends in Improved Mobility of the Visually Impaired* (pp. 237–271). Springer.

Imran, M., & Kunwar, F. (2016). A hybrid path planning technique developed by integrating global and local path planner. In *2016 International Conference on Intelligent Systems Engineering (ICISE)* (pp. 118–122). IEEE.

Jain, S., & Malhotra, I. (2020). A Review on Obstacle Avoidance Techniques for Self-Driving Vehicle. *International Journal of Advanced Science and Technology, 29*(6), 5159–5167.

Joseph, S. L., Xiao, J., Zhang, X., Chawda, B., Narang, K., Rajput, N., et al. (2015). Being aware of the world: Toward using social media to support the blind with navigation. *IEEE Transactions on Human-Machine Systems, 45*(3), 399–405.

Kamil, F., Tang, S., Khaksar, W., Zulkifli, N., & Ahmad, S. (2015). A review on motion planning and obstacle avoidance approaches in dynamic environments. *Advances in Robotics & Automation, 4*(2), 134–142.

Kim, P., Chen, J., Kim, J., & Cho, Y. K. (2018). *SLAM-driven intelligent autonomous mobile robot navigation for construction applications* (pp. 254–269). Springer.

Kudriashov, A., Buratowski, T., Giergiel, M., & Malka, P. (2020). *SLAM Techniques Application for Mobile Robot in Rough Terrain*. Springer.

Lee, J. (2017). Heterogeneous-ants-based path planner for global path planning of mobile robot applications. *International Journal of Control, Automation and Systems, 15*(4), 1754–1769.

Lentzas, A., & Vrakas, D. (2020). LadyBug. An intensity based localization bug algorithm. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)* (pp. 682–689). IEEE.

Li, B., Muñoz, J. P., Rong, X., Chen, Q., Xiao, J., Tian, Y., et al. (2018). Vision-based mobile indoor assistive navigation aid for blind people. *IEEE transactions on mobile computing, 18*(3), 702–714.

Li, Y., Dai, S., Shi, Y., Zhao, L., & Ding, M. (2019). Navigation simulation of a Mecanum wheel mobile robot based on an improved A* algorithm in Unity3D. *Sensors, 19*(13), 2976.

Liu, D., Gao, S., Chi, W., & Fan, D. (2021). Pedestrian detection algorithm based on improved SSD. *International Journal of Computer Applications in Technology, 65*(1), 25–35.

Liu, H., Xu, B., Lu, D., & Zhang, G. (2018). A path planning approach for crowd evacuation in buildings based on improved artificial bee colony algorithm. *Applied Soft Computing, 68*, 360–376.

Liu, J., Yang, J., Liu, H., Tian, X., & Gao, M. (2017). An improved ant colony algorithm for robot path planning. *Soft Computing, 21*(19), 5829–5839.

Liu, Y., Wang, Y., Wang, S., Liang, T., Zhao, Q., Tang, Z., & Ling, H. (2020). Cbnet: A novel composite backbone network architecture for object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 11653–11660).

Lu, H., Wang, X., Fei, Z., & Qiu, M. (2014). The effects of using chaotic map on improving the performance of multiobjective evolutionary algorithms. *Mathematical Problems in Engineering, 2014*.

Luckcuck, M., Farrell, M., Dennis, L. A., Dixon, C., & Fisher, M. (2019). Formal specification and verification of autonomous robotic systems: A survey. *ACM Computing Surveys (CSUR), 52*(5), 1–41.

Mac, T. T., Copot, C., Tran, D. T., & De Keyser, R. (2016). Heuristic approaches in robot path planning: A survey. *Robotics and Autonomous Systems, 86*, 13–28.

Mahida, P. T., Shahrestani, S., & Cheung, H. (2017). Localization techniques in indoor navigation system for visually impaired people. In *2017 17th International Symposium on Communications and Information Technologies (ISCIT)* (pp. 1–6). IEEE.

Mahida, P. T., Shahrestani, S., & Cheung, H. (2018). Comparision of pathfinding algorithms for visually impaired people in IoT based smart buildings. In *2018 28Th International Telecommunication Networks and Applications Conference (ITNAC)* (pp. 1–3). IEEE.

Mamdani, E. H. (1974). Application of fuzzy algorithms for control of simple dynamic plant. In *Proceedings of the institution of electrical engineers, 121* pp. 1585–1588). IET.

Manjari, K., Verma, M., & Singal, G. (2020). A survey on assistive technology for visually impaired. *Internet of Things, 11*, Article 100188.

McGuire, K. N., de Croon, G., & Tuyls, K. (2019). A comparative study of bug algorithms for robot navigation. *Robotics and Autonomous Systems, 121*, Article 103261.

Mellal, M. A., & Williams, E. J. (2018). A survey on ant colony optimization, particle swarm optimization, and cuckoo algorithms. In *Handbook of research on emergent applications of optimization algorithms* (pp. 37–51). IGI Global.

Minetto, R., Kozievitch, N. P., da Silva, R. D., Almeida, L. D. A., & de Santi, J. (2016). Shortcut suggestion based on collaborative user feedback for suitable wheelchair route planning. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 2372–2377). IEEE.

Muhawenayo, G., & Gkioxari, G. (2021). Compressed Object Detection. arXiv preprint arXiv:2102.02896.

Nandini, D., & Seeja, K. (2019). A novel path planning algorithm for visually impaired people. *Journal of King Saud University-Computer and Information Sciences, 31*(3), 385–391.

Ntakolia, C., Dimas, G., & Iakovidis, D. K. (2020). User-centered system design for assisted navigation of visually impaired individuals in outdoor cultural environments. *Universal Access in the Information Society*, 1–26.

Ntakolia, C., & Iakovidis, D. K. (2021a). A route planning framework for smart wearable assistive navigation systems. *SN Applied Sciences, 3*(1), 1–18.

Ntakolia, C., & Iakovidis, D. K. (2021b). A swarm intelligence graph-based pathfinding algorithm (SIGPA) for multi-objective route planning. *Computers & Operations Research, 133*, Article 105358.

Ntakolia, C., & Lyridis, D. V. (2021). A swarm intelligence graph-based pathfinding algorithm based on fuzzy logic (SIGPAF): A case study on unmanned surface vehicle multi-objective path planning. *Journal of Marine Science and Engineering, 9*(11), 1243. https://doi.org/10.3390/jmse9111243

Ntakolia, C., & Lyridis, D. V. (2022). A comparative study on Ant Colony Optimization algorithm approaches for solving multi-objective path planning problems in case of unmanned surface vehicles. *Ocean Engineering, 255*, Article 111418. https://doi.org/10.1016/j.oceaneng.2022.111418

Patle, B., Pandey, A., Parhi, D., Jagadeesh, A., et al. (2019). A review: On path planning strategies for navigation of mobile robot. *Defence Technology, 15*(4), 582–606.

Pratama, P. S., Nguyen, T. H., Kim, H. K., Kim, D. H., & Kim, S. B. (2016). Positioning and obstacle avoidance of automatic guided vehicle in partially known environment. *International Journal of Control, Automation and Systems, 14*(6), 1572–1581.

Purian, F. K., & Sadeghian, E. (2013). Mobile robots path planning using ant colony optimization and Fuzzy Logic algorithms in unknown dynamic environments. In *2013 International Conference on Control, Automation, Robotics and Embedded Systems (CARE)* (pp. 1–6). IEEE.

Raja, P., & Pugazhenthi, S. (2012). Optimal path planning of mobile robots: A review. *International Journal of Physical Sciences, 7*(9), 1314–1320.

Saeedi, S., Trentini, M., Seto, M., & Li, H. (2016). Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics, 33*(1), 3–46.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510–4520).

Sangeetha, V., Krishankumar, R., Ravichandran, K. S., Cavallaro, F., Kar, S., Pamucar, D., et al. (2021). A fuzzy gain-based dynamic ant colony optimization for path planning in dynamic environments. *Symmetry, 13*(2), 280.

Tapu, R., Mocanu, B., & Zaharia, T. (2018). *Wearable assistive devices for visually impaired: A state of the art survey*. Pattern Recognition Letters.

Vaishnav, D., Rao, B. R., & Bade, D. (2021). Wearable assistance device for the visually impaired. In *Advances in Machine Learning and Computational Intelligence* (pp. 667–676). Springer.

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2020). Scaled-YOLOv4: Scaling Cross Stage Partial Network. arXiv preprint arXiv:2011.08036.

Wolf, D. F., & Sukhatme, G. S. (2005). Mobile robot simultaneous localization and mapping in dynamic environments. *Autonomous Robots, 19*(1), 53–65.

Xiao, J., Joseph, S. L., Zhang, X., Li, B., Li, X., & Zhang, J. (2015). An assistive navigation framework for the visually impaired. *IEEE transactions on Human-Machine Systems, 45*(5), 635–640.

Yen, C.-T., & Cheng, M.-F. (2018). A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance. *Microsystem Technologies, 24*(1), 125–135.

Yufka, A., & Parlaktuna, O. (2009). Performance comparison of bug algorithms for mobile robots. Proceedings of the 5th international advanced technologies symposium, Karabuk, Turkey (pp. 13–15).

Zhang, H., Butzke, J., & Likhachev, M. (2012). Combining global and local planning with guarantees on completeness. In *2012 IEEE International Conference on Robotics and Automation* (pp. 4500–4506). IEEE.

Zhang, X., Xiao, J., Li, B., Muñoz, P., Joseph, S. L., Sun, Y., et al. (2016). A Wearable Indoor Navigation System with Context Based Decision Making for Visually Impaired. *Int. J. Adv. Robot. Autom, 1*, 1–11.

Zhang, X., Yao, X., Zhu, Y., & Hu, F. (2019). An ARCore based user centric assistive navigation system for visually impaired people. *Applied Sciences, 9*(5), 989.