International Journal of Robotics and Automation, Vol. 39, No. 5, 2024

BEHAVIOUR-DEFINED NAVIGATION FRAMEWORK FOR DYNAMICAL OBSTACLE AVOIDANCE IN MULTI-ROBOT SYSTEMS CONSISTING OF HOLONOMIC ROBOTS

Tahniat Khayyam,[*] S.G. Ponnambalam,[**]
Mukund Nilakantan Janardhanan,[***] and Izabela Ewa Nielsen[****]

Abstract

Dynamical obstacle avoidance is a challenging problem in the field of autonomous robot navigation. Current research in this field has been mostly limited to single robots, thus, there exists a gap in research in the field of dynamical obstacle avoidance in multi robot systems. While rich literature is available on multirobot systems, this paper attempts to propose a novel navigation framework for an environment which includes multiple robots. The proposed navigation framework applies certain behaviours to ensure a safe trajectory for the multi-robot systems. As opposed to other reported literature which focused on implementing their algorithms on non-holonomic robots, the proposed navigation framework is implemented on several holonomic robots. Simulations and real-life experiments were carried out using the proposed framework. Dynamic obstacles are considered in the environment and Khepera IV robots are used to conduct real-life experiments. Two dynamic obstacles were placed at different positions in the workspace. These obstacles had linear movements, whereby each robot could move horizontally and vertically across the workspace. Three experimental trials were performed. Results show that the proposed navigation framework is successful in navigating the multi-robot system to their respective target locations while avoiding obstacles.

Key Words

Multi-robot systems, holonomic robots, dynamic obstacle, obstacle avoidance

* Multronica Solutions LLP, 522-C Gulgasht Colony, Multan, Pakistan; e-mail: tkhay200@gmail.com
** School of Mechanical Engineering, Vellore Institute of Technology, Vellore, India; e-mail: ponnambalam.g@vit.ac.in
*** Warwick Manufacturing Group, International Digital Laboratory, University of Warwick, Coventry CV4 7AL, UK; e-mail: mukund-nilakantan.janardhanan@warwick.ac.uk
**** Department of Materials and Production, 9220 Aalborg, Denmark; e-mail: izabela@mp.aau.dk
Corresponding author: S.G. Ponnambalam

(DOI: 10.2316/J.2024.206-1055)

1. Introduction

Navigation of multiple robots in a dynamic environment with moving obstacles are exposed to the likelihood of collisions. Most research reported focuses on dynamic obstacle avoidance for single robots rather than multi-robot and mostly considers static obstacles rather than dynamical obstacles. This paper proposes a behaviour-defined navigation framework for a multi-robot system consisting of holonomic robots. The navigation framework allows the robots to reach their destination while avoiding collisions with the dynamic obstacles present in the environment. The dynamical obstacles move in a linear manner across the environment, and the multi-robot system consists of two to three robots. Simulation results are conducted, and the results show that the proposed navigation framework is successful in guiding the multi-robot system to their respective targets. The holonomic robots being used for the experiment trials are the Khepera IV robots. These are equipped with IR sensors, Ultrasonic sensors, and Wi-Fi connectivity.

2. Literature Review

Rich literature is available in the field of navigation in multi-robot systems. Literature survey had been conducted and the literature collected has been grouped into five categories, articles based on evolutionary and swarm-based approaches, articles with the use of potential field method, articles use the concept of multi-agent systems, articles based on reinforcement learning approaches, and a collection of other articles with other approaches are grouped under collision avoidance in uncertain multi-robot systems.

2.1 Evolutionary and Swarm-based Approach

Particle swarm optimisation was used to carry out obstacle avoidance in multi-robot systems in a work reported by Di Mario and Martinoli [1]. They implemented an adaptation process to optimise the algorithmic parameters and the experiments are conducted by considering the wheel speed, the wheel speed difference, and the activation value of proximity sensors using Khepera III robots. An Improved artificial immune algorithm to avoid obstacles in multi-robot systems was proposed by Yuan *et al.* [2]. Once an obstacle attacks the network of B-cells (multi-robot system), it triggers the formation of antibodies which could result in stimulation and suppression. The authors also consider deadlock situations in multi-robot systems, whereby, a roulette wheel is used to choose another antibody and to move out of a deadlock situation. A combination of the artificial potential field (APF) method and enhanced genetic algorithm (EGA) is used to plan collision free paths for a multi-robot system by Nazarahari *et al.* [3]. Researchers used bio-inspired algorithms to handle dynamic obstacles in multi-robot systems. Xue and Ma [4] used ant colony optimisation (ACO) to form a navigation algorithm. Savkin and Wang [5] used simple navigation rules that are often found in biology, which use the concept of a vision cone to maintain a constant avoiding angle from all obstacles.

2.2 Potential Field Method

Path planning using artificial potential field method is implemented on multi-robot systems by Zhaofeng and Ruizhe [6]. The destination unreachable problem arises when an obstacle or several obstacles are close to a target. Thus, making the robot unable to reach the destination as the repulsive forces of the obstacles are then larger than the gravitational force. APF method was extended to have dynamical obstacle avoidance properties by Mbede *et al.* [7] hereby a hybrid algorithm was presented which used fuzzy logic and APFs together for robot navigation. The APF method is prone to the local minima problem. This issue was addressed by Ge and Cui [8] where a decision-making capability was incorporated once the robot and target are close to an obstacle. The vector field histogram method was extended to avoid non-static obstacles by adding a time dependent look ahead tree by Ge and Cui [8]. Pseudo-bacterial potential field (PBPF) method created by Orozco-Rosas *et al.* [9]. This method allows for faster convergence to an optimised path that avoids collisions with obstacles.

2.3 Multi-agent Systems

Cifuentes *et al.* [10] proposed an algorithm wherein the multi-robot systems maintain the formations by using virtual fields. The virtual fields also consider the possibility of collisions with obstacles. Pan *et al.* [11] proposed an approach to maintain the formations in a multi-robot system with a virtual spring model between the robots, targets, and the static obstacles. Nascimento *et al.* [12] used a nonlinear model predictive formation control (NMPFC) as a controller for the multi-robot system, which has two sub-blocks, namely an optimiser and a predictor. The optimiser's aim is to minimise the cost function which includes the obstacle avoidance ability and formation maintenance ability. Li *et al.* [13] proposed an algorithm which focused on the navigation control of a multi-robot system in an environment with unknown static obstacles. The multi-robot system maintains formation while navigating through the environment. This method also eliminates the deadlock and local minima problem. Dai *et al.* [14] and De La Cruz and Carelli [15] proposed an approach to maintain the multi-robot system formation whilst moving to their target locations. Both approaches could only avoid static obstacles, not dynamical obstacles. Xu *et al.* [16] suggested to use artificial moments in the robots' controllers to induce repulsive moments away from static obstacles. Simulation results involving three robots, shows that the proposed algorithm successfully navigates all three robots to their target points without any collisions with the static obstacles in the environment.

Wei *et al.* [17] proposed an algorithm for a multi-robot system's self-assembly. An initial virtual region, where robots are randomly placed, is shrunken until the target virtual region is obtained. Cena *et al.* [18] assigned priorities to the robots in the multi-robot system in their paper. Each individual agent uses the A* algorithm for path planning.

Savkin and Wang [19] created navigation algorithm whereby the robot uses a laser range finder and ultrasonic sensors to determine if an obstacle falls within its sensing range. Divya Vani *et al.* [20] proposed an autonomous navigation system for a multi-robot transportation application in indoor environments. Lafmejani and Berman [32] presented an online nonlinear Model Predictive Control (MPC) method for collision-free, deadlock-free navigation by multiple autonomous nonholonomic Wang *et al.* [21] proposed a collision avoidance algorithm based on velocity obstacles (VO) is proposed for distributed mobile robots to achieve oscillation-free autonomous navigation, which is called as repulsion-oriented reciprocal collision avoidance (RORCA). Wen Pang *et al.* [22] proposed a time-varying formation reconfiguration and obstacle avoidance strategy are investigated for a fleet of autonomous underwater vehicles (AUVs) in the three-dimensional (3-D) ocean environment.

2.4 Reinforcement Learning-based Approach

Neural networks and Q-learning were used to plan safe trajectories for multi-robot systems in dynamic environments [23]. Fan *et al.* [24] used a reinforcement learning to plan the individual trajectories of robots in a multi-robot system. These robots possessed no a priori knowledge of the dynamic and unstructured environment they were in. 3-D metric maps are updated using RGB-data obtained by the robots in [25].

2.5 Collision Avoidance in Multi-robot Systems

Fiorini and Shiller [26] considered prediction of the obstacles' trajectories in the VO approach. This approach involves using a velocity obstacle set that contains all the velocities which may result in collision with an obstacle, within a given time duration. Wu and How [27] improved the velocity obstacle approach in their work. The velocity obstacle approach was merged with a reachability set. The reachability sets find the regions in the environment that will result in a collision. This method allowed for several obstacle trajectories' to be included in the set rather than a single trajectory. Otte and Frazzoli [28] extended the RRT algorithm to the RRTX algorithm whereby the robot generates a search-tree at each sampling time, rather than iteratively growing the search-tree. The search tree contains information about the obstacles' potential trajectories. Aoude *et al.* [29] proposed an algorithm known as the RR-GP solution, wherein, the RRT method is implemented with a Gaussian Process which gives it the ability to predict the obstacle's future trajectories by using pattern-based approach. Shahriari and Biglarbegian [30] developed a navigation methodology for a distributed scheme by incorporating the robots' dynamics through calculating the time to collision (TTC) and designing a new controller accordingly that avoids collisions.

Lafmejani and Berman [31] presented an online nonlinear MPC method for collision-free, deadlock-free navigation by multiple autonomous nonholonomic wheeled mobile robots (WMRs). Zhu *et al.* [32] proposed a decentralised and communication-free method for probabilistic multirobot collision avoidance in cluttered environments. The method considers robot localisation and sensing uncertainties and relies on the computation of buffered uncertainty-aware voronoi cells (B-UAVC). Xin Li *et al.* [33] proposed an adaptive sliding mode method to track the formation for a multi-AUV system in the water flow environment. Moghaddam *et al.* [34] addressed the trajectory planning of a serial robot which is used in the spot-welding process of an automobile industry.

A concurrent multi-objective dynamic optimisation method is proposed for optimal selection and control of synchronous ac servomotors. They optimised three main objectives, such as energy consumption, tracking error, and total weight of motors [35]. Optimisation algorithms such as grey wolf optimisation have been utilised to fine tune parameters in the field of mechanical applications.

Rigorous mathematical analyses are provided by Belkhouche [36] where a concept of virtual plane is suggested. Sezer and Gokasan [37] presented a mathematical analysis-based algorithm where robots seek to navigate through the obstacles rather than avoid them completely. This is done by finding the largest gap between obstacles and heading at an angle halfway through the gap.

2.6 Research Gap

Based on an extensive literature review, it is observed that there exists a gap in research in the field of autonomous robot navigation, with regards to dynamical obstacle avoidance involving multi-robot systems. Most of the research involving dynamical obstacle avoidance has been limited to single robot systems. In addition, the literature considering multi-robot systems is largely focused on maintaining formations rather than the issue of dynamical obstacle avoidance. Further research should be done to create a navigation framework that allows dynamical obstacle avoidance in a multi-robot system.

2.7 Objectives of the Research

The main objective of this paper is to propose a navigation framework for dynamical obstacle avoidance in multi-robot systems. The proposed framework uses certain behaviour-defining rules which set each robot's motion to move closer to the target and avoid dynamical obstacles. In case of imminent collisions with the obstacles, a heading is calculated so that the robot can safely move away from the obstacle. This navigation framework also considers a method to avoid robot-robot collisions. This navigation framework was tested in real-life experiments with Khepera IV robots. The following objectives have also been considered in this paper:

1. Navigate a holonomic robot to its target location without colliding with the moving obstacles and other robots in the environment.
2. The path planning should be done in real-time at a local level, without prior knowledge of the obstacles' trajectories.

A few assumptions have been made which are summarised below:

1. Each robot has a different target and starting location.
2. There are two moving obstacles in the workspace, and they have simple motions. That is, they move only in left and right directions.
3. There are only two robots being considered for the multi-robot system.

This paper is structured as follows. Section 3 discusses the Navigation Framework, Section 4 presents the simulation of the navigation framework and Section 5 discusses the implementation of the framework in real-life experiments.

3. Navigation Framework

3.1 Sensing Disk Model

The navigation framework models the sensing region of the robot as a disk, as shown in Fig. 1. The sensing region is the region in which the sensors on the robot can detect obstacles. The origin of the sensing region is denoted by $Disk_x$and $Disk_y$. The coordinates of $Disk_x$ and $Disk_y$ are found by (1) and (2), where x and y are the coordinates of the robot and r represents the radius of the sensing disk. θ_r is the current heading of the robot. In this case, the sensing disk radius r is taken to be twice of the radius of the robot.

$$\text{Diskx} = x + r * \cos\theta r \tag{1}$$

$$\text{Disky} = y + r * \sin\theta r \tag{2}$$

3.2 Linear Navigation Strategy

At every step of the movement, the framework checks if there could be imminent collisions with the moving obstacles present in the environment. If there aren't any dynamical obstacles near the robot, the robot's next heading is calculated using the equations shown in

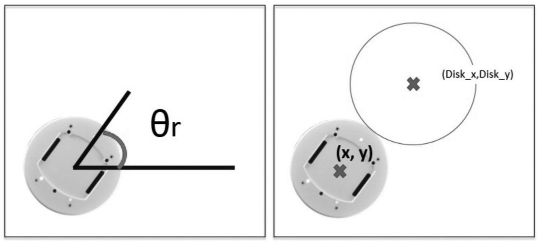


Figure 1. Sensing disk model.

(3) and (4).

$$\text{Horizontal Distance} = x \text{ robot} - x \text{ target} \quad (3)$$
$$\text{Vertical Distance} = y \text{ robot} - y \text{ target} \quad (4)$$

Where, x_{robot} and y_{robot} are the current coordinates of the robot. (x_{target}, y_{target}) represent the target position for the robot in question. The behaviour for the linear strategy is summarised in Figs. 2 and 3. Figure 2 shows how the linear navigation strategy chooses the heading for the robot. The horizontal and vertical distance refers to the distance to the target [shown in (3) and (4)]. Figure 3 shows how the navigation strategy allows for linear motion in the leftwards, rightwards, upwards, and downwards direction. This is done to approach the target location and is only used if there aren't any obstacles nearby. The angles in Fig. 3 represent the robot's future headings in each of the cases is shown in Fig. 2.

3.3 Obstacle Avoidance Strategy

When obstacles are nearby and there is a likelihood of collision, the robot must calculate a new heading that effectively removes the risk of collision. At every step, the sensing disk and obstacle disk are checked to see if they are intersecting, if they are, then, the respective x–y coordinates of those intersection points are calculated. The concept of "circle-circle" intersection was used. The equations were derived using the methods explained in Mathworld [38]. The distance between both circles is found using the square root of *dist* in (5). There exists a relationship among the *a, b, h,* and *dist* variables (annotated on the diagram in Fig. 4). r_1 and r_2 are the respective radiuses of the circles. The relationship among the variables is defined in (6) and (7). The obstacle radius (r_1) is taken to be twice the radius of the robot (r_2). The obstacle's center coordinates (e, f) are known in this environment; hence, those values are used in the equations. The rest of the parameters of an obstacle disk such as intersection coordinates are then determined using (16)–(19).

$$dist = (c - e)2 + (d - f)2 \quad (5)$$
$$dist = a + b \quad (6)$$
$$a = \frac{r_1^2 - r_2^2 + dist}{2\sqrt{dist}} \quad (7)$$

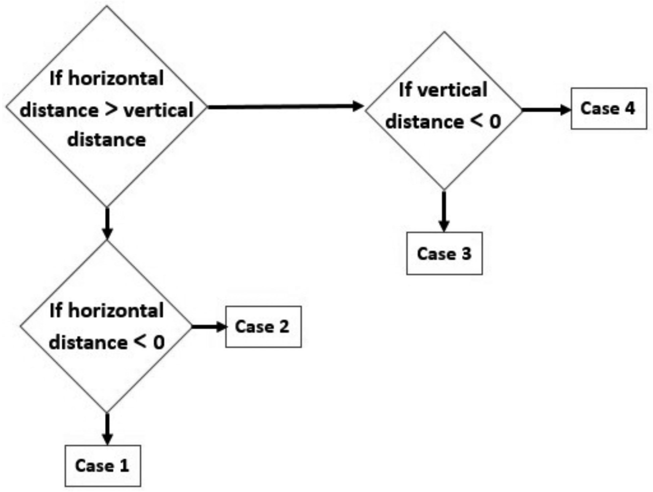

If horizontal distance > vertical distance
If vertical distance < 0
Case 4
Case 3
If horizontal distance < 0
Case 2
Case 1

Figure 2. If-Else conditions for linear navigation strategy.

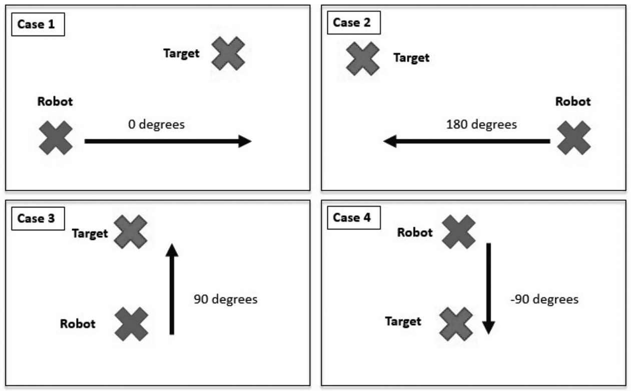


Figure 3. The simple linear navigation strategy's various possible headings.

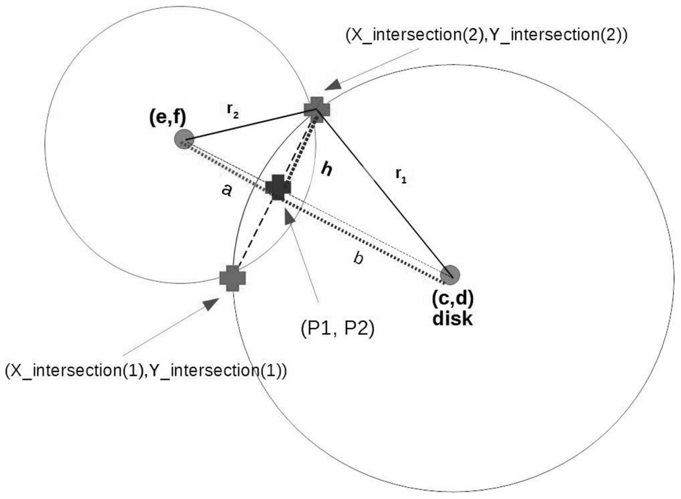

(X_intersection(2),Y_intersection(2))
(e,f)
r2
h
r1
a
b
(c,d)
disk
(P1, P2)
(X_intersection(1),Y_intersection(1))

Figure 4. Annotated diagram of two circles intersecting.

The h is further simplified in terms of r_1, r_2, and *dist* variables.

$$h^2 = r_2^2 - a^2 = r_1^2 - b^2 \quad (8)$$

Where, $h^2 = \frac{2*\text{dist}\left(r_1^2+r_2^2\right)-(r_2^2-r_2^2)^2-\text{dist}^2}{\text{dist}}$, and it could be written as,

$$h = \sqrt{\frac{2*\text{dist}\,(r_1^2+r_2^2)-(r_2^2-r_1^2)^2-\text{dist}^2}{\text{dist}}} \tag{9}$$

The numerator of h can be further factorised to the equation denoted by the variable $\boldsymbol{t}$ (10). The numerator was factorised for the purpose of simplicity. The relationship between h and t is summarised with the (10).

$$h = \frac{\sqrt{t}}{\sqrt{\text{dist}}} \tag{10}$$

Where,

$$t = ((r_1+r_2)^2 - \text{dist}) * (\text{dist} - (r_2-r_1)^2) \tag{11}$$

The intersection points are defined by (12) and (13). The variables *(c,d)* and *(e,f)* refer to the coordinates of each circle's origin.

$$X_{\text{intersection}} = P1 \pm \frac{h}{2\sqrt{\text{dist}}}(d-f) \tag{12}$$

$$Y_{\text{intersection}} = P2 \pm \frac{h}{2\sqrt{\text{dist}}}(e-c) \tag{13}$$

P1 and *P2* refer to the coordinates that fall halfway through the intersection points. These are calculated using (14) and (15).

$$P1 = \frac{c+e}{2} + \frac{(e-c)(r_2^2-r_1^2)}{2*\text{dist}} \tag{14}$$

$$P2 = \frac{d+f}{2} + \frac{(f-d)(r_2^2-r_1^2)}{2*\text{dist}} \tag{15}$$

Finally, after substituting t and simplifying the equations in (12) and (13), the resulting equations (16)–(19) can help find the coordinates of the two intersection points.

$$X1_{\text{intersect}} = P1 + (d-f)\left(\frac{\sqrt{t}}{2*\text{dist}}\right) \tag{16}$$

$$X2_{\text{intersect}} = P1 - (d-f)\left(\frac{\sqrt{t}}{2*\text{dist}}\right) \tag{17}$$

$$Y1_{\text{intersect}} = P2 + (e-c)\left(\frac{\sqrt{t}}{2*\text{dist}}\right) \tag{18}$$

$$Y2_{\text{intersect}} = P2 - (e-c)\left(\frac{\sqrt{t}}{2*\text{dist}}\right) \tag{19}$$

Where c, d correspond to the (x,y) coordinates of the sensing disk, and e and f correspond to the (x,y) coordinates of the obstacle. r_1 and r_2 correspond to the radius of the circle.

The angle between the points of intersection and the robot's current position has to be determined because this helps in calculating the robot's heading for the next iteration. The central angle theorem helps in determining those angles. The theorem dictates that the central angle will be twice the size of the angle in the triangles drawn on the circle. The angles are found using the cosine formula.

The next step involves calculating the heading of the robot using the angles α and β. α refers to the angle between the nearest intersection point and the robot, and $(\alpha+\beta)$ refers to the angle between the furthest intersection point and the robot. The two possible formulas are given below. The latter one (21) is only applicable when the robot isn't intersecting the first quadrant of the sensing disk.

$$\text{Heading} = \alpha + \beta \tag{20}$$

$$\text{heading} = \frac{2\pi - 2\theta_r - \alpha - \beta}{4} \tag{21}$$

Where, θ_r is the robot's current orientation, α is the angle to the nearest intersection point, and β is the angle between both intersection points. The heading chosen by the robot will ensure that no collision takes place with any nearby dynamic obstacle in the next iteration.

The heading equation (20) is only chosen when the obstacles are in the 4^{th} quadrant. In all other cases, (21) is used to calculate the heading.

3.4 Robot-Robot Collision Avoidance Strategy

As each robot in the system plans their path individually, there is a likelihood of collisions among themselves. Hence, an avoidance strategy is implemented to mitigate that possibility. In the proposed framework, one robot has a higher priority than the other one. The robot's priority is assigned arbitrarily beforehand. The Euclidean distance between the higher priority robot's future location and lower priority robot's future location is calculated, if it is less than a safe distance, then, the lower priority robot stops and recalculates its heading at the next iteration. However, if the higher priority robot has reached its location, then, the navigation framework will treat them like an obstacle and calculate their new heading using the "obstacle avoidance strategy".

3.5 Go-To-Target Mode

Afterwards, the angle between the target location and robot's current location is calculated using (22)–(25). The variables used are shown in Fig. 5.

$$A = \sqrt{(T_x - x + 2)^2 + (T_y - y)^2} \tag{22}$$

$$B = \sqrt{(x - x + 2)^2 + (y - y)^2} \tag{23}$$

$$C = \sqrt{(T_x - x)^2 + (T_y - y)^2} \tag{24}$$

$$\theta_t = \frac{C^2 + B^2 - A^2}{2*B*C} \tag{25}$$

The robot then changes its current heading to move towards the target location. The Go-To Target mode is only applied if the robot is sufficiently close to the target location.

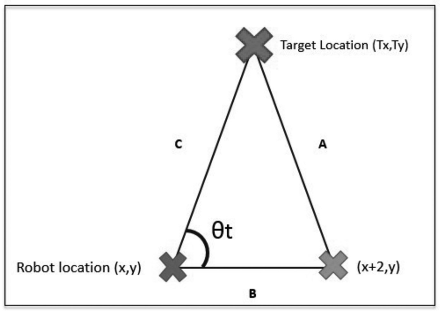


Figure 5. Image displaying the variables used to calculate the angle θ_r.

3.6 Novelty of Proposed Navigation Framework

The proposed navigation framework was established while considering the capabilities of the Khepera IV robot and the size of the workspace. It alters differently from the other algorithms as it uses IR sensors, as opposed to laser range finders and isn't computationally expensive. In addition, it can cater for multi-robot systems and as the Khepera IV robot is holonomic, the consideration of angular velocities is eliminated as the robot can rotate on its own axis. One lesser known and recently proposed algorithm in Savkin and Wang [24] is also used for comparison purposes. This is used because it serves as an inspiration to the proposed navigation framework, however, the proposed navigation framework has considerable differences from it. Such as the original algorithm works on the condition that the obstacles' velocities are less than that of the robots. This condition doesn't exist in the Navigation Framework, as unlike the algorithm mentioned, the robot avoids the obstacle by moving away at a certain angle. In addition, as the navigation framework will be implemented on a holonomic robot, it does not require the need to consider angular velocities, but rather moves in a linear motion parallel to either the X or Y axis.

4. Simulation of Proposed Navigation Framework

Simulations were carried out using C++ on a 4GB RAM computer powered with Intel(R) Core i5-4200U CPU with 1.6GHz. In all simulations, the robots could converge close to their respective target locations. If a robot is within 0.2 cm of the target location, then, it has reached its target. The dynamical obstacle's motions differ in each simulation. Each simulation has different starting positions and target positions. In simulation 1, the obstacles move upwards and downwards and move parallel to each other. In simulations 2 and 3, the obstacles move in different directions. One moves upwards/downwards and the other leftwards and rightwards. The experiments are repeated for 20 runs for all the three simulations. The average results of 20 runs obtained from these experiments are reported in this study.

4.1 Simulation 1

It takes a total of 25 steps for the simulation to conclude. Robot 1 reaches first, robot 2 reaches second followed by robot 3. In step 3, robot 2 and 3 have moved away from obstacle 1. In step 11, robot 1 has reached their target. In step 22, robot 2 has reached its target. Meanwhile, robot 3 keeps moving closer to its target until step 25.

Table 1 shows the initial positions and the results of simulation 1. It is observed that the standard deviation for the distance to target values for Robots 2 and 3 are very small, this is because the robots are able to consistently converge towards to the target. Robot 1's distance to target values have a slightly higher standard deviation because there are more fluctuations in the values.

4.2 Simulation 2

Table 2 shows the initial positions and the results of simulation 2. Robot 3's standard deviation is slightly on the higher side. This can be owed to the fact there were some runs where Robot 3 would converge to values larger than threshold (0.2 m). Robot 2's values are consistent and always reach very close to the target. The values of the final positions have been rounded in the table. Robot 1's average distance to target was less than the threshold, yet the standard deviation was low as the values were consistent.

4.3 Simulation 3

Table 3 shows the initial positions and the results of simulation 3. It takes a total of 24 steps for all the robots to reach their target locations. At iteration 24, robot 2 undergoes the "Go-To-Target" mode and reaches its target location. There were some runs in Simulation 3 where Robot 2 and 3's final distance was exceeding the threshold. Robot 3's standard deviation is 0.1660.

4.4 Effects of Velocity on Number of Steps

While carrying out the simulations, it is observed that sometimes the robots had trouble converging to their respective target positions. Hence, the following simulation results aim to identify the relationship between the velocity values and the number of steps it took for the robot to reach the target position. Robots' velocities were altered in simulations as shown in Table 4. If the distance to target was greater than 2 units, the speed was set to 2 unit/step. In this case the robot converges to the target location. However, when the speed was set to 5 units/step, to robot doesn't seem to converge and keeps on oscillating between two locations. Thus, showing that the velocity of the robot's also affects the efficiency of the navigation framework.

5. Implementation of Navigation Framework

The robots used to conduct the experiments are Khepera IV. The programming was carried out in the Eclipse IDE for C/C++ Programmers. The codes were then sent *via* Wi-Fi to the robot's Linux environment. Also, the robots

Table 1
Initial Positions and Results of Simulation 1

Robot Number	Starting Point (cm)	Target Location (cm)	Final Position of in Simulation (cm)	Number of Steps	Final Distance to Target (cm)	Average Distance to Target (cm)	Standard Deviation
Robot 1	20,12	8,10	7.89,10.06	11	0.1253	0.1308	0.0160
Robot 2	18,20	9,12.5	9.02,12.58	22	0.0825	0.0809	0.0085
Robot 3	1,5	10,25	9.86,24.95	25	0.1487	0.1480	0.0029

Table 2
Initial Positions and Results of Simulation 2

Robot Number	Starting Point (cm)	Target Location (cm)	Final Position in Simulation (cm)	Number of Steps	Final Distance to Target (cm)	Average Distance to Target (m)	Standard Deviation
Robot 1	13,2	4,10	4.21,10.05	48	0.2159	0.1388	0.0686
Robot 2	12,2	12,12	11,12	9	3.5900E-08	3.5900E-08	0.0000
Robot 3	18,12	10,5	10,5.3	15	0.0900	0.2000	0.2966

Table 3
Initial Positions and Results of Simulation 3

Robot Number	Starting Point (cm)	Target Location (cm)	Final Positioning Simulation (cm)	Number of Steps	Final Distance to Target (cm)	Average Distance to Target(m)	Standard Deviation
Robot 1	1,2	4,20	4,20	7	3.5900E-08	3.5900E-08	6.8344E-16
Robot 2	2,4	17,12	16.83,12.10	24	0.1972	0.1995	0.0916
Robot 3	10,1	13,10	12.67,10.71	9	0.7829	0.1722	0.1660

sent their coordinates *via* Wi-Fi using a TCP/IP protocol. The Khepera IV robots can carry out speed control and has Wi-Fi connectivity as well. They are also equipped with sensors, such as gyroscopes, IR sensors, ultrasonic sensors, and encoders.

The multi-robot system has levels of priority assigned to each robot. These allow for a robot-robot collision avoidance strategy to be enabled. In the experiments, only two robots were being used, one of them has the higher priority and the other one has a lower priority with slight variations in their navigation framework.

5.1 Navigation Framework for Higher Priority Robot

Once the IR sensors detect the presence of an obstacle, the dynamical obstacle avoidance mode is initiated, whereby, it either rotates at a heading of 135 degree or 225 degrees. If the IR sensors do not detect the presence of obstacles, the navigation framework checks to see it is close to the target, if that is not the case either, but the robot's distance to the target is less than 15 cm, then, the speed is reduced. If the robot's distance to the target is more than 15 cm, then, it enters the simple linear navigation mode, where it tries to cover the maximum horizontal or vertical distance to reach the target.

Table 4
Effect of Changing the Velocity Affects Results.

Velocity if distance to target is >2	Velocity if distance to target is <2	Number of steps
2	1	33
5	1	Doesn't Converge

5.2 Navigation Framework for Lower Priority Robot

It is like the navigation framework of the higher priority robot, however, the only difference is that, if the obstacle detection condition is not true, then, it calculates and checks if the distance between the lower and higher-priority robot is less than a threshold distance, which is set to 50 cm. If it is less than the threshold distance, the lower priority robot stops for a single step, and recalculates its heading in the next step.

5.3 Minor Adjustments to Theoretical Navigation Framework to Conduct Real Life Experiments

To implement the Navigation Framework in real life experiments, the constraints of the experimental workspace

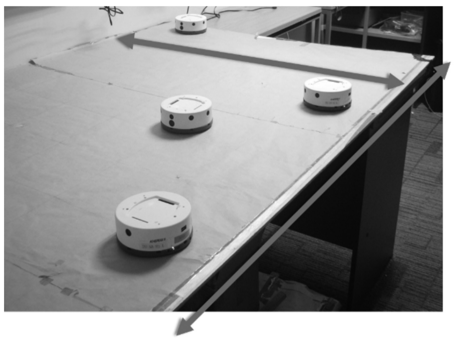


Figure 6. Workspace where the robots moved around.

▬ Higher Priority Robot

◇ Lower Priority Robot

✸ Dynamical Obstacle

Figure 7. Icons for robots and obstacles.

have to be taken into consideration. In addition, during the experimentation phase, there were certain inaccuracies that were discovered, which led to the navigation framework for real-life experiments is to be altered slightly, this involved simply eliminating the Go-To Target Mode, having the lower priority robot stop if there is imminent danger of it colliding with the higher priority robot, and restricting the obstacle avoidance mode to only two possible orientations. The navigation frameworks use the simple linear navigation strategy to get closer to the target position.

The sensing region was modelled as a disk in the theoretical framework, but in the experiments the sensing region was as per the IR sensor's sensing region. The former was modelled as a disk so that it would be easier to calculate obstacles' position in a MATLAB simulation as there is no sensor doing the "detection" in the virtual world. In experiments, this constraint is no longer there, IR sensors can directly sense distance to the obstacles. Ultimately, the guiding principle in both are the obstacles' positions. That is what triggers the dynamical obstacle avoidance mode. Hence, minor adjustments made do not alter the working principle of navigation framework.

5.4 Experimental Results

Three real-life experiments were carried out using Khepera IV robots. The initial positions of the robots were changed each time. The threshold for the minimum distance between the target and robots was set to 15 cm. The experiments were carried out on the work bench made from wooden board which is shown in Fig. 6. Two robots were placed at differing locations in the workspace, and the other two served as the dynamical obstacles. The dynamical obstacles moved in a linear motion, moving from left side of the board to the right end constantly through each trial. The icons used to distinguish higher priority robot, lower priority robot, and dynamic obstacles for the robots used in the experiments are shown in Fig. 7.

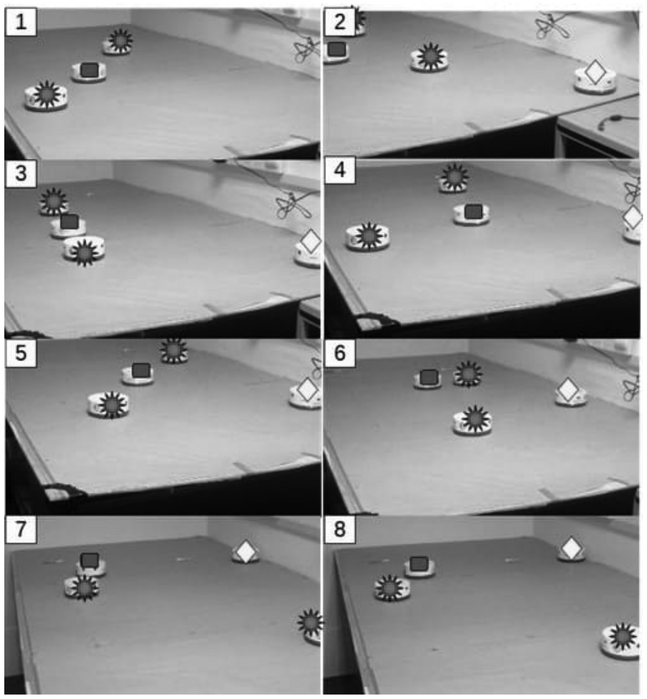


Figure 8. Images of Experiment 1.

5.4.1 Experiment 1

In Fig. 8, one can see that the higher priority robot was placed in the top-middle part of the workspace, while the lower priority robot was placed at the bottom of the workspace. In image 2, the higher priority robot moves forward. Images 3 and 4 show the higher priority robot detecting the dynamical obstacle in front and moving away at 225 degrees. In images 4–8, both robots start moving closer to their target using the simple linear navigation strategy. Throughout the trial, the lower priority robot keeps checking the risk of imminent collisions and as there are none, it completes the whole trial run using only the simple linear navigation strategy. The final distance between the lower priority robot and target was 12.076 cm. While the distance between the higher priority robot and its target was noted to be 9.610 cm.

5.4.2 Experiment 2

In Fig. 9, both the higher and lower priority robots were placed at the bottom of the workspace. Initially, the higher priority robot moves first and then goes into the obstacle avoidance mode in image 2. It then moves upwards again and engages in the obstacle avoidance mode again, as shown in image 6. This was a false positive because of the noise present on the IR sensors. It then engages in the Simple Linear Navigation Mode and reaches its target in image 10. Likewise, the lower priority robot uses the same strategy and gets close to its target in image 9. However, there is some distance between the two because the robot accumulated rotation errors during the trial, as sometimes it wasn't parallel to the X or Y axis. In addition, the robot goes into obstacle avoidance mode once again because of

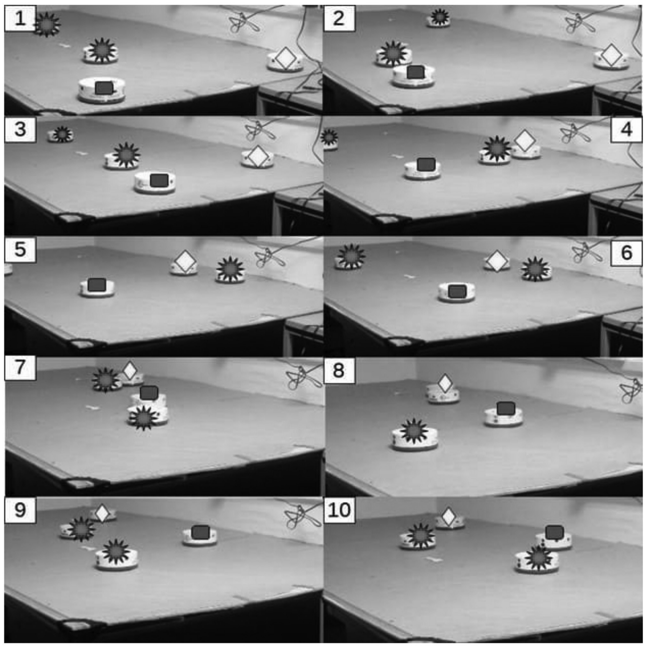


Figure 9. Images of Experiment 2.

the fluctuations in the IR sensor values. It is observed that there were rotation errors during the run and thus, there is still some gap between the robot and target at the end of the run.

5.4.3 Experiment 3: Trials without Obstacle Avoidance Modes

Two experiment trial images are shown in Figs. 10 and 11. This is to show two instances where the robots moved around the workspace and reached their respective targets without engaging into the obstacle avoidance modes. In Fig. 10, in images 1–4, both the lower priority and higher priority robots are moving upwards in the workspace. In image 5, the higher priority robot uses the simple linear navigation strategy and moves rightwards, to get closer to its target. In image 6, the lower priority robot has reached its target. Throughout the whole run, neither robot goes into obstacle avoidance modes. The robots had the same initial positions as in experiment 2, yet their trajectory was different. Thus, every trial run would be different each time. Figure 11's first image shows the initial positions of the two robots. In this trial run, they all move upwards and reach their respective targets. Once again, they do not encounter any dynamical obstacles in their path and thus do not engage in the obstacle avoidance mode.

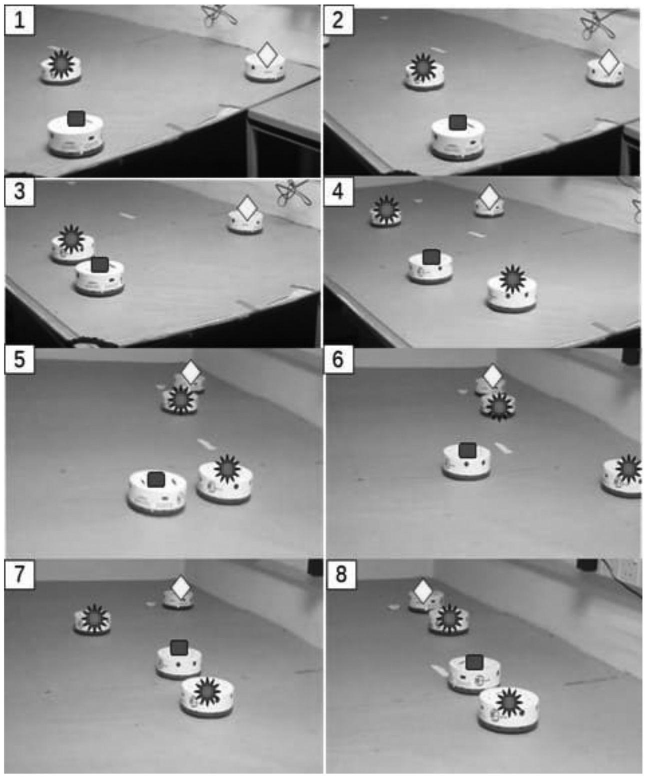


Figure 10. Images of trial where robots did not encounter obstacles.

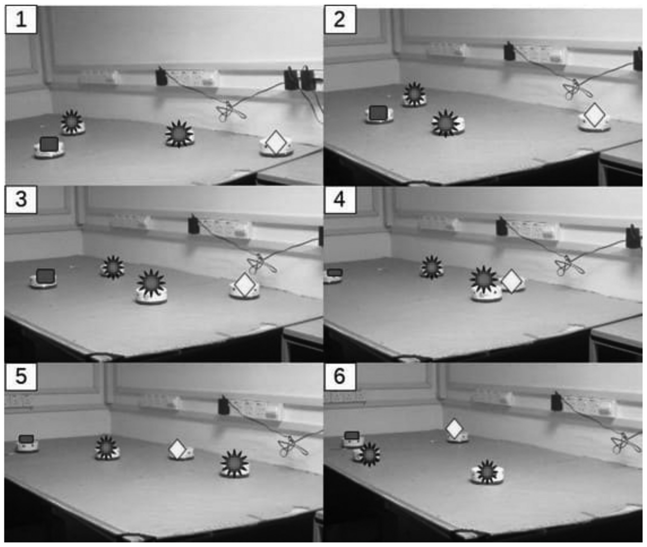


Figure 11. Images of trial where robots moved in a straight line.

6. Discussion

The robots were able to reach their target locations without colliding with the moving obstacles or other robots in all the three experiments. Furthermore, robots were able to navigate in the environment without any prior knowledge of the obstacles' trajectories. There are cases where the robots don't fully converge to their target locations. This can be attributed to the rotation inaccuracies that don't allow the robot to fully align itself parallel to the axes, which would lead to the accumulation of errors. This causes the robot to drift from its perceived path. The odometer sensors sometimes couldn't perceive their exact position as they were unable to discern if the robot had made errors in its rotation. The "distance-to-target" values calculated did not take the rotation errors and position errors into account and thus, the robots were not always able to converge completely to their target locations. In addition, there was noise present on the IR sensors which gave fluctuating results and sometimes gave false positives. Eventually leading the robots to start the obstacle avoidance mode, even if there were no robots present in its path.

7. Conclusion

The challenge in the field of autonomous robot navigation with regards to dynamical obstacle avoidance was addressed in this paper. The navigation framework was implemented on Khepera IV robots and the constraints of the workspace and inaccuracies in rotation were considered, and alterations were made in the theoretical navigation framework to make it suitable for implementation in a real-life experiment. In most cases, the robots converge to the target location, however, there have been cases where the rotation errors have hindered the robot from reaching the target location. There also have been rare instances in real-life experiments where collisions occurred with one of the dynamical obstacles. These are due to the noise present on the IR sensors. The theoretical navigation framework, however, fares better in navigating the robots in a dynamic environment.

There is an accumulation of rotation errors that affect the accuracy of the results. A trade-off had to be found between the sampling time and the speed of the robot. If the robot's speed is kept slow, it may be more likely to detect the presence of obstacles nearby, however that increases the duration of the experiment. If it is too fast, it may fail to detect the presence of obstacles, and this might result in collision. Due to workspace constraints and the rotation errors that arose, there was a considerable amount of difference between the experimental navigation framework and the theoretical navigation framework.

For future improvements, one could consider eliminating the limitations of the experimental setup to improve the accuracy of the navigation framework and implement the framework in its true form. Using a smoothing filter such as a Kalman filter on the IR sensors would most likely reduce the inaccuracies by a large margin. The current study considers the obstacle movement in horizontal and vertical movements only. Obstacle movements in arbitrary directions could be considered in future work. To conclude, the navigation framework is mostly successful in guiding a multi-robot system to its target locations while moving in a dynamic environment.

References

[1] E. Di Mario and A. Martinoli, Distributed particle swarm optimization for limited-time adaptation with real robots, *Robotica, 32*, 2014, 193–208.

[2] M. Yuan, Y. Jiang, X. Hua, B. Wang, and Y. Shen, A real-time immune planning algorithm incorporating a specific immune mechanism for multi-robots in complex environments, *Proc. of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering,* 2017, 29–42.

[3] M. Nazarahari, E. Khanmirza, and S. Doostie, Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm, *Expert Systems with Applications*, *115*, 2019, 106–120.

[4] H. Xue and H.-X. Ma, Swarm intelligence based dynamic obstacle avoidance for mobile robots under unknown environment using WSN, *Journal of Central South University of Technology, 15*, 2008, 860–868.

[5] A.V. Savkin and C.A. Wang, Simple biologically inspired algorithm for collision-free navigation of a unicycle-like robot in dynamic environments with moving obstacles, *Robotica, 31*, 2013, 993–1001.

[6] Y. Zhaofeng and Z. Ruizhe, Path planning of multi-robot cooperation for avoiding obstacle based on improved artificial potential field method, *Sensors & Transducers, 165*, 2014, 221.

[7] J.B. Mbede, X. Huang, and M. Wang, Fuzzy motion planning among dynamic obstacles using artificial potential fields for robot manipulators, *Robotics and Autonomous Systems*, *32*, 2000, 61–72.

[8] S.S. Ge and Y.J Cui, Dynamic motion planning for mobile robots using potential field method, *Autonomous Robots, 13*, 2002, 207–222.

[9] U. Orozco-Rosas, O. Montiel, and R. Sepúlveda, Pseudo-bacterial potential field based path planner for autonomous mobile robot navigation, *International Journal of Advanced Robotic Systems, 12*, 2015, 81.

[10] S. Cifuentes, J.M. Girón-Sierra, and J. Jiménez, Virtual fields and behavior blending for the coordinated navigation of robot teams: Some experimental results, *Expert Systems With Applications, 42*, 2015, 4778–4796.

[11] Z. Pan, D. Wang, H. Deng, and K.A. Li, Virtual spring method for the multi-robot path planning and formation control, *International Journal of Control, Automation and Systems, 17*, 2019, 1272–1282.

[12] T.P. Nascimento, A.G. Conceiçao, and A.P. Moreira, Multi-robot nonlinear model predictive formation control: the obstacle avoidance problem, *Robotica, 34*, 2016, 549–567.

[13] Y. Li, J. Gao, X. Su, and J. Zhao, Cooperation control of multiple miniature robots in unknown obstacle environment, *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 229*, 2015, 202–214.

[14] Y. Dai, Y. Kim, S. Wee, D. Lee, and S. Lee, A switching formation strategy for obstacle avoidance of a multi-robot system based on robot priority model, *ISA Transactions, 56*, 2015, 123–134.

[15] C. De La Cruz and R. Carelli, Dynamic model based formation control and obstacle avoidance of multi-robot systems, *Robotica, 26*, 2008, 345–356.

[16] W.-B. Xu, X.-B. Chen, J. Zhao, and T.-Y. Huang, A decentralized method using artificial moments for multi-robot path-planning, *International Journal of Advanced Robotic Systems, 10*, 2013, 24.

[17] H.-X. Wei, Q. Mao, Y. Guan, and Y.-D. Li, A centroidal Voronoi tessellation based intelligent control algorithm for the self-assembly path planning of swarm robots, *Expert Systems With Applications*, *85*, 2017, 261–269.

[18] C.G. Cena, P.F. Cardenas, R.S. Pazmino, L. Puglisi, and R.A. Santonja, A cooperative multi-agent robotics system: Design and modelling, *Expert Systems With Applications, 40*, 2013, 4737–4748.

[19] A.V. Savkin and C. Wang, Seeking a path through the crowd: Robot navigation in unknown dynamic environments with moving obstacles based on an integrated environment representation, *Robotics and Autonomous Systems*, *62*, 2014, 1568–1580.

[20] G. Divya Vani, S.R. Karumuri, and M. Chinnaiah, Hardware schemes for autonomous navigation of cooperative-type multi-robot in indoor environment, *Journal of The Institution of Engineers (India): Series B*, *103*, 2021, 1–12.

[21] S. Wang, X. Hu, J. Xiao, and T. Chen, Repulsion-oriented reciprocal collision avoidance for multiple mobile robots, *Journal of Intelligent & Robotic Systems, 104*, 2022, 1–21.

[22] W. Pang, D. Zhu, and S.X. Yang, A novel time-varying formation obstacle avoidance algorithm for multiple AUVs, *International Journal of Robotics and Automation, 38*(3), 2023, 194–207.

[23] M. Duguleana and G. Mogan, Neural networks based reinforcement learning for mobile robots obstacle avoidance, *Expert Systems With Applications*, *62*, 2016, 104–115.

[24] T. Fan, P. Long, W. Liu, and J. Pan, Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios, *The International Journal of Robotics Research*, *39*, 2020, 856–892.

[25] K. Charalampous, I. Kostavelis, and A. Gasteratos, Robot navigation in large-scale social maps: An action recognition approach, *Expert Systems with Applications*, *66*, 2016, 261–273.

[26] P. Fiorini and Z. Shiller, Motion planning in dynamic environments using velocity obstacles, *The International Journal of Robotics Research*, *17*, 1998, 760–772.

[27] Wu, and J.P. How, Guaranteed infinite horizon avoidance of unpredictable, dynamically constrained obstacles, *Autonomous Robots*, *32*, 2012, 227–242.

[28] M. Otte and E. Frazzoli, RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning, *The International Journal of Robotics Research*, *35*, 2016, 797–822.

[29] G.S. Aoude, B.D. Luders, J.M. Joseph, N. Roy, and J.P. How, Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns, *Autonomous Robots*, *35*, 2013, 51–76.

[30] M. Shahriari and M. Biglarbegian, Toward safer navigation of heterogeneous mobile robots in distributed scheme: A novel time-to-collision-based method, *IEEE Transactions on Cybernetics, 52*(9), 2021, 9302–9315.

[31] A.S. Lafmejani and S. Berman, Nonlinear MPC for collision-free and deadlock-free navigation of multiple nonholonomic mobile robots, *Robotics and Autonomous Systems*, *141*, 2021, 103774.

[32] H. Zhu, B. Brito, and J. Alonso-Mora, Decentralized probabilistic multi-robot collision avoidance using buffered uncertainty-aware Voronoi cells, *Autonomous Robots, 46,* 2022, 1–20.

[33] X. Li, D. Zhu, B. Sun, Q. Chen, W. Gan, and Z. Li, Formation tracking for a multi-AUV system based on an adaptive sliding-mode method in the water flow environment, *International Journal of Robotics and Automation*, *38*(5), 2023, 352–366.

[34] E.S. Moghaddam, M.G. Saryazdi, and A. Taghvaeipour, Trajectory optimization of a spot-welding robot in the joint and cartesian spaces, *International Journal of Robotics and Automation*, *38*(2), 2023, 109–125.

[35] E.A. Padilla-Garcia, A. Rodriguez-Angeles, J.R. Resendiz, and C.A. Cruz-Villar, Concurrent optimization for selection and control of AC servomotors on the powertrain of industrial robots, *IEEE Access, 6*, 2018, 27923–27938.

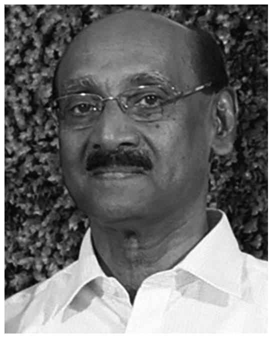
[36] F. Belkhouche, Reactive path planning in a dynamic environment, *IEEE Transactions on Robotics*, *25*, 2009, 902–911.

[37] V. Sezer and M. Gokasan, A novel obstacle avoidance algorithm: "Follow the Gap Method", *Robotics and Autonomous Systems*, *60*, 2012, 1123–1134.

[38] W. Mathworld, Circle-circle intersection, Accessed on 15 September 2023. Available Online: https://mathworld.wolfram.com/Circle-CircleIntersection.html

Biographies

Tahniat Khayyam received the B.Eng. degree in mechatronics engineering from Monash University, Australia, and the M.Sc. degree in robotics from King's College London, UK. She is currently the Technical Director with Multronica Solutions, Multan, Pakistan. Her research interests include multi-robot systems, path planning, and teleoperation.



S. G. Ponnambalam hasforty-two years of experience with forty years in academia and two years in a manufacturing company as a Production Engineer. He is currently a Senior Professor with the School of Mechanical Engineering, VIT University, India. He has previously served as a Full Professor with the Faculty of Manufacturing and Mechatronics Engineering Technology, University Malaysia Pahang from 2018– to 2020 and with the School of Engineering, Monash University, Malaysia, from 2002– to 2017. His areas of expertise include robotic assembly line systems, manufacturing, supply chain management, optimisation, and swarm robotics. He has published over 300 articles in various refereed journals, conferences, and chapters in edited books. He edited *Industry 4.0 and Hyper-Customized Smart Manufacturing Supply Chains* (IGI Global, 2019) with N. Subramanian, M. K. Tiwari, and W. A. Yusoff and *Innovation Analytics: Tools for Competitive Advantage* (World Scientific, 2023). He had also edited special issues for *International Journal of Advanced Manufacturing Technology* (Springer), *Flexible Services and Manufacturing Journal* (Springer), *International Journal of Robotics and Automation* (Acta Press), and *International Journal of Advanced Mechatronics Systems* (Inderscience). He is a Fellow of the Institution of Mechanical Engineers (UK) and a Chartered Engineer registered with the Engineering Council, UK. He is also a Senior Member of IEEE, USA.

Mukund Nilakantan Janardhanan is currently working with the Warwick Manufacturing Group, University of Warwick as an Associate Professor in Intelligent Manufacturing Systems. He has around 10 years of experience including 8 years in academia and 2 years in a Engineering consultancy company. He received the Ph.D. degree in manufacturing optimization from Monash University, Malaysia. Prior to his current job at Warwick, he was the Director of Education with the University of Leicester and also has worked as a Postdoctoral Fellow with Aalborg University, Denmark, and has also previously worked in product development in an Engineering Design company. He is an ambitious researcher with special interests in Industry 4.0, manufacturing system optimisation and simulation. Till date, to his credit he has published more than 60 referred research papers in leading manufacturing and operations journals. He is involved in funded research projects and currently leading a novel research project in using robotics for disaster management. He has also edited special issues in reputed international journals and has served as programme committee members of reported international conferences. He is also a Chartered Engineer and Fellow of Higher Education academy, UK.

Izabela Ewa Nielsen was born on December 22, 1977, in Poland. She received the Engineering and M.Sc. degrees from the Faculty of Management and Production Engineering, Opole University of Technology in 2001, and the Ph.D. degree with honors in the application of constraint logic programming techniques in production flow planning from the Faculty of Production Engineering, Warsaw University of Technology in 2005. She is currently a Professor with the Department of Materials and Production, Aalborg University, Denmark. Her research is primarily in the areas of planning, scheduling, optimisation problems, and decision support systems. She has a special emphasis on automated manufacturing, transportation, and production systems. She has published over 140 articles in journals, books, and conferences. In 2006, she received an award for her research work from the Polish Ministry of Science and Higher Education. She is an Associate Editor of the *International Journal of Industrial Engineering: Theory, Applications and Practice*; and *European Journal of Industrial Engineering*. Furthermore, she is an Editorial Board Member in several reputed journals as: *International Journal of Advanced Logistics* and *International Journal: Production and Manufacturing Research*.