# The Boundary Node Method for Multi-Robot Multi-Goal Path Planning Problems

R. A. Saeed[1], Diego Reforgiato Recupero[1], and Paolo Remagnino[2]

[1]Department of Mathematics and Computer Science, University of Cagliari, Via Ospedale, 72, 09124, Cagliari, Italy
[2]Computer Science Department, School of Computer Science and Mathematics, Kingston University, KT1 2EE, London, UK

**Abstract**

In this paper, we investigate the multi-goal path planning problem to find the shortest collision-free path connecting a given set of goal points located in the robot working environment. This problem combines two sub-problems: first, optimize the sequence of the goal points located in the free workspace; second, compute the shortest collision-free path between goal points. In this study, a genetic algorithm is used to optimize the sequence of the goal points that the robot should visit. Once the sequence of the goal points is available, our developed method called Boundary Node Method ($BNM$) is applied to generate an initial collision-free path between every pair of the sequenced goal points. Subsequently, an additional developed method called Path Enhancement Method ($PEM$) is used to find an optimal or near-optimal path by reducing the overall initial path length. In the Boundary Node Method, the robot and its immediate surroundings are simulated by a nine-node quadrilateral element, where the centroid node represents the robot's position. The robot moves between goals with boundary nodes in the working environment depending on the boundary node's potential value. The potential value of each point in the working environment is calculated based on the proposed potential function. Additionally, this article investigates the multi-goal path planning problem for multi-robot systems, when each goal reached by several robots. Finally, simulations and experiments are performed on a real mobile robot to demonstrate the effectiveness of the developed methods.

Keywords: Multi-goal path planning, Robot Path Planning, Boundary Node Method, Path Enhancement Method

## 1 Introduction

In robotic planning, the problem of finding the shortest collision-free path connecting a given set of goal points located in the robot working environment is called a multi-goal path planning problem ($MTP$) Glorieux et al., 2020; Saha et al., 2006; Wurll et al., 1999. In $MTP$, the task to find a sequence of the goal points located in the robot working environment can be solved as an instance of the Traveling Salesman Problem($TSP$) Vonásek and Pěnička, 2019 in which paths between goal points have to be traverse-able by the robot. The requirement for a collision-free path connecting goal points is the main reason why the problem is called $MTP$ rather than $TSP$ to emphasize its difficulty Faigl, 2016. Finding the collision-free paths between the goal points is essential in this problem Vonásek and Pěnička, 2019. In the $TSP$, a salesman has to visit several cities (locations) with the constraint that the salesman should visit each city once. A salesman is interested in travelling on the shortest route linking a set of cities based on the given distance between them Sun, 2020. The shortest tour starts from a given city, passing through all the other cities and returning to the home city Wurll et al., 1999. There are many methods to solve $TSP$, such as heuristic algorithms, genetic algorithms, simulated annealing algorithm, and ant colony optimization Hongyun et al., 2013. Furthermore, the $TSP$ has been well studied, and efficient algorithms are available to solve the $TSP$ problem Spitz and Requicha, 2000. The $TSP$ has been well studied and powerful algorithms are available Spitz and Requicha, 2000. In a simple case for the mobile robot, optimizing the sequence of the goal points in free working space and the path between the goal points can be modeled by the $TSP$ Vicencio et al., 2014; Vonásek and Pěnička,

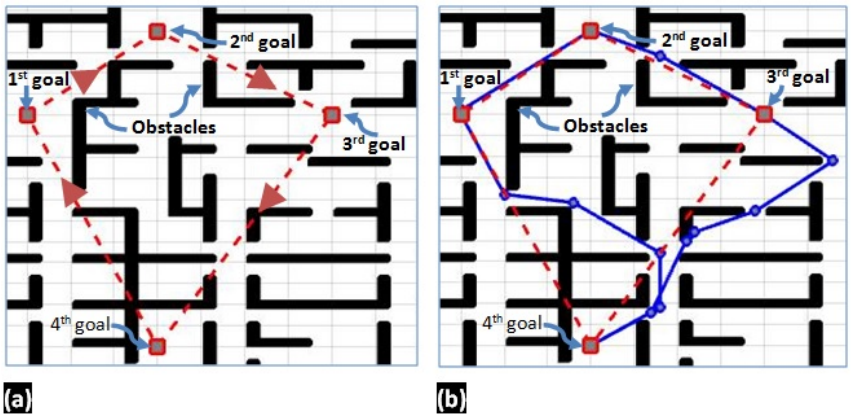


Figure 1: A simple example of multi-goal path planning problem in a $2D$ workspace. ($a$) The sequence of goal points obtained from the implementation of $GA$, the path marked in red dashed line and the goals are shown in grey square object. ($b$) Results of $BNM\&PEM$ to generate the collision-free path for linking the goal points, plotted in blue solid line.

2019. However, in the basic $TSP$, the probability of colliding obstacles along the path between every pair of the sequenced goal points are not considered Noormohammadi-Asl and Taghirad, 2019, and the $TSP$ is not a suitable model. In this case, not only it is necessary to optimize the sequence of the goal points, but an optimal collision-free path between goals must be generated as well (see Shiltagh and Jalal, 2013).

Figure 1 illustrates a simple example of $MTP$ in a two-dimensional ($2D$) robot working environment with obstacles. As shown in the figure, the navigable area is shown in white and the space occupied by obstacles is shown in black, while obstacles are assumed to be fixed in their position. The goals (4 in Figure 1) are located randomly in the robot working environment, where each goal is represented by a single point. The robot has to reach all goal points to fulfill a given task, however, the sequence in which they should be reached is not given. This scenario illustrates the $MTP$ to find an optimal or near-optimal collision-free path in two steps: first, using a Genetic Algorithm ($GA$) to find the optimal sequence of the goal points over the free working space (see Figure 1$a$) to minimize the length of the path. As shown in the figure, the arrows indicate the optimal goals sequence where the path starts from the $1^{st}$ $goal$, passing through the $2^{nd}$ $goal$, $3^{rd}$ $goal$, and $4^{th}$ $goal$ and return to the $1^{st}$ $goal$ (similar to the $TSP$) such that each goal is visited once. Previously, the use of $GA$ to solve the $TSP$ problem has been reported by many researchers (e.g., Bonert et al., 2000; Sun, 2020; Wurll et al., 1999). Moreover, in the fields of mobile robotics, $GA$ has been widely used in many studies to solve the $TSP$ problem by taking advantage of its strong optimization capability Wurll et al., 1999; Yu et al., 2002; Zacharia and Aspragathos, 2005. $GA$ is an adaptive search technique based on natural selection and a genetic reproduction mechanism, that is widely applied to perform a random search to solve optimization problems.

Once the sequence of the goal points is available, the Boundary Node Method ($BNM$) is applied to construct the initial collision-free path between every pair of the sequenced goal points. Consequently, an additional developed method, Path Enhancement Method ($PEM$), is used to find an optimal or near-optimal collision-free path from the initial feasible path by minimizing the overall path length. Both methods, $BNM\&PEM$, are introduced to provide an optimal path between every two goal points, as illustrated in Figure 1$b$. The path starts from the initial goal point, passing through all intermediate goal points, and returning to the initial goal point is called a multi-goal path. A multi-goal path is assembled by simply connecting goal-to-goal points, extracted between goals iteratively until the path (tour) is completed. The length of the multi-goal path is the sum of the lengths of the goal-to-goal paths.

Recently, using multiple mobile robots rather than a single mobile robot has attracted many researchers' attention Huang et al., 2019, and it has been widely used in industrial plants and warehouses Liu et al., 2010.Optimizing the paths of the mobile robots that move simultaneously is

a challenging problem Nazarahari et al., 2019. The multi-robot path planning problem has been studied since the 1980s Chen et al., 2009; Hvězda et al., 2019; Pellegrinelli et al., 2017, and several methods have been proposed during this period Hvězda et al., 2019. Multi-robot path planning methods have been used to determine the collision-free path for multiple robots from their given start positions to their goal positions. Two types of collisions have to consider to plan collision-free paths for multiple mobile robots: robot–obstacle and robot–robot. In general, the multi-robot path planning problem is considered as an optimization problem, the solution to the optimization problem provides the optimal path Hvězda et al., 2019.

In many applications, the computational cost of constructing a collision-free path over every goal-to-goal pairings in obstacle-filled environments with many goals is high Englot and Hover, 2011; Faigl, 2016; Saha et al., 2006, because of both the dimensionality of the configuration space and the geometric complexity of the obstacles and the robot Saha et al., 2006. In robotic planning, multiple-goal collision-free path planning has not been studied extensively in previous literature because of the complexity that it implies Diaz-Arango et al., 2020. Depending on the problem complexity, finding the shortest collision-free path between goal points as well as optimizing the sequence of goal points will be very difficult Wurll et al., 1999, and computationally hard problems Saha et al., 2003.

The difficulty of determination of a collision-free path for a mobile robot that visits a set of locations in the robot working environment has also been observed in Glover et al., 2001, and also poor quality solutions are found for this kind of problem. Furthermore, sequencing goal points before the search for the collision-free path may generate a near-optimal path, because goals are arranged according to a distance function that ignores the obstacles. Additionally, the path planning between two sequenced goals may be very difficult Spitz and Requicha, 2000. While significant progress has been made in this area, there are still no effective approaches Edelkamp et al., 2018.

The main objectives of this paper are stated below:

1. Solve a multi-goal path planning problem for single and multiple mobile robots by generating the shortest collision-free path connecting a given set of goal points located in the robot working environment.
2. Perform an experimental study on a real robot to verify the performance and effectiveness of the proposed method for solving the multi-goal path planning problems, and illustrate how the robot navigates in the working environment.

Based on these difficulties and to effectively solve this challenging problem, this paper investigates the implementation of our developed method, $BNM$, to find optimal goal-to-goal path between a set of goal points in the obstacle-filled environment. A valuable benefit of $BNM$ is its simplicity and can be applied in a grid environment efficiently. Moreover, this method is capable of finding the path for mobile robots effectively and efficiently in terms of path length and computational time, even if the complexity of the environment is increased, as explained in detail in our previous work Saeed and Recupero, 2019; Saeed et al., 2020. Moreover, this study contributed to extending the $BNM$ method further to solve the multi-goal path planning problem for multiple mobile robot systems by generating the shortest collision-free path connecting every pair goal points sequentially among obstacles.

The rest of the paper is organized as follows: a brief overview of the background work within the domain of $MTP$ is introduced in the this section. The problem statement is introduced in Section 3. In Section 4, a brief description of the developed methods for generating the shortest feasible multi-goal path is introduced. The implementation of the developed methods is presented in Section 5 followed by the evaluation of results and a discussion. The experimental study is presented in Section 6. Final conclusions are provided in Section 7.

## 2 Related Work

Multi-goal planning has been widely used in many robotics applications, such as surveillance, manufacturing, autonomous inspection, and assembly. Moreover, the multi-goal path-planning

is important for autonomous mobile robots that perform tasks in industrial environments Diaz-Arango et al., 2020. The existing methods solve the $MTP$ in a two-step process Englot and Hover, 2011: first, constructing a path by optimizing the sequence of goal points in a robot working environment without considering obstacles; then, constructing a path (tour) when obstacles are introduced into the environment, in which the robot must reach all sequenced goal points and it visits each goal once. The $MTP$ in the presence of obstacles have been discussed in literature and many solution have been found in this area, i.e, self-organizing map ($SOM$) Faigl, 2016; Vaněk et al., 2014, branch-detected algorithm and heuristic algorithm Hongyun et al., 2013, boundary iterative-deepening depth-first search ($BIDDFS$) algorithm Lim et al., 2015, Lin-Kernighan Heuristics ($LKH$) algorithm Hernandez et al., 2017, generalized traveling salesman problem with neighborhoods Vicencio et al., 2014, probabilistic roadmap ($PRM$) planner Spitz and Requicha, 2000, ant colony optimization ($ACO$) combined with a sampling-based point-to-point planning algorithm Englot and Hover, 2011, branch-detected algorithm and heuristic algorithm Hongyun et al., 2013, adaptive neural network Burke, 1996, Partially Observable Markov Decision Process ($POMDP$) framework Noormohammadi-Asl and Taghirad, 2019, hierarchical distance computation based on the $A^*$ search algorithm Wurll and Henrich, 2001, Artificial Potential Field ($APF$) Nazarahari et al., 2019. The authors in Pandey et al., 2020; Patle et al., 2019 provide the various soft computing techniques applied by different researchers for mobile robot navigation. The main characteristics of the most used collision-free path planners are reported in Diaz-Arango et al., 2020.

The $MTP$ has been previously investigated, and proposed approaches are motivated by many practical problems, i.e. planning for a robotic arm Saha et al., 2006; Spitz and Requicha, 2000, hexapod walking robot Vaněk et al., 2014, Industrial manipulators Zacharia et al., 2013, spot welding task Saha et al., 2003; Wurll and Henrich, 2001, inspection planning Faigl et al., 2011, search and rescue mission Kulich et al., 2005, planetary exploration Hongyun et al., 2013, inspection and surveillance applications Englot and Hover, 2011, coordinate measuring machines ($CMMs$) Spitz and Requicha, 2000, finding cracks in large structures Danner and Kavraki, 2000, office-like environments to perform common tasks of picking up, and delivering things such as mail, goods, trash recycled paper, etc Hernandez et al., 2017. The author in Saha et al., 2006 considered a motion planning problem which occurs often in practice, e.g., in spot-welding, car-painting, inspection, and measurement tasks Saha et al., 2006 to compute a near optimal path of the mobile robot.

In previous research work, different methods were used to solve the multi-robot multi-goal path planning problem, i.e., the authors in Huang et al., 2019 considered the cooperative path planning problem of multiple mobile robots in an unknown indoor environment. They proposed a novel obstacle avoidance and real-time navigation algorithm. This algorithm consists of global path planning and local path planning via a hybrid artificial fish swarm algorithm ($HAFSA$) and an expansion logic strategy. The application of adaptive Charged System Search ($CSS$) algorithms has been usedPrecup et al., 2015 to find an optimal path for multiple mobile robots . They examined these algorithms on holonomic wheeled platforms in static environments. A new planning algorithm for multi-goal path planning, called Space-Filling Forest (SFF), is proposed by Vonásek and Pěnička, 2019. The single robot ceiling vision $SLAM$ has been extended by Chen et al., 2009 to multi-robot formations to address global localization problems in unknown environments. The author in Das et al., 2016 proposed improved classical $Q$-learning and improved ($PSO$) with perturbed velocity ($QIPSO\text{-}DV$) algorithm to construct an optimal collision-free path multi-robots path planning from predefined starting and goal positions for each robot in the robot working environment. The authors in Edelkamp and Lee, 2019; Warsame et al., 2020 addressed the multi-robot multi-goal motion planning to solve the vehicle routing problem for mobile robots by using Monte-Carlo search. Additionally, new path planning and collision-avoidance methods were introduced by Sagar et al., 2017 to solve multi-robot multi-goal path planning problems. The motion planning problem for multi-robot spot-welding cells in the construction of car doors studied by Pellegrinelli et al., 2017. The authors in Kala, 2016 present a solution to an overall mission wherein a team of robots visit several mission sights and carry some operation. The multiple mobile robot solutions are extremely useful in spacecraft, rescue, transportation, etc Chen et al., 2009.

# 3 Problem formulation

The multi-goal path planning problem is formulated as follows. For a given robot working environment $C = R^2$, the region of the working space free of obstacles is represented by $C_{free} = C - C_{obs}$ and the region of the working space occupied by obstacles is denoted by $C_{obs}$. Each position within $C_{free}$ is reachable by the robot. The robot's working space is decomposed into rectangular grid cells. Each grid cell has integer coordinates of the form $C(x, y) \in C, 1 \leq x \leq n,\ 1 \leq y \leq m$, which either corresponds to a navigable area $C(x, y) \in C_{free}$ or a space occupied by obstacles $C(x, y) \in C_{obs}$. We assume that all information related to the workspace is known in advance, and obstacles are also assumed to be stationary. Each grid cell $C(x, y)$ in $C_{free}$ has a potential value $E(x, y) \in E$, which is calculated based on given potential function. In this study, a set of goal points $g_i, (i = 1...n)$ are located randomly inside the free space $C_{free}$, however the sequence of goals is not known. In this study, the problem formulation have extended further to solve the multi-goal path planning problem $MTP$ for multiple mobile robot systems. The robot is requested to visit all goal points and find the shortest path starting from the first goal $g_1$ and tracking it through all intermediate goal points $g_i, (i = 1...n)$ then return to $g_1$, such that each goal is visited by the robot once. The robot must not collide with any obstacle and must optimize the path between waypoints point. In this study, obstacles are assumed to be static, and the robot is treated as a dynamic obstacle for other robots. To solve the above planning problem, we compute an optimal or near-optimal path for each robot, so that the robot reach all goal points and visit each goal once.

# 4 Overall method

This section describes the implementation of the developed methods to construct the shortest path between goal points $g_i, (i = 1...n)$. The first step of the developed method is constructing a $2D$ grid model of the robot working environment, and then calculate the potential value of the grid cells based on the new proposed potential function. In the second step, the initial feasible path $IFP$ between goal points is generated by using a new developed method called Boundary Node Method $(BNM)$. In this method, the $IFP$ is generated from a sequence of waypoints $w$ that the robot has to traverse as it moves toward the destination point without colliding with any obstacles. The obtained $IFP$ between goal points is not optimal. Therefore, to reduce the overall path length in the third step, an additional developed method called path enhancement method $(PEM)$ is used to construct an optimal or near-optimal path from $IFP$ by reducing the number of waypoints and the overall path length.

In this study, the robot's working space is decomposed into rectangular grid cells, and the center of all grid cells in the given workspace meets the Equation 1. Each grid cell is considered as either an obstacle $C_{obs}$ or a non-obstacle $C_{free}$. An example of three different workspace scenarios with different obstacle layouts is shown in Figure 2. In these scenarios, the workspace consists of $(67 \times 109)$ grid cells. As shown in Figure 2, static obstacles $(1 \times 1\ unit)$ are distributed at different locations in the workspace, where the number of the obstacles in these workspaces are 1682, 1060 and 956 grid cells, respectively. The white region represents the $C_{free}$ and the black objects represent the $C_{obs}$. The goal points are positioned randomly in the free workspace $C_{free}$.

After constructing a model for each of the workspace, one-dimensional $(1D)$ potential value $E(k),\ k = 1...N\ (N = n \times m)$ is calculated for each grid cell $C(h, k)$, with $(h = 1...2)$, and $(k = 1...N)$ in the workspace $C$ based on the new proposed potential function, and their values formulated by using Equation 2. Subsequently, the obtained $1D$ array of $E(k)$ is converted into $2D$ array of potential value $E(x, y) \in E, 1 \leq x \leq n,\ 1 \leq y \leq m$ by using Equation 3. The potential function is used to direct the robot from the starting goal point $g_1(x_{g1}, y_{g1})$ toward the destination goal point $g_2(x_{g2}, y_{g2})$. This function has the lowest potential value at $g_2$ and the potential value increases as the robot moves further away. As shown in the Figure 2, the line's color represents the potential value, i.e. the blue line corresponds to the lowest potential value and the yellow line corresponds the highest potential value.

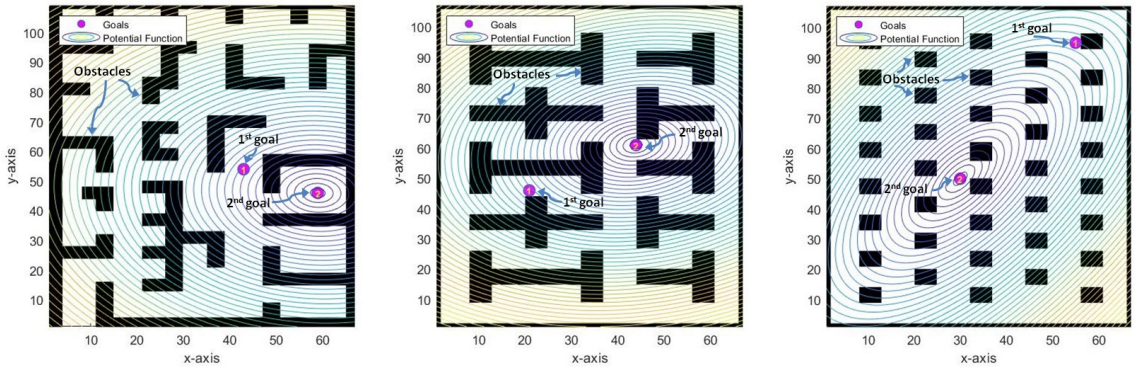


Figure 2: $2D$ models of the robot's workspaces with obstacles. Contour lines represent the potential values. The pink circle objects are goals, $g_1$ and $g_2$. The rectangular black objects represent the obstacles. The size of the workspace is $67 \times 109$ and the starting goal point $g_1$ and destination goal point $g_2$ are located randomly in the free space of the environment.

$$C = \sum_{x=1}^{n} \sum_{y=1}^{m} C(x, y) \tag{1}$$

$$\begin{aligned} E(k) &= \sqrt{(d_l(1,k))^2 - (d_p(1,k))^2}, \\ d_l(1,k) &= |m \times C(1,k) + b \times C(2,k) + c|/ll, \\ d_p(1,k) &= \sqrt{(C(1,k) - x_{g2})^2 + (C(2,k) - y_{g2})^2}^2, \end{aligned} \tag{2}$$

$$E(x, y) = \sqrt{(E(k)/D)^2 + (d_1(k))^2} \tag{3}$$

where $C(h,k), (h = 1...2), (k = 1...N)$ represents the location of the grid cells in the workspace, and the length of each grid cell is equal to 1 *unit*. The value of constants $m$, $b$, $c$, and $ll$ can be determined as follow: $m = ((y_{g1} - y_{g2})/(x_{g1} - x_{g2}))$, $c = (y_{g1} - m * x_{g1})$, $ll = \sqrt{m^2 + b^2}$, $(b = -1)$. Additionally, $D$ represents the distances between $g_1$ and $g_2$, is calculated as follow: $D = \sqrt{(x_{g1} - x_{g2})^2 + (y_{g1} - y_{g2})^2}$), where the slope of a straight line $D$ is denoted by $m$. The distance between the goal point $g_2(x_{g2}, y_{g2})$ and surrounding point $C(h,k)$ in the workspace is represent by $d_p(1,k)$.

Afterward, the $BNM$ is used to find the initial feasible path ($IFP$) for the mobile robot to move between goal points in the workspace without colliding with any obstacles. In this study, the given $2D$ workspace is completely known in advance, and also obstacles are assumed to be static. In $BNM$, the robot is simulated by a nine-node quadrilateral element (see Figure 3$a$). The nodes are denoted by $p(q), (q = 1...9)$, and their location can be formulated by using Equation 4. In the simulated model, the centroid node $p(5)$ represents the robot's location (see Figure 3$b$) and the other nodes $p(1 \rightarrow 4)$ with $p(6 \rightarrow 9)$ represent the 8-boundary nodes. As shown in Figure 3$b$, in each iteration $t$, the robot and boundary nodes $p(q), (q = 1...9)$ are restricted to move in 8-possible directions $e(u), (u = 1...8)$ in the workspace.

The potential values $E(q), (q = 1...9)$ for the robot and boundary nodes are equivalent to the potential value of the corresponding generated point in the workspace (see Figure 3$c$), and their values are calculated based on the proposed potential function. Characteristics of the boundary nodes, their positions and potential values, guide the robot to move forward and change its motion direction in the workspace (see Figure 3$c$). On the other hand, these characteristics can help the robot to avoid obstacles as well.

In the workspace that contains no obstacles, the robot will reach the destination point along a straight line from any starting point. As obstacles exist, the robot interferes with obstacles when the distance between the robot and the obstacles is less than $d$, a safety distance between

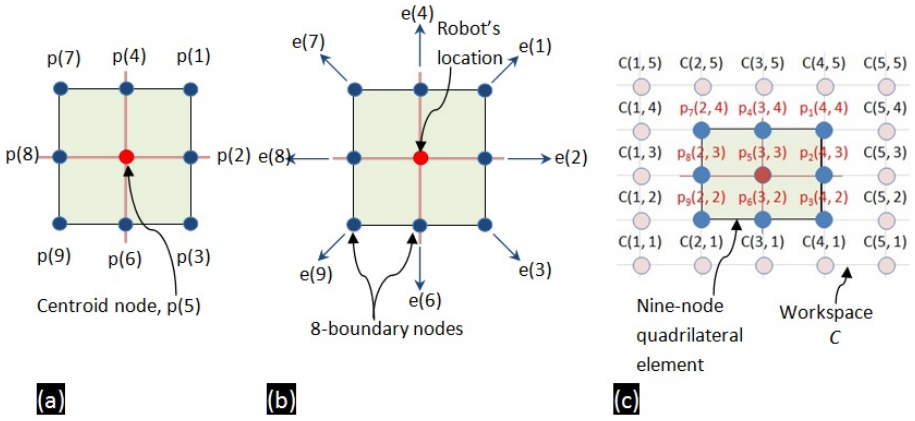


Figure 3: 2D illustration of a nine-node quadrilateral element ($a$) along with its motion directions ($b$) and exploration location in the workspace ($c$).

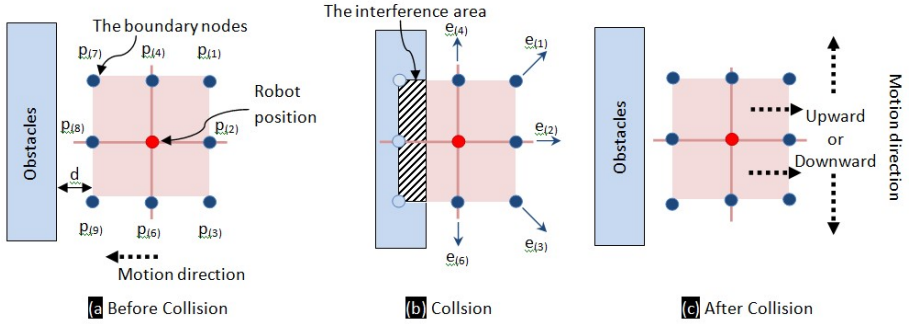


Figure 4: Collision avoidance with static obstacles, ($a$) the initial position of the robot, ($b$) the new updated position of the robot, ($c$) obstacles avoidance and change motion direction.

the center and the edge of the obstacle, $d = 0.5$ *unit*. As the robot might move very close to the obstacle, they should keep a certain margin for safety. Therefore, the robot and boundary nodes require to avoid obstacles and change their moving direction by selecting a new position in the $C_{free}$.

To explain the obstacle avoidance, suppose that the long vertical set of obstacles block the robot path as presented in Figure 4. As shown in the Figure 4$a$, the boundary nodes $p(1 \rightarrow 4)$ and $p(6 \rightarrow 9)$ are generated around the robot position $p(5)$ by using Equation 4, the red object represents the robot and the blue objects represent the boundary nodes. At iteration $t$, the robot with boundary nodes change their positions from the current position (see Figure 4$a$) to the new updated position (see Figure 4$b$). As a result, the nodes $p(7)$, $p(8)$, and $p(9)$ interfere with the obstacles (see Figure 4$b$). Therefore, the robot needs to investigate the workspace to find the next position without colliding obstacles. In this case, the robot will move along the $y$-axis either in the upward or downward direction until the robot passes the block of obstacles. The motion direction depends on the characteristic of the boundary nodes (see Figure 4$c$).

$$p(q) = \begin{cases} x, y & q{=}5 \\ (x + v_x, y), (x, y + v_y), (x - v_x, y), (x, y - v_y) & q = 2,\ 4,\ 6, \text{ and } 8 \\ (x + v_x, y + v_y), (x - v_x, y + v_y), (x - v_x, y - v_y), (x + v_x, y - v_y) & q = 1,\ 3,\ 7, \text{ and } 9 \end{cases} \tag{4}$$

where $x$ and $y$ are the distances between grid cell's center and $x$ and $y - axis$, $v_x$ is the horizontal distance and $v_y$ is the vertical distance between the robot and boundary nodes, we assume that $v_x = v_y = 1$ *unit*.

Furthermore, in order to demonstrate how the robot avoids colliding with the other robots and

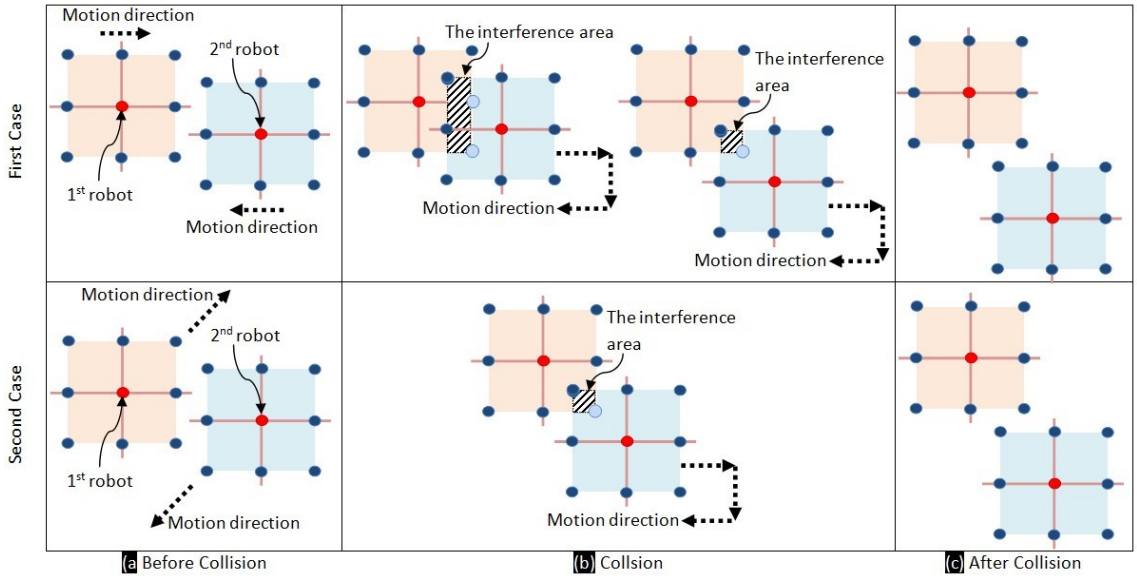


Figure 5: Collision avoidance between two moving robots, $(a)$ the initial positions of both robots, $(b)$ the new updated positions of both robots, $(c)$ robot-robot collision avoidance and change motion direction.

changes its motion direction by using $BNM$, consider the two simple cases shown in Figure 5. As illustrated in figure 5$a$, both robots start to move with boundary nodes in different directions toward each other. While the robot moves forward in the first case, nodes $p_(2)$ and $p_(3)$ of the first simulated robot interfere with the second robot, and nodes $p_(7)$ and $p_(8)$ of the second simulated robot interfere with the first robot. In the same way, in the second case, node $p_(3)$ of the first simulated robot interfere with the second robot and node $p_(7)$ of the second simulated robot interfere with the first robot (see Figure 5$b$). In order to avoid collision, the robot needs to change its motion direction along the $y$-axis. The motion direction depends on the characteristics of the boundary nodes, their positions and potential values. Then the robot moves forward and changes its motion direction by shifting the robot toward a downward direction until the robot passes the other robot, as shown in Figure 5$c$.

As the robot moves from $g_i$ to $g_{i+1}$, $(i = 1...n)$ in the workspace, it generates a set of waypoints $w(j), (j = 1...J)$ that the robot visits sequentially. Then, the $IFP$ is generating from a set of waypoints $w$ that the robot visits before reaching the final destination point. For better clarity, the waypoints are connected into a continuous path, the line segment that connects two waypoints in sequence is represented by $P_{l,l+1}, 1 \leq l \leq w-1$. The path $IFP$ between goals consists of a finite number of straight-line segments joining the way-points. The length of all line segments that connect all waypoints sequentially to each other is representing the length of the $IFP$. The path length between $g_i$ and $g_{i+1}$ is formed by concatenation of all inter-line segments $P_{l,l+1}$ as follows: $IFP = [P_{1,2}, P_{2,3} \ldots, P_{w-1,w}]$. The extracted path between each pair of goals is assembled by simply connecting to another goal, iteratively, until the path is completed into a single connected component. The complete path which starts in $g_1$, and passes through all goal points $g_i, (i = 2, ..., n)$, and then returns to $g_1$ is called the multi-goal path.

The obtained $IFP$ for a mobile robot between goal points is a safe path, however, it is not the shortest path. In order to reduce the overall path length, the $PEM$ is used to generate the shortest path from $IFP$ by reducing the number of waypoints $w$. To explain the $PEM$, consider the robot moves from the first goal point $g_1$ toward the second goal point $g_2$, and the generated $IFP$, consists a number of waypoints $w(j), (j = 1...J)$. The $PEM$ connect $g_1$ with the intermediate waypoints $w(j)$ by a line-segments $U$, iteratively. For the first line-segments $U_1$, $g_1$ is connected to the first waypoints $w(1)$. Then, the line-segments $U_1$ between these two points is checked for feasibility. If a collision is not found, then $g_1$ is connected to $w(2)$, and this procedure continues in the same way for all waypoints. In case the line between these two points collides with obstacles,

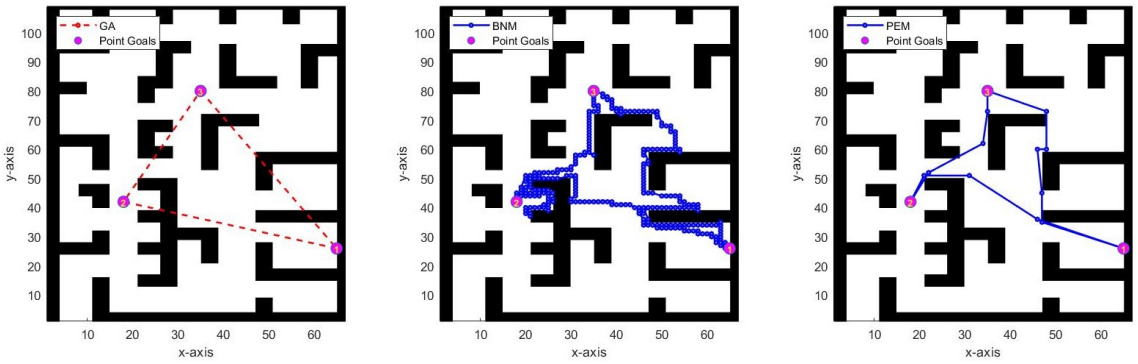


Figure 6: A multiple-goals path planning problem ($a$) the path formulated from linking all sequenced goals, plotted in red dashed line. ($b$) the $IFP$ for the robot in a given workspace, plotted in blue. ($c$) optimize the initial path.

the starting point for the line-segments $U_2$ is placed in the last waypoint. This procedure continues until the robot reaches $g_2$. The total length of the shortest path $U$ between $g_1$ and $g_2$ is calculated by summing the length of all the line segments $U(i)$ in the path between $g_1$ and $g_2$. The same procedure is repeated for every pair of goal points sequentially until the robot returning to $g_1$. More details about these methods can be found in Saeed et al., 2020.

# 5 Simulation

In order to validate the performance of the developed methods, different simulated configurations with different parameters have been conducted, with different number of robots, obstacles layout, number of goal points and their positions. Three examples of obstacle-filled environment scenarios with complex obstacle layout used in this study are presented in Figure 2. In these scenarios, the given workspace is completely known in advance, and also obstacles are assumed to be static. The number of goal points $n$ are positioned randomly in the free space of the working environment, where each goal is represented by a single point. In the process of implementing and testing of the developed methods, we compute a shortest collision-free path for the robot to reach all goal points and visit each goal once. All simulations have been carried out on a laptop Intel($TM$) Core($TM$) i5-8300$H$ $CPU$, 2.3$GHz$, and 8$GB$ $RAM$. The proposed model is implemented in the $Python$ and $Matlab$ programming language.

In this study, the multiple-goals path planning problem in a given $2D$ workspace with obstacles (see an example in Figure 6) is solved as follow: first, the optimal sequence of the goal points (3 $goals$) located in a $C_{free}$ is found by using $GA$ regardless of the obstacles(see Figure 6$a$). As shown in the figure, the path (tour) starts from $g_1$, passing through $g_2$ and $g_3$ and returns to $g_1$, where the goals are marked in pink circle object and the red dashed lines represent the shortest path between sequenced goals. Next, the $BNM$ is used to find the initial collision-free path connecting every pair of the sequenced goal points, as illustrated in Figure 6$b$. It can be seen clearly that the $BNM$ successfully generated the $IFP$ for the robot to move from $g_1$ to $g_2$ (see Figure 7$a$), from $g_2$ to $g_3$ (see Figure 7$b$), and from $g_3$ return to $g_1$ (see Figure 7$c$). As the robot moves from $g_i$ to $g_{i+1}$, $(i = 1...n)$ in the workspace, the $IFP$ is constructed from the way-points $w(j), (j = 1 \rightarrow J)$ that visited by the robot, where $J$ represents the time required by the robot to reach the final destination. As shown in that figure, the way-points $w$ are represented by blue circles objects, each new way-point position $w(j + 1)$ is allocated after the current point position $w(j)$ on the path. The obtained $IFP$ is a collision-free path, where the way-points do not fall on any obstacle and also the line segments that connect all way-points do not intersect with any obstacle.

The $BNM$ can generate the $IFP$ safely and efficiently, but the path is not optimal in terms of the total path length. Therefore, the $PEM$ is applied to reduce the number of way-points as well as the overall length of $IFP$. The obtained result of $PEM$ (see Figure 6$c$) is an optimal or close-to-optimal path for the robot which is represented by a thick blue line object. A complete multi-goal path can be formulated by joining all line segments obtained between $g_1$ and $g_2$ (Figure 7$d$), $g_2$ and

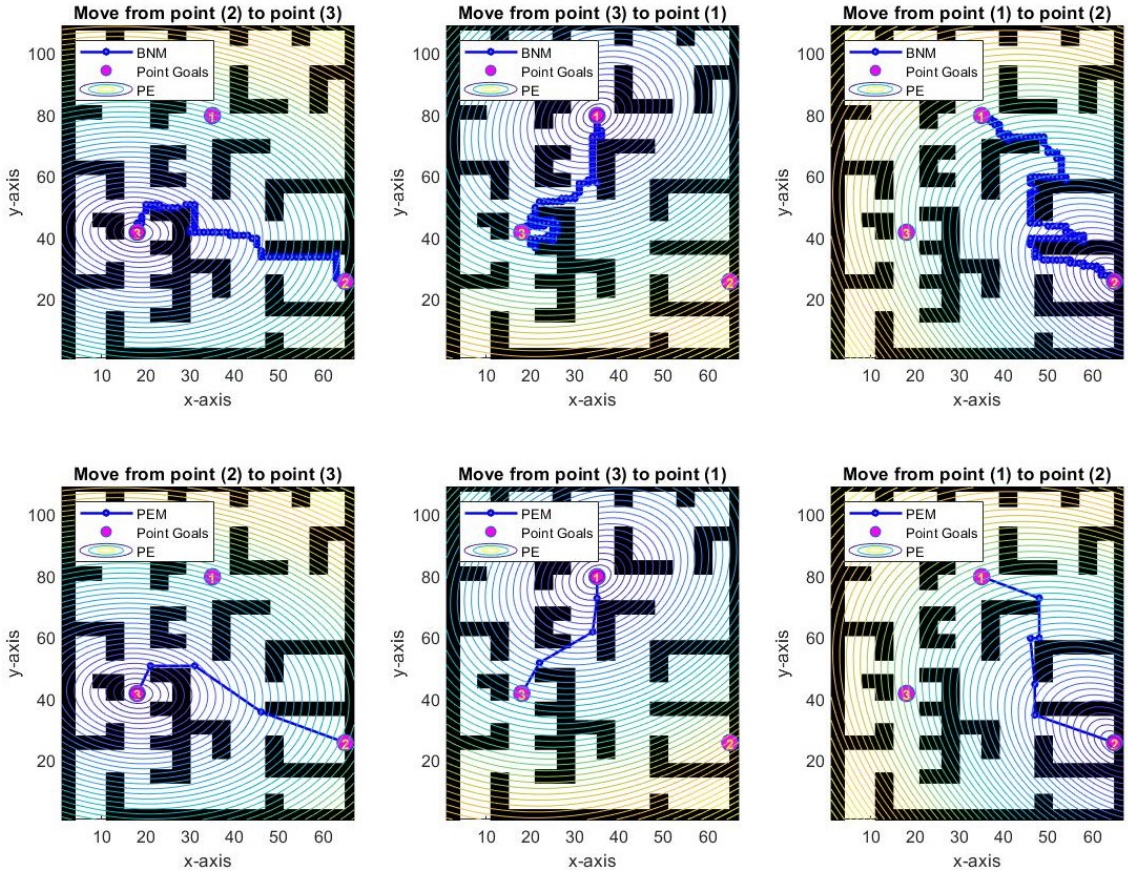


Figure 7: : A $2D$ illustration of the path planning algorithm

Table 1: Simulation and performance evaluation: Mean and standard deviation of the computational time (in *seconds*) to find the feasible path by a single robot for each instance on different working environment.

| $No.of\,Goals$ | $First\,Environment$ | | $Second\,Environment$ | | $Third\,Environment$ | |
|---|---|---|---|---|---|---|
| | $Mean_{CT}$ | $Std_{CT}$ | $Mean_{PL}$ | $Std_{PL}$ | $Mean_{PL}$ | $Std_{PL}$ |
| 2 | 4.9125 | 0.1966 | 4.8895 | 0.1249 | 4.5685 | 0.1641 |
| 4 | 5.1960 | 0.2057 | 5.1235 | 0.0953 | 4.8500 | 0.1126 |
| 6 | 5.3210 | 0.1501 | 5.2915 | 0.1200 | 5.0000 | 0.09531 |
| 8 | 5.4410 | 0.1210 | 5.4935 | 0.3305 | 5.0970 | 0.1425 |
| 10 | 5.6395 | 0.1245 | 5.7085 | 0.4026 | 5.1665 | 0.1111 |
| 12 | 5.6585 | 0.1608 | 5.8520 | 0.1461 | 5.2430 | 0.1246 |
| 14 | 5.7020 | 0.2240 | 5.9955 | 0.2518 | 5.3095 | 0.1236 |
| 16 | 5.7240 | 0.2368 | 6.0505 | 0.1719 | 5.3790 | 0.1587 |
| 18 | 5.7670 | 0.2157 | 6.1275 | 0.1831 | 5.4735 | 0.2103 |
| 20 | 5.8170 | 0.1787 | 6.1575 | 0.3323 | 5.5105 | 0.1694 |

$g_3$ (see Figure 7*e*), $g_3$ and $g_1$ (see Figure 7*f*). The exact path length $L$ can be found as the sum of path length between goal points sequentially $l_1, \ldots, l_n$. In the simulated environment, the problem formulation to find the shortest path for the single and multi-robot by avoiding the collision with obstacles are elaborated in subsections 5.1 and 5.2.

## 5.1 Simulation results of single-robot system

This section presents the obtained simulation results of multiple-goals path planning problem $MTP$ for a single mobile robot whose task requires visiting multiple-goal points. In this study, $GA$ and $BNM\&PEM$ were implemented for solving the $MTP$ in several simulated scenarios with different obstacles layout and the different number of goal points. All simulations were implemented in $MATLAB$. Throughout this section, we consider three examples (Figure 2) with the different numbers of goal points (2, 4, 6, 8, 10, 12, 14, 16, 18, and 20 goals) which are located randomly in the robot working environments. For each example, the proposed methods implemented on 200 randomly generated instances to find the initial feasible path. At each instance, the goal points are placed randomly in the working environment, each random placement of the goal points led to a different result.

The performance of the proposed method is calculated in terms of the total execution time required to find the shortest path for the robot to reach the final goal point. The mean and the standard deviation ($Std$) of the total computational time for each instance are calculated and presented in Table 1, and the graphical representation of the simulation results illustrated in Figure 8. The obtained results reveal that the proposed methods can determine the shortest path between goal points for each given instance within a reasonable computational time. In particular, for the high number of goal points in a highly complex environment, the total computing time to find the shortest path is around 6 *second*. Moreover, the mean value of the computational time required to obtain the final path is not increased significantly with increasing the number of goal points and the complexity of the simulated working environment (see Table 1). Obviously, the computational time taken by this operation depends on the number of goals, their ordering, geometric complexity of the obstacles, and also the problem complexity. The results agree with Saeed et al., 2020, which stated that the computational time required to solve the path planning problem by using $BNM$ does not increase significantly with the increase of the environment's complexity.

An example of the obtained results for each simulated scenario with 20 goal points located randomly in the working environment are presented in Figures 9, 10, and 11, respectively. For each tested scenario, the $GA$ is used to optimize the sequence of the goal points with absence of obstacles, and the obtained results are presented in Figures 9*a*, 10*a* and 11*a*. The goal points are marked in pink circle object and the red dashed lines represent the shortest path between sequenced goal points. As shown in figures, the robot visits all given goal positions located in the robot working environment with minimizes the path length and the total execution time.

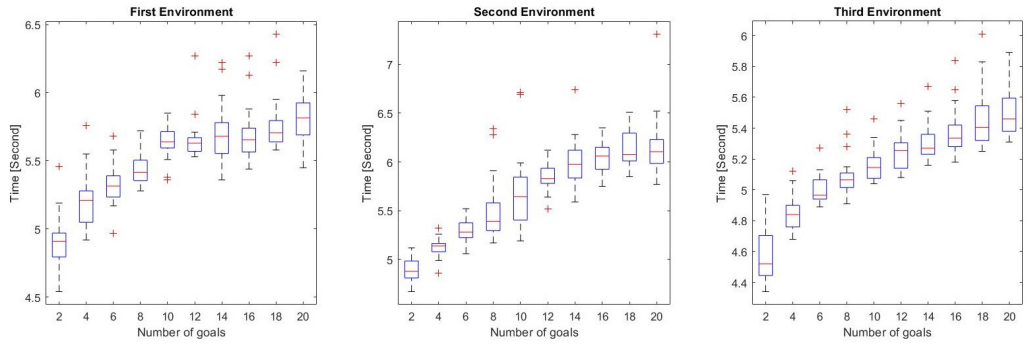


Figure 8: Influence of the number of goal points on the total computational time required to solve $MTP$ using the proposed method for each simulated working environment.

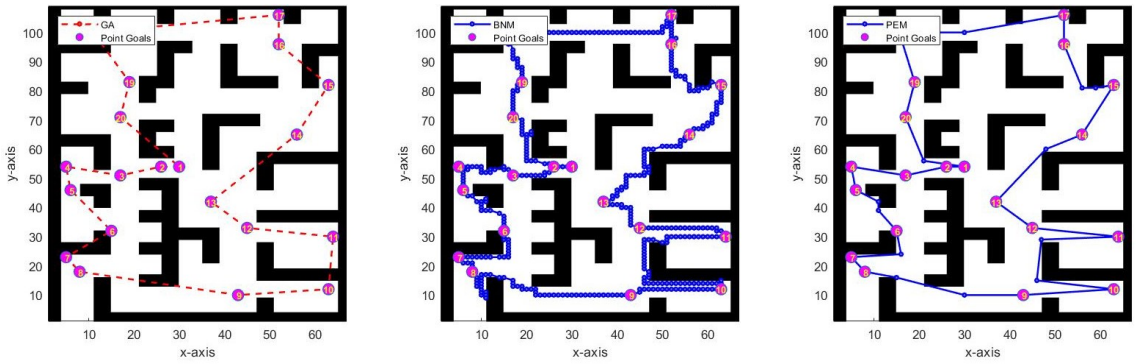


Figure 9: A first scenario of the multi-goal path for 20 randomly-selected goal points. ($a$) the path formulated from linking all sequenced goals plotted in the red dashed line. ($b$) the $IFP$ for the robot in a given workspace, plotted in blue. ($c$) optimize the initial path, plotted in the thick blue line.

Subsequently, the $BNM$ is used for generating the $IFP$ between every pair of given goal points, and the simulation results are presented in Figures 9$b$, 10$b$ and 11$b$. The achieved result of $IFP$ is represented by a set of way-points $w$, the way-points of the robot's motion path are marked by the blue circle objects. For better clarity, these way-points are connected into a continuous path. As observed from the figures, the final path allows the robot to move from goal to goal sequentially, and avoid obstacles successfully.

From the obtained results, it can be clearly seen that the $BNM$ has been well applied to generate $IFP$, and reached important achievements in terms of safety and short computational time. However, the path is not optimal in terms of the total path length. Therefore, the $PEM$ is used to reduce the overall length of $IFP$, and the obtained results for all tested scenarios are presented in Figures 9$c$ and 10$c$ and 11$c$, where the solid blue lines between goal points represent the final solution. As shown in the figures, the $PEM$ can find the collision-free path that covers the lowest number of waypoints to reach all of the goal points, and the total path length for each tested scenario is reduced significantly. Furthermore, the obtained results clearly show that the $BNM\&PEM$ provide the short and safe path for the robot to visit a given set of goal points (see Figures 9, 10 and 11).

## 5.2 Simulation results of multi-robot system

This section presents the implementation of the proposed method for solving the multi-robot multi-goal path planning problems. Each robot has to visit a number of goal points in the robot working environment with obstacles, and each robot has to find its path independently without collision with either static obstacles or other robots. Moreover, we have considered the robot working en-

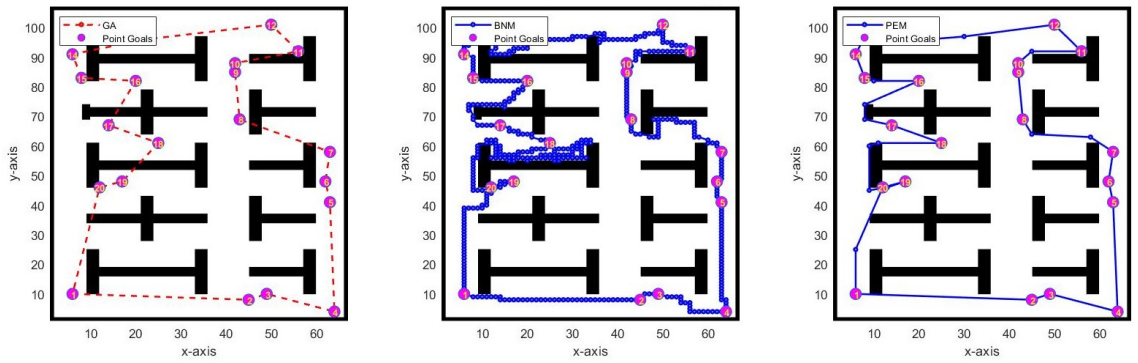


Figure 10: A second scenario of the multi-goal path for 20 randomly-selected goal points. ($a$) the path formulated from linking all sequenced goals plotted in the red dashed line. ($b$) the $IFP$ for the robot in a given workspace, plotted in blue. ($c$) optimize the initial path, plotted in the thick blue line.

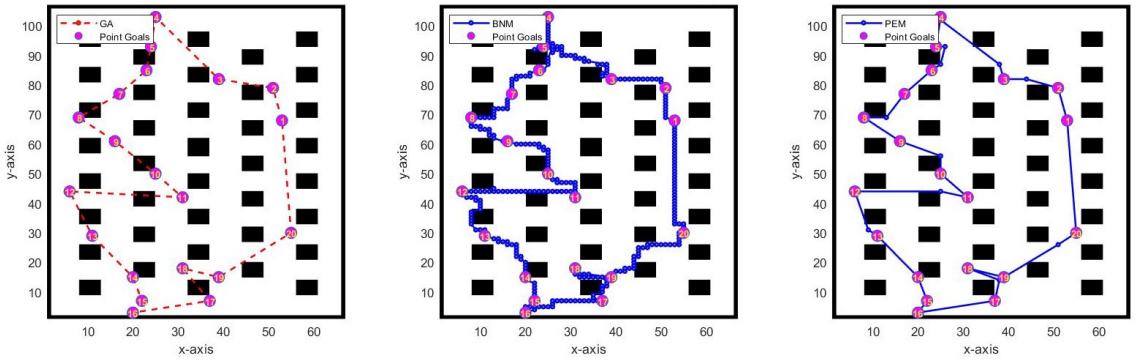


Figure 11: A third scenario of the multi-goal path for 20 randomly-selected goal points. ($a$) the path formulated from linking all sequenced goals plotted in the red dashed line. ($b$) the $IFP$ for the robot in a given workspace, plotted in blue. ($c$) optimize the initial path, plotted in the thick blue line.

Table 2: Characteristics of three different example scenarios

| Scenarios | Goal points |
|---|---|
| First Scenario (four goal points) | [[14,5],[37,20],[29,41],[19,23]] |
| Second Scenario (three goal points) | [[14,5],[29,41],[19,23]] |
| Third Scenario (two goal points) | [[14,5] ,[19,23]] |

Table 3: Simulation and performance evaluation: Mean and standard deviation of the computational time (in *seconds*) to find the feasible path for each instance on different simulated scenario.

| $No. of\ Robots$ | $First\ Scenario$ | | $Second\ Scenario$ | | $Third\ Scenario$ | |
|---|---|---|---|---|---|---|
| | $Mean_{CT}$ | $Std_{CT}$ | $Mean_{PL}$ | $Std_{PL}$ | $Mean_{PL}$ | $Std_{PL}$ |
| 1 | 12.6479 | 0.6727 | 11.8793 | 0.7119 | 6.0778 | 0.4230 |
| 2 | 25.3430 | 1.1184 | 24.2406 | 1.3893 | 12.1266 | 0.7627 |
| 3 | 38.3126 | 1.1113 | 36.4111 | 1.1663 | 18.5977 | 1.0296 |
| 4 | 52.5791 | 2.1483 | 48.9389 | 1.6548 | 25.3420 | 1.5899 |
| 5 | 66.2990 | 2.0072 | 61.7195 | 1.9908 | 31.8260 | 1.4962 |
| 6 | 80.8070 | 2.1736 | 76.3014 | 2.7425 | 38.7869 | 1.5057 |

vironment and obstacles are completely known in advance and the goal points scattered randomly in the working environment.

For the formulation of the multi-robot path-planning problem, we consider a group of mobile robots planning their path in the same simulated working environment ($48 \times 44$ square grid cells). All simulations were implemented in *Python*. At any instant of time $t$, a number of robots $m$ at positions $R(t) = r_1, r_2, ..., r_m$ start to move to visit a set of goal positions $G(t) = g_1, g_2, ..., g_n$ scattered randomly in the $2D$ robot working environment with static obstacles. Each robot moves from the starting goal position, through all of the intermediate goal points, then return to the starting goal position, such that each goal is reached by all robots and each goal $g \in G(t)$ is visited by each robot $r \in R(t)$ once. In our research, each robot uses the $BNM$ to find $IFP$ from any goal point to the next goal point in the workspace without colliding with any obstacle or other robots. $IFP$ is generated from a set of waypoints $w$ that the robot visits before reaching to its final destination.

An analysis of the computational complexity of the proposed was carried out by calculating the execution time versus the number of robots. We conducted different simulated scenarios with a different number of goal points and robots, an example of three scenarios with the different number of goal points are presented in Table 2. As shown in the table, the number of goal points in different locations have been varied in all the tested scenarios. The problem formulation is to find the path of each robot in the simulated environment by avoiding the collision with static obstacles and other moving robots in the system. Each robot moves from the first goal point, through all the intermediate goal points until it returns to the first position. Each robot uses the $BNM$ to find the shortest collision-free path to visit all goal points in the workspace.

The total computational time required to find the shortest path for each scenario has been computed for 240 runs of the proposed methods. The mean and the standard deviation ($Std$) of the total computational time for each instance are calculated and presented in Table 3, and the graphical representation of the simulation results illustrated in Figure 12. The simulation results are shown in the table and the figure reveals that all the robots reached their final destination positions within a reasonable computational time without any collision with either static obstacles or other robots. Moreover, it is clear from the obtained results that the computational time is commensurate to the number of deployed robots, and the mean value of the computational time to find a feasible path increases linearly with the number of robots.

An example of the simulation results for the multi-robot multi-goal path planning problem shown in Figures 13, with five-robots ($m = 5$), four-goal points ($n = 4$), and 304 static obstacles. In the first step, $GA$ is used to optimize the sequence of the goal points scattered randomly in a

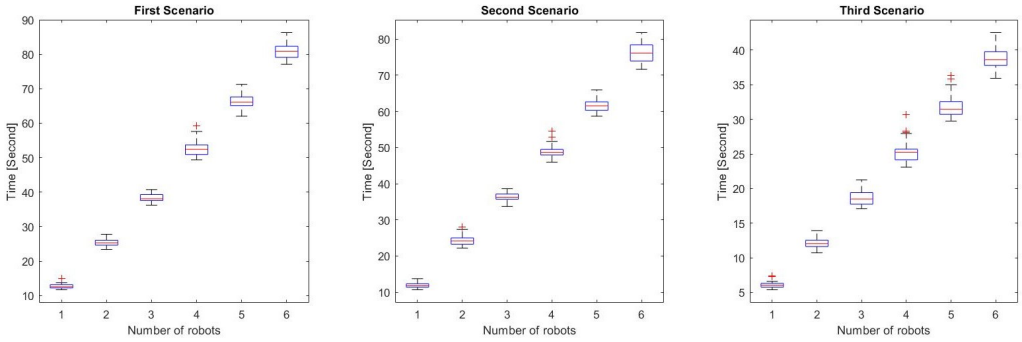


Figure 12: Influence of the number of robots on the total computational time required to solve $MTP$ using the proposed method for each simulated scenario.

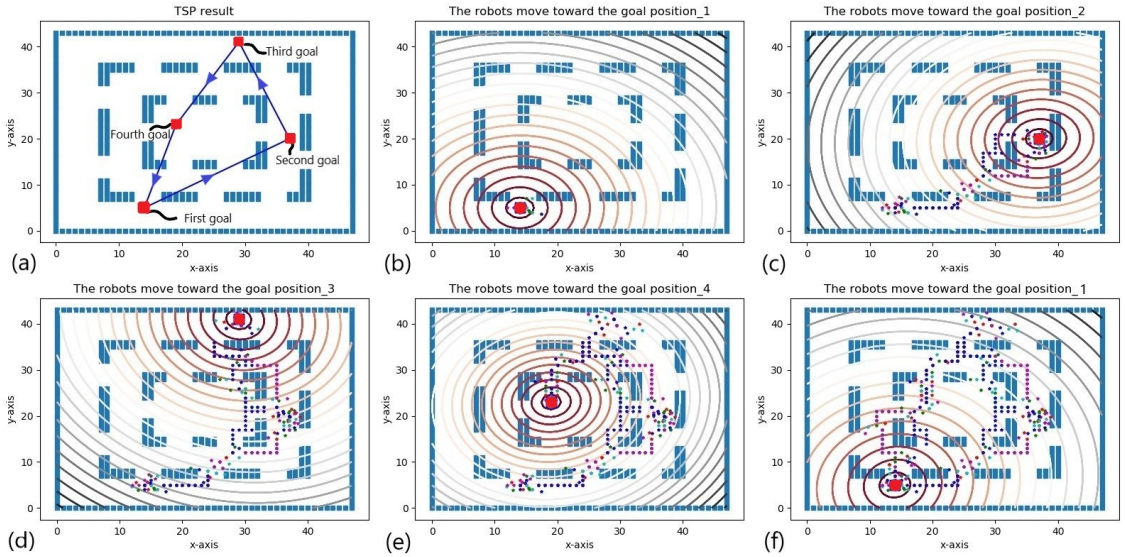


Figure 13: The simulation results for solving the $MTP$: $(a)$ The sequence of goal points obtained from the implementation of $GA$, the red square objects represent the goal points that have been visited by each robot. $(b \rightarrow f)$ Five robots move to visit multiple-goal points sequentially in a simulated working environment with obstacles.

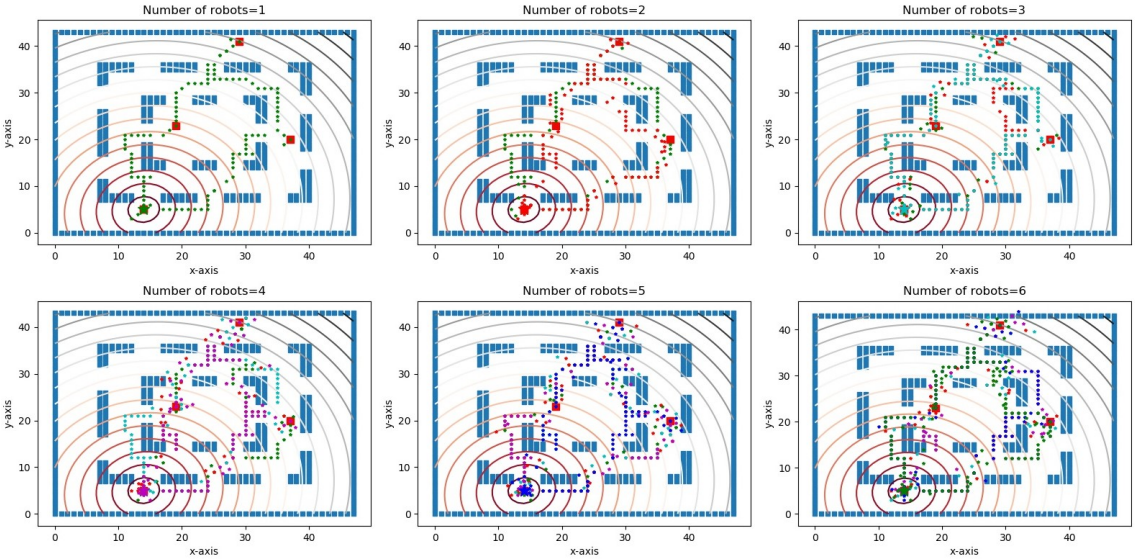


Figure 14: The simulation results for solving the $MTP$: the number of goal points is fixed (four-goal points) and we vary the number of robots from 1 to 6.

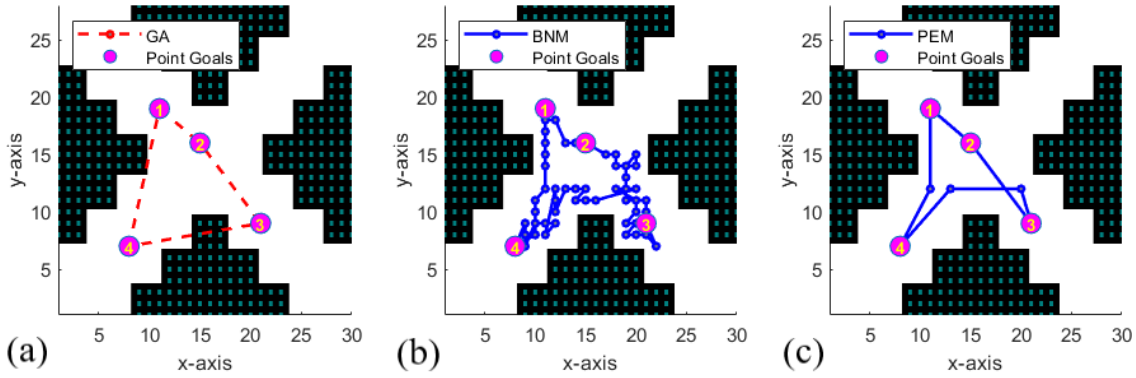


Figure 15: The simulation results to generate the shortest path by using $BNM\&PEM$.

$2D$ simulated working environment with the absence of obstacles, as shown in Figure 13$a$, whereas a group of robots (5 *robots*) has to move along a line (denoted by the blue solid line) to visit a group of goal points (4 *goals*), marked in the red square objects. Then, the $BNM$ is used to generate a feasible goal-to-goal path to direct each robot from each goal position toward the next goal position by avoiding collision with either static obstacles or other robots in the system, and the final simulation result is illustrated sequentially in Figure 13$b \rightarrow f$. Different colors of the small colored circles represent the position of the robots. Additionally, an example of the simulation results for the different numbers of the robot is presented in Figure 14, whereas the number of goal points is fixed (four-goal points) and we vary the number of robots from 1 to 6. The simulation results demonstrate that each robot constructs an individual path independently by satisfying the optimum path length, and all robots reach to their final destination position successfully without any collision with either static obstacles or other moving robots.

# 6 Experimental evaluation

In this section, validation of the developed method for solving multiple-goals path-planning problems is studied by performing a set of experimental tests with the e-puck robot. Several experimental scenarios have been carried out with different positions of the goal points and different obstacles configuration. An example of the experimental scenario for a single mobile robot that visits four-goal positions in the robot working environment is illustrated in Figure 15.

In the first step, $GA$ is used to optimize the sequence of the goal points scattered randomly in a $2D$ simulated working environment with the absence of obstacles, as shown in Figure 15$a$, the red dashed-line is represented the shortest obtained path. Then, the $BNM$ is used to generate $IFP$ to direct the robot from its current goal position toward the second goal position by avoiding collision with obstacles, as shown in Figure 15$b$. Finally, $PEM$ is used to find an optimal or near-optimal collision-free path from $IFP$ by reducing the number of waypoints and the overall path length (see Figure 15$c$).

Consequently, the constructed shortest collision-free path obtained from the simulation is used to direct the real physical robot to move from the first goal point, passing through all intermediate goal points, and return to the first goal point. An e-puck robot, shown in Figure 16, is used for the experimental test. The experimental set-up and the robot working environment with obstacles for conducting the experiment are presented in Figure 17, where the goal points (four-goal points) are located on the robot working environment. The e-puck robot has a diameter of 75 $mm$, and it has two actuators that control the robot's movement speed and direction. We have chosen an e-puck robot because the e-puck robot is a very compact, small, and flexible Mondada et al., 2009. There is also a library extension to $MATLAB$ to program the robot and integrate the robot with the developed methods. The e-puck robot uses Bluetooth to connect to a computer, which allows control programs to be remotely uploaded to the robot.

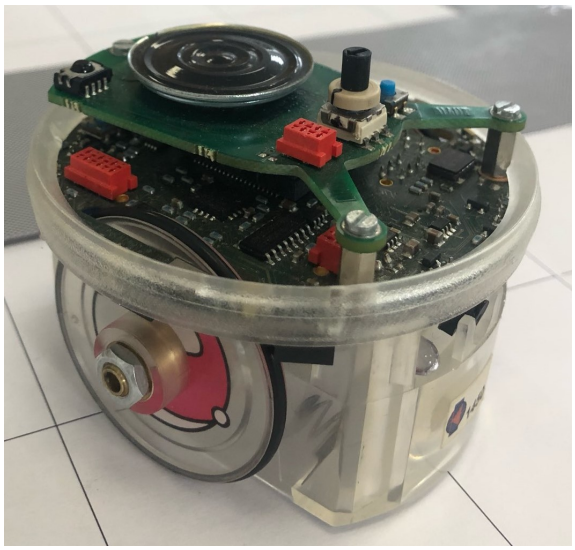

Figure 16: The e-puck robot used for the experimental test

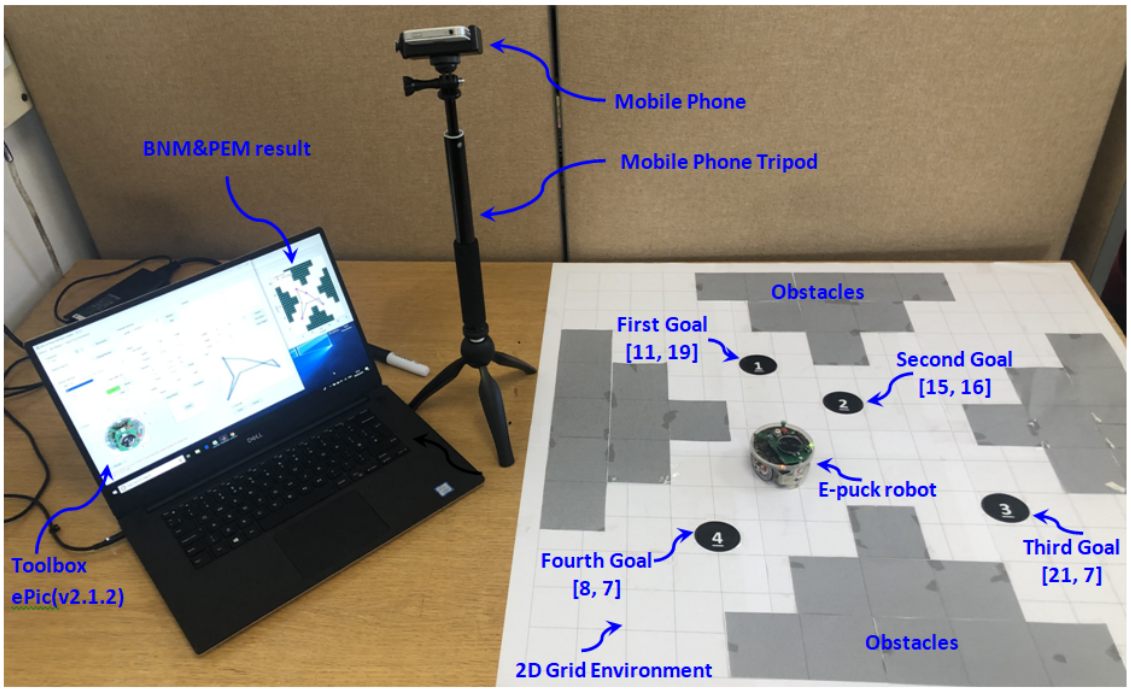


Figure 17: The experimental set-up

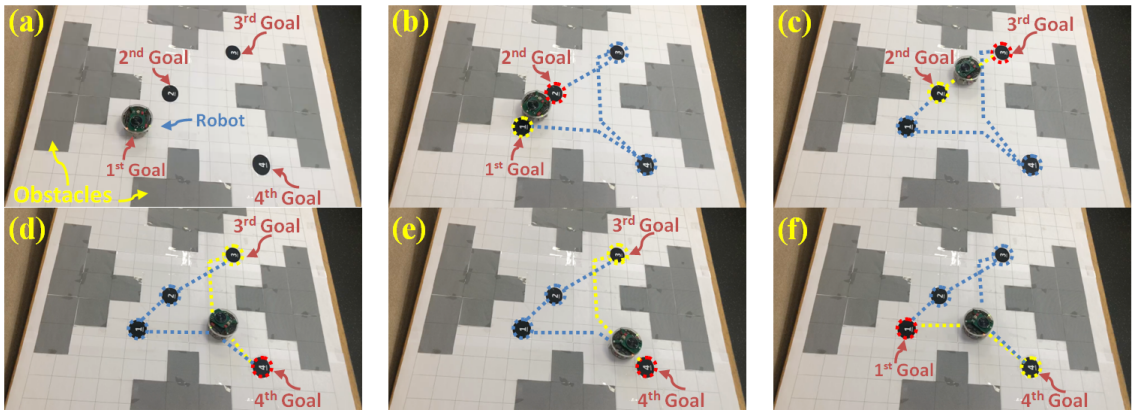


Figure 18: The experimental results: $(a)$ shows a graph representation of e-puck robot location in the robot's working environment, and the movement of the robot illustrate in $(b)$ from $g_1$ to $g_2$, $(c)$ from $g_2$ to $g_3$, $(d \,\&\, e)$ from $g_3$ to $g_4$, and $(f)$ from $g_4$ to $g_1$. The obtained shortest path by using $BNM\&PEM$ is represented by a blue dashed-line, and the yellow and red dashed circles represent the starting and destination goal points, respectively.

Table 4: The travelling path length and computational time from the starting point to the final destination point for the $e-puck$ robot

| The movement of the robot | Travelling path length $[cm]$ | Computational time $[second]$ |
|---|---|---|
| move from $g_1$ to $g_2$ | 3.92 | 6.54 |
| move from $g_2$ to $g_3$ | 6.18 | 12.91 |
| move from $g_3$ to $g_4$ | 23.12 | 24.72 |
| move from $g_4$ to $g_1$ | 13.46 | 18.09 |

Based on the generated data from the simulation results (see Figure 15$c$), the e-puck robot motion data is determined. As shown in the figure, the shortest path consists of the number of waypoints $w(j), (j = 1...J), (J = 7)$, represented by blue circle objects. Then the e-puck robot is connected to the computer via Bluetooth and the generated motion data are transmitted to the robot via a toolbox $ePic(v2.1.2)$, where $ePic(v2.1.2)$ allows the user to control the robot in $MATLAB$. Let $(w_{1x}, w_{1y})$ and $(w_{2x}, w_{2y})$ be the centroid coordinates of the first $w_1$ and second $w_2$ waypoint, respectively, of the shortest path for the robot in the simulated workspace, as shown in Figure 15. In the experimental test, in order to move the e-puck robot from first waypoint $w_1$ towards the second waypoint $w_2$ as illustrated in Figures 18, the orientation of the robot is calculated in $MATLAB$ by using $atan2(w_{2y}\text{-}w_{1y}, w_{2x}\text{-}w_{1x})$. Thereafter, the e-puck robot starts to move from $w_1$ to $w_2$ and so on until the robot visit all of the goal points. Figures 18$(a \rightarrow f)$ show the intermediate moment of the robot's positions at different locations in the robot working environment during the experimental test. A summary of the obtained results of the travelling path length and computational time from the starting point to the final destination point for the $e-puck$ robot is provided in Table 4. The test results demonstrate that the developed methods can generate a goal-to-goal path to direct the e-puck robot to move from $g_1$, passing through all intermediate goal points $g_i, (i = 2...n), (n = 4)$, and returning to $g_1$ such that each goal is visited by the robot once. In spite of the efficiency and optimality guarantee of the $BNM$, the quality of the constructed path obtained from the simulation depends on the resolution of the grid map. This means that it can generate the optimal path in the high-resolution map, while in the low-resolution map the path is not optimal. Besides, in a real environment, there are uncertainties such as friction and slippery surfaces which can lead to non-optimal results.

# 7 Conclusions

In this paper, we have extended the newly developed path planning method called Boundary Node Method $BNM$ further to solve the multi-goal path planning problem. The objective of this study

is to find the shortest collision-free path connecting a given set of goal points scattered randomly in a $2D$ robot working environment, without colliding with any obstacles. The shortest collision-free path is found in a two-step process. First, we applied a Genetic Algorithm $GA$ to find an optimal goal sequence. Second, we applied $BNM$ to generate an initial collision-free path between every pair given goal points, which is followed by using an additional developed method called Path Enhancement Method $PEM$ to extract an optimal or near-optimal collision-free path from the initial path by minimizing the overall path length. Simulation results show that the $BNM\&PEM$ can find the optimal or near-optimal collision-free path connecting the selected goal points located in a robot working environment, efficiently. Additionally, the $BNM\&PEM$ takes a relatively short computational time to solve $MTP$ for generating the shortest path for all examined scenarios which makes it easy to implement in real-time navigation.

Furthermore, instead of single robot, the developed method is also extended successfully to solve the multi-goal path planning problem for multiple mobile robot systems. While each robot has to find its path independently without collision with either static obstacles or other robots in the system. The simulation results demonstrate the effectiveness of the developed method for constructing the multi-goal path for multi-robot systems,

Finally, simulations and experiments are performed on a real mobile robot to demonstrate the effectiveness of the developed methods for solving multiple-goals path planning problems. Several experimental tests on the $e-puck$ robot have been carried out with the different positions of the goal points and different obstacles configuration. The results obtained from the experimental tests show that the proposed method can construct the shortest collision-free path, and direct the real physical robot to the final destination goal point.

Further study is required to address several research issues related to autonomous navigation of mobile robots in unknown environments, where the robot does not have full knowledge about its environment.

# References


Bonert, M., Shu, L., & Benhabib, B. (2000). Motion planning for multi-robot assembly systems. *International Journal of Computer Integrated Manufacturing*, *13*(4), 301–310.

Burke, L. (1996). "conscientious" neural nets for tour construction in the traveling salesman problem: The vigilant net. *Computers & operations research*, *23*(2), 121–129.

Chen, H., Sun, D., & Yang, J. (2009). Global localization of multirobot formations using ceiling vision slam strategy. *Mechatronics*, *19*(5), 617–628.

Danner, T., & Kavraki, L. E. (2000). Randomized planning for short inspection paths. *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, *2*, 971–976.

Das, P., Behera, H., & Panigrahi, B. (2016). Intelligent-based multi-robot path planning inspired by improved classical q-learning and improved particle swarm optimization with perturbed velocity. *Engineering science and technology, an international journal*, *19*(1), 651–669.

Diaz-Arango, G., Vazquez-Leal, H., Hernandez-Martinez, L., Jimenez-Fernandez, V. M., Heredia-Jimenez, A., Ambrosio, R. C., Huerta-Chua, J., Cos-Cholula, D., Hernandez-Mendez, S., et al. (2020). Multiple-target homotopic quasi-complete path planning method for mobile robot using a piecewise linear approach. *Sensors*, *20*(11), 3265.

Edelkamp, S., Lahijanian, M., Magazzeni, D., & Plaku, E. (2018). Integrating temporal reasoning and sampling-based motion planning for multigoal problems with dynamics and time windows. *IEEE Robotics and Automation Letters*, *3*(4), 3473–3480.

Edelkamp, S., & Lee, J. (2019). Multi-robot multi-goal motion planning with time and resources. *Annual Conference Towards Autonomous Robotic Systems*, 288–299.

Englot, B., & Hover, F. (2011). Multi-goal feasible path planning using ant colony optimization. *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2255–2260.

Faigl, J. (2016). An application of self-organizing map for multirobot multigoal path planning with minmax objective. *Computational intelligence and neuroscience*, *2016*.

Faigl, J., Kulich, M., & Přeučil, L. (2011). A sensor placement algorithm for a mobile robot inspection planning. *Journal of Intelligent & Robotic Systems*, *62*(3-4), 329–353.

Glorieux, E., Franciosa, P., & Ceglarek, D. (2020). Coverage path planning with targetted viewpoint sampling for robotic free-from surface inspection. *Robotics and Computer-Integrated Manufacturing*, *61*, 101843.

Glover, F., Gutin, G., Yeo, A., & Zverovich, A. (2001). Construction heuristics for the asymmetric tsp. *European Journal of Operational Research*, *129*(3), 555–568.

Hernandez, K., Bacca, B., & Posso, B. (2017). Multi-goal path planning autonomous system for picking up and delivery tasks in mobile robotics. *IEEE Latin America Transactions*, *15*(2), 232–238.

Hongyun, L., Xiao, J., & Hehua, J. (2013). Multi-goal path planning algorithm for mobile robots in grid space. *Control and Decision Conference (CCDC), 2013 25th Chinese*, 2872–2876.

Huang, Y., Li, Z., Jiang, Y., & Cheng, L. (2019). Cooperative path planning for multiple mobile robots via hafsa and an expansion logic strategy. *Applied Sciences*, *9*(4), 672.

Hvězda, J., Kulich, M., & Přeučil, L. (2019). Improved discrete rrt for coordinated multi-robot planning. *arXiv preprint arXiv:1901.07363*.

Kala, R. (2016). Sampling based mission planning for multiple robots. *2016 IEEE Congress on Evolutionary Computation (CEC)*, 662–669.

Kulich, M., Faigl, J., & Preucil, L. (2005). Cooperative planning for heterogeneous teams in rescue operations. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, 230–235.

Lim, K. L., Seng, K. P., Yeong, L. S., Ang, L.-M., & Ch'ng, S. I. (2015). Uninformed pathfinding: A new approach. *Expert Systems with Applications*, *42*(5), 2722–2730.

Liu, S., Sun, D., & Zhu, C. (2010). Coordinated motion planning for multiple mobile robots along designed paths with formation requirement. *IEEE/ASME transactions on mechatronics*, *16*(6), 1021–1031.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., & Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. *Proceedings of the 9th conference on autonomous robot systems and competitions*, *1*(CONF), 59–65.

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

Noormohammadi-Asl, A., & Taghirad, H. D. (2019). Multi-goal motion planning using traveling salesman problem in belief space. *Information Sciences*, *471*, 164–184.

Pandey, A., Panwar, V. S., Hasan, M. E., & Parhi, D. R. (2020). V-rep-based navigation of automated wheeled robot between obstacles using pso-tuned feedforward neural network. *Journal of Computational Design and Engineering*, *7*(4), 427–434.

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.

Pellegrinelli, S., Pedrocchi, N., Tosatti, L. M., Fischer, A., & Tolio, T. (2017). Multi-robot spot-welding cells for car-body assembly: Design and motion planning. *Robotics and Computer-Integrated Manufacturing*, *44*, 97–116.

Precup, R.-E., Petriu, E. M., Radae, M.-B., Voisan, E.-I., & Dragan, F. (2015). Adaptive charged system search approach to path planning for multiple mobile robots. *IFAC-PapersOnLine*, *48*(10), 294–299.

Saeed, R., & Recupero, D. R. (2019). Path planning of a mobile robot in grid space using boundary node method. *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2019 - Volume 2*, 159–166.

Saeed, R., Recupero, D. R., & Remagnino, P. (2020). A boundary node method for path planning of mobile robots. *Robotics and Autonomous Systems*, *123*, 103320.

Sagar, K., Zlatanov, D., Zoppi, M., Nattero, C., & Muthuswamy, S. (2017). Multi-goal path planning for robotic agents with discrete-step locomotion. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, *58172*, V05AT08A033.

Saha, M., Roughgarden, T., Latombe, J.-C., & Sánchez-Ante, G. (2006). Planning tours of robotic arms among partitioned goals. *The International Journal of Robotics Research*, *25*(3), 207–223.

Saha, M., Sánchez-Ante, G., & Latombe, J.-C. (2003). Planning multi-goal tours for robot arms. *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, *3*, 3797–3803.

Shiltagh, N. A., & Jalal, L. D. (2013). Path planning of intelligent mobile robot using modified genetic algorithm. *International Journal of Soft Computing and Engineering (IJSCE)*, *3*(2), 31–36.

Spitz, S. N., & Requicha, A. A. (2000). Multiple-goals path planning for coordinate measuring machines. *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, *3*, 2322–2327.

Sun, C. (2020). A study of solving traveling salesman problem with genetic algorithm. *2020 9th International Conference on Industrial Technology and Management (ICITM)*, 307–311.

Vaněk, P., Faigl, J., & Masri, D. (2014). Multi-goal trajectory planning with motion primitives for hexapod walking robot. *2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, *2*, 599–604.

Vicencio, K., Davis, B., & Gentilini, I. (2014). Multi-goal path planning based on the generalized traveling salesman problem with neighborhoods. *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, 2985–2990.

Vonásek, V., & Pěnička, R. (2019). Space-filling forest for multi-goal path planning. *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1587–1590.

Warsame, Y., Edelkamp, S., & Plaku, E. (2020). Energy-aware multi-goal motion planning guided by monte carlo search. *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, 335–342.

Wurll, C., & Henrich, D. (2001). Point-to-point and multi-goal path planning for industrial robots. *Journal of Robotic Systems*, *18*(8), 445–461.

Wurll, C., Henrich, D., & Wörn, H. (1999). Multi-goal path planning for industrial robots. *International Conference on Robotics and Application (RA'99), Santa Barbara, USA*.

Yu, Z., Jinhai, L., Guochang, G., Rubo, Z., & Haiyan, Y. (2002). An implementation of evolutionary computation for path planning of cooperative mobile robots. *Proceedings of the 4th World Congress on Intelligent Control and Automation (Cat. No. 02EX527)*, *3*, 1798–1802.

Zacharia, P. T., & Aspragathos, N. (2005). Optimal robot task scheduling based on genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, *21*(1), 67–79.

Zacharia, P. T., Xidias, E. K., & Aspragathos, N. A. (2013). Task scheduling and motion planning for an industrial manipulator. *Robotics and computer-integrated manufacturing*, *29*(6), 449–462.