IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 10, 2020, accepted December 1, 2020, date of publication December 8, 2020,
date of current version December 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043333

HPPRM: Hybrid Potential Based Probabilistic Roadmap Algorithm for Improved Dynamic Path Planning of Mobile Robots

ANKIT A. RAVANKAR[1], (Member, IEEE), ABHIJEET RAVANKAR[2], (Member, IEEE), TAKANORI EMARU[1], (Member, IEEE), AND YUKINORI KOBAYASHI[1,3], (Member, IEEE)

[1]Division of Human Mechanical Systems and Design Engineering, Research Faculty of Engineering, Hokkaido University, Sapporo 080-8628, Japan
[2]Faculty of Engineering, School of Regional Innovation and Social Design Engineering, Kitami Institute of Technology, Kitami 090-8507, Japan
[3]National Institute of Technology, Tomakomai College, Tomakomai 059-1275, Japan

Corresponding authors: Ankit A. Ravankar (ankit@eng.hokudai.ac.jp) and Abhijeet Ravankar (aravankar@mail.kitami-it.ac.jp)

This work was supported by the "SOUSEI Support Program for Young Researchers" at Hokkaido University, Japan, under Grant FY2020.

ABSTRACT Path planning and navigation is a very important problem in robotics, especially for mobile robots operating in complex environments. Sampling based planners such as the probabilistic roadmaps (PRM) have been widely used for different robot applications. However, due to the random sampling of nodes in PRM, it suffers from narrow passage problem that generates unconnected graph. The problem is addressed by increasing the number of nodes but at higher computation cost affecting real-time performance. To address this issue, in this paper, we propose an improved sampling-based path planning method for mobile robot navigation. The proposed method uses a layered hybrid Probabilistic Roadmap (PRM) and the Artificial Potential Field (APF) method for global planning. We used a decomposition method for node distribution that uses map segmentation to produce regions of high and low potential, and propose a method of reducing the dispersion of sample set during the roadmap construction. Our method produces better goal planning queries with a smaller graph and is computationally efficient than the traditional PRM. The proposed planner called the Hybrid Potential based Probabilistic Roadmap (HPPRM) is an improved sampling method with respect to success rate and calculation cost. Furthermore, we present a method for reactive local motion planning in the presence of static and dynamic obstacles in the environment. The advantage of the proposed method is that it can avoid local minima and successfully generate plans in complex maps such as narrow passages and bug trap scenarios that are otherwise difficult for the traditional sample-based methods. We show the validity of our method with experiments in simulation and real environments for both local and global planning. The results indicate that the proposed HPPRM is effective for autonomous mobile robot navigation in complex environments. The success rate of the proposed method is higher than 95% both for local and global planning.

INDEX TERMS Probabilistic roadmap, artificial potential field, motion planning, path planning, dynamic obstacle avoidance, robot navigation.

I. INTRODUCTION

Autonomous mobile robot navigation and planning has gained much interest in recent years for different applications where the robot has to operate in challenging and harsh environments. It is a fundamental problem for robot navigation in areas such as search and rescue, deep-sea explorations, mining, and service robots in industries, homes, and offices. Motion planning for such robots is, therefore, crucial for the safety, inspection, and monitoring of harsh environments and in areas where human exploration is not possible. In mobile robots, path planning is a technique to find a collision-free path in a known or unknown environment from any specified location to the given goal position. It is a fundamental problem in robotics, and numerous solutions in the past have been proposed to solve motion planning for different types of robots such as industrial manipulators, unmanned aerial vehicles (UAV), unmanned ground vehicles (UGV), and very

The associate editor coordinating the review of this manuscript and approving it for publication was Liang Hu.

recently autonomous driving vehicles [1], [2]. Path planning techniques are also prevalent in computer graphics and game design, where a path needs to be constructed between different points. When planning paths, different parameters such as the obstacle and map boundaries and vehicle's motion constraints should be carefully taken into consideration to accomplish the final goal. The problem is challenging, and the complexity grows exponentially with the size of the environment or the *configuration space (C-space)*. Moreover, conditions such as static or dynamic environment, wholly and partially unknown environment, adds to the complexity of solving the path planning problem. In the case of an unknown environment, the robot has to perform localization and mapping simultaneously while exploring the environment and commonly called the SLAM problem in robotics [2]. Whereas planning in a completely known environment where the position of the obstacles are static and do not change over time is relatively more straightforward, planning under dynamic and completely unknown environment with uncertainty is a challenging and an active area of research in robotics community.

The work in this paper presents a solution for mobile robot navigation and path planning in complex environments with static and dynamic obstacles. Our proposed method uses a sampling-based probabilistic roadmap (PRM) planner for global planning combined with a classical reactive local planning using the artificial potential field (APF) method for efficient path construction. Our proposed method, called the Hybrid Potential based Probabilistic Roadmap (HPPRM), can eliminate some of the drawbacks of PRM and APF in planning paths in cases such as the narrow passages, bug-traps, and local minima problem where these traditional methods generally fail to produce a solution. We also present an improved local planning for APF using virtual force that is more robust to avoiding the deadlock conditions, and plan paths in the presence of dynamic obstacles. We present a layered planner that can generate paths with fewer nodes and is computationally efficient with a higher success rate.

II. BACKGROUND AND RELATED WORKS

Path planning can be treated as a sub-problem of the broader motion planning family where the goal is to find a collision-free path from a start configuration to the goal configuration without considering the dynamics, control inputs, motion constraints, and duration of motion [1], [3], [4]. In terms of scope, the planning algorithms are classified into *global* path planning and *local* path planning. Global planning methods (hierarchical paradigm) generates a collision-free path of the mobile robot in a known map. Global planning methods find a completion path, which means they tend to produce a complete path if it exists otherwise outputs failure. They are very effective in static environments and are convergent in nature. Classical roadmap methods such as the Voronoi diagrams [5], [6], visibility graphs [7], [8], adaptive roadmaps, virtual force field (VFF) [9], and virtual field histogram (VHF) [10], [11] are some of the popular methods. On the other hand, local planning is based on a reactive paradigm and highly depends on local sensing rather than knowing the complete map. Because the environment is unknown, local planning invokes the robots to sense the obstacles in the vicinity and tightly act based on this learned information. The reactive planning needs to be fast and work in real-time to avoid unknown obstacles that can be static or dynamic. For global planning, such situations generally result in failure. For e.g., when moving on a planned global path, an unforeseen obstacle is placed in the planned path, re-planning is required to avoid hitting the obstacle [12]. Planning in a dynamic environment can be computationally expensive as the map needs to be updated at each step. Many robotic applications combine the advantages of both the local and global methods to present a complete solution to the navigation problem [13].

Artificial Potential Field or APF method first introduced by Khatib [14] in 1986 is a classical reactive local navigation method inspired by the potential energy in nature, such as the gravity or magnetic field for motion. Utilizing this method, a point robot in the configuration space can move under the influence of repulsive and attractive potential fields generated by the obstacles and target position. The approach has been widely used in the robotics community and is known for its mathematical elegance, real-time performance, and simplicity to determine the path from start to goal in minimum computation time. Earlier works on the potential field include [15] that presented search-based planning using the potential field to guide the search. Other series of work described in [16]–[19] discussed the potential field function with a unique local minima and construction of navigation functions. Despite the many advantages of APF due to its simplistic model, there are some disadvantages. One major problem with APF is that the robot can get stuck in the local minima of the potential field at a point in the map that is far from the actual goal even if there exist a motion plan. Such local minima occur in the map due to the distribution of obstacles and the potential field generated by them. When the robot navigates in such a potential field, there are chances of getting caught in the local minima. To counter this problem many researchers have proposed different strategies to avoid local minima trap in APF with several improvements [20]–[24]. Other researches for avoiding local minima by adopting force distribution were presented in [21], [25]–[28]. Approaches similar to APF that are popular are the Virtual Field Histogram (VFH) [10], [11] and Enhanced VFH+ [29] where the uncertainty from the sensor are modeled in histogram grids generated by the distance information from ultrasound sensor or range sensors. Unlike other obstacle avoidance algorithms, VFH considers the shape and dynamics of the robot, such as steering and control for navigation [9], [11]. APF and VFH can both work effectively in static and dynamic obstacles in limited conditions. Other planning methods considering dynamic obstacles in uncertain environments have been previously presented in works like [13], [30]–[32]. A recent interesting work by Ayawli et. al. presented VD-CGT based

planning in dynamic environment using Voronoi diagram and computational geometry where a small rectangular region is used around the robot to check for collisions [33]. Collision check based on virtual obstacles was presented in [34]. Distributed obstacle information sharing for collision detection and navigation in complex scenarios using single and multiple-robot systems is presented in [35], [36].

In recent years, sampling-based planning (SBP) methods are gaining much interest because of its efficiency and success rate. Sampling-based methods are unique in the sense that planning occurs by the connectivity of the C-space through deterministic or random function sampling. These functions build a graph or tree of all the possible feasible motions of the robot in the C-space [37]. Probabilistic Road Map (PRM) [38] and Rapidly Random Exploring Trees (RRT) [39], [40] are the most popular sampling-based planners. The former is a C-space planner that uses multiple-query planning, while the RRTs uses a tree representation for single-query planning in the C-space. The key idea of the PRM is to randomly distribute the nodes across the C-space and then connect these nodes using a simple local planner and straight lines to form a graph roadmap. By connecting the free space (C_{free}), the PRM is successful in finding faster paths by reducing the search to a graph. However, PRM does not perform well in narrow configuration spaces due to the random sampling that distributes nodes with constant density in C_{free} thereby reducing the volume spanned by the narrow spaces as compared to the total C_{free} [41]. Solution to this problem includes sampling more densely near obstacle boundaries since points in narrow passages lie close to the obstacle. But this is not always true and results in many points distributed away from narrow passage resulting in broken roadmaps [42]. A similar problem exists with the RRT, and many researchers have proposed variants to these planners to overcome this problem. For instance, the RRT* algorithm [43] is a variation of the original single-tree RRT that continually rewires the search tree for the shortest path search. Bidirectional RRT [44], [45] uses a bi-directional tree search for faster route planning. Similarly, [46], presented Lazy PRM, an improved PRM that minimizes the number of collision checks performed during the planning and therefore minimizes the running time of the planner. [47] presented improved connectivity of the roadmap by connecting previously generated connected components. PRM* an optimal variant of the original PRM algorithm was presented in [44]. The connection radius in PRM* is chosen as a function of number of sampling nodes. This makes the connection radius to decrease with the increase in the number of samples ensuring an optimal path between start and goal position. Also, PRM* is asymptotically optimal algorithm whereas PRM is only probabilistic complete. SBP methods uses a local planner for moving from one node to another on the constructed roadmap. The planning problem performs by finding the simple path from the start node to the roadmap and consequently from the roadmap to the goal node. Search based algorithms such as the Dijkstra [48] and the A* [49] algorithm are commonly used methods that find the minimum-cost paths on the constructed graph. For example, the A* algorithm operates by exploring from the node that is unexplored and has the minimum estimated cost [50], [51]. The total cost is estimated by accounting the weights on the edges that the robot takes while reaching to that particular node from the start node along with the cost-to-goal weight. These search planners are relatively fast and can work on medium to large scale maps. Once the plan is generated on the graph, path smoothing methods can be used to generate a smooth trajectory from start to goal by carefully considering the kinematics of the robot platform [52], [53].

In this work, we present a navigation solution for a mobile robot in 2D space with static and dynamic obstacles. We combine local reactive and global planning for efficient navigation in complex spaces. The local reactive method is based on the APF, and we introduce a new virtual force to the robot system that guarantees that the robot never gets stuck into the local minima. The new contribution of this work are: (1) Introduction of the new virtual force for local reactive navigation that works in sync with global planner. The proposed local planning algorithm always sway the robot away from the local minima and moves it towards the goal. The proposed local planner works in the presence of dynamic obstacles. (2) HPPRM with improved sampling using potential field to solve narrow passage problem by distributing nodes away from the obstacles and generates path that is shorter and much safer for the robot. Our method is capable of finding paths in complex maps with a fewer number of nodes with comparable computation cost. (3) We present a layered planner that combines local reactive and global planning in presence of dynamic obstacle with higher success rate.

The rest of the paper is structured as follows. Section III provides the problem formulation. Section IV-A gives the introduction to the artificial potential field method with mathematical formulation and introduces the virtual force based local minima avoidance algorithm. Section IV-B gives the introduction to the original PRM algorithm and discusses the disadvantages of the original PRM. Section IV-C introduces the proposed HPPRM algorithm with dynamic obstacles. Section V presents simulation and experimental results for local reactive navigation using the proposed virtual force and global planning using HPPRM. We also show the results of HPPRM in real environment considering dynamic obstacles. Finally, Section VI concludes the paper.

III. PROBLEM FORMULATION

This section defines the formulation used in the paper and defines the motion planning problem that is addressed for mobile robot navigation and path planning. Let Q be the set of sequence of an ordered list denoted as $\{q_i = Q(i)\}_{i \in N}$ of length $N \in \mathbb{R}$, and is mapped from $i \in \mathbb{N}$ to the i^{th} element of Q. Also, let $X \subset \mathbb{R}^d$ and represent the given state space and $d \in \mathbb{N}$, $d \geq 2$. Let C be the configuration space of all possible placements of moving object. The obstacle and obstacle-free state spaces are defined by C_{obs} and C_{free} such

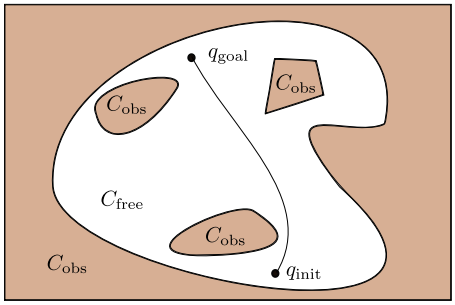


FIGURE 1. Configuration space.

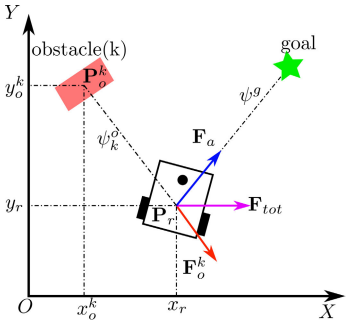


FIGURE 2. Force distribution model for artificial potential field on a 2D robot.

that $C_{\text{obs}} \subset C$ and $C_{\text{free}} = C \backslash C_{\text{obs}}$. Let the initial state is given by $q_{\text{init}} \in C_{\text{free}}$ and goal state is given as $q_{\text{goal}} \in C_{\text{free}}$ (Figure 1). Let τ be the non-zero scalar path length such that τ: $[0, 1] \rightarrow C_{\text{free}}$. An accurate geometric map $\mathcal{M}$ of the environment is also assumed to be available to evaluate C_{free} during the motion planning. A solution path is *feasible* if the connected path between $\tau(0) = q_{\text{init}}$ and $\tau(1) \in q_{\text{goal}}$ lies in the obstacle-free path C_{free}. Let U: $\mathbb{R}$ denotes the artificial potential function. There are three main problems addressed for path planning, namely: *feasibility*, *optimality*, and *speed*.

1) Feasibility: Let γ_{f} be the feasible trajectories that are represented by the cost function $c(.)$ and finds the Euclidean distance function between two positions $q_1, q_2 \in C_{\text{free}}$ denoted by $d(q_1, q_2) \in \mathbb{R}$ in the obstacle free configuration space C_{free}. The aim is to find the feasible path with minimum cost function $c_{\min}$.
2) Optimality: Given that all the feasible trajectories from the previous condition exists and an optimal path $\tau(.) \in \gamma_{\text{f}}$ exists for the distance function $c\{\tau(.)\}_{\min}$.
3) Speed: Given condition (1) and (2) are true, then find the path from $q_{\text{init}} \in C_{\text{free}}$ to $q_{\text{goal}} \in C_{\text{free}}$ in minimum time $T \in \mathbb{R}$ such that a set of controls $u : [0, T] \rightarrow \mathcal{U}$ and the motion satisfies $x(T) \in Q_{\text{goal}}$ and $q(x(t)) \in C_{\text{free}}$.

IV. HYBRID POTENTIAL BASED PROBABILISTIC ROADMAP (HPPRM)

Our proposed HPPRM method combines local and global navigation method for robot navigation in static and dynamic environment. First we describe the artificial potential based local navigation method and discuss the formulation and problems with the traditional APF. We present our solution to overcome some of these problems using the virtual force. Next, we describe the PRM method and the narrow passage problem and discuss reasons for poor performance of PRM in such scenarios. Finally, we present the HPPRM algorithm and its mathematical formulation.

A. LOCAL NAVIGATION USING ARTIFICIAL POTENTIAL FIELD METHOD

We first describe the local planner used for the navigation of the robot around obstacles. In this work, the artificial potential field (APF) method is used for the local reactive navigation of the mobile robots [14]. In this method, the robot is modeled as a moving point particle inside an artificial potential field that uses an attractive potential function to pull the robot towards the goal configuration and a repulsive potential function that pushes the robot away from the obstacles. The negative gradient of this potential function generates a force that can be used to guide the robot towards the goal while avoiding obstacles. Typically, the function consists of two components, attractive and repulsive. The attractive potential ($\mathbf{U}_a$) generated between the robot current position $\mathbf{p}_r = [x_r, y_r]^{\text{T}}$, and the desired goal position, $\mathbf{p}_g = [x_g, y_g]^{\text{T}}$, is as follows (see, Figure 2):

$$\mathbf{U}_a = c_g \left(1 - e^{\frac{-\|\psi^g\|^2}{l_g^2}}\right).\vec{n}_g, \tag{1}$$

where, ψ^g is the Euclidean distance between the goal and robot position in the space and is given by $\psi^g = d(\mathbf{p}_g, \mathbf{p}_r) = \mathbf{p}_r - \mathbf{p}_g$, and c_g and l_g are the strength and correlation distance for the goal destination. The first term c_g acts to make the attractive potential $\mathbf{U}_a$ zero when $\psi_g = 0$. The corresponding force is equal to the negative gradient of the potential, i.e $\mathbf{F} = -\nabla\mathbf{U}$, and is given by,

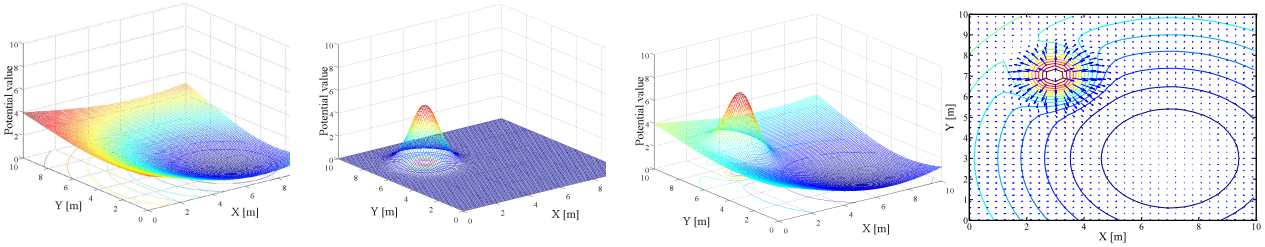
$$\mathbf{F}_a = -\nabla\mathbf{U}_a = -\frac{2c_g\vec{n}_g\psi^g}{l_g^2} e^{\frac{-\|\psi^g\|^2}{l_g^2}}, \tag{2}$$

$\vec{n}_g$ is the unit vector towards the goal destination and is given by,

$$\vec{n}_g = \frac{(x_g - x_r)\vec{i} + (y_g - y_r)\vec{j}}{\psi^g}. \tag{3}$$

The parameter c_g also defines the strength radius of a circular boundary around the goal position $\mathbf{p}_g$ in quadratic range. It allows the robot to move towards the goal position because of the highly attractive force generated between the robot position and goal position quickly. This force gradually decreases as the robot approaches the goal and varies conically allowing the robot to slow down when it is very close to the goal and thereby avoiding it to overshoot the goal position.

Similarly, the repulsive potential ($\mathbf{U}_o$) generated by any k^{th} obstacle whose position is given by $\mathbf{p}_o^k = [x_o^k, y_o^k]^{\text{T}}$ and the



(a) Attractive potential, $\mathbf{x}_g = (7, 3)$, $c_o = 5$, and $l_o = 8$ **(b)** Repulsive potential, $\mathbf{x}_o = (3, 7)$, $c_o = 5$, and $l_o = 1$ **(c)** Combined potential. **(d)** Gradient of the potential field

FIGURE 3. Example of Artificial Potential Field method.

robot is given by,

$$\mathbf{U}_o^k = \begin{cases} c_o \left(e^{\frac{-\|\psi_k^o\|^2}{l_o^2}} \right) .\vec{n}_o^k, & \psi_k^o \leq \psi_i \\ 0, & \psi_k^o > \psi_i \end{cases} \tag{4}$$

where, ψ_k^o is the Euclidean distance between the k^{th} obstacle and robot position and is given by $\psi_k^o = d(\mathbf{p}_o^k, \mathbf{p}_r) = \mathbf{p}_r - \mathbf{p}_o^k$. The parameters, c_o, l_o and ψ_i are the strength, correlation distance for obstacle avoidance, and influence distance, respectively. The repulsive potential is considered zero if the minimum distance ψ_o^k is greater than the constant factor ψ_i. Such a condition implies that the robot is at a large distance from the nearest obstacle and therefore in order for the robot to move towards the goal quickly, the repulsive potential is made equal to zero in Equation 4. The unit vector $\vec{n}_o$ towards the robot from the detected k^{th} obstacle is given by,

$$\vec{n}_o = \frac{(x_r - x_o^k)\vec{i} + (y_r - y_o^k)\vec{j}}{\psi_k^o}. \tag{5}$$

The corresponding force generated by the negative gradient is given by,

$$\mathbf{F}_o = -\nabla \mathbf{U}_o = \begin{cases} \frac{2c_o \vec{n}_o \psi_o^k}{l_o^2} \left(e^{\frac{-\|\psi_k^o\|^2}{l_o^2}} \right) .\vec{n}_o^k, & \psi_k^o \leq \psi_i \\ 0 & \psi_k^o > \psi_i. \end{cases} \tag{6}$$

The net overall repulsive potential from all the obstacles around the robot can then be calculated as,

$$\mathbf{U}_o = \sum_k \mathbf{U_o^k}, \tag{7}$$

and, the net repulsive force from all the obstacles is given by,

$$\mathbf{F}_o = -\nabla \sum_k \mathbf{U_o^k} = \sum_k \mathbf{F_o^k}. \tag{8}$$

The net overall potential generated by the attractive and repulsive potential is given by the force $\mathbf{U}_{tot}$ and is computed as,

$$\begin{aligned} \mathbf{U}_{tot} &= \mathbf{U}_a + \mathbf{U}_o \\ &= c_g \left(1 - e^{\frac{-\|\psi^g\|^2}{l_g^2}} \right) .\vec{n}_g + \sum_k c_o \left(e^{\frac{-\|\psi_k^o\|^2}{l_o^2}} \right) .\vec{n}_o^k. \end{aligned} \tag{9}$$

The net overall potential force is then given by,

$$\begin{aligned} \mathbf{F}_{tot} &= -\nabla \mathbf{U}_{tot} = -\nabla \mathbf{U}_a - \nabla \mathbf{U}_o \\ &= -\frac{2c_g \vec{n}_g \psi^g}{l_g^2} e^{\frac{-\|\psi^g\|^2}{l_g^2}} - \frac{2c_o \vec{n}_o \psi_o^k}{l_o^2} \left(e^{\frac{-\|\psi_k^o\|^2}{l_o^2}} \right) .\vec{n}_o^k. \end{aligned} \tag{10}$$

In general, the APF algorithm generates a smooth function over the robot's configuration space (C_{space}) that has higher values near the obstacles and lower values when the robot is further away from the obstacles. The forces acting on the robot under the influence of the obstacle and goal are depicted in Figure 2. The net total force $\mathbf{F}_{tot}$ moves the robot around the obstacles and towards the goal. The influence of the different parameters when drawing the potential is explained in Figure 3. When $\mathbf{p}_g = (7, 3)$, $c_g = 5$, and $l_g = 8$, the attractive potential ($\mathbf{U}_a$) is generated as shown in Figure 3a. When the obstacle position is $\mathbf{p}_o = (3, 7)$, $c_o = 5$, and $l_o = 1$, the repulsive potential is generated as shown in Figure 3b. The combined attractive and repulsive potential ($\mathbf{U}_{tot}$) is shown in Figure 3c. The gradient of the potential field represent the force vector which departs from the obstacles and into the goal position is as shown in Figure 3d. It shows the influence of the circular regions with respect to the distance to the obstacles. The outward force magnitude from the obstacles are stronger near the obstacles and weaker near the goal position thereby creating a force under the influence of which the robot can move down the slope and reach the destination safely, without hitting any obstacle.

The APF method works in real-time and performs best in environments where the obstacles are well distributed and wide enough with clearance for the robot to navigate around [21], [54], [55]. Moreover, since both the fields are independent of each other, it exhibits parallelism. Despite the efficiency of the APF method in finding the goal, there are many challenges when implementing it in the real environment, and it has been addressed in various literatures [56]–[58]. One of

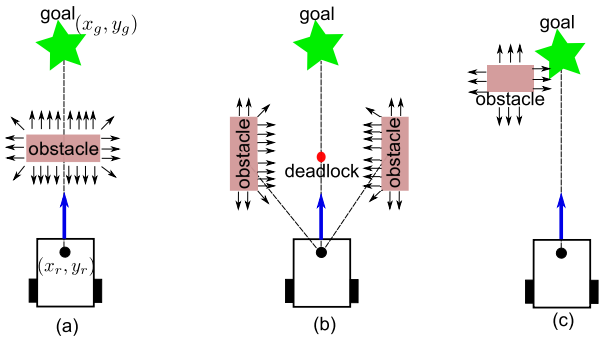


FIGURE 4. APF deadlock cases: (a) Goal-robot in line, (b) Symmetric obstacle, (c) Obstacle near the goal.

the main drawbacks is that the robot can get trapped in a local minima or deadlock position. This problem occurs when the attractive and repulsive forces generated by the goal and the obstacles nullify each other. These situations are depicted in Figure 4, and shows different cases where the robot might get trapped into local minima and did not find the goal. The deadlock situation can occur mainly when (a) the robot position and the goal position are collinear and equidistant to an obstacle (robot-obstacle-goal) position, (b) when the robot is around symmetrically arranged obstacles along the robot-goal line, and (c) when the goal position is very near to a large obstacle also called as a goal not reachable due to obstacle nearby or GNRON problem [56], [59] (Figure 4(c)). This is a very common problem with APF, and many researchers have tried to solve it [25].

1) LOCAL MINIMA AVOIDANCE

To solve the local minima problem, we present a new approach to avoid deadlock. To reach its local goal, it is enough to understand the relative position of obstacles and the goal concerning the robot i.e., the direction of the obstacles and the angle at which the goal is positioned from the robot. For this reason, we create a local motion planning method based on APF in a polar coordinate system. In this method, a robot acquires information of the surrounding environment at regular intervals, e.g., using a scanning range sensor or sonar sensor, and then creates an artificial potential field only in the range of sensing. After that, the robot only moves in a direction where the potential value is minimum. By repeating this motion, the robot can arrive at its destination without colliding with obstacles.

First, the Euclidean distance values from the sensor are converted into polar coordinates and the distance to potential function are plotted. For obstacles closer to the sensor, the potential value becomes higher, and smaller for obstacles that are further away. We do not consider the geometry of the obstacles at this stage, but only see the existence and non-existence of the obstacles. The two potentials (attractive and repulsive) are plotted against the angle of scanning by the sensor (θ_i to θ_g), ($i \in 1, \cdots, 180$) where θ_g is the angle of the goal position and the attractive potential field $\mathbf{F}_a$ is zero at this angle. Similarly, for the repulsive potential field $\mathbf{F}_o$ exhibited by the sensors field, the value of scanning angle (θ_i) where the obstacles (θ_o) are present will give a high potential. In general, as the scanning range of most range finders and integrated sonars are wide, obstacles in the near vicinity of the robot can be detected in one scan. As the robot continuously moves around and gather more scanning data, it can detect additional regions of scans where the obstacles are present and move towards the goal using reactive navigation. Figure 5a and Figure 5b shows the polar plot of the repulsive and attractive potential field for the scanning angle from the sensor in a real environment. In this example, the obstacles are placed at a scan angle of 10 deg, 40 deg, 95 deg, and 170 deg with respect to the sensor position. This is represented as repulsive peaks (red) in Figure 6a. The goal potential is located at 80 deg (θ_g) and is shown as the minimum value in Figure 6a (blue). The same goal angle in the polar coordinate system is shown in Figure 5b close to 80 degrees. The green line is the threshold value of δ, which can be set for different sensor configurations. Figure 6b, shows the combined cumulative potential for the threshold value. The regions where the cumulative value falls below the threshold represent open spaces in that scanning angle (θ_i). The regions whose cumulative potential value is higher than the threshold represent areas that are filled with obstacles. Following this approach, the region at scan angle 50 deg and 140 deg are found to be open areas, and the robot can pass through these regions in the map easily after carefully checking the passability. Based on the lowest potential value in the region, the robot will choose the direction as marked with a blue arrow in the polar coordinate plot as it is the direction without any obstacles and closer to the goal position (Figure 5c).

2) VIRTUAL FORCE DIRECTED LOCAL MINIMA AVOIDANCE

By constructing the potential functions for each obstacle for the sensing angle, we can determine the open and obstacle-rich areas in the map and program the robot to move towards the goal. There are likewise situations when the robot cannot completely avoid the deadlock situation because of equal force distribution from the obstacle and goal placement. In this condition, the robot will not move at all and get stuck in the deadlock position. To avoid trapping into the local minima or deadlock positions while navigating, we introduce a virtual force in addition to the existing potential forces to the robot that always ensures that the robot will move away from the local minima.

In the proposed method, the new virtual force and obstacle repulsive force are perpendicular to each other. The magnitude of the virtual force is proportional to the angle between robot heading and goal position (θ_g). This virtual force generates a repulsive rotational control that will move the robot from the local minima when it is trapped. The force act in such a way that it will always direct the robot towards the goal. Figure 7, shows the force distribution after introducing the virtual force ($\mathbf{F}_v$). When the robot detects any obstacle

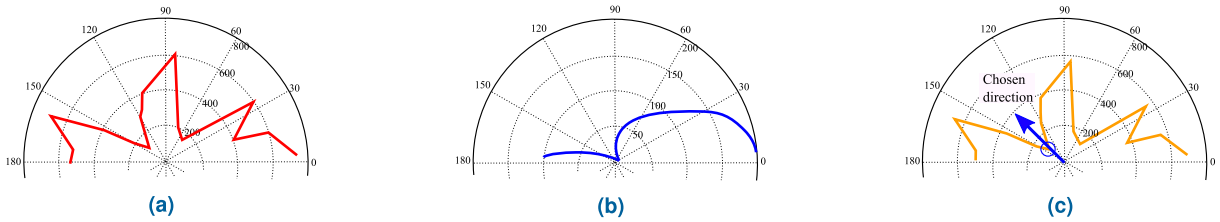
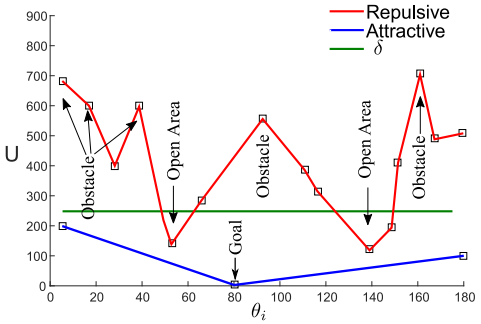
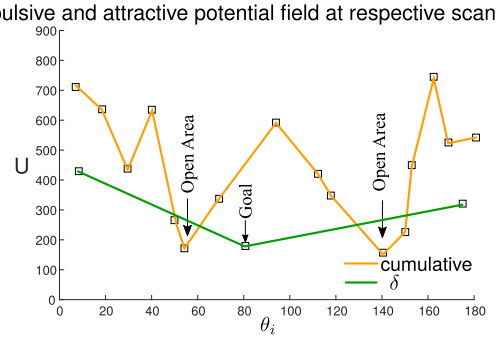


FIGURE 5. (a)Repulsive potential in polar coordinate system, (b)Attractive potential in polar coordinate system, (c) Cumulative potential in polar coordinate system. The arrow represents the chosen direction for robot motion.



(a) Repulsive and attractive potential field at respective scan angles.



(b) Cumulative potential values at respective scan angles.

FIGURE 6. Potential value (U) vs scanning angle θ_i at one sensor scan.

$[x_o, y_o]^{\mathrm{T}}$ in its sensing range, the new virtual force ($\mathbf{F}_v$) is added perpendicular to the original repulsive force $\mathbf{F}_o$. The virtual force $\mathbf{F}_v$ is defined as,

$$\mathbf{F}_v = M(\theta_g)\begin{bmatrix}\mathbf{F}_o^x & \mathbf{F}_o^y\end{bmatrix}^{\mathrm{T}}, \tag{11}$$

where, $\mathbf{F}_o^x$ and $\mathbf{F}_o^y$ are the x and y components of the repulsive force $\mathbf{F}_o$ generated by the obstacle at position $[x_o, y_o]^{\mathrm{T}}$. And the function $M(\theta_g)$ is given as,

$$M(\theta_g) = M_{max}(1 - e^{\mu.\theta_g})^{-1}. \tag{12}$$

The value of function $M(\theta_g)$ depends on the robot heading angle and goal position given by θ_g. When the angle is zero degrees, the value of the function $M(\theta_g) = M_{max}$, i.e., maximum value, and it decreases exponentially as the angle θ_g is increased, reaching zero at some predefined θ that is decided by the user. The value of μ is experimentally determined and is based on the type of sensor.

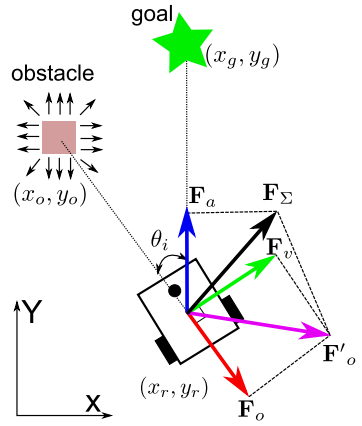


FIGURE 7. Virtual repulsive force for local minima avoidance.

The total force from Equation 10 becomes,

$$\mathbf{F}_\Sigma = \begin{cases} \mathbf{F}_a + \sum_k (\mathbf{F}_o^k + \mathbf{F}_v) & \text{if } (\psi_o^k \le \psi_g \;\&\; \psi_o^k \le \psi_i) \\ \mathbf{F}_a & \text{else } (\psi_o^k > \psi_g). \end{cases} \tag{13}$$

The net force $\mathbf{F}_\Sigma$ generated by the introduction of virtual force causes a rotational moment and moves the robot away from the obstacle. This is graphically shown in Figure 7 with the new force distribution on the robot body. The direction of rotation (left or right) is carefully selected and chosen so that the motion always leads towards the goal. To understand the effect of virtual force, consider the example, as shown in Figure 8 of local minima trap due to symmetric obstacle configuration. The repulsive forces due to the obstacle at position $[x_o, y_o]^{\mathrm{T}}$, and the attractive force generated by a goal at $[x_g, y_g]^{\mathrm{T}}$ on the robot body located at $[x_r, y_r]^{\mathrm{T}}$ position 'A' cancels each other. In this situation, traditionally, the robot is called to be in the local minima. However, by introducing the virtual force that is acting perpendicular to the repulsive force, the generated net force $\mathbf{F}_\Sigma$ will rotate the robot towards the right and away from the obstacle. Similarly, when the robot is positioned at 'B', 'C', and 'D' the net force moves the robot from the obstacle and towards the goal. The angle of goal heading θ_g at each position will determine the magnitude of the repulsive force. For E.g. when at position 'A', the goal position and robot heading are in straight line ($\theta_g = 0$). Therefore, the maximum repulsive force is generated at this

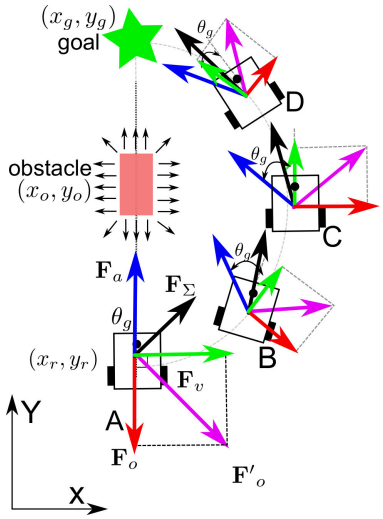


FIGURE 8. Virtual repulsive force for local minima avoidance in case of robot-goal in line situation.

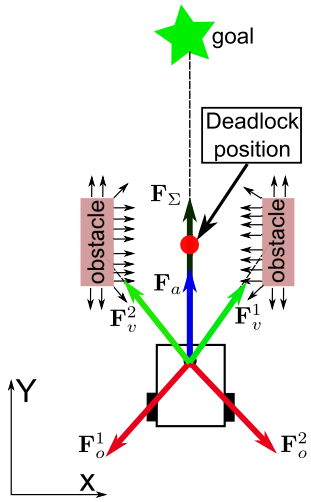


FIGURE 9. Deadlock avoidance in case of symmetric obstacles.

position. At positions 'B' and 'C' the magnitude of virtual force gradually decreases, and at 'D' the angle is near maximum making the repulsive force negligible. At this particular position, just by using the nearby goal's attractive potential, the robot can successfully reach the target position. By carefully determining obstacles and open areas the local minima is avoided.

Similarly, for symmetric obstacle case as shown in Figure 4b, where two obstacles are arranged symmetric to each other and a possibility of the robot to be trapped at local minima exists as shown by the red dot due to the force distribution. In this case, as well, using the virtual force method, the robot can easily avoid this situation, as indicated in Figure 9. Here the repulsive forces generated by the two obstacles are shown by forces $\mathbf{F}_o^1$ and $\mathbf{F}_o^2$. The corresponding repulsive forces are shown by $\mathbf{F}_v^1$ and $\mathbf{F}_v^2$ that acts perpendicular to the obstacle repulsive forces. The attractive force $\mathbf{F}_a$ and the resultant net force $\mathbf{F}_\Sigma$, acts such that the robot can tackle the deadlock and move in a straight line between the two obstacles to reach the goal. The results for the virtual force directed local minima avoidance are discussed in the results section.

B. GLOBAL PATH PLANNING METHOD

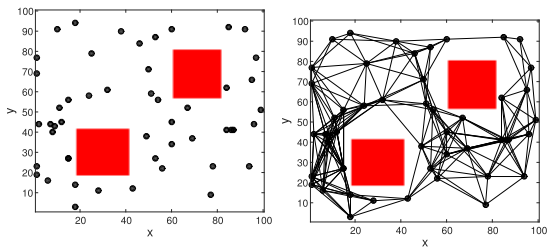
The local reactive navigation method using the artificial potential field works well in small to medium-sized maps and where global information is not available. However, for large environments, local reactive navigation becomes very difficult. Hence, we build a global path planning method under the assumption that the robot has a map of the environment in advance and uses it to guide the robot toward the goal. We used the multi-query probabilistic roadmap method for global navigation. We first give a brief overview of the classical PRM technique with its advantages and disadvantages. After that, we present our proposed hybrid potential based probabilistic roadmap (HPPRM) method.

1) PROBABILISTIC ROADMAP METHOD

For global planning, we used the modified probabilistic roadmap method. The PRM is one of the global planning methods that determine a path from the initial state to the global state in a given map. The PRM samples the given configuration space into free configurations and then tries to connect them using a roadmap for feasible motions. PRM is a multi-query sampling algorithm and works in two phases: the construction phase, and the query phase. At first, a roadmap R is constructed in the free configuration space C_{free} at the construction phase. The roadmap $R = (N, E)$ is a unidirected graph of sets of nodes N and a set of edges E. Here an edge corresponds to a simple, feasible path connected by a line segment between two nodes. These paths are also called as local paths and are calculated extremely fast by a powerful local planner. Initially, the roadmap R is empty. The sequence of the construction phase are as follows:

(i) Distribute node q in the free configuration space C_{free} as shown in Figure 10a and add to N.
(ii) Check the connectivity between q and its neighbor $N(q_i)$. If connectivity exists, a new edge (q, q') is added to the set of edges E as shown in Figure 10b.
(iii) Iterate (i) and (ii) V times where V is the number of all nodes.

Next, in the query phase, the roadmap is used to solve the individual path planning problem from a given initial state space configuration q_{init} to the goal state-space configuration q_{goal}. At first it tries to connect the initial and goal state-space configuration to some nodes q_{init} and q' in N. If a successful connection is found it searches the roadmap configuration R for sequence of edges connecting the nodes q' and q_{goal}. The connections are attempted between roadmap vertices that are within a fixed radius r from one another. This search can be done by depth-first search or breadth-first search or both, e.g., A* algorithm or Dijkstra algorithm. Finally a feasible path is obtained and concatenated to corresponding



(a) Node distribution ($V = 70$). (b) Connecting by edges.

FIGURE 10. Construction phase: The black dots, line segments and red objects represents nodes, edges, and obstacles, respectively.

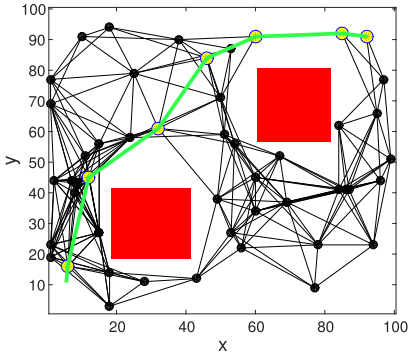


FIGURE 11. Shortest path calculated shown by green line.

local paths. The sequence of the query phase is as follows:

(i) The shortest distance D called distance to goal from an arbitrary node q_n to the goal is calculated. For any pair (q, q') of configurations, the shortest distance $D(q, q')$ can be defined as Euclidean distance and given as,

$$D(q, q') = \max_{q \in C_{\text{free}}} \left\| q - q' \right\|. \tag{14}$$

(ii) q' is added to array of path and set as a next closest neighbor of q_{init} in $N(q_{\text{init}})$ whose D is minimum among the neighboring node.

(iii) The shortest path from the initial point q_{init} to the goal point q_{goal} on the roadmap is obtained by iterating the operation ii) from the initial point until the value of D becomes zero (Figure 11).

Algorithm 1 (Construction phase) and Algorithm 2 (Query phase) describes the complete PRM algorithm.

PRM is a relatively simple method among the sampling-based path planning methods, and therefore it can generate the path efficiently in terms of calculation cost. However, there are some problems with PRM, especially when handling narrow passages that lead to (i) unconnected and fractured graph, and (ii) increased calculation cost concerning the increase in the number of nodes. The unconnected problem occurs when a path between the initial and goal points are unconnected. This occurs due to the random sampling of the nodes in the PRM initial stage. The nodes are generally sampled uniformly and randomly for the whole configuration space (C_{free}), and therefore the possibility of sampling from narrow and complex spaces is relatively low. Figure 12, explains an example of this problem when the start and goal point are $s = (5, 5)$, and $g = (95, 95)$, respectively. It can be seen that PRM did not sample any nodes in the narrow region, and therefore no connectivity between the nodes on the left and right side of the obstacles could be generated, resulting in failed plan.

The second problem is of increased calculation cost with an increase in the number of nodes. Assuming the worst case condition, the nodes and number of edges are given as N and E, respectively. The computation cost of PRM in the construction phase is $\mathcal{O}(N^2)$ at maximum. In the query phase of the PRM cycle, the calculation cost depends on finding the shortest path. For the Dijkstra method, the calculation cost is $\mathcal{O}(N^2)$, and if the improved Dijkstra method with the Fibonacci heap is used, the calculation cost is $\mathcal{O}(E + N)\log N$. If we adopt the A* method for the shortest

Algorithm 1 Construction Phase Algorithm

```
N ← 0
E ← 0
q_i ← sample from C_free
k: number of closest neighbors to choose in each configuration
N(q_i) ← closest neighboring nodes of q chosen from N
for i = 1, ··· , V do
    q ← a node randomly sampled from free space
    N(q_i) ← k closest neighboring nodes of q chosen from N
    for each q ∈ N(q_i) do
        if there is no collision between q and q' from q to q_i and there is not already an edge from q to q_i to the roadmap R then
            E ← E ∪ (q, q')
        end if
    end for
end for
return R
```

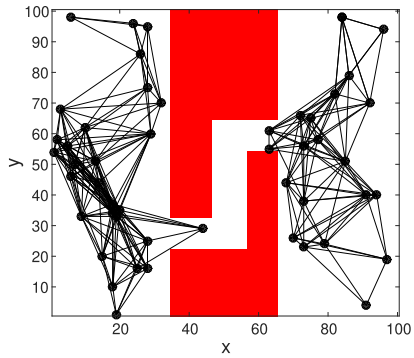


FIGURE 12. Unconnected problem in PRM.

path, the calculation cost is $\mathcal{O}|E|$. When enough nodes are distributed to make a path, the number of edges exceeds the number of nodes, and therefore the computation cost increases significantly for the number of nodes. For e.g., in Figure 11, if the number of nodes is increased from $N = 40$ to $N = 160$ at intervals of 20, the calculation time will rapidly increase with the number of nodes. This is shown in Figure 13. Here the calculation time is the meantime measured 20 times of successful path planning.

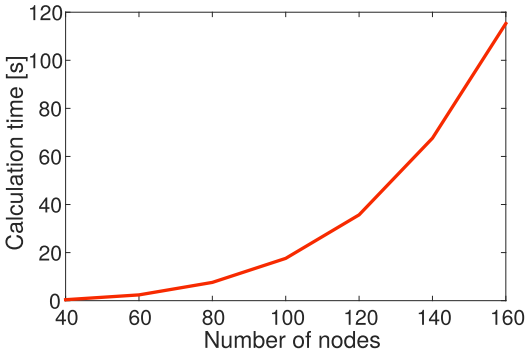


FIGURE 13. Relation between number of nodes and calculation cost in PRM.

C. HYBRID POTENTIAL BASED PROBABILISTIC ROADMAP (HPPRM) ALGORITHM

In this section, we present an improved Hybrid Potential based Probabilistic Roadmap (HPPRM) method to overcome the problems discussed in the previous section. The proposed method uses the artificial potential field method and map segmentation to improve the sampling of nodes during the PRM process. The outline of our proposed method is as follows:

1) First, a potential map is created for a given map based on the obstacle information. The equation to generate the repulsive potential field $\mathbf{U}_o$ is the same as that described in Equation 4. Note that, here, we only use repulsive potential for obtaining regions of obstacles. Here $\mathbf{U}_o$ takes a real number from [0, 1], and the value is stored into all grids.
2) Next, we decompose the potential map into equal m_x and m_y grids in x and y directions, respectively (Algorithm 3).
3) Each grid area is classified into two areas, and the total number of nodes to be placed in each region is decided. The total potential value of each area given as $T_P(i, j)$, where $(1 < i < m_x, 1 < j < m_y)$ is calculated as,

$$T_P(i,j) = \sum_{k=\frac{1+(i-1)X}{m_x}}^{\frac{iX}{m_x}} \left(\sum_{l=\frac{1+(j-1)Y}{m_y}}^{\frac{jY}{m_y}} p(k,l) \right), \quad (15)$$

where, X and Y are the number of nodes in x-direction and y-direction in the given map, and $p(k, l)$ is the potential value at a grid (k, l). Next, we define the median value of the total potential $T_P(i, j)$ as a global threshold as below,

$$T_G = \text{median}\,(T_P(i,j))\,. \quad (16)$$

The areas whose $T_P(i, j) > T_G$ are defined as 'high potential' regions, and the areas whose $T_P(i, j) < T_G$ are defined as 'low potential' regions. Coefficients a and b are added to high potential and low potential regions respectively, where a and b satisfies the following relation,

$$a = 1 + k, \quad b = 1 - k (0 < k < 1), \quad (17)$$

Multiplying a and b to $meanNum = V/(m_x m_y)$ depending on the region, we finally get the number of nodes in each region as,

$$numNodes(i,j) = \begin{cases} \dfrac{aV}{m_x m_y} & \text{at high potential region} \\ \dfrac{bV}{m_x m_y} & \text{at low potential region} \end{cases} \quad (18)$$

4) Node distribution: We define the mean potential value of each region as $LocalThreshold(i, j)$, and distribute the

Algorithm 2 Query Phase Algorithm

```
N(q_init) ← k closest neighboring nodes of q_init} chosen from N
N(q_goal) ← k closest neighboring nodes of q_goal chosen from N
N ← {q_init} ∪ {q_goal} ∪ N
set q' to be the closest neighbor of q_init in N(q_init)
repeat
    if D(q_init, q') ≠ 0 then
        E ← (q_init, q') ∪ E
    else
        set q' to be the next closest neighbor of q_init in N(q_init)
    end if
until a connection was successful or N(q_init) is empty
set q' to be the closest neighbor of q_goal in N(q_goal)
repeat
    if D(q_goal, q') ≠ 0 then
        E ← (q_goal, q') ∪ E
    else
        set q' to be the next closest neighbor of q_goal in N(q_goal)
    end if
until a connection was successful or N(q_goal) is empty
P ← shortest path (q_init, q_goal, R)
if P is not empty then
    return P
else
    return failure
end if
```

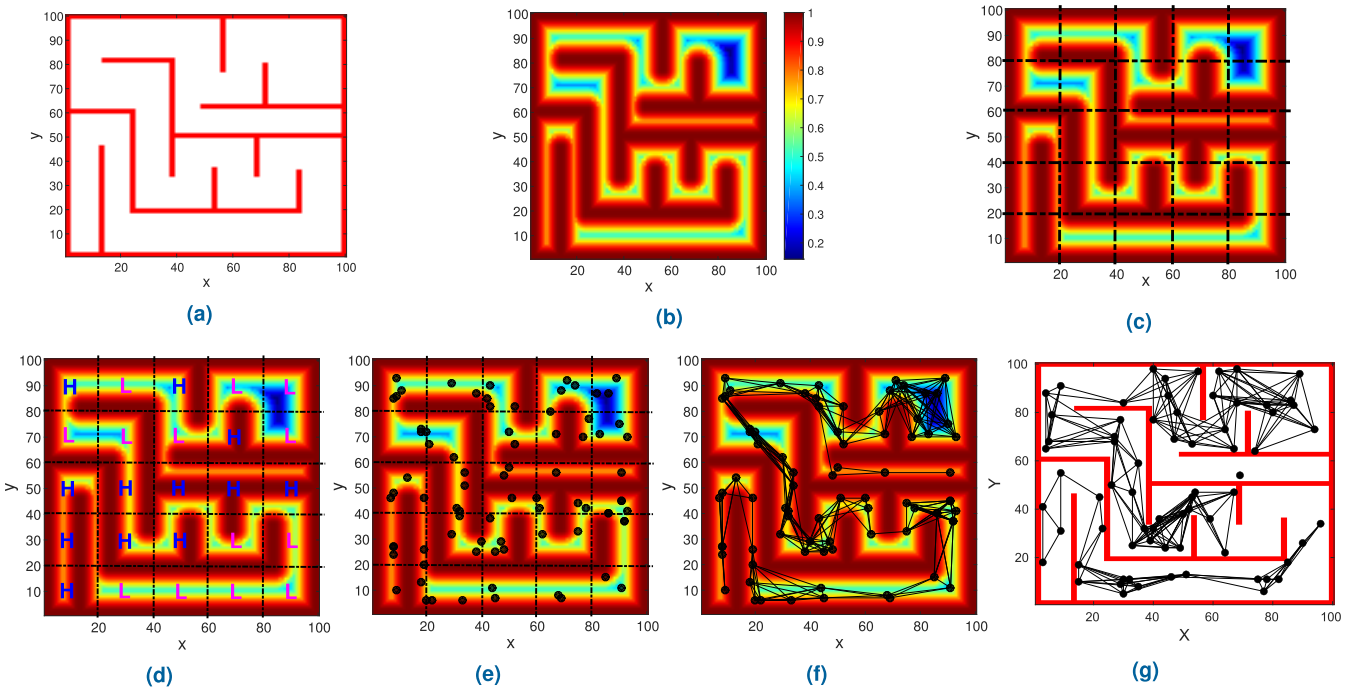


FIGURE 14. HPPRM process, (a) Map environment, (b) Repulsive potential map, (c) Grid decomposition, (d) High (H) and low (L) potential area classification, (e) Node distribution based on potential, (f) Roadmap construction by PRM, (g) Node distribution in original PRM.

Algorithm 3 HPPRM - Map Decomposition and Node Distribution

$U_0 \leftarrow$ Repulsive potential
$V \leftarrow$ Number of nodes
$(m_x, m_y) \leftarrow$ Decomposition of grids in x and y direction

$T_P(i,j) \leftarrow$ Sum potential of each grid
$T_G(i,j) \leftarrow$ Global potential threshold
for $m_x \times n_y$ areas **do**
  calculate $T_P(i,j)$
end for
$T_G(i,j) \leftarrow median(T_P(i,j))$
$meanNum \leftarrow V/(m_x m_y)$
for $m_x \times m_y$ areas **do**
  if $T_P(i,j) > T_G(i,j)$ **then**
    $numNodes(i,j) \leftarrow a \times meanNum$;
  else
    $numNodes(i,j) \leftarrow b \times meanNum$;
  end if
end for

Algorithm 4 HPPRM - Sampling Algorithm

$V \leftarrow$ Number of nodes
$(m_x, m_y) \leftarrow$ Decomposition of grids in x and y direction

$T_P(i,j) \leftarrow$ Sum potential of each grid
$T_G(i,j) \leftarrow$ Global potential threshold
for $m_x \times n_y$ areas **do**
  calculate $T_P(i,j)$
  for $k = 1$ to $numNodes(i,j)$ **do**
    while *True* **do**
      $n \leftarrow$ randomly choose from the area
      if $occgrid(n) \neq T_P(i,j)$ **then**
        print
      end if
    end while
  end for
  add n to roadmap as a new node
end for

nodes to the grids whose potential value is less than $LocalThreshold(i,j)$.

5) The method to connect the nodes by edges and query phase is same as the original PRM.

The complete process of the proposed HPPRM is shown in Figure 14 and explained in Algorithm 3 and Algorithm 4. The complex map region with obstacles in red is shown in Figure 14a. The corresponding potential map of the obstacles is given in Figure 14b. Next, the potential map decomposition into equal m_x and m_y grid regions in respective x and y directions is shown in Figure 14c. Based on the calculated total potential for each region $T_P(i,j)$, and the global threshold T_G, regions of 'high potential' and 'low potential' are marked on the map as H and L, respectively and shown in Figure 14d. Next, node distribution based on the mean potential value of each region as $LocalThreshold(i,j)$ is calculated, and nodes are distributed to high potential and low potential regions, and the result is shown in Figure 14e. The node-edge joining

and query phase of the original PRM on the new distributed nodes is presented in Figure 14f. For comparison, the node distribution on the same map for the original PRM is shown in Figure 14g. The random distribution of nodes in the original PRM resulted in areas where no nodes are dispersed. This results in a unconnected graph, and therefore the PRM fails to produce any results. From the results, it can be seen that for the proposed method, the nodes distribution is improved based on the obstacle position.

D. HPPRM WITH DYNAMIC OBSTACLES

Path planning generally considers obstacles in the map to be static. However, in real environments, the obstacle position is not fixed and may change with time. Such changes occur when obstacles are displaced, or new obstacles are added to the environment. In such situations, the robot needs to replan its path considering such dynamic changes. In our case, dynamic changes refer to obstacles that are displaced and not part of the initial map used for path planning, e.g., boxes or chair added during the planning phase. Other types of moving dynamic obstacles, such as humans and animals, are not considered in our research work. Avoidance considering dynamic obstacles has been previously studied in [52], [60], [61].

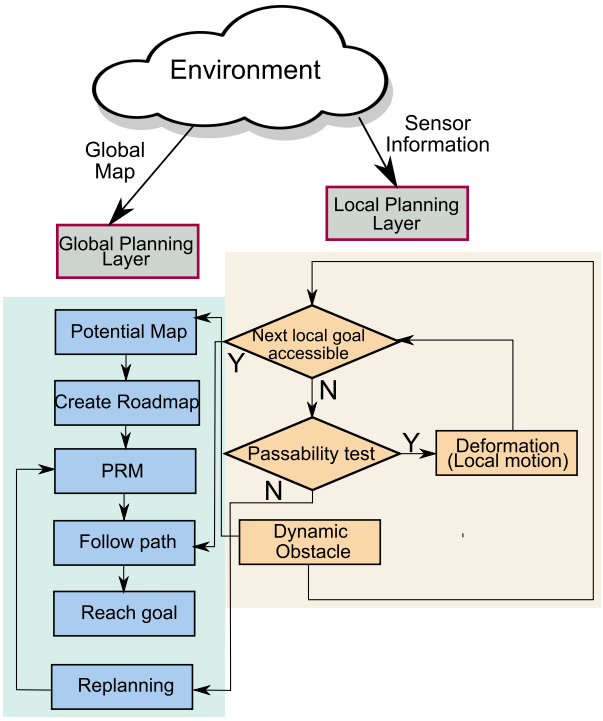


FIGURE 15. Flowchart of the layered planner.

HPPRM can generate effective path planning with dynamic obstacles in the map. For this purpose, HPPRM adopts a layered planning method for tackling dynamic obstacles that appear during the planning phase. The flowchart of the layered planner is presented in Figure 15. It is divided into two planning layers, the global planning layer, and the local

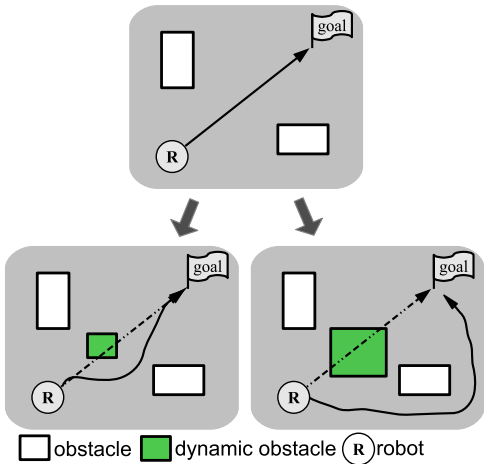


FIGURE 16. Deformation and replanning in dynamic environment.

planning layer. In the global planner, the HPPRM first creates the potential map of the obstacles using the global map information and then creates the roadmap for the PRM. If there are no dynamic obstacles in the scene, the PRM planner follows the path and reach the goal. The local planner, on the other hand, uses sensor information for reactive local navigation. If during the global planning, dynamic obstacles are detected by the planner, the potential map is updated with new obstacle information. The robot continuously checks the accessibility to the next local goal. If it can access the next local goal, the robot continues to follow the path. If not, it then checks whether the space between the obstacles are wide enough for the robot to pass through. When the robot can pass around the obstacles, it uses the local motion planning method and deforms its path to avoid the obstacle, as described in Figure 16. Otherwise, a new path is planned considering the given map and dynamic obstacle since it is difficult move in the direction of the earlier planned path. The process is iterated until the robot reaches the goal position.

V. RESULTS

In this section, we summarize the simulation and real experiment results for virtual force directed local reactive navigation and the proposed HPPRM for global planning.

A. LOCAL PLANNING USING VIRTUAL FORCE

In this section, we present the results of local reactive navigation in the simulation environment. Simulation tests were performed on five sample maps with different levels of obstacle density. The test also includes the local minima trap configuration for confirming the validity of the method. The local planning results are given in Figure 17. Map A and B show the situations where the local minima trap occurs in the case of traditional APF. By using the virtual force configuration, the robot could successfully avoid getting into the deadlock and arrive at the goal from the test results. In map B, due to the cumulative forces generated from the potential field, the robot could pass the narrow passage (symmetric obstacles)

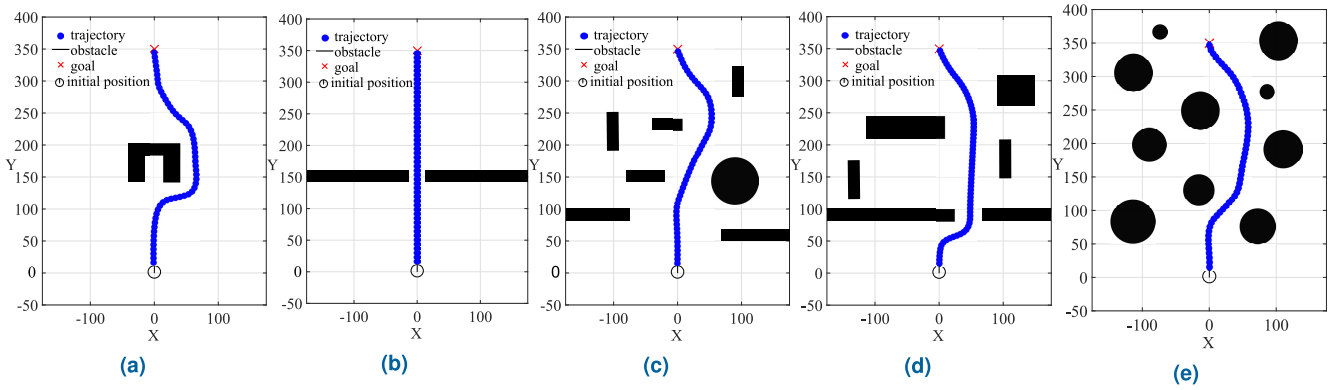


FIGURE 17. Simulation results of local reactive planning using the virtual force. (a) Map A (Deadlock), (b) Map B (Symmetric), (c) Map C (complex 1), (d) Map D (complex 2), (e) Map E (circles).

TABLE 1. Local reactive navigation path length and execution time(s) (Simulation Test), Proposed vs TAPF.

Environment	d_{mean}(Proposed)	d_{mean} (TAPF)	T_{mean} (s) (Proposed)	T_{mean} (s) (TAPF)	S_r (%) (Proposed)	S_r(%)(TAPF)
Map A	389	467	3.6	13.2	96 %	6 %
Map B	350	432	2.8	12.8	95%	18 %
Map C	391	497	4.2	9.5	100 %	78 %
Map D	394	502	4.4	11.4	100 %	39 %
Map E	397	486	4.1	13.8	100 %	63 %

right through the narrow gap and reached the goal. Map C, D, and E present a varying levels of obstacle density and the proposed method could find the goal successfully in all the cases. In all the test environments, the start and goal positions are fixed at $(0, 0)$ and $(0, 350)$, respectively. The value of parameter c_g was determined from Equation 1. The value of c_g is kept low when obstacle density is high, such that the priority is for obstacle avoidance. However, keeping a large value of c_g is not beneficial in dense obstacle scenario as it may lead to robot hitting the obstacle. Therefore we varied the value of c_g for each configuration before the experiment. For the experiment tests the value of c_g ranges from $[0.5, 2.1]$. The robot checks the passability by calculating the attractive and repulsive potential from the obstacles and determines the open areas by using the method described in Figure 6 (Section IV-A1). The mean path lengths (d_{mean}) and mean execution time (T_{mean}) for each environment is listed in Table 1. The simulation tests were performed 30 times, and the mean path length and execution time along with the success rate (S_r) for both the proposed and traditional APF method (TAPF) were recorded. The motion model of the robot in the simulation test is omnidirectional, meaning it can move in 360-degree regions. The success rate of the proposed method in all the five environments was over 95%, whereas for the TAPF, the simulation failed in most environments as the robot collided into obstacles or got stuck at local minima. The mean path lengths and execution time are also smaller for the proposed method than the TAPF. Moreover, using the virtual force, the robot will always escape the local minima trap by carefully selecting open spaces by thresholding the repulsive and attractive potentials at each step of sensor measurement.

B. EXPERIMENTS IN REAL ENVIRONMENT: LOCAL REACTIVE NAVIGATION WITH VIRTUAL FORCE

The local reactive navigation using virtual force was tested in a real environment with varying obstacle densities and local minima trap configurations. Tests were performed in three environments with an actual robot. Performance was evaluated on 3 parameters namely: mean path length (d_{mean}), mean execution time (T_{mean}), and success rate (S_r). The three test environment are shown in Figure 18. A grid map of the environment was constructed using the 2D laser range sensor (LRF), and the obstacle information during the motion was sensed using the LRF sensor. Figure 19 shows the results of the local reactive navigation using virtual force. The actual trajectories of the robot plotted on the grid map of the environments are shown in Figure 20. The trajectories of the robot are shown in blue (Figure 19) and green (Figure 20), respectively. Table 2, shows the performance evaluation of the test in a real environment. The tests were conducted 15 times for each environment, and the mean path length and average execution time with success rate were recorded. The value of $c_g = 1.2$, $c_g = 1.9$, and $c_g = 2.1$ were selected for Environment 1, Environment 2, and Environment 3, respectively. The values are determined experimentally. Again, in the real environment test, the proposed method's success rate is higher than 95%. The mean path lengths and execution time is also

TABLE 2. Real environment local reactive navigation path length(m) and execution time(s), Proposed vs TAPF.

Environment	Start	Goal	d_{mean} (Proposed)	d_{mean} (TAPF)	T_{mean}(s) (Proposed)	T_{mean}(s) (TAPF)	$S_r(\%)$ (Proposed)	$S_r(\%)$(TAPF)
Env. 1 (Fig. 18a)	(-20,20)	(25,370)	412	491	7.6	16.9	98 %	72 %
Env. 2 (Fig. 18b)	(50,10)	(25,380)	362	412	5.1	23.3	95%	18 %
Env. 3 (Fig. 18c)	(20,20)	(25,380)	401	448	6.9	28.1	97 %	23 %

FIGURE 18. Real test environment for the improved virtual force directed APF local navigation.

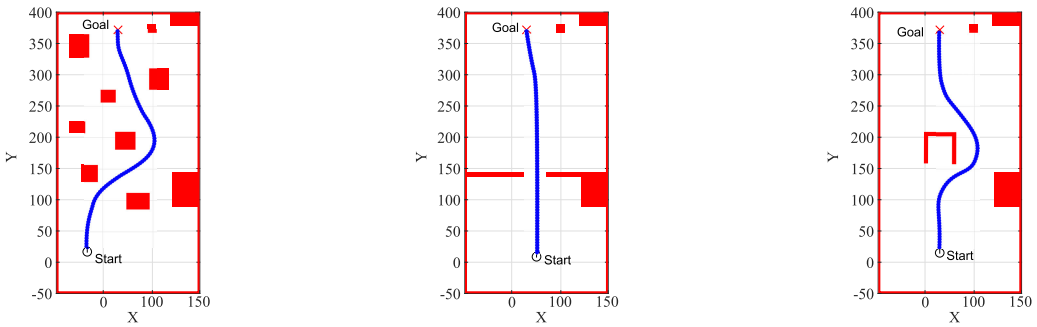


FIGURE 19. Real local reactive navigation results.

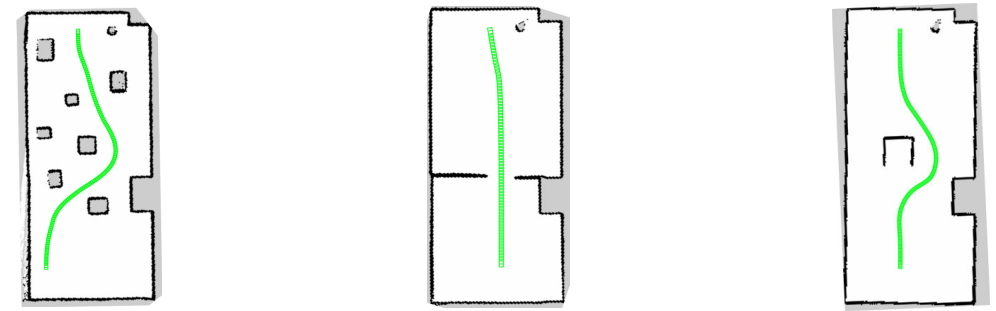
shorter compared to the traditional method. In many test runs, the TAPF could not complete the total run due to crashing into obstacles. Such test runs were excluded from calculating the mean values.

C. DYNAMIC OBSTACLE TEST

We tested the proposed method on randomly moving obstacle with varying speed and obstacle numbers. The simulation test environment consists of a single map with 8 large static obstacles. Tests were done with 8, 15, 30, 45 and 60 moving obstacles in the map. The obstacles are generated randomly with respect to their positions in the map and move with different velocities in each simulation run. The simulations were performed on MATLAB and Python programming language. The global planner is used to construct an initial path for the robot and reactive local navigation is performed as the robot encounters moving obstacles during the navigation. The robot recalculates the path towards the goal by calculating the difference between current heading and goal heading. APF around the obstacles is constructed and virtual force directed local reaction method is used to avoid collision with the moving obstacles.

Figure 21 shows the navigation result of the robot with 50 dynamic obstacles shown in red. The robot start position (0.2,2) and the goal position (0,-2) are represented by yellow and green circles, respectively. All the obstacles are randomly generated and are modeled to pass through the static obstacles. Moreover, every dynamic obstacle exhibit its own repulsive field while moving.

Firstly, PRM algorithm is used to generate the roadmap from start to goal position and the global path is calculated. A repulsive potential field around the static and dynamic obstacles is produced and a low potential path towards the goal is calculated. At each step of the iteration, the robot checks the new repulsive force mainly generated by the dynamic obstacles in the map and a cumulative force is calculated based on the virtual force directed reactive navigation policy discussed earlier. The gradient of the calculated force vector determines the velocity at which the robot should cross the dynamic obstacle. By continuously following this policy,



(a) Environment 1 (b) Environment 2 (c) Environment 3

FIGURE 20. Actual trajectories plotted on the grid map of the test environment.

TABLE 3. Path planning result in dynamic environment.

Method	Obstacles	Mean Path Length (D_{mean})	Mean Cost (T_{mean})	$S_r(\%)$
TAPF	15	11.8	3.51	87
	30	13.5	3.67	69
	45	18.9	3.51	53
	50	21.1	3.72	31
	60	28.1	3.48	19
PRM	15	10.3	1.85	83
	30	14.4	2.41	74
	45	18.9	2.86	48
	50	23.6	2.63	27
	60	26.8	2.79	14
A*	15	10.2	1.13	89
	30	13.1	1.25	71
	45	16.5	1.78	63
	50	19.3	2.14	47
	60	24.7	2.38	24
Proposed	15	9.42	2.57	94
	30	11.6	2.41	82
	45	13.9	2.63	76
	50	14.3	2.81	72
	60	18.9	2.59	67

the robot is able to reach the goal position. The velocity plot of robot during the navigation is shown in Figure 22. Maximum velocity during the simulation run was 1.23 m/s while the average velocity during the run was 0.58 m/s. The variation in velocity during the entire run from start to goal confirms the APF response as it gets closer to the goal approaching zero velocity. Also, the virtual force creates sudden transitions in the velocity to overcome hitting the dynamic obstacle during its motion. The complete trajectory of the robot with final obstacle position is shown in Figure 21 with start and goal positions denoted by 'S' and 'G', respectively.

A comparative analysis of the dynamic obstacle simulation test was done with three well known methods, the traditional APF (TAPF) [14], PRM [38], and A* algorithm [49]. A total of 250 simulations were done for each of the method and the mean path length (D_{mean}), mean calculation time (T_{mean}), and success rate $S_r(\%)$ were calculated. The results are described in Table 3. From the results, TAPF performs fairly well when the number of obstacles are lower but the performance deteriorate as the number of obstacles increases. Since TAPF depends on the potential field distribution for safe navigation, increased number of obstacles generate low potential areas in the map resulting in failure. This results in longer duration to complete the planning and in many cases hitting the obstacle. The PRM algorithm exhibits a similar results with a faster calculation cost. For PRM test, the number of nodes were fixed at $N = 100$, and hence the same roadmap is used with different number of obstacles. During the dynamic navigation the switching of nodes for recalculated path results in longer time to complete the mission. When obstacles are increased to over 45, in most cases the robot cannot complete the planning due to collision or robot stuck situations. The

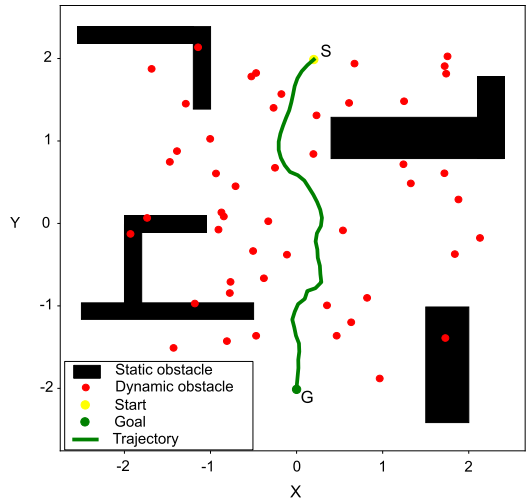


FIGURE 21. Trajectory (green) for the robot during the navigation.

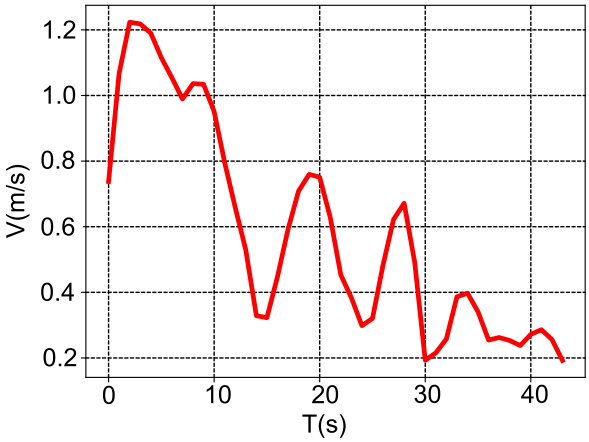


FIGURE 22. Velocity plot of the robot. Average velocity (0.58 m/s).

A* algorithm is computationally efficient in finding faster plan due to its heuristic characteristics but exhibits poor performance as the number of obstacles increases. Here the recalculation of heuristic cost to reach the goal increases as the number of obstacle increases. In comparison the proposed method generated most number of successful planning in all cases with a success rate of 94%, 82%, 76%, 72%, and 67%, respectively for different number of obstacles. With the virtual force assisted APF, the algorithm can avoid robot trapping scenarios effectively giving higher success ratio as compared to other methods. The time cost and mean path length are also the best among other methods.

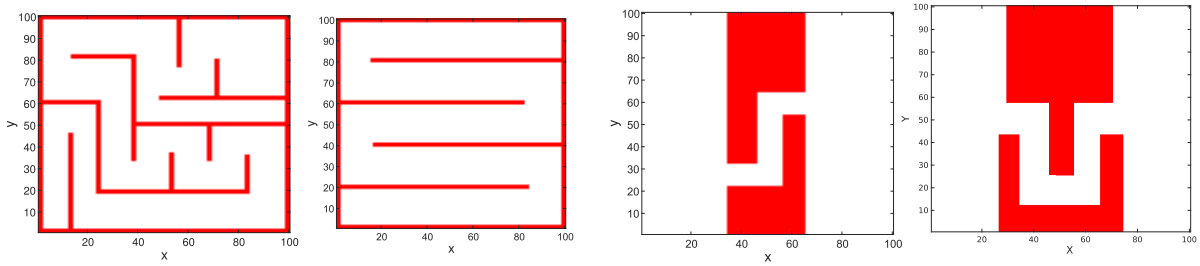
D. HPPRM-EFFECTIVENESS

We present the effectiveness of the proposed HPPRM with the original PRM mainly with respect to complex maps and narrow regions. As mentioned earlier, PRM suffers from unconnected problem in narrow areas since there are not enough nodes distributed in the narrow space, and many nodes are distributed in the open regions of the map. Furthermore, increasing the number of nodes will result in higher computation cost and our goal is to keep the nodes as few as possible for faster plan generation. Using HPPRM, the nodes can be distributed in narrow spaces since there must be more than one node in each region. Also, because our method uses the grid regions' potential values and divides them into high and low potential regions, nodes can be distributed with more confidence. For complicated maps with many obstacles, it is relatively difficult to join the edges with only a few nodes. On the other hand, it is easier to connect edges in regions with fewer obstacles. Therefore our method can solve the narrow region problem of PRM without increasing the number of nodes. Using potential field distribution and proposed node distribution method, a path can be found with fewer nodes, even in relatively low potential areas. The proposed method puts relatively more nodes at turning areas than straight areas based on the region potential value. Owing to this node distribution based on potential field, the nodes get closer to each other, and we can avoid the disconnected problem associated with the original PRM.

To test the connectivity of HPPRM in narrow spaces, we tested our proposed algorithm on four maps, as given in Figure 23. The four environment maps (complex map, corridor map, narrow map (1), and narrow map (2)) typically represent narrow passages where PRM suffers from connectivity problem. In all the four maps, same number of nodes $N = 70$ with start $= (5, 5)$, goal $= (95, 95)$ are used. The result is described in Figure 24. In all the cases, with node $N = 70$, the original PRM failed to produce results in maximum trial runs. On the other hand, HPPRM could successfully find the path in narrow spaces with a higher success rate than the original PRM. The node distribution using potential field generates nodes at narrow spaces, and the final path is then calculated using the PRM method.

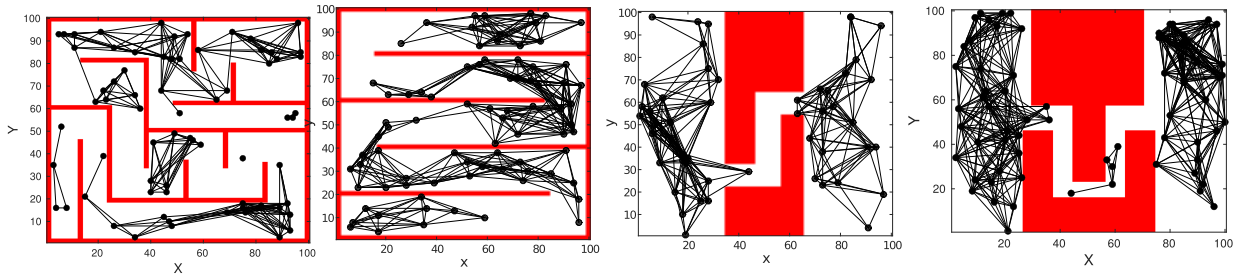
E. HPPRM- EVALUATION

We evaluated the effectiveness of the HPPRM on mean calculation time $t_{\text{mean}}(s)$, success rate $S_r(\%)$, and mean distance for the path d_{mean} on four environments and compared it with the original PRM by changing the number of nodes. HPPRM is a probabilistic complete method as it uses the original PRM, however it is not asymptotically optimal. On the other hand PRM* is asymptotically optimal [44]. Therefore, to test the optimally complete solution we also made tests by combining potential method with the PRM* algorithm and naming it HPPRM*. PRM* is an optimal variant of the original PRM algorithm with the only difference that the connection radius r is used as a function of number of nodes N that have already been placed. This allows the connection radius to decrease with the increase in the number of samples ensuring that the final path connecting q_{init} and q_{goal} is optimal. The test environments are given in Figure 23 and comprises different scenarios. In each of the given maps, both the PRM and PRM* has trouble with the narrow passage problem. We evaluated HPPRM based on mean computation time, mean path length, and success rate with the original PRM and PRM*.

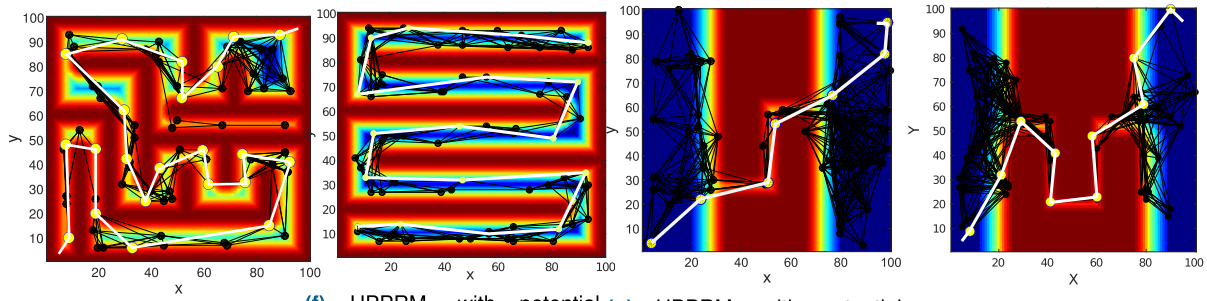


(a) Complex map (b) Corridor map (c) Narrow passage map (1) (d) Narrow passage map (2)

FIGURE 23. Example environment for connectivity test of the HPPRM with $N = 70$, start $= (5, 5)$, goal $= (95, 95)$.



(a) Complex map (A)- Original PRM (failed) (b) Corridor map (B)- Original PRM (failed) (c) Narrow map (C)- Original PRM (failed) (d) Narrow map (D)- Original PRM (failed)



(e) HPPRM- Complex map (A) - Success (f) HPPRM- with potential (success) (g) HPPRM- with potential (success) (h) HPPRM- with potential (success)

FIGURE 24. Results of PRM and HPPRM in test environments. The success rate of HPPRM in number of trials is over 90% for ($N = 80$, *start* $= (5, 5)$, *goal* $= (95, 95)$).

TABLE 4. Experiment parameters.

Parameter	Value
c_o	1
l_o	10
m_x	5
m_y	5
a	1.3
b	0.7
start	$(5, 5)$
end	$(95, 95)$

A total of 100 tests for each environment were performed, and the mean values were calculated. The values of different parameters set for the simulation is listed in Table 4.

The evaluation result for environment A (Figure 23a) is listed in Table 5. The map is a complex environment with many passages and straight walls. When the number of nodes is set as $N = 70$, the success rate of PRM is 35.7% and that of PRM* is 42.7%, while the success rate for HPPRM is 89.3% and that of HPPRM* is 87.6%. The calculation cost (s) is 0.79s in PRM and 2.11s in PRM*, which is better than that of HPPRM (0.84 s) and HPPRM*(1.80s). The total length of the path for PRM is 449 and PRM* is 456. The path length for HPPRM is 416 while HPPRM* is 433. When increasing the number of nodes to $N = 100$, the success rate of HPPRM and HPPRM* is higher at 88.3% and 89.9% respectively, compared to PRM at 65.3% and PRM* at 68.1%. The mean calculation time for HPPRM (2.76s) is higher than HPPRM* (4.81s). The mean path length for HPPRM (427) is lower in all cases.

The evaluation result for environment B (Figure 23b) is listed in Table 5. When the number of nodes are fixed at $N = 70$, the success rate of HPPRM is 92.1%, HPPRM* is 90.8%, while PRM is 18.3% and PRM* is 29.1%. The mean calculation time for HPPRM is at 2.16s, HPPRM* is at 5.50, PRM is at 0.94s, and PRM* is at 2.17s, respectively. The path length for HPPRM (436) is shorter than all cases. When the number of nodes is increased to $N = 100$, the success rate for the HPPRM is 98%, HPPRM* is 98.3%, PRM is 48.8%, and PRM* is 58.1%. The mean calculation time for HPPRM is 7.72s while that of HPPRM* is 11.94s.

Similar results are reflected in environment C (Figure 23c) and D (Figure 23d) in all the tests. The success rate for the HPPRM when the number of nodes is $N = 100$ in the case of Environment C reaches 100%. In all cases, the mean path length is shorter for HPPRM.

The key observations from the evaluation test are as follows:

1) The success rate in case of HPPRM was better in all the cases when the number of nodes were same. The reason for the higher success rate in the case of HPPRM can be related to the fact that nodes are distributed uniformly in both the maps using the map segmentation method. Since enough nodes are not distributed in narrow regions in the case of PRM and PRM*, the start and goal configuration could not be connected by edges, and therefore the path planning failed. Another reason for the success of HPPRM can be attributed to the distribution of more nodes in complex areas of the map with obstacles. At the same time, a fewer number of nodes are distributed in large open areas, thereby increasing the edge connectivity between the nodes. For e.g. if we calculate the distribution of nodes for $N = 100$ with constant $k = 0.3$, and $(m_x, m_y) = (5, 5)$, the number of nodes in high potential areas are $1.3 \times 100/(5 \times 5) = 5.2 \approx 5$, and in the low potential area are $0.7 \times 100/(5 \times 5) = 2.8 \approx 3$. Thus, we can see that more nodes are distributed in high potential areas or near the obstacles. The third reason is that in low potential areas that are more open or safe, the nodes are distributed closer to each other while in a high potential area, the nodes are further to each other and make it easier to connect the edges. The success rate of HPPRM* is closer to that of the HPPRM.
2) The mean calculation time(s) in the case of HPPRM is slightly higher than PRM when the nodes are the same. However, the mean calculation time required to achieve the success rate is higher in case of HPPRM. The higher mean calculation time (s) for HPPRM compared to PRM is due to the higher cost during the query phase. As discussed earlier in Section IV-B1, the calculation cost largely depends on the local planner used to connect the nodes and edges e.g., Dijkstra or A* method. When the number of nodes are same in PRM and HPPRM, the number of edges E are higher in case of HPPRM, and the node distribution is closer than PRM. As the number of edges E increases, the total cost increases too. This is the main reason for slower performance of HPPRM compared to PRM when the number of nodes is same. The difference in the mean calculation time is not that large. However, if the number of nodes are increased, HPPRM will be slower than the PRM. Similar reason can be given for PRM* as there is not a lot of difference between the two algorithm however PRM* is computationally expensive than the original PRM. This is because in the connection phase of PRM*, the connection radius decreases as the number of sampled states increases. Therefore, the computation cost for HPPRM* is higher in all cases as it produces more edges in the connection phase than the PRM. The success rate for both HPPRM and HPPRM* are high. So if a quicker solution is preferable rather than optimal one HPPRM is the chosen method. Furthermore, we believe that the success rate is more important for planning in complex environments, and that result is higher in case of HPPRM.
3) The length of the path for HPPRM is shorter in all cases. In the case of HPPRM, we could achieve lower mean path length because zig-zag paths are prevented as nodes are more concentrated near the center of the narrow passages. On the other hand, in case of PRM, nodes are dispersed near the walls, which results in longer path length. As straight-line paths are shorted than a zig-zag path, the edges follow a straight-line path. Therefore, the mean path length is better in case of the proposed HPPRM than PRM. In the case of PRM* there is a significant difference in the number of edges produced than the PRM giving shorter connections between nodes that generates a finer path. This has certain advantages in environments with complex geometry.
Figure 25 shows the result of PRM and HPPRM in a new map with many narrow passages and higher obstacle density. The start and goal position is (5, 5) and (95, 95), and the number of nodes is ($N = 100$). It can be seen (blue circled regions) in Fig. 25 that the path generated by HPPRM is centered and away from obstacles as compared to the PRM path, which is closer to obstacles and walls. Moreover, the node distribution in the case of HPPRM is better due to the segmentation of regions based on potential values. The mean path length is also shorter since the generated path in HPPRM has less sharp turns. We also found that with an increased number of nodes for HPPRM, the path almost follows the center of passages implying safety for the robot motion.

F. REAL ENVIRONMENT TEST: HPPRM WITH DYNAMIC OBSTACLES

Path planning is generally considered assuming static obstacles on the map. This means that once the planner generates the path, the robot will follow the path and reach the goal. However, there can be dynamic obstacles in the map that should be avoided by the robot during the planning phase.

TABLE 5. Evaluation result for Figure 24.

Environment		$(N = 70)$ $S_r(\%)$	$t_{\text{mean}}(\text{s})$	d_{mean}	$(N = 100)$ $S_r(\%)$	$t_{\text{mean}}(\text{s})$	d_{mean}
Complex map A (Fig. 23a)	PRM	35.7	0.79	449	65.3	1.55	445
	PRM*	42.7	2.11	456	68.1	3.56	432
	HPPRM	89.3	1.40	416	88.3	2.76	427
	HPPRM*	87.6	0.97	433	89.9	4.81	441
Corridor map B (Fig. 23b)	PRM	18.3	0.94	443	48.8	2.82	447
	PRM*	29.1	2.17	437	58.1	5.03	439
	HPPRM	92.1	2.16	436	98.0	7.72	429
	HPPRM*	90.8	5.50	442	98.3	11.94	436
Narrow map C (Fig. 23c)	PRM	52.4	4.2	154	79.9	24.22	149
	PRM*	56.9	8.8	162	77.2	33.17	159
	HPPRM	97.6	5.3	143	100	25.10	141
	HPPRM*	96.2	13.4	145	99.8	46.29	149
Narrow map D (Fig. 23d)	PRM	37.2	0.83	207	59.1	5.97	200
	PRM*	39.5	4.68	218	63.04	14.36	210
	HPPRM	94.8	0.56	202	98.7	3.97	193
	HPPRM*	95.6	6.75	208	98.2	18.59	200

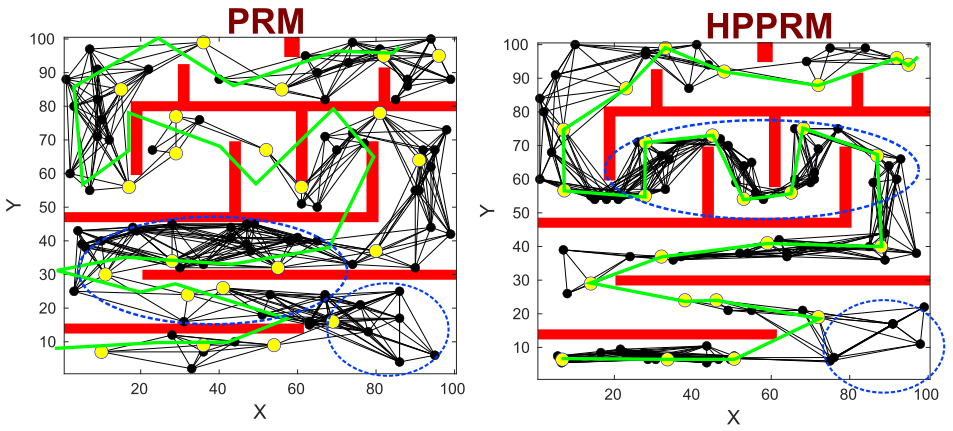


FIGURE 25. Key comparison between PRM and HPPRM in a complex narrow passage map ($N = 100$).

HPPRM can plan a path in the presence of dynamic obstacles. By dynamic, we mean obstacles that were not present before the planning and were later added to the static map. Traditionally the PRM would fail in such a case, as the plan is already built, and there is no additional check for dynamic obstacles added on the map. In such situations, the robot would collide with the obstacles, and the navigation will fail. It is also essential to consider such dynamic elements during the planning since the robot might have to return to the original position after completing the task, and therefore planning based on dynamic obstacles is crucial for safe robot navigation. We considered the dynamic obstacle scenario in the real environment and tested the feasibility of HPPRM to deal with safe planning under dynamic obstacles.

1) SYSTEM SPECIFICATION

A differential drive robot was used for the experiment. The robot used is the Pioneer P3DX from Adept Mobile Robotics Inc. Figure 26b shows the robot model used for the experiment. It is equipped with a 2D SICK LMS111-LRF with a maximum range of 20 m, angular scanning resolution of 270 degrees, and 0.36 degrees/pitch. A 3D vision sensor (Asus Xtion Pro Live) is also used for recording RGB and depth images. All computations were performed on a robot

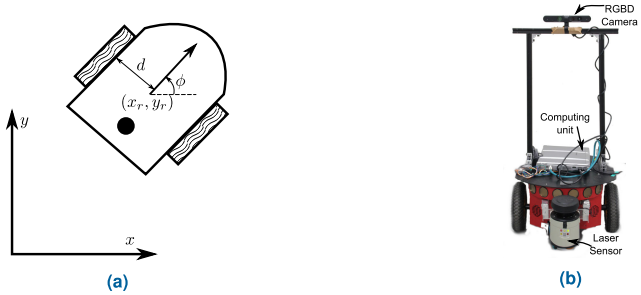


FIGURE 26. (a) Differential kinematics model (b) Pioneer P3DX robot.

PC running 64-bit Ubuntu Linux 16.04 operating system with Intel Core i7 processor and 8 GB of RAM. All programming was done on MATLAB and Kinetic version of the Robot Operating System (ROS), a software middleware for robotics. Prior to testing the navigation and planning, a grid map of the environment was built by utilizing the wheel odometry and LRF scan data.

2) MOTION MODEL OF DIFFERENTIAL DRIVE ROBOT

It is important to consider the motion model of the differential drive since its motion will be different from the simulated omnidirectional point robot that has been considered so far in the simulation tests. We first derive the motion model of the differential Pioneer P3DX robot. The differential drive configuration consists of two independently driven wheels of radius r and one or more low friction castor wheels for balancing the robot in a horizontal plane. Let the distance between the two wheels be $2d$ and let (x_r, y_r) represent the robot coordinate frame halfway between the wheels, as shown in Figure 26a. If u denotes the configuration of the robot kinematic motion model given by $u = (\phi, x_r, y_r, \theta_L, \theta_R)$, where ϕ is the heading angle that the robot body makes with the global coordinate frame (x, y). The parameters, θ_L and θ_R are the rolling angles of the left and the right wheels, respectively. The kinematic equation of the differential drive robot is given by,

$$\dot{u} = \begin{bmatrix} \dot{\phi} \\ \dot{x_r} \\ \dot{y_r} \\ \dot{\theta_L} \\ \dot{\theta_R} \end{bmatrix} = \begin{bmatrix} -r/2d & r/2d \\ \frac{r}{2}\cos\phi & \frac{r}{2}\cos\phi \\ \frac{r}{2}\sin\phi & \frac{r}{2}\sin\phi \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_L \\ v_R \end{bmatrix}, \tag{19}$$

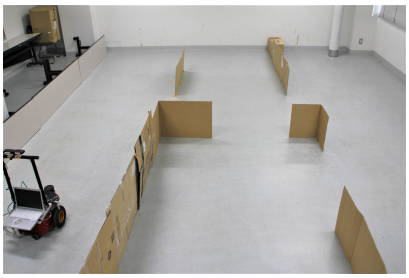
where, v_L and v_R is the angular speed of the left and right wheel, respectively and the positive angular speed corresponds to the forward motion of the wheel. The control value for each wheel lies in the interval $[-v_{\max}, v_{\max}]$. The simplified control scheme for the differential drive system ignoring the rolling angles of the two wheels become,

$$\dot{u} = \begin{bmatrix} \dot{\phi} \\ \dot{x_r} \\ \dot{y_r} \end{bmatrix} = \begin{bmatrix} -r/2d & r/2d \\ \frac{r}{2}\cos\phi & \frac{r}{2}\cos\phi \\ \frac{r}{2}\sin\phi & \frac{r}{2}\sin\phi \end{bmatrix} \begin{bmatrix} v_L \\ v_R \end{bmatrix}. \tag{20}$$

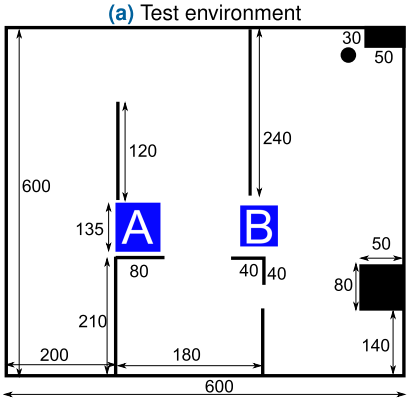
3) HPPRM-DYNAMIC ENVIRONMENT

In this section, we present the HPPRM performance in a dynamic environment. In a dynamic environment, HPPRM uses the combined local reactive and global navigation layered planner to overcome the obstacles in the map and reach the goal position. The environment setup for the real test is as shown in Figure 27a. The dimensions of the map are given in Figure 27b. The process of HPPRM with dynamic obstacles is as follows:

1) The potential map and the PRM path is generated for the given map with a start and goal positions.



(a) Test environment



(b) Map dimensions (cm)

FIGURE 27. Dynamic obstacle test environment.

2) The robot initiates the plan and follows the path along the generated planner trajectory.
3) When the robot is about to take the first turn at point 'A', a new dynamic obstacle is added to the map (shown as blue box, Figure 27b). The obstacle position in the map is sensed using the 2D LRF.
4) The potential map with a new obstacle is updated, and the robot checks for the accessibility to the next local goal using the node information available. It searches for the nearest nodes in the vicinity to continue using the planned path. If the robot is able to access the next local goal, it checks for the passability of the robot to pass around the obstacle. If the path is not passable, it replans the path since it is difficult to find a new path in the direction of the planned path anymore.
5) The robot follows the new path, and once again at point 'B' in the map, another dynamic obstacle is added which the robot senses using the LRF sensor.
6) Step 4 is repeated, and a new plan is generated considering the passability test.
7) Robot reaches the goal position following the new path.

The initial motion planning policy generated by the PRM with no dynamic obstacle information is shown in Figure 29a. The number of nodes, start and goal positions are $N = 120$, start $= (50, 50)$, and goal $= (250, 280)$ respectively. The green trajectory represent the generated path for PRM. The updated PRM path with the potential map and dynamic obstacles is shown in Figure 29b, and the new path generated on the PRM nodes is shown in Figure 29c on the grid map of the test

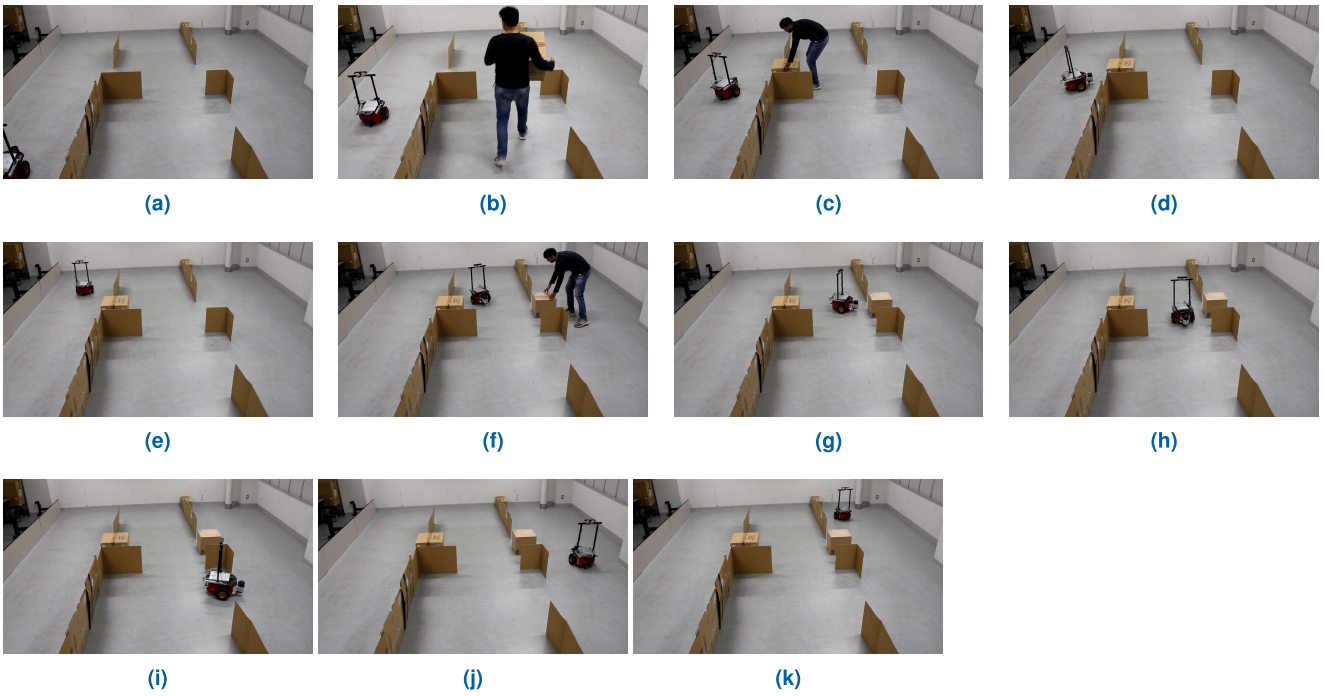


FIGURE 28. Sequence of events for the dynamic obstacle test, (a) Robot starts from initial position (50, 50), (b)-(c) First dynamic obstacle (A) added at the first open space, (d)-(e) Robot takes the new path towards goal, (f) Second dynamic obstacle (B) added to the map, (g)-(k) Robot takes the new path to goal.

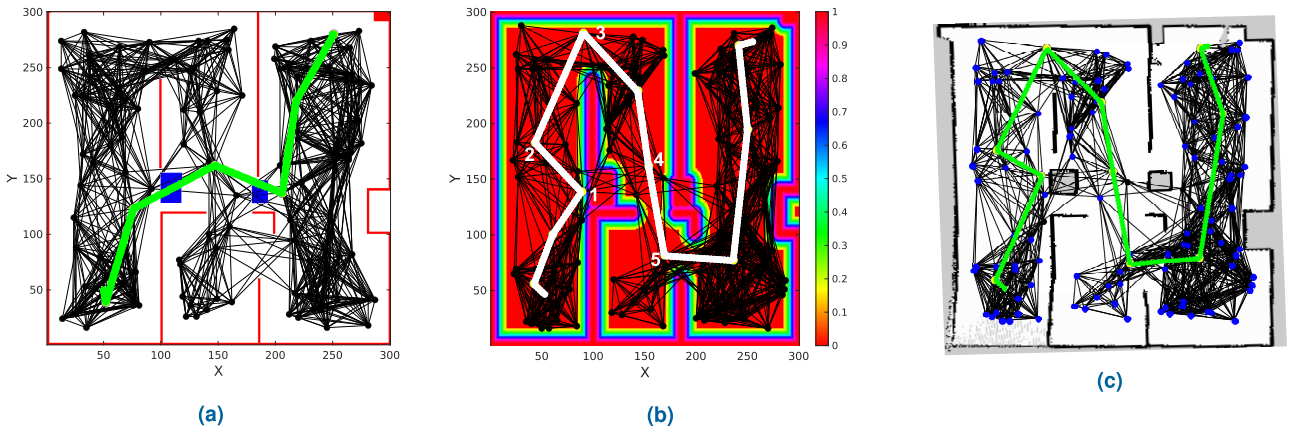


FIGURE 29. (a) PRM result without considering dynamic obstacles. (b) HPPRM with dynamic obstacle. (c) Actual plan on the grid map of the test environment.

environment. The sequence of adding the dynamic obstacles to the map during the robot run is shown in Figure 28.

For the original PRM, once the roadmap is constructed and the path is generated, the plan is set for the robot. The planner will not consider any new obstacles added to the map, and the robot will collide with the obstacle at position 'A' as shown in Figure 29c. The HPPRM uses mixed reactive navigation combining global and local planners. Therefore, when a new obstacle is detected on the global path, the planner switches to the local reactive planner and replans its path to avoid the collision with the obstacle. In Figure 29b, when the robot reaches the position (95,130) at point '1' from the start position it senses an obstacle in the path by scanning the environment using the LRF sensor. The robot decides its behavior based on the local sensor information gathered during the local motion planning. The potential value measured at this position is shown in Figure 30a. The direction towards the goal of checkpoint '4' is 18.2 degrees. Considering the parameters for the robot width, the passability test fails to directly approach the checkpoint '4' around the obstacle. The only open space that the robot can take is at 90 degrees to 230 degrees. Therefore, the robot moves to the direction of open space at 180 degrees and replans the motion to point '2'. From checkpoint '2' to '4', the motion is a straight-line with an

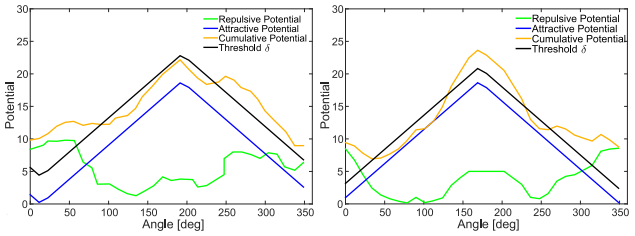


FIGURE 30. Potential values calculated at (a) checkpoint 1, and (b) checkpoint 4 as shown in Figure 29b.

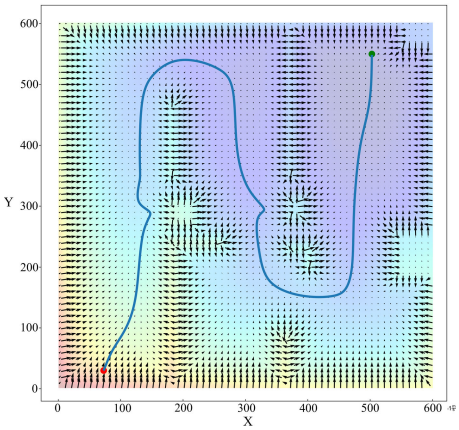


FIGURE 31. Force distribution around the obstacles and robot trajectory shown in blue in dynamic obstacle environment test.

intermediate node in between. The robot follows the renewed path until it reaches checkpoint '4'. At this point (168,162), a new obstacle is detected, and the direction towards the goal is at 253.8 degrees. The cumulative potential values are shown in Figure 30b. Based on the sensing data, there is a single opening space between 40 degrees to 130 degrees and this region also has a lower potential which is closer to the goal; therefore the robot moves in the direction of 96 degrees towards checkpoint '5' based on the potential gradient plotted at point '4'. A new route towards the goal is calculated, and the robot continues towards the goal. Potential force vectors around the obstacles are presented in Figure 31. The size of the arrows represent the magnitude of force generated by the obstacle. The blue color represents the trajectory taken by the robot during the dynamic obstacle experiment. The colors represents the total potential from the start to the goal position. The robot trajectory mostly follows the low potential areas as indicated by force vectors in the figure. Since the new map is updated with the dynamic obstacle positions, if the goal and start positions are reversed, the robot can easily come back to the original position following the HPPRM map.

VI. CONCLUSION

In this paper, we present an improved sampling based path planning method for mobile robots in complex environment using the artificial potential method. A new sampling strategy that combines probabilistic roadmap (PRM) and the artificial potential field for node distribution in narrow spaces is presented. By segmenting regions of the map into areas of high and low potential the node distribution of the PRM is improved significantly. A layered path planner combining reactive local and global planners that allows the robot to plan the trajectories on the map in presence of dynamic obstacles without suffering from the local minima problem is presented. A virtual repulsive force is introduced for avoiding the local minima trap problem when navigating in areas with static and dynamic obstacles. A comparative analysis of the proposed method with state of the art methods both in real and simulation environments is performed for speed and path length. From the results, we can say that the proposed HPPRM is superior to the original PRM in terms of shorter path length and success rate. A success rate of over 95% is achieved for both local and global planning. It performs better than PRM in narrow passages and generates paths that are safer and shorter by utilizing node dispersion using the potential field values. In addition, navigation method was tested for local and global planning using simulation and real experiments, and in all the cases the proposed method produced higher success rate and better plans. In future, we plan to expand our algorithm for multi-robot scenarios with formations of different number of robots and geometrical shapes. Future work also includes expanding the proposed algorithm for outdoor environments.

REFERENCES

[1] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. Cambridge, MA, USA: MIT Press, 2005.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. Cambridge, MA, USA: MIT Press, 2005.

[3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009.

[4] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Berlin, Germany: Springer-Verlag, 2016.

[5] O. Takahashi and R. J. Schilling, "Motion planning in a plane using generalized Voronoi diagrams," *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 143–150, Apr. 1989.

[6] M. R. H. Al-Dahhan and K. W. Schmidt, "Voronoi boundary visibility for efficient path planning," *IEEE Access*, vol. 8, pp. 134764–134781, 2020.

[7] T. Maekawa, T. Noda, S. Tamura, T. Ozaki, and K.-I. Machida, "Curvature continuous path generation for autonomous vehicle using B-spline curves," *Comput.-Aided Des.*, vol. 42, no. 4, pp. 350–359, Apr. 2010.

[8] T. Asano, T. Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility-polygon search and Euclidean shortest paths," in *Proc. 26th Annu. Symp. Found. Comput. Sci. (SFCS)*, Oct. 1985, pp. 155–164.

[9] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, no. 5, pp. 1179–1187, Sep. 1989.

[10] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE Trans. Robot. Autom.*, vol. 7, no. 3, pp. 278–288, Jun. 1991.

[11] I. Ulrich and J. Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification," in *Proc. ICRA. Millennium Conf. IEEE Int. Conf. Robot. Automat., Symposia*, Apr. 2000, pp. 2505–2511.

[12] U. Orozco-Rosas, K. Picos, and O. Montiel, "Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots," *IEEE Access*, vol. 7, pp. 156787–156803, 2019.

[13] A. Ravankar, A. A. Ravankar, Y. Kobayashi, and T. Emaru, "SHP: Smooth hypocycloidal paths with collision-free and decoupled multi-robot path planning," *Int. J. Adv. Robotic Syst.*, vol. 13, no. 3, p. 133, Jun. 2016.

[14] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1986, pp. 396–404.
[15] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, no. 2, pp. 224–241, Mar. 1992.
[16] E. Rimon and D. E. Koditschek, "The construction of analytic diffeomorphisms for exact robot navigation on star worlds," *Trans. Amer. Math. Soc.*, vol. 327, no. 1, pp. 71–116, Jan. 1991.
[17] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 5, pp. 501–518, Oct. 1992.
[18] D. E. Koditschek and E. Rimon, "Robot navigation functions on manifolds with boundary," *Adv. Appl. Math.*, vol. 11, p. 412, 1990.
[19] D. E. Koditschek, "Applications of natural motion control," *ASME J. Dyn. Syst., Meas. Control*, vol. 113, no. 4, pp. 552–557.
[20] P. Shi and Y. Zhao, "An efficient path planning algorithm for mobile robot using improved potential field," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2009, pp. 1704–1708.
[21] J. Sfeir, M. Saad, and H. Saliah-Hassane, "An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment," in *Proc. IEEE Int. Symp. Robotic Sensors Environ. (ROSE)*, Sep. 2011, pp. 208–213.
[22] R. Raja, A. Dutta, and K. S. Venkatesh, "New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover," *Robot. Auto. Syst.*, vol. 72, pp. 295–306, Oct. 2015.
[23] M. A. K. Jaradat, M. H. Garibeh, and E. A. Feilat, "Autonomous mobile robot dynamic motion planning using hybrid fuzzy potential field," *Soft Comput.*, vol. 16, no. 1, pp. 153–164, Jan. 2012.
[24] O. Cetin, I. Zagli, and G. Yilmaz, "Establishing obstacle and collision free communication relay for UAVs with artificial potential fields," *J. Intell. Robotic Syst.*, vol. 69, nos. 1–4, pp. 361–372, Jan. 2013.
[25] T. Weerakoon, K. Ishii, and A. A. F. Nassiraei, "An artificial potential field based mobile robot navigation method to prevent from deadlock," *J. Artif. Intell. Soft Comput. Res.*, vol. 5, no. 3, pp. 189–203, Jul. 2015.
[26] Y. Yongjie and Z. Yan, "Collision avoidance planning in multi-robot based on improved artificial potential field and rules," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Feb. 2009, pp. 1026–1031.
[27] J. Zhang, J. Yan, and P. Zhang, "Fixed-wing UAV formation control design with collision avoidance based on an improved artificial potential field," *IEEE Access*, vol. 6, pp. 78342–78351, 2018.
[28] P. T. Zhang, Y. Zhu, and J. Song, "Real-time motion planning for mobile robots by means of artificial potential field method in unknown environment," *Ind. Robot, Int. J.*, vol. 37, no. 4, pp. 384–400, 2010.
[29] I. Ulrich and J. Borenstein, "VFH+: Reliable obstacle avoidance for fast mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1998, pp. 1572–1577.
[30] A. Ravankar, A. A. Ravankar, M. Watanabe, Y. Hoshino, and A. Rawankar, "Multi-robot path planning for smart access of distributed charging points in map," *Artif. Life Robot.*, pp. 1–9, Jul. 2020, doi: 10.1007/s10015-020-00612-8.
[31] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Auto. Robots*, vol. 13, no. 3, pp. 207–222, 2002.
[32] A. Tuncer and M. Yildirim, "Dynamic path planning of mobile robots with improved genetic algorithm," *Comput. Electr. Eng.*, vol. 38, no. 6, pp. 1564–1572, Nov. 2012.
[33] B. B. K. Ayawli, X. Mei, M. Shen, A. Y. Appiah, and F. Kyeremeh, "Mobile robot path planning in dynamic environment using Voronoi diagram and computation geometry technique," *IEEE Access*, vol. 7, pp. 86026–86040, 2019.
[34] A. Ravankar, A. Ravankar, Y. Hoshino, and Y. Kobayashi, "Virtual obstacles for safe mobile robot navigation," in *Proc. 8th Int. Congr. Adv. Appl. Informat. (IIAI-AAI)*, Jul. 2019, pp. 552–555.
[35] A. Ravankar, A. Ravankar, Y. Kobayashi, and T. Emaru, "Symbiotic navigation in multi-robot systems with remote obstacle knowledge sharing," *Sensors*, vol. 17, no. 7, p. 1581, Jul. 2017.
[36] A. Ravankar, A. Ravankar, Y. Kobayashi, and T. Emaru, "Hitchhiking robots: A collaborative approach for efficient multi-robot navigation in indoor environments," *Sensors*, vol. 17, no. 8, p. 1878, Aug. 2017.
[37] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.
[38] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.
[39] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.
[40] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic Comput. Robot., New Directions*, vol. 5, pp. 293–308, Jul. 2000.
[41] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. H. Reif, "Narrow passage sampling for probabilistic roadmap planning," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1105–1115, Dec. 2005.
[42] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1999, pp. 1018–1023.
[43] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.
[44] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, Jun. 2011.
[45] W. Xinyu, L. Xiaojuan, G. Yong, S. Jiadong, and W. Rui, "Bidirectional potential guided RRT* for motion planning," *IEEE Access*, vol. 7, pp. 95046–95057, 2019.
[46] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proc. ICRA. Millennium Conf., IEEE Int. Conf. Robot. Automation. Symposia*, Apr. 2000, pp. 521–528.
[47] M. Morales, S. Rodriguez, and N. M. Amato, "Improving the connectivity of PRM roadmaps," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 2003, pp. 4427–4432.
[48] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.
[49] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.
[50] J.-C. Latombe, *Robot Motion Planning*, vol. 124. New York, NY, USA: Springer, 2012.
[51] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[52] A. Ravankar, A. Ravankar, Y. Kobayashi, Y. Hoshino, and C.-C. Peng, "Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges," *Sensors*, vol. 18, no. 9, p. 3170, Sep. 2018.
[53] A. Ravankar, A. Ravankar, A. Rawankar, Y. Hoshino, and Y. Kobayashi, "ITC: Infused tangential curves for smooth 2D and 3D navigation of mobile robots," *Sensors*, vol. 19, no. 20, p. 4384, Oct. 2019.
[54] P. Vadakkepat, K. Chen Tan, and W. Ming-Liang, "Evolutionary artificial potential fields and their application in real time robot path planning," in *Proc. Congr. Evol. Comput., CEC*, Jul. 2000, pp. 256–263.
[55] G. Li, A. Yamashita, H. Asama, and Y. Tamura, "An efficient improved artificial potential field based regression search method for robot path planning," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Aug. 2012, pp. 1227–1232.
[56] D. H. Kim and S. Shin, "Local path planning using a new artificial potential function composition and its analytical design guidelines," *Adv. Robot.*, vol. 20, no. 1, pp. 115–135, Jan. 2006.
[57] R. R. Murphy, *Introduction to AI Robotics*. Cambridge, MA, USA: MIT Press, 2019.
[58] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 1991, pp. 1398–1404.
[59] S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Trans. Robot. Autom.*, vol. 16, no. 5, pp. 615–620, Oct. 2000.
[60] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, Jul. 1998.
[61] K. Fujimura, *Motion Planning in Dynamic Environments*. Tokyo, Japan: Springer, 2012.

ANKIT A. RAVANKAR (Member, IEEE) received the bachelor's degree in production engineering from the University of Pune, India, in 2009, and the M.S. and Ph.D. degrees in robotics engineering from Hokkaido University, Sapporo, Japan, in 2012 and 2015, respectively. In 2016, he joined Hokkaido University, where he is currently an Assistant Professor with the Department of Human Mechanical Systems and Design Engineering and works with the Research Laboratory of Robotics and Dynamics. His research interests include mobile robot navigation, SLAM, multi-robot systems, computer vision, machine learning applications to robotics, and aerial vehicles. He was a recipient of the MEXT Scholarship (Mombukagakusho) by the Ministry of Education, Culture, Sports and Technology, Japan, for his graduate studies, from 2010 to 2015.

ABHIJEET RAVANKAR (Member, IEEE) received the M.S. degree in computer science and engineering from the University of Aizu, Japan, in 2011, and the Ph.D. degree in robotics engineering from Hokkaido University, in 2017. He was a Postdoctoral Fellow of Hokkaido University working on navigation of multi-robots and self-driving cars. He worked with the Research and Development Section, Panasonic Corporation, Osaka, Japan, as an Engineer working on computer vision, optimization, and big data problems. He is currently working as an Assistant Professor with the Department of Mechanical Engineering, Kitami Institute of Technology, Japan. His research interests include robot navigation, multi-robot systems, computer vision, deep learning, and artificial intelligence. He was a recipient of the MEXT Scholarship (Mombukagakusho) by the Ministry of Education, Culture, Sports and Technology, Japan for his graduate studies.

TAKANORI EMARU (Member, IEEE) received the M.E. and Ph.D. degrees in electrical engineering from Hokkaido University, Japan, in 1998 and 2002, respectively. He was a Research Fellow of the Japan Society for the Promotion of Science, The University of Electro-Communications, Japan, from 2004 to 2006. He was also an Assistant Professor with Osaka Electro-Communication University from 2006 to 2007. He is currently an Associate Professor with Hokkaido University. His research interests include robotics, navigation, sensor, and nonlinear signal processing.

YUKINORI KOBAYASHI (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in mechanical engineering from Hokkaido University, Sapporo, Japan, in 1981, 1983, and 1986, respectively. He is currently the President with the National Institute of Technology, Tomakomai College, Japan, and an Emeritus Professor with Hokkaido University. His research interests include vibration control of flexible structures, control problem of robots having flexibility, path planning and navigation of mobile robots, vibration analysis, and nonlinear vibrations of continuous systems.