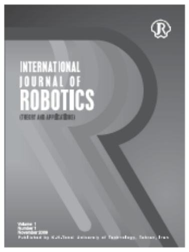

INTERNATIONAL JOURNAL OF ROBOTICS
(THEORY AND APPLICATIONS)
E-mail: IJR@kntu.ac.ir

A Navigation System for Autonomous Robot Operating in Unknown and Dynamic Environment: Escaping Algorithm

F. AdibYaghmaie [a], A. Mobarhani [b], H. D. Taghirad [c,*]

a. Member IEEE, Faculty of Electrical Engineering, K. N. Toosi Univ. of Technology, Tehran, Iran
b. Member IEEE, Faculty of Electrical Engineering, K. N. Toosi Univ. of Technology, Tehran, Iran
c. Senior Member, IEEE, Faculty of Electrical Engineering, K. N. Toosi Univ. of Technology, Tehran, Iran

ARTICLE INFO

Article history:

Received: June 20, 2015.
Received in revised form:
October 19, 2015.
Accepted: October 23, 2015.

Keywords:

Index Terms
Dynamic Environment
Mobile Robot Navigation
Potential Filed

ABSTRACT

Abstract—In this study, the problem of navigation in dynamic and unknown environment is investigated and a navigation method based on force field approach is suggested. It is assumed that the robot performs navigation in unknown environment and builds the map through SLAM procedure. Since the moving objects' location and properties are unknown, they are identified and tracked by Kalman filter. Kalman observer provides important information about next paths of moving objects which are employed in finding collision point and time in future. In the time of collision detection, a modifying force is added to repulsive and attractive forces corresponding to the static environment and leads the robot to avoid collision. Moreover, a safe turning angle is defined to assure safe navigation of the robot. The performance of proposed method, named Escaping Algorithm, is verified through different simulation and experimental tests. Besides, comparison between Escaping Algorithm and Probabilistic Velocity Obstacle (PVO), based on computational complexity and required steps for finishing the mission is provided in this paper. The results show Escaping Algorithm outperforms PVO in term of dynamic obstacle avoidance and complexity as a practical method for autonomous navigation.

1. Introduction

2MOBILE robots find their path to the daily life of humans by the means of navigation. This general concept transforms a mobile robot from a "toy" to an autonomous system able to operate in unknown environment. The motivation of developing navigation systems lies in two important aspects: assistive robotics and industrial applications. Examples of autonomous systems for assistive purposes are automated wheelchairs, robots able to find and manipulate objects, searching and rescue after catastrophe [1] and performing dangerous tasks instead of human like searching mines and monitoring the Fukushima nuclear site [2]. Such applications need both the ability to move in crowded environment and optimal path planning. Besides, industry benefits from the concept of navigation in different ways. The usage of autonomous systems decreases the expenses of health insurance (due to usage of robots instead of human) and raises the production rate due to operation with higher speed.

There is a rich literature on the concept of navigation. Different navigation approaches are suggested by researchers for static or dynamic environment. The complete literature review on the navigation problem is mentioned in Section 2. A navigation scheme which

* Corresponding author
Tel.: +982182883358, E-mail address: taghirad@kntu.ac.ir

supposes complete knowledge of environment and dynamic obstacles will not provide a realistic framework for practical applications. This issue attracts researchers concern for further extension of navigation systems. Navigation in dynamic environment is a challenging topic from two aspects:

1- Dynamic obstacles in the environment have unknown future motion.
2- The robot is equipped with sensors with limited range of view. The robot does not have complete knowledge about environment.

In this paper, these two challenges will be addressed. The main purpose of current study is suggesting a method for navigation in dynamic environment without considering any prior information about environment and dynamic obstacles. The first challenging topic will be addressed by categorizing observations to static and dynamic and tracking dynamic ones by the means of Kalman filter. The main contribution of this paper lies in solving the second issue and that is using Escaping Algorithm (EA) for navigation. In the mentioned method which is an extension of force field approach, the aim is leading the robot to the target while trying to avoid local obstacles.

In our problem it is assumed that the environment is unknown to the robot. Hence the robot performs SLAM and navigates toward the target simultaneously. In each iteration of our routine, the robot incrementally completes its local map by performing SLAM and uses the local map to distinguish between dynamic and static obstacles. The robot predicts the path of dynamic obstacles and based on it, the modifying force is applied to the robot. The resultant total force leads the robot toward the target while avoids the local obstacles. The robot moves for a while (i.e. duration of the loop) and then the loop starts again by performing SLAM. This procedure continues until the robot reaches the target.

This paper which is an extension to our previous work [35] is organized as follows: In section 2, a complete literature review of the navigation systems is presented. Section 3 provides environment representation and formulates the problem. The definitions of attractive and repulsive force in static environment are mentioned in this part. In section 4, escaping algorithm and its components are enlightened and the required steps to obtain repulsive force of dynamic obstacles are discussed. Section 5 contains simulation and experimental results of EA. Besides, EA is compared to PVO and the complexity analysis of both is included in this section. Finally, concluding remarks are given in section 6.

2. Related Works

Navigation systems have been extensively studied before. The problem of navigation in static environment is solved and saturated with different approaches like A* [3], [4] and force field method [5]. However, navigation in dynamic environment still attracts researchers' attention and motivates them for better and further developments.

Some navigation systems divide the navigation problem into two parts: the first part is path planning from the start point to the target and the second part is designing a controller for tracking the path. Modified A* [6] and D* [7] are examples of these groups which search the cost map to generate the path. These methods usually suppose complete knowledge about environment or consider unknown locations as free and plan the path toward the goal. The expensive computation cost of these methods and the need of re-planning due to the change in the environment, limit their usage as an applicable navigation method.

Another part of research involves navigation systems which solve for optimal or near optimal trajectory. Even though these navigators are fast, they usually plan locally and it is possible to trap in local minima. In [8], a horizon limited trajectory is produced by minimizing a utility function and considering obstacles with very low velocities. The idea of dynamic window is introduced in [9] and tries to find the control inputs by maximizing the cost function which contains the robot heading, distance to obstacles and the robot velocity.

Rapidly exploring Random Trees (RRTs) method is a tool able to search high dimensional input space and consider vehicle dynamic. The method is suitable for searching complex environment [10]. In [11], partial path planning is done based on RRT that considers vehicle model constraints such as acceleration, steering velocity, and steering angle bounds and the real-time operation.

A set of techniques named Velocity Obstacle (VO) exist that compute safe velocities for the robot based on obstacle velocities and the selected time horizon [12]. The main problem of this set is that they assume a complete knowledge about moving objects like their velocities. Recently, some navigation algorithms based on VO developed that consider the avoidance possibilities of workspace objects like [13] and [14]. These methods may trap in local minima and fail in some situations.

The concept of Inevitable Collision State (ICS) introduced by [15] guarantees the motion safety criteria which is listed in [16]. The output of mentioned approach is a set of states for the mobile robot which leads to a collision and then, they are dangerous and should be avoided. This approach requires a complete environment model which is not realistic. In [17], the Probabilistic Inevitable Collision State (PICS) which is applicable for probabilistic settings is suggested. One of disadvantages of this method is that there is no reliable long-term motion prediction for humans.

Potential field is one of the most common methods for path planning. The popularity of this method is due to its simplicity for implementation. This method was introduced by [5] and then improved for real time implementations in [18]. First, potential field approach was suggested for navigation in static environment [19], [20], [21], [22]. However, the real world is not stationary, and the robot moves in dynamic environment and encounters dynamic obstacles like moving humans. Researchers started to develop potential field methods for navigation in dynamic environment. In [23], the velocity of dynamic obstacles is included in the definition of potential function. The basic problem is that the collision depends on the velocity of both robot and obstacle; however, [23], considers only the speed of the robot. In [24] relative positions and velocities of the mobile robot with respect to the obstacles are considered in definition of potential function. However, this method needs exact knowledge of velocity of dynamic objects, which is not available in practice. Potential function to reach a moving target is defined in [25], but the velocities of the robot, obstacles and target are assumed to be known.

The contribution of this paper is an extension of potential field approach for navigation in dynamic environment without considering any prior information about environment and moving objects. The proposed method provides safe motion for the robot operating in dynamic environment. This paper is an extension to our previous work [35] and describes more details about Escaping Algorithm. Besides, the current study includes comparison to another navigation system (PVO).

2.1. The Environment Modeling and Problem Formulation

2.1.1. The Environment Modeling

In this paper, the popular occupancy grid map is used for environment modeling [26]. The occupancy grid map uses a matrix to represent obstacles. Each entity in the matrix is a symbol of one square part of environment and its quantity shows the confidence of the obstacle lying at this location. In this context, the size of square sides is set to 10 cm. Certainty values range from $-\infty$ to $+\infty$ in occupancy grid maps. As the possibility of existence of an obstacle in specific square increases, the certainty values goes to $+\infty$; while for a free cell, this value approaches to $-\infty$. The robot is equipped by one laser range finder and two encoders. Since laser is used to obtain information from environment, this map is suitably adapted to our system. In each range reading, the values of certainties are updated and used for navigation purposes.

2.1.2. Problem Formulation

The motion planning problem of a mobile robot is to plan and control it such that it reaches the target while avoiding obstacles. In Force Field Method (FFM) the obstacles exert repulsive force to the robot while the target attracts the robot to itself. The total force determines the direction of movement for the robot. The definitions of attractive and repulsive forces are not unique, and different definitions can be found in [27] and [25].

To use FFM for the mobile robot navigation, it is enough to suitably define the repulsive force of static objects, the attractive force of the target and the repulsive force of the dynamic objects and use their total direction for mobile robot steering. The last component will be explained in next section with more details.

I. The repulsive force of static objects

Since gird map is used for environment representation, the repulsive forces of static objects are defined for each occupied cell in grid map. To accomplish this task, it is needed to define a window on the robot coordinates and calculate the repulsive forces through it. This window is called active window and is used to avoid local obstacles while moving toward the target. For each cell in the active window, the repulsive force of static cell is calculated as follows.

$$f_r(i,j) = -\frac{F_{cr}C(i,j)}{d_{ij}^2}\left[\frac{x_{i,j}-x_r}{d_{ij}}\hat{x} + \frac{y_{i,j}-y_r}{d_{ij}}\hat{y}\right] \quad (1)$$

in which, F_{cr} is the repulsive force constant, $C(i,j)$ denotes the certainty of cell (i,j), d_{ij} denotes the Euclidean distance between the robot and cell (i,j), $(x_{i,j}, y_{i,j})$ is the position of cell (i,j), and (x_r, y_r) is the position of the robot. Total repulsive force is calculated by adding all repulsive forces in active window:

$$F_r = \sum_{i,j \in ActiveWin} f_r(i,j) \quad (2)$$

II. The attractive force of the target

The robot moves toward the target while avoiding obstacle. As a result, it is not important that whether the target is located in the active window or not, it always exerts its attractive force to the robot by the following relation:

$$F_a = F_{ca}\left[\frac{x_t - x_r}{d_t}\hat{x} + \frac{y_t - y_r}{d_t}\hat{y}\right] \quad (3)$$

In which, F_{ca} denotes attractive force constant, (x_t, y_t) is the target position, and d_t denotes the Euclidean distance between the robot and the target.

III. The total force

The total force is determined as the sum of the repulsive and attractive forces. The total force in static environment is derived from following equation:

$$F_t = F_a + F_r \quad (4)$$

The above force is used for avoiding local static objects while moving toward the target. If a collision is predicted, the repulsive force of dynamic object is added to the total force and the resultant force is used for navigation. Definition of repulsive force of dynamic objects denoted by F_m and the required steps are explained in details in the next section.

$$F_f = F_t + F_m \quad (5)$$

The direction of F_f is used as steering rate command. In static environment, F_m is equal to zero; in the other words F_f and F_t are the same. Let δ shows direction of F_f. If robot direction is denoted by θ, angular velocity can be given by:

$$\omega = k_s(\delta - \theta) \quad (6)$$

In which, k_s is the steering constant whose dimension is s^{-1}. This constant is set as the inverse of the sampling time.

3. Escaping Algorithm: A Strategy for Navigation in Dynamic Environment

In static environment, the mobile robots can reach the target by using repulsive forces of static objects and attractive force of the target. However, it needs to perform four steps sequentially, to move in dynamic environment safely. These steps are as follows: moving objects detection, motion prediction, collision detection and velocity planning for obstacle avoidance. These sub programs are executed within the Simultaneous Localization and Mapping (SLAM) loop and use the grid map obtained from SLAM in their calculations. Each sub program is explained in the following sections with more details.

3.1. Moving Object Detection

Moving object detection is one of the most important parts of planning in dynamic environment. The objective is to classify observations as static or dynamic. Researchers developed several methods for this classification. One common method is Expectation Maximization Algorithm (EMA) [28], [29]. EMA is a two steps maximization process which solves incomplete data optimization problem [30]. Another method is sample-based variant of probabilistic data association filter. This method filters dynamic observations like human and results robust scan matching [31]. Besides, non-probabilistic methods are also developed. For example, [32] suggested a simple rule for classification. This method is extended for grid map and is used for dynamic observation mapping in this paper.

In this paper a three-state map is generated and used for dynamic object detection. The three-state map has similar structure to grid map and represents environment by set of cells. Each cell in this map can be labeled as free, occupied or unknown. A reading is associated to dynamic object if it locates in a free cell. The three-state map is generated by following formula in each SLAM loop:

$$c_d(i,j) = \begin{cases} \text{free}, & if\ c(i,j) < c_{\min}, c_{\min} < 0 \\ \text{Occupied}, & if\ c(i,j) > c_{\max}, c_{\max} > 0 \\ \text{Unknown}, & otherwise \end{cases} \quad (7)$$

In the above relation, $c_d(i,j)$ shows the cell (i,j) in three-state map. The values of $c_{\min}$ and $c_{\max}$ are tuned practically. Using three-state map, observations are divided to dynamic and static. Static observations are used in gird map, while dynamic observations are predicted and special strategy (EA) is used to avoid them. Please note that several sequential dynamic observations refer to one moving object because moving objects like human reflects several beams of laser to the robot. As a result, it is required to group sequential dynamic readings. The center of each group is used as pose of dynamic object and the distance between center and dynamic border is considered as obstacle radii, R_{obs}.

3.2. Motion Prediction

In this context, a set of Kalman filters is defined. Each Kalman filter predicts next poses and velocities of one moving object. The state vector of each Kalman filter is defined as $X = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T$. The initial guess of state vector is set to $X = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^T$ for all moving objects. As it can be seen, the constant velocity model is used to represent moving object movement. It is important to note that movement of moving obstacles especially human is unpredictable; however, the constant velocity model with noisy acceleration may be suitably used to predict this behavior [28]. The discrete space state equation is shown as follows:

$$X_k = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x_{k-1} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} n_{w1} \\ n_{w2} \end{bmatrix} \tag{8}$$

In which T, denotes the sampling time. Dynamic readings which obtained from previous routine are considered as new positions of moving objects. Therefore, the observation equation is as follows.

$$Z_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x_k + \begin{bmatrix} n_{v1} \\ n_{v2} \end{bmatrix} \tag{9}$$

In Eq. (8) and Eq. (9), n_w and n_v are process and observation noises. The variance of process noises have to be carefully tuned in practice to provide desire performance in tracking. The variance of observation noises is related to sensor properties and should be determined based on them. In each sensor range readings, observations are classified into static and dynamic. Dynamic observations are used for updating Kalman filters. Since there are several dynamic observations and moving object, nearest neighbor algorithm is used to match dynamic observations and obstacles. If there is not any observation for one moving obstacle, our algorithm only performs the prediction step. If this happens several times, it means that the obstacle moves out of the robot vision and it is not necessary to predict its motion anymore. As a result, the corresponding filter is eliminated.

A Kalman filter is also used for the robot position and velocity prediction. The prediction of this filter is used for collision detection between the robot and the obstacles. Therefore, in each SLAM loop, the following set of Kalman filters is updated.

$$\{(X_k, Z_k)\}_i, \quad i = 1, \dots, N+1 \tag{10}$$

In this equation, N denotes the number of predictable moving objects, while the last Kalman filter is for the estimation of robot pose in future. By this means, N may differ in each iteration, according to the number of visible dynamic obstacles.

3.3. Collision Detection

The set of Kalman filters and their predictions are used to find possible collisions. To perform that, each Kalman filter predicts the next poses and velocities of one dynamic obstacle up to max predictions time ($T_{\max}$). Similar prediction is done for the robot. Moreover, suppose that dynamic obstacles and the robot can be modeled by circles with radius R_{obs} and R_{rob} respectively. In k th prediction step the distance between the robot pose and the obstacle is calculated. If this distance, denoted by d, is less than the summation of obstacle and robot radii, then the collision will probably happen. The distance is calculated between the robot and all moving obstacles in order to find all possible collisions. It is important to note that as prediction step increases, the uncertainty grows, as well. As a result, a confidence factor, denoted by λ_{conf}, is defined and used in collision detection. The following condition holds if a collision is possibly happening:

$$d_k < \lambda_{conf} \times (R_{rob} + R_{obs}), \quad \lambda_{conf} > 1 \tag{11}$$

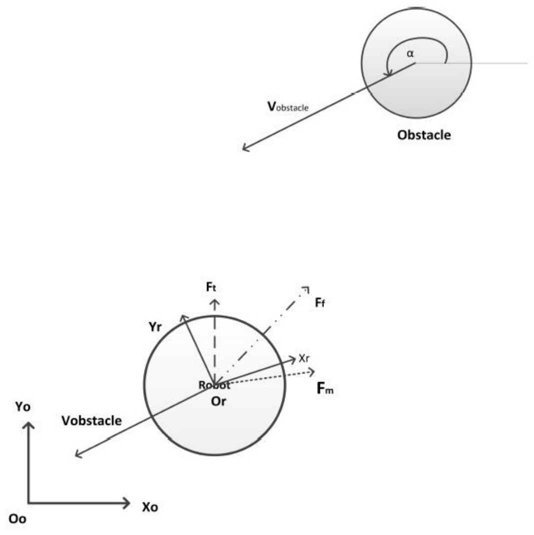

Figure 1- Escaping Algorithm

3.4. Velocity Planning for Obstacle Avoidance

Navigation in dynamic environment using potential field method is widely studied in literature. In this method, the target exerts attractive force to the robot, while static and dynamic objects apply repulsive forces. There are different definitions for the repulsive and attractive forces. For example in [25] a moving target is considered and the repulsive and attractive forces are derived by supposing full knowledge of the target and obstacle's positions and velocities. The final output of method suggested in [25] is a function of the target velocity and the relative positions between the obstacles, the target and the robot. Similarly, by considering full knowledge about moving objects and the target, in [24] relative distance and velocity between the robot and the obstacle are used in repulsive force definition. The attractive force is defined based on relative distance and velocity between the robot and the target. One key problem of this method is that they need exact knowledge of position and velocity of dynamic obstacle and the target. However, this is an unrealistic assumption as none of them is known in practice.

In this paper, Escaping Algorithm is suggest for obstacle avoidance. This algorithm is originated from a common behavior of human. A person usually intends to move in opposite direction in order to avoid colliding with a moving person. The same strategy can be used for mobile robot navigation. In our approach, the robot tries to moves in opposite direction if possibility of collision is detected. Consider Fig. 1; in this figure, obstacle direction in global frame is denoted by angle α . The projection of this direction in robot frame is a suitable direction to align a new repulsive force. In order to perform that, the velocity of obstacle is expressed in the robot frame by the following rotation.

$$V_{obs}^{r} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V_{obs}\cos\alpha \\ V_{obs}\sin\alpha \end{bmatrix} = \begin{bmatrix} V_{obs}\cos(\alpha-\theta) \\ V_{obs}\sin(\alpha-\theta) \end{bmatrix} \quad (12)$$

In Fig. 1, F_t denotes the total force considering static environment. The obstacle velocity which is projected in the robot frame is used to define modifying force F_m. This new force has the same norm as F_t and its direction is opposite to the obstacle direction in x , and parallel to it in y direction. Any time that possibility of a collision is detected, the modifying force F_m is added to the total force and the final force F_f , is used for steering the robot toward the target.

To address the safety of motion, consider Fig. 2. As it is mentioned before, it is supposed that the robot and the obstacle can be modeled by circles with radius R_{obs} and R_{rob} respectively. It is similar to consider the robot as a point and enlarge the radius of obstacle to $R_{obs} + R_{rob}$. As a result, the minimum turning angle for the robot is φ to avoid collision. This angle is shown in Fig. 2 and may be derived by the following equation.

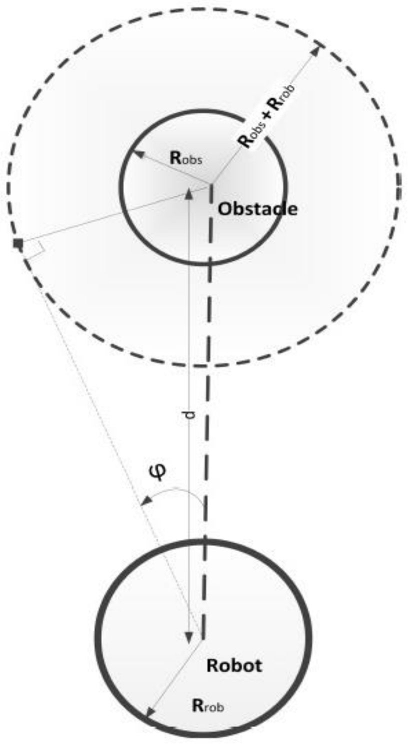

Figure 2- Definition of safe turning angle

$$\varphi = \arcsin\left(\frac{R_{rob} + R_{obs}}{d}\right) \quad (13)$$

In the above equation, R_{rob} is a known parameter and can be measured before test. R_{obs}Is calculated as it is explained in section 3-A. to assure safety, if the turning angle $(\delta-\theta)$, is less than φ, the turning angle φ is considered for the robot steering.

4. Results

In this part the simulation and experimental results of using EA in dynamic environments are presented. It is important to note that navigation by using force field family suffers from trapping in local minima. This problem occurs when the robot direction differs more than 90 degrees off target or F_f is equal to zero.

$$|\delta-\theta| > \pi/2 \quad (14)$$

$$|F_f| = 0 \quad (15)$$

Researchers develop several recovery methods to encounter local minimum problem. For example recovery methods based on electromagnetic field and modification of repulsive potential functions are suggested in [33] and [24] respectively. One of the most popular recovery methods is Wall Following Method

(WFM) which is suggested in [5]. In this context, WFM is used for mobile robot navigation to avoid trapping in local minima.

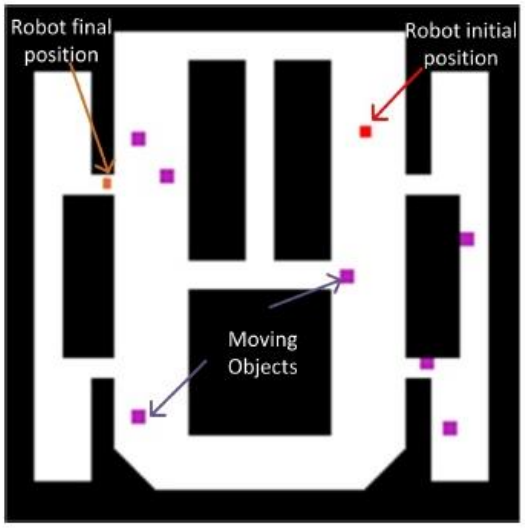

Figure 3- The global map in which the robot reads observation

4.1. Simulation Results

In this section, simulation result of using EA in a dynamic environment is described. For simulation, seven dynamic objects are considered whose positions and motion directions are randomly selected. The robot does not have any prior knowledge about moving objects and their paths are estimated by Kalman filter. The robot obtains its observation from global map shown in Fig. 3. In this figure, the start point, the target point and seven dynamic objects are shown. The robot navigates from the start point to the target point successfully, provided that it does not collide with static and dynamic objects and it reaches the target. Through different tests, it is possible that several collisions are detected at some occasions by collision detection algorithm. In these situations, our algorithm considers only the nearest obstacle. The reason is that the nearest obstacle is more dangerous than others and hence the danger of it should be removed first. After eliminating the nearest danger, the next hazardous collision is considered.

Fig. 4 shows one of the several simulation tests and it is selected because it shows different aspects of EA algorithm. Through moving from the start point to the target point, the robot builds the map of environment incrementally and localizes itself simultaneously in SLAM loop as it is shown in Fig. 4 (a)-(f). As it can be seen in this figure, the robot starts moving to the target in (a). The wall w_0 bans the robot path. In this point the wall following method is activated to preserve the robot from trapping in local minima. As a result of this method, the robot follows wall w_0. Continuing its movement in wall following, the robot faces a moving object in (b). Here, first collision detection algorithm predicts a possible collision and then EA is activated. The modifying force is executed on the robot and it causes that the robot moves in opposite direction of the moving object. When the threat of collision is removed, the robot turns to the target in (c). Due to the absence of dynamic obstacles and the good alignment of the robot toward the target, the robot continues its path to the target in (d) by potential field method (since in this step, the environment is static). However, appearance of another moving object in (e) and the possibility of collision cause that our EA algorithm is activated again and the robot tries to avoid the obstacle by turning it. This movement is depicted in (f). Again, by disappearing dynamic obstacles, the robot uses potential field method to reach the target in (f).

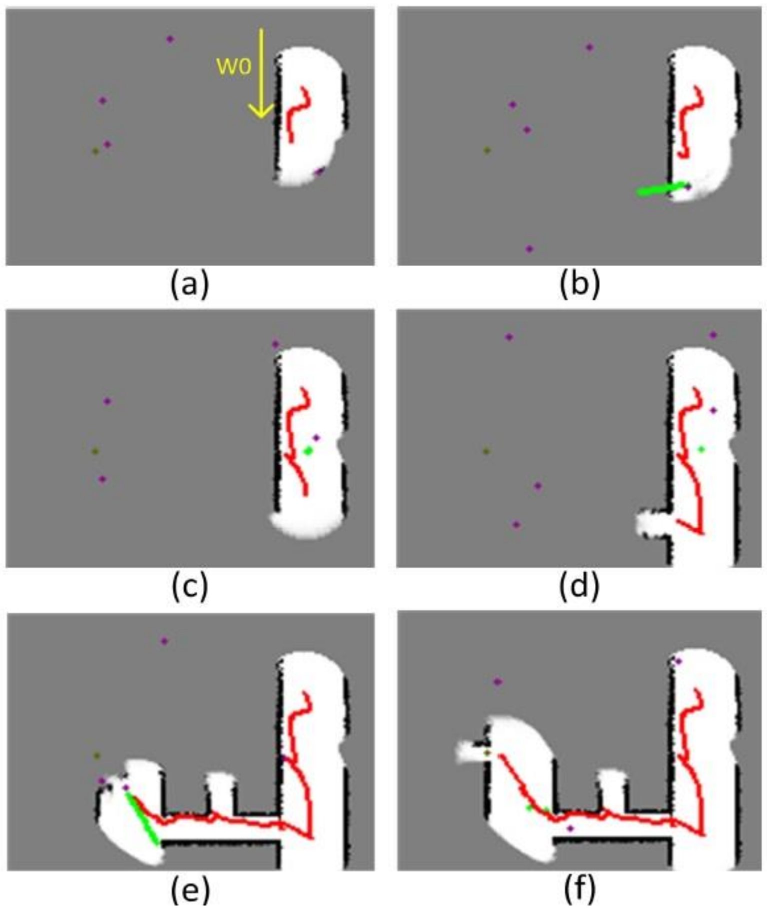

Figure 4- The robot path in dynamic environment using Escaping Algorithm (EA)

Figure 4 shows only one of several simulation tests. Due to the fact that moving objects are located and directed randomly, almost all scenarios have been tested on the robot. These set of simulations provides us enough assurance to implement this algorithm in the following real time experiments.

4.2. Experimental Results

EA for navigation in dynamic environment is implemented on KNTU Mellon mobile robot. The mobile robot perceives environment through a laser range finder whose maximum range reading is 8 meter. The laser scan data is used for map building in a SLAM environment. The ego motion estimation is done by well-known ICP algorithm [34]. To achieve more accurate result in the robot localization, the odometry information by encoders mounted on two wheels, serves as the initial guess for ICP algorithm. A computer with core i5 processor is used for online execution of the algorithm. The algorithm contains a loop for simultaneous localization and mapping (SLAM) which contains the robot navigation routine. For the control of Mellon mobile robot two commands are prompted, namely the velocity of the right and left wheels, denoted by ω_r , and ω_l , respectively. These velocities are easily obtained by linear transformation of linear and angular velocities as bellow:

$$\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 1 & b/2 \\ 1 & -b/2 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \tag{16}$$

where, r and b denote radii of wheel and wheel base, respectively. Different methods are suggested by researchers [25], [18] for linear velocity definition. In this context, since fast performance of the mobile robot is desired, it is good that the mobile robot moves as fast as possible. However, it should start deceleration before an obstacle appears in its path in order to avoid collision. Hence, it is suitable that the mobile robot moves with its maximum linear velocity until it reaches the region that should decelerate to stop completely near an obstacle. Hence,

$$V = \begin{cases} V_{\max} & if \;\; d_{\min} > d_{eff} \\ \dfrac{d_{\min}}{d_{eff}} V_{\max} & otherwise \end{cases} \tag{17}$$

In Eq. (17), $d_{\min}$ is the minimum range reading in range scanning and d_{eff} is the distance passed by the robot when it decelerates from maximum velocity to zero.

In what follows, experimental result of using EA in dynamic environment is given. In this experiment, the target is located in front of the robot with a relative distance of $4m$, while the robot tries to reach target with a distance less than $30cm$. In the terms of static environment, the robot goes straight to the target until it approaches static obstacle and tries to avoid it. However, the robot motion in dynamic environment is different. The dynamic scenario is run twenty two times and the result of one of them is shown in Fig. 5.

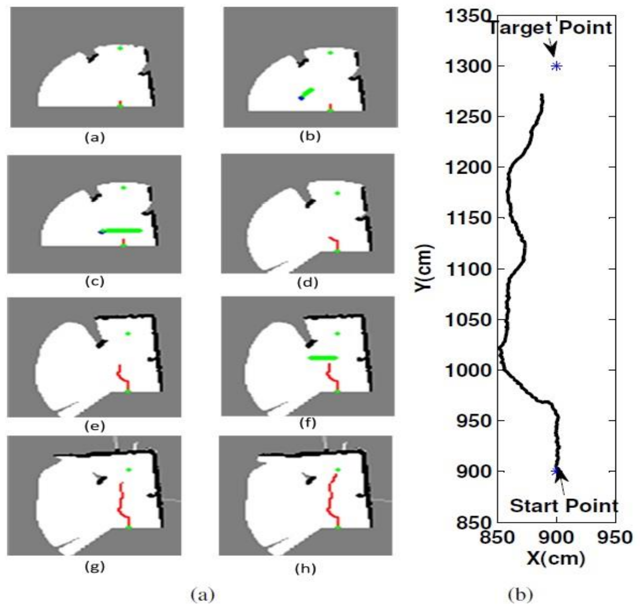

Figure 5- Experimental results of Escaping Algorithm (EA) in a real dynamic environment

To examine the effect of dynamic environment, two moving objects approach the robot from the left. The robot tries to find a collision free path for safe navigation. As indicated in Fig. 5 and in part (a), the robot starts moving toward the target, while in (b) a moving obstacle appears and in (c) a collision possibility is detected. In (d) the robot turns left to avoid collision and again in (e) it moves back toward the target. Moreover, in (f) another collision possibility is detected and the modifying force is added to the total force, and hence, the robot turns to the left to avoid collision in (g). Finally, in (h) it reaches the target. Robot path using EA in this experiment can be observed in more detail in Fig. 5. From the promising results observed in the set of twenty two experiments, we may conclude that EA algorithm is suitable to be used in further development of autonomous robots.

4.3. Probabilistic Velocity Obstacle: A Method for Comparison

Velocity Obstacle (VO) method first introduced in [12] and supposed deterministic knowledge about the velocity of obstacles to produce control inputs for navigation. This method was extended to probabilistic framework in [13] and is used in this paper for the sake of comparison. The idea is describing all velocities in the robot frame and finding those velocities which yield to collision in a predefined time horizon. The Collision Cone CC_{ro} of the robot r relative to the obstacle o, is the set of all relative dangerous velocities $v_r' = (v_r - v_o)$ which finally cause a collision:

$$CC_{ro} = \left\{ v_r' \mid \exists 0 < t < t_h, \quad \left(x_r + v_r' \vec{i} \times t,\, y_r + v_r' \vec{j} \times t \right) \in \Theta \right\} \qquad (18)$$

In which, t_h shows the time horizon, and Θ denotes any type of obstacle (dynamic or static). Any velocity is safe for the robot if the relative velocity does not belong to the collision cone. To compute the probability of collision $P_{coll}(V_R)$ of the robot velocity, all the possible velocities of obstacles have to be considered. Since in this paper Kalman filter is used to predict next poses and velocities of dynamic obstacles, the estimated state of dynamic object and its relevant covariance matrix is used to compute the probability of collision. To obtain permitted velocity for the robot, two constraints are considered: minimizing the risk of collision and reaching the goal position.

$$U(v_r) = \mathrm{dsit}\left(xr[k+1], x_{goal}, v_r\right) \tag{19}$$

$$T_{safe}(v_r) = t_h + T_{break}(v_r) \tag{20}$$

Equation (19) calculates the distance between the robot next pose and the goal position applying velocity v_r to the robot. A velocity can be applied to the robot for the time interval t_h, if the robot does not collide with any object up to T_{safe}; as it can be seen in Eq. (20), T_{safe} contains the required time for deceleration from v_r to zero which is $T_{break}(v_r)$.

Minimizing the distance between the robot and the goal position yields to high velocities and thus, increases the risk of collision. In other words, these two cost functions act reversely. PVO navigates in the following order: First, velocity with maximum utility is considered. Second, the collision time $T_{collision}$ for the selected velocity is calculated. Third, the following equation is evaluated:

$$T_{safe}(v_r) > T_{collision} \tag{21}$$

If Eq. (21) holds, it means that a collision will be happen for the selected velocity. As a result, this velocity is unsafe for the robot. Forth, until finding a safe velocity, iteration will be done.

There are some problems with PVO. One of them is trapping in local minima. The author in [13] mentioned that in the presence of local minima, some optimization parameters are considered. However, the recovery method is not mentioned in either [13] or [12]. The velocity searching space is the second problem of this method. The complexity of computation grows with the size of velocity sampling set. The mentioned issues limit the performance of PVO in complex scenarios and in the real word applications.

4.4. Escaping Algorithm vs Probabilistic Velocity Obstacle

To validate Escaping Algorithm and demonstrate its performance, Probabilistic Velocity Obstacle (PVO) method is implemented and tested through simulation tests. The global map shown in Fig. 6 is used for testing mentioned algorithms in both dynamic and static forms. For dynamic scenario, four moving objects are located and directed randomly. Both EA and PVO were simulated 20 times for static and dynamic environment and the reporting results in Table I shows the average value of them.

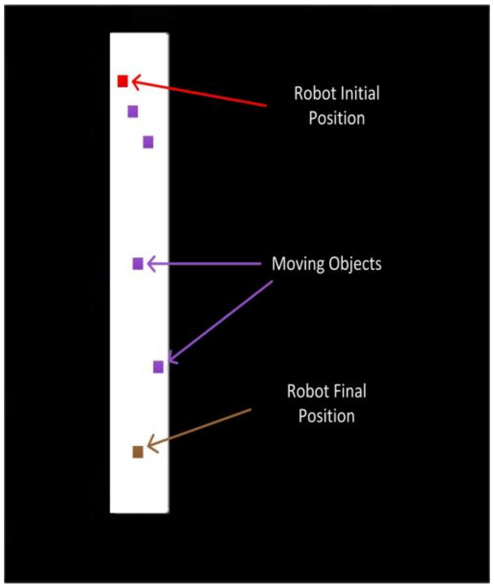

Figure 6- Complete simulated environment

Our study analyses the performance of EA and PVO from two important aspects. The first one is the number of iterations required to complete the mission and the second one is the complexity of algorithm. Here, we analysis the first metric and in the next subsection the complexity issue is extensively discussed.

In our study we found that EA algorithm terminates its mission in smaller number of iterations than PVO. In dynamic environment, EA takes 165 steps in average to finish its task while this value for PVO is 173. While EA algorithm produces continuous values for linear and angular velocity, in PVO the velocity is discretized. Even though it is possible to assume smaller sampling interval for velocity, it causes the complexity grows up polynomially. Assuming a reasonable value for velocity intervals, the selected velocity differs from the optimal value and hence, the required number of iteration is larger than EA. Similarly for static environment, the number of iterations of PVO is more than EA. Please note that in static environment EA reduces to potential field method. The required number of steps is reported in Table 1.

Table 1- Performance analysis of Escaping Algorithm (EA) and Probabilistic Velocity Obstacle (PVO) in static and environment

	EA			PVO		
	Steps	**Running Time (s)**	**Step/Running Time**	**Steps**	**Running Time (s)**	**Step/Running Time**
Static Environment	116	25.63	0.221	167	38.29	0.229
Dynamic Environment	165	36.25	0.220	173	39.756	0.229

4.5. Complexity Analysis

The computational time complexity of EA is

$$O\left(n_a^2\right) \tag{22}$$

In which n_a stands for the size of the active window in force field method. The complexity of PVO grows with the size of control space sampling set. Let's show the size of control space sampling set with n_v. Since in this paper a planar robot is assumed, the control input has two components: $\left[v_x, v_y\right]^T$ and they will be selected from $n_v \times n_v$ space. Besides, PVO needs to check next poses of the robot for the selected control input and defined time horizon. Up to $\max\left\{\frac{(v_r - v_o) \times t_h}{d_{scale}}\right\}$ grids with size d_{scale} will be check for each selected velocity. Hence, the overall complexity of PVO is:

$$O\left(n_m^2 \times \max\left\{\frac{(v_r - v_o) \times t_h}{d_{scale}}\right\}\right) \tag{23}$$

Table 2 shows the considered parameters in EA and PVO. For PVO as it is mentioned in [13], only integer values for the robot velocity is considered. Besides, $5s$ is selected as the time horizon as it is in [13]. Using these values, the complexity of EA is extremely lower than PVO and it is in compliance with the reported values in Table I. The concept of "Step/Running time" is a rough measure of complexity in the absence of mathematical analysis of complexity. Here, our measured values confirm that the complexity of EA is less than PVO and it makes EA suitable for online implementation.

It is possible to choose some of PVO parameters such that it has lower complexity. It can be done by 1- decreasing number of control input samples and 2- by reducing the time horizon. By choosing option 1, the precision of control input selection decreases and may result to improper results in the term of obstacle avoidance and running time. Decreasing time horizon in option 2 is not suitable since this enlarges the danger of collision. Hence, mindful selection of the variables is a must to have the desired behavior.

Table 2- Selected Parameters in Escaping Algorithm (EA) and Probabilistic Velocity Obstacle (PVO)

EA	PVO			
n_a	n_v	v_r	v_o	t_h
31	21	-10< ·<10	-10<·<10	5

5. Discussion and Conclusion

This paper deals with two challenging issues in the navigation problem. First, the method introduced in this paper provides solution for handling unknown observations of dynamic environment and determining the source of observations. This is achieved by defining the three-state map and categorizing data into static and dynamic. As dynamic obstacles move, the trajectory of them is required to completely describe their motion through time. Hence, Kalman filter is used to track and predict the dynamic obstacle motions. The motion prediction of dynamic obstacles helps us to address the second challenging issue and that is using Escaping Algorithm strategy for navigation in dynamic environment. As it is mentioned before, EA is modeled based on force field approach. EA concept originates from the common behavior of human; it is a frequent behavior for us to turn another person in the opposite direction of his movement to avoid collision.

The performance of EA is checked under different metrics. First, several simulation tests in several environments with different number of robots are checked. Some of the considered environments are U shape environment to check the performance of the system in local minimum situations. Then, the proposed method was implemented on Mellon platform to assure the performance of the algorithm in the real implementation. After that, EA is compared to Probabilistic Velocity Obstacle method in the required number of steps for finishing the task and the complexity of computation. Our results show that using typical parameters in PVO, our algorithm has lower complexity and smaller time for completing the mission.

To compare PVO and EA fairly, instead of assuming the velocity and trajectory of dynamic obstacle as a priori, which is one of the basic assumptions in PVO, we predicted them by Kalman filter. This helped us to evaluate two algorithms neutrally. However, substituting the prediction values instead of exact values caused some problem in PVO. Due to use of imperfect knowledge about moving obstacles, PVO failed in situations that the prediction precision is not good enough. Similarly, EA may fail in a situation such that moving objects start their motions in a close vicinity of the robot and Kalman filter does not predict their motion precisely.

Even though the performance of EA is checked through different scenarios, the next step is providing concrete stability analysis of the proposed framework and finding the exact conditions for performance guarantee. Another possible direction of future work is developing this method for navigation of swarm of robots toward their goals considering the global and limited communication links.

References

[1] A. Ferworn, J. Tran, A. Ufkes, and A. D’Souza, Initial experiments on 3d modeling of complex disaster environments using unmanned aerial vehicles, in Safety, Security, and Rescue Robotics (SSRR) of IEEE International Symposium, (2011) 167–171.

[2] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi and M. Fukushima, Emergencyresponse to the nuclear accident at the fukushimadaiichi nuclear power plants using mobile rescue robots,” Journal of Field Robotics, 2012.

[3] X. Lai, S. Ge, and A. Al Mamun, Hierarchical incremental path planning and situation-dependent optimized dynamic motion planning considering accelerations, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, Vol. 37(6), (2007) 1541–1554.

[4] G. Oriolo, G. Ulivi, and M. Vendittelli, Real-time map building and navigation for autonomous robots in unknown environments, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, Vol. 28(3), (1998) 316-333.

[5] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, International journal of robotics research, Vol. 5(1), (1986) 90-98.

[6] Y. Lu, X. Huo, O. Arslan, and P. Tsiotras, Incremental multi-scale search algorithm for dynamic path planning with low worst-case complexity, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, Vol. 41(6), (2011), 1556– 1570.

[7] A. Yahja, S. Singh, and A. Stentz, An efficient on-line path planner for outdoor mobile robots, Robotics and Autonomous systems, Vol. 32 (2), (2000)129-143.

[8] L. B. Cremean, T. B. Foote, J. H. Gillula, G. H. Hines, D. Kogan, K. L. Kriechbaum, J. C. Lamb, J. Leibs, L. Lindzey, C. E. Rasmussen et al., Alice: An information-rich autonomous vehicle for high-speed desert navigation, Journal of Field Robotics, Vol. 23(9), (2006) 777–810.

[9] D. Fox, W. Burgard, and S. Thrun, The dynamic window approach to collision avoidance, Robotics and Automation Magazine, IEEE, Vol. 4 (1), (1997) 23-33.

[10] N. A. Melchior, J. y. Kwak, and R. Simmons, Particle RRT for path planning in very rough terrain, in NASA Science Technology Conference (NSTC), (2007).

[11] G. Chen, T. Fraichard, A real-time navigation architecture for automated vehicles in urban environments, in Intelligent Vehicles Symposium of IEEE, (2007) 1223–1228.

[12] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using velocity obstacles, The

International Journal of Robotics Research, Vol. 17 (7), (1998) 760–772.

[13] C. Fulgenzi, A. Spalanzani, and C. Laugier, Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid, in Robotics and Automation of IEEE International Conference. (2007) 1610–1616.

[14] B. Kluge and E. Prassler, Recursive probabilistic velocity obstacles for reflective navigation, in Field and Service Robotics. Springer, (2006) 71–79.

[15] T. Fraichard, H. Asama, Inevitable collision statesa step towards safer robots? Advanced Robotics, Vol. 18 (10), (2004) 1001–1024.

[16] T. Fraichard, A short paper about motion safety, in Robotics and Automation of IEEE International Conference, (2007) 1140–1145.

[17] A. Bautin, L. Martinez-Gomez, and T. Fraichard, Inevitable collision states: a probabilistic perspective, in Robotics and Automation of IEEE International Conference (ICRA), (2010) 4022–4027.

[18] J. Borenstein and Y. Koren, Real-time obstacle avoidance for fact mobile robots, Systems, Man and Cybernetics, IEEE Transactions, Vol. 19 (5), (1989) 1179-1187.

[19] J. Chuang, N. Ahuja, An analytically tractable potential field model of free space and its application in obstacle avoidance, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions, Vol. 28 (5), (1998)729–736.

[20] E. Rimon, D. Koditschek, Exact robot navigation using artificial potential functions, Robotics and Automation, IEEE Transactions on, Vol. 8(5), (1992) 501–518.

[21] J. Kim, P. Khosla, Real-time obstacle avoidance using harmonic potential functions, Robotics and Automation, IEEE Transactions, Vol. 8(3), (1992) 338–349.

[22] P. Veelaert, W. Bogaerts, Ultrasonic potential field sensor for obstacle avoidance, Robotics and Automation, IEEE Transactions, Vol. 15(4), (1999) 774–779.

[23] N. Ko, B. Lee, Avoidability measure in moving obstacle avoidance problem and its use for robot motion planning, in Intelligent Robots and Systems, Proceedings of IEEE/RSJ International Conference IROS 96, Vol. 3, (1996) 1296–1303.

[24] S. Ge, Y. Cui, Dynamic motion planning for mobile robots using potential field method, Autonomous Robots, Vol. 13(3), (2002) 207–222.

[25] L. Huang, Velocity planning for a mobile robot to track a moving target–a potential field approach, Robotics and Autonomous Systems, Vol. 57(1), (2009) 55–63.

[26] G. Grisetti, C. Stachniss, and W. Burgard, Improved techniques for grid mapping with rao-blackwellized particle filters, Robotics, IEEE Transactions, Vol. 23(1), (2007) 34–46.

[27] S. Ge, Y. Cui, New potential functions for mobile robot path planning, Robotics and Automation, IEEE Transactions, Vol. 16(5), (2000) 615–620.

[28] L. Montesano, J. Minguez, and L. Montano, Modeling dynamic scenarios for local sensor-based motion planning, Autonomous Robots, Vol. 25(3), (2008) 231–251.

[29] S. Thrun, C. Martin, Y. Liu, D. Hahnel, R. Emery-Montemerlo, D. Chakrabarti, and W. Burgard, A real-time expectation-maximization algorithm for acquiring multiplanar maps of indoor environments with mobile robots, Robotics and Automation, IEEE Transactions, Vol. 20(3), (2004) 433–443.

[30] G. McLachlan, T. Krishnan, The EM algorithm and extensions. Wiley New York, Vol. 274, (1997).

[31] I. Cox, A review of statistical data association techniques for motion correspondence, International Journal of Computer Vision, Vol. 10(1), (1993) 53–66.

[32] M. Lindstrom, J. Eklundh, Detecting and tracking moving objects from a mobile platform using a laser range scanner, in Proceedings of Intelligent Robots and Systems, IEEE/RSJ International Conference, Vol. 3. (2001) 1364–1369.

[33] V. Petridis, T. Tsiboukis, An optimal solution to the robot navigation planning problem based on an electromagnetic analogue, Robotic systems: advanced techniques and applications, Vol. 10, (1992) 297–303.

[34] P. Besl, N. McKay, A method for registration of 3-d shapes, IEEE Transactions on pattern analysis and machine intelligence, Vol. 14(2), (1992) 239–256.

[35] F Adib Yaghmaie, A Mobarhani, HD Taghirad, A new method for mobile robot navigation in dynamic environment: Escaping algorithm, in IEEE Robotics and Mechatronics (ICROM), First RSI/ISM International Conference on, (2013) 212-217.

Biography

Farnaz Adib yaghmaie received her B.S. and M.S. degrees in electrical engineering-control in 2009 and 2011 from K. N. Toosi University of Technology, Tehran, Iran. From 2009 to 2011, she was a member of Industrial Control lab (ICL) and worked on the navigation systems for mobile robot. She is currently working toward her Ph.D. degree at Nanyang Technological University (NTU). Now, she is a member of Intelligent Robotic Lab at NTU and her research interests are Swarm Modeling, Coordination System and Task allocation.

Amir Mobarhani received his B.S. degree in Computer Engineering from K. N. Toosi University of Technology, Tehran, Iran, in 2010. He also received his M.S. in Artificial Intelligence at the same university. He has been a member of Advanced Robotic and Automated System (ARAS) since 2008. His research interests include SLAM, Mobile Robot Navigation, Computer Vision and Artificial Intelligence and currently his research is focused on Semantic Maps and Mobile Robot Task Planning.

Hamid D. Taghirad has received his B.Sc. degree in mechanical engineering from Sharif University of Technology, Tehran, Iran, in 1989, M.Sc. in mechanical engineering in 1993, and Ph.D. in electrical engineering in 1997, both from McGill University, Montreal, Canada. He is currently the chairman of the Faculty of Electrical Engineering, and a Professor with the Department of Systems and Control and the Director of the Advanced Robotics and Automated System (ARAS) at K.N. Toosi University of Technology, Tehran, Iran. He is a senior member of IEEE, and member of the board of Industrial Control Center of Excellence (ICCE), at K.N. Toosi University of Technology, editor in chief of Mechatronics Magazine, and Editorial board of International Journal of Robotics: Theory and Application and International Journal of Advanced Robotic Systems. His research interest is robust and nonlinear control applied to robotic systems. His publications include five books, and more than 190 papers in international Journals and conference proceedings.