Navigating Multiple Mobile Robots without Direct Communication

Rahul Kala*
School of Cybernetics, School of Systems Engineering,
University of Reading, Whiteknights, Reading, RG6 6AY, UK
rkala001@gmail.com, http://rkala.99k.org/

This is a pre-peer reviewed version of the following article: R. Kala (2014) Navigating Multiple Mobile Robots without Direct Communication. International Journal of Intelligent Systems, 29(8): 767–786, which has been published in final form at: http://onlinelibrary.wiley.com/doi/10.1002/int.21662/abstract

* **Current Address:** Rahul Kala, Robotics and Artificial Intelligence Laboratory, Indian Institute of Information Technology Allahabad, Devghat, Jhalwa, Allahabad, India, Ph: +91-532-292 2117

Abstract: A bulk of research is being done for the autonomous navigation of a mobile robot. Multi-robot motion planning techniques often assume a direct communication amongst the robots, which makes them practically unusable. Similarly approaches assuming the robot moving amidst humans assume cooperation of humans which may not be the case if the human is replaced by a robot. In this paper a deliberative planning at the higher level with a new cell decomposition technique is presented, along with a reactive planning technique at the finer level which uses fuzzy logic. Coordination amongst the robots in the absence of direct communication and knowledge of other robot's intent is a complex research question, which is solved using a simple fuzzy based modelling. Experimental results show that the multiple robots maintain comfortable distances from the obstacles, robots navigate by near optimal paths, robots can easily escape previously unseen obstacles, and robots coordinate with each other to avoid collision as well as maintain a large separation. This work displays a simple and easy to interpret system for solving complex coordination problem in multi-robotics.
Keywords: Muti-robot motion planning, path planning, cell decomposition, fuzzy inference system, coordination

1. Introduction

The task of path planning for mobile robotics deals with navigating a robot within a map amidst static and dynamic obstacles [1]. The planning techniques may be classified as deliberative and reactive. The deliberative planning techniques (e.g. [2]) construct a trajectory for the robot to follow. A separate control algorithm moves the robot as per the trajectory. For planning in dynamic maps, the trajectory needs to be constantly monitored, and any change in the map resulting in potential collision results in the planning algorithm being re-called to modify the trajectory as per the new map. The reactive techniques (e.g. [3]) on the other hand analyze the map and decide the next immediate action of the robot. This is converted into control signals for a unit move of the robot.

The generated robot motion as per any technique needs to ensure a short path length and a high clearance, which together denote the optimality of the technique. The deliberative techniques are mostly optimal and complete but have a high computation time which makes them unlikely for the dynamically changing maps. The reactive techniques have a small computation time and enable the robot to escape from any new or suddenly appearing obstacles, but the approach is neither optimal nor complete. The robots can many times be struck at a point.

The advantages and disadvantages of the two techniques motivate their hybridization [4]. Many techniques divide the entire planning task at two levels. The higher level deals with the production of a coarser map and planning by a deliberative technique, which produces a coarser level path of the robot. The same is used to guide the robot motion which runs in a dynamic environment with a reactive planning technique. The resultant approach is hence near optimal and near complete, as well as enables the robots to react to unseen obstacles.

The production of a coarser level map is commonly carried by the technique of cell division [5]. The entire map is assumed to be available as a grid of cells at a finer level. Multiple cells of finer level are grouped to form a coarser level cell, which then converts the map into a graph based on cell connectivity. Graph search is a usual technique for computing the coarser level path. Higher cell sizes lead to a very vague coarser path, while low sizes lead to high computational costs. The coarser planning must enable high clearance and short paths.

Planning of multiple robots [6] is a more complex problem as compared to planning a single robot in a dynamic environment. A simple technique may be to treat the other robots as moving obstacles, however since the motion of the other robots is unknown (no direct communication assumed), deliberative techniques cannot be used, and the robots can by their characteristic motions block each other. The optimal navigation plan on the contrary would be much simpler. Coordination deals with devising a strategy to optimally move the robots such that no collision occurs.

Planning of multiple robots may be centralized or decentralized [7]. Centralized techniques create a complex coordination space which accounts for all the possible combination of robots. These techniques are usually time-consuming for even a low number of robots. Decentralized techniques plan each robot independently and use a separate coordination technique for avoiding the potential collisions.

Direct communication is mandatory in the centralized techniques and a common assumption to be made in the decentralized techniques. The robots can cooperatively decide the strategy of motion in such a case. In the simplest form, the robots may be prioritized and then planned using priority based coordination [8-9]. Here every robot avoids collisions with the robots having a higher priority. Communication and similarity of robots are impractical assumptions in such a technique. Further the paths of lower priority robots are usually poor.

Many robots deployed in similar environments are directed to stop in case no path seems possible for navigating to goal, and move normally otherwise [10]. This is a valid strategy in the environments involving dynamic obstacles and humans. The dynamic obstacles may clear with time, while the humans are usually cooperative in motion and move in order to create space for other’s motion [11-12]. Stopping is however not a good strategy when operating in an environment with multiple different robots. Two robots may wait indefinitely for each other to move. Such situations need to be avoided, which forms a major issue in multi-robotic planning without direct communication.

In this paper first a new cell decomposition technique is proposed for planning at the coarser level. The technique ensures high clearance and completeness in cases of obstacles placed with narrow gaps. Further a fuzzy inference system is proposed for the motion of the robot, guided by the coarser path. Then the approach is extended to the presence of multiple robots. The approach extrapolates the potential motion of the other robots, to decide the robot’s immediate move using fuzzy inference system. This when repeated for all the robots at all times of motion creates paths with high clearance and short length.

The key contributions of the paper, in lieu of literature presented in section 2, are: (i) Study of the problem of planning of multiple robots in the absence of any direct communication, within a complex obstacle framework of known and unknown obstacles. This kind of problem has not been sufficiently studied in the literature as most methods either assume some communication, or assume the robot in a dynamic environment which differs due to stated reasons. (ii) Proposing a new cell decomposition method for planning at the coarser level, such that the coarser level path has a high clearance and a short path length. Both objectives cannot be simultaneously ascertained using hexagonal, voronoi, rectangular, quad-tree, probabilistic, or similar approaches. (iii) Design of a fuzzy inference system for the navigation of a robot within the obstacle framework. Developed fuzzy system, under the guidance of coarser path, can also escape the robot from any newly discovered obstacle. (iv) Development of a coordination scheme for the case of multiple robots. Coordination is purely on the basis of assessment of nearby robots without the need of any direct communication. Previously studied fuzzy approaches either coordinate robots using a deliberative technique with fuzzy logic doing lower level planning, or use fuzzy logic as a controller when plan is already formulated. Here fuzzy inference system is used for coordination without inputs from any other technique.

The paper is organized as follows. First some related works are presented in section 2. Section 3 presents the algorithm with coarser level planning, planning in case of a single robot, and planning with multiple robots. Simulation results are given in section 4 and discussions are done in section 5. Section 6 gives the conclusion remarks.

2. Related Works

This work is largely inspired by one of my earlier works for planning of a single robot using fuzzy inference system [13]. In this work a probabilistic map was built using square cells. The coarser graph search algorithm attempted to find paths with low probability of collision and small path length. In this paper a better coarser path planning strategy is used with multi-sized cells that guarantee a high clearance and a collision-free path to goal. Further additional inputs are added to the fuzzy planner to enable the robot maintain a high clearance. The approach is extended to the presence of multiple robots.

A similar problem is navigation of robot amongst humans. Sgorbissa and Zaccaria [12] presented a similar approach in which Voronoi graph was created using an initial map on which deliberative planning was done. Reactive planning was done through artificial potential field and trajectory smoothing. The authors identified scenarios called roaming trails, where a robot was disallowed to move to a position from which it cannot proceed as per the deliberative plan. In a similar work Alvarez-Sanchez et al. [11] placed the heuristic that robot's immediate move is towards the centre of area of obstacle free cone bounded by humans and obstacles. This was inspired by human motion amidst a large crowd. Both approaches assume human cooperation in case the robot is struck. While these methods have a high clearance, the path length may not always be short.

Fuzzy and potential methods are widely used for reactive planning of a single robot. Selekwa et al. [14] presented a fuzzy system for planning of a single robot. The authors modelled multiple behaviours as multiple fuzzy systems with each system responsible for some kind of action. For a given scenario all the systems produced different outputs which were integrated on the basis of their weights. Tu and Baltes [15] presented a fuzzy based potential field method for navigation of a single robot. Like the potential methods, navigation was through the potential gradient. However the computations were through fuzzy arithmetic and modelling. Parhi and Mohanta [16] used a hybrid potential and fuzzy based controller for navigating various robots. The robots could escape static obstacles and avoid each other while reaching the goal.

Motlagh et al. [17] identified the problem of formulating rules for the fuzzy inference system and hence proposed the use of fuzzy cognitive maps for the reactive planning of a single mobile robot. The authors, besides information about obstacle and target, fed in the system the slippage of the two wheels. This meant that their approach could adapt to actuation uncertainties. Similarly Jolly et al. [18] used fuzzy neural network for planning of robots in robot soccer. The field was divided into multiple regions which became fuzzy sets. The fuzzy neural network could learn the direction in which the ball needs to be played. Ng and Trivedi [19] presented use of a neuro-fuzzy controller. A limitation of such learning based approach is however that the robot performs well only to the scenarios similar to those in learning. In a multi-robotic scenario where there are large ways in which robots interact, producing scenarios for exhaustive learning may be difficult.

Yang et al. [20] fused the outputs of two fuzzy systems, one being responsible for obstacle avoidance and the other for reaching the target. In another approach Aguirre and Gonzalez [21] used a rule based deliberative planner to instruct a lower level fuzzy based planner for navigation of a single robot. The higher level planner could give a context of operation for the fuzzy planner which is a big boon for structured environments, while the environment used in this paper was unstructured. Other notable works include [22-24].

Baxter et al. [25] used the concept of shared potential to coordinate multiple robots. Here each robot computed the potential of any point. The different potentials generated by the different robots were shared. The computation of the potential by different robots at same point in the space sometimes led to different recordings. As a result the conflict was resolved by an optimistic or a pessimistic approach. While the optimistic approach assumed the obstacle to be always absent in case of conflict in opinions, the pessimistic approach assumed the obstacle to be present. Sharing implies communication which is a limitation of the work. Further, dealing other robots as static environment is not a powerful coordination technique.

Snape et al. [26] presented hybrid reciprocal velocity objects for planning of multiple robots. The algorithm identified two colliding robots and modified their velocities in the opposite directions, such that half the modifications were for the first colliding robot and the other half for the second colliding robot. If any robot crossed the other robot on the wrong side, the robot being planned assumed complete priority. The approach is not fit for scenarios where obstacles are of any shape and size, which may not entirely be known at the start.

Chand and Carnegie [27] also used a 2-tier approach for problem solving in case of multiple robots. The authors used A* algorithm at both coarser and finer level. Unlike the previous approaches, the author's modelling used the finer level A* algorithm to compute cost between two neighbouring nodes of coarser level A* algorithm. Hence the approach may be seen as a single hierarchical graph search which is entirely deliberative.

A* algorithm gives optimal paths which are short in length with needed clearance. A number of approaches are made to make them perform well in small execution times. Kala et al. [28] converted the finer map into a probabilistic map with a multi-level tree based representation. At any level the algorithm computed a path from source to goal which has a high probability of non-collision as per the probabilistic map as well as small path lengths. The cells along the path were decomposed and a new path was computed along the lines of the older path. The process was iteratively repeated.

In a similar approach Zhang et al. [29] used a hybrid of approximate cell decomposition approach along with probabilistic roadmaps, where the two approaches were alternatively used till the algorithm was able to find a path where all the cells were obstacle free. Similarly Lu et al. [30] iteratively divide and search the graph. The authors used Lifelong Planning A* algorithm to make the resultant approach fit for the dynamic cases. These approaches are extendable to the case of multiple robots. However the deliberative nature restricts their use to planning with no communication and unknown obstacles.

3. Algorithm

The problem deals with the motion of multiple robots within a given map. The algorithm consists of two stages. The first stage is offline and consists of decomposition of the map into a number of cells. This map is used for online planning by all the robots. Each robot first plans a coarser level path. This path is used to guide the robot using a fuzzy inference system. The fuzzy inference system attempts to move the robot towards the goal, while avoiding any previously known or new obstacles. At the same time the fuzzy inference system is given inputs regarding the motion of the other robots. This is responsible for the coordination between the robots. The fuzzy planner attempts to escape the robot from collision with the other robots, while maintaining sufficiently large gaps. The robots have no direct communication and hence they need to track each other, which is solely what the coordination is based upon. The algorithm is summarized in figure 1. Details are provided in subsequent sub-sections.

3.1 Coarser Level Planning

The coarser level planning is responsible for the generation of a vague path for the motion of the robot. It is assumed that a map is already available which details the regions of obstacles and navigable regions. The map is common for all the robots. Such maps can easily be built offline for any home, office or similar environments. The map would broadly capture the room size along with pathways as navigable regions and other objects (tables, chairs, etc.) as obstacles. There may be changes in the room like a new chair being placed, or a chair being moved, which may not be reflected in this map.

Consider that a map of size m x n is available as a grid with $map(x,y)$ denoting the presence ($map(x,y)$=1) or absence ($map(x,y)$=0) of an obstacle at the location (x,y), $1 \leq x \leq m$, $1 \leq y \leq n$. The first step is to reduce the dimensionality of the map which is done by fitting large sized cells into the free regions of the map. A number of approaches exist in the literature for the task which includes cell shapes like triangular cells, hexagonal cells [31], Voronoi based cells [32], rectangular cells with quad-tree map representation [33-34], framed quad-tree [35-36] etc. The choice is chiefly governed by the fact that in the absence of new obstacles and other robots, the path traced by the robot would be similar to the one returned by the coarser level planning. Hence high clearance and short path length are desirable. Voronoi and similar approaches may not guarantee a small path length. Probabilistic representation may require a high magnitude of computation at the coarser planning to ensure a collision free path, at the same time not ensuring a high clearance. My prior approach of probabilistic representation [13] used a single graph search at a specific resolution of map, generating probabilistic path which is neither guaranteed to be collision free nor to maintain a high clearance. The method generates good results for maps with wide spaces and highly cluttered environments, but may fail in cases with narrow corridors.

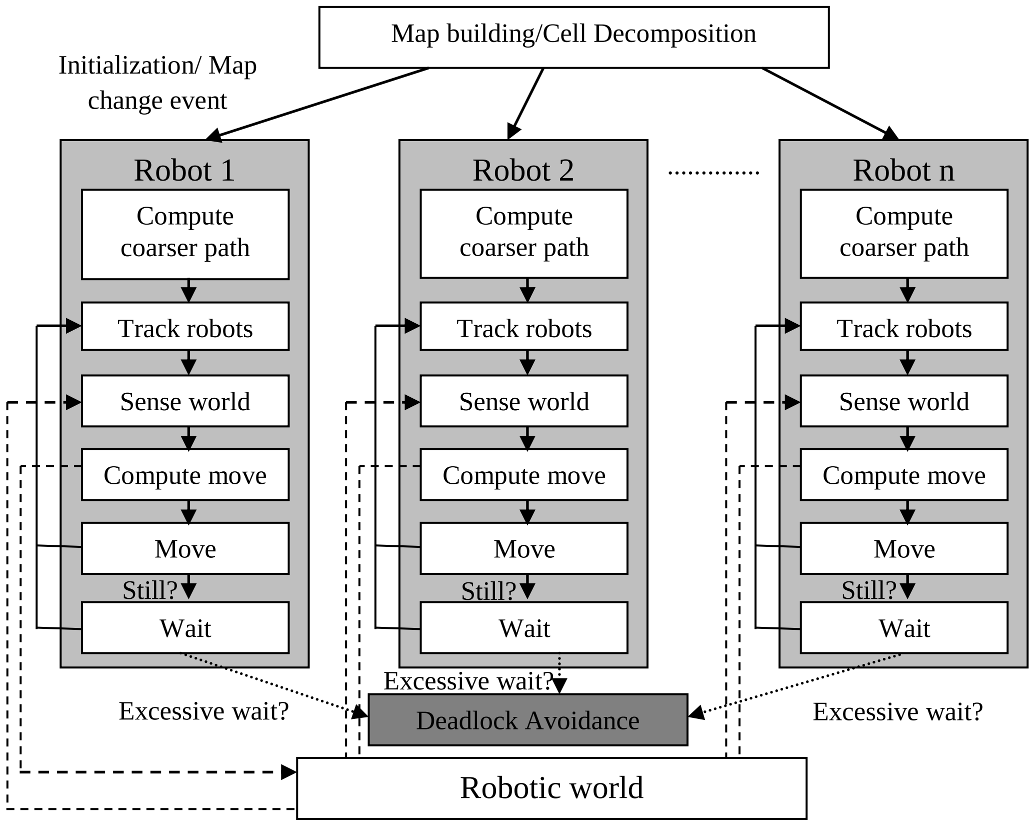


Figure 1: Algorithm Framework. An initial map is available for the environment. A cell decomposition technique is applied which produces a low resolution graph for all the robots. Each robot plans a coarser path, which is used to guide the robot motion using fuzzy inference system. In case of no possible motion, robots first wait and then an external deadlock avoidance strategy is used. In case the coarser map changes (big change with static obstacles), coarser planning repeats.

Considering the nature of the problem that the coarser level map is built offline with an online planning, a new strategy for map decomposition is devised. Let the collection of cells be V. Consider the i^{th} cell V_i be a square of size $a_i \times a_i$ with centre at (x_i, y_i). For the cell to be valid the following points must hold: (i) The complete cell must lie inside the map that is $1 \leq x_i\text{-}a_i/2 \leq x_i+a_i/2 \leq m$, $1 \leq y_i\text{-}a_i/2 \leq y_i+a_i/2 \leq n$. (ii) The complete cell must be obstacle free that is $map(x_i+t_x, y_i+t_y)=0\ \forall\ \text{-}a_i/2 \leq t_x, t_y \leq a_i/2$. (iii) No two cells must overlap that is $area(V_i) \cap area(V_j) = \phi$, $\forall\ V_i, V_j \in V, V_i \neq V_j$. Here $area(V_i)$ denotes the area of map occupied by the cell V_i. (iv) The maximum and minimum size of any cell is bounded that is $sizeMin \leq a_i \leq sizeMax$. Here *sizeMax* is dependent on the desired maximum clearance, while *sizeMin* is for computational speedup of map building and search.

Consider the robot follows the path of the coarser search. Consider a single rectangular shaped obstacle. The optimal path would make the robot keep a fixed maximum clearance C_{max}, the value of which depends upon the desired safety and possible uncertainties. This emphasizes that a coarser map cell be located such that its centre lies at a separation of C_{max} from the obstacle plus the robot size. A cell more distant from obstacle would result in a clearance of more than C_{max} but a longer path length. A nearer cell may not provide a clearance of C_{max} and hence the robot may have to use a more distant cell if available, increasing the path length. *sizeMin* eliminates too small cells being placed in the map. Instead a part of obstacle free map may be without being covered by any cell.

As per the strategy used in this paper, for an optimal cell placement the following objectives need to be fulfilled: (i) Maximize the area covered by the cells that is $\max\left(\left|\bigcup_i area(V_i)\right|\right)$. (ii) Minimize the number of cells that is min(|V|). In order to meet these objectives a simple heuristic approach is used. The algorithm fills the cells in the entire map starting from the largest size first *sizeMax* to the smallest size *sizeMin*. The following is the general algorithm. The cell breakup for a synthetic map is given in figure 2.

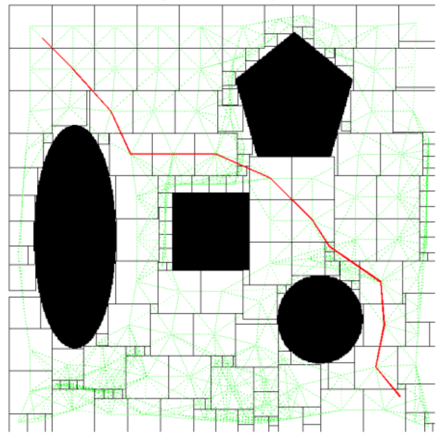


Figure 2: Coarser Level Planning. The map is first decomposed into cells by filling in square cells from large sized to small sized in the free regions (black rectangles). The neighbouring cells are connected to produce a graph (green lines). This is used for coarser level planning (red line).

Clearance is an objective of the search algorithm and hence must explicitly be known for each cell. The computation task is not much time consuming since it is guaranteed that no obstacle exists in the square of size a_i around the cell V_i. Any cell V_i may have either an obstacle just outside this square or it may be completely surrounded by other similar cells.

CellDecomposition(map)

V← ϕ

used ←map

for a from *sizeMax* down to *sizeMin*

    for all valid cells V_i with centre (x_i, y_i) and size a

        C ← clearance(V_i)

        V ← V ∪ V_i <x_i, y_i, a, C>

        used(x_i+t_x, y_i+t_y) ← 1 $\forall$ $-a/2 \le t_x, t_y \le a/2$

The data structure *used*() is used to compute the area occupied by some cell or obstacle. The function *clearance*() returns the clearance of a point sized robot placed at centre of cell, with clearance measured as minimum of straight and diagonal directions along the axis system.

The other task is to convert the resultant map into a graph. The set of cells V denote the vertices of the graph. Any cell V_i is connected to a cell V_j if $\| V_i - V_j \| \le sizeMax \times \eta$ and V_i to V_j is a collision free path. Here η determines the connectivity of the coarser map. Higher values of η favour path length, but clearance may not be guaranteed.

Every robot searches the graph so formed. The source and the goal are added as additional vertices and the corresponding neighbouring vertices are added as edges. A* algorithm is used as the search algorithm. The historical cost of any vertex V_j (say $g(V_j)$) is given by (1) while the heuristic cost is the Euclidian distance to the goal.

$$g(V_j) = \begin{cases} g(V_i) + \left\| V_j - v_i \right\| + \alpha.\max\left(sizeMax/2 - C(V_j), 0\right) & V_j \neq Source \\ \alpha.\max\left(sizeMax/2 - C(V_j), 0\right) & V_j = Source \end{cases} \quad (1)$$

The second factor is the path length, while the third factor penalizes any clearance less than *sizeMax*/2 by a scale of α.

The searched path is returned. Figure 2 shows the path for a synthetic scenario. If the environment has static sensors and computing devices which monitor the change in environment (excluding dynamic and small obstacles) at some low frequency, and there exists some mechanism to inform the same to the robot, the search performed may be repeated at every change informed [37]. The immediate position may be used as the source.

3.2 Planning of a single robot

The coarser level returns a vague path for guiding the robot to follow. The path cannot be used for the actual motion as it may not consider the nonholonomic constraints of the robot, new obstacles may have emerged in the environment making the previously collision free path as collision prone, some obstacles may have changed in the course of time affecting the path, other moving robots or dynamic obstacles were not considered [4, 12]. The finer level planning, for which fuzzy inference system is used, has all these challenges to cater to. This is studied in the head of planning with a single robot in this sub-section, and planning in presence of multiple robots in the next sub-section.

For a single robot the challenge is the design of a fuzzy inference system that takes the current scenario as input and generates as output the immediate move of the robot. The designed fuzzy system takes 10 inputs, out of which 6 are used for planning of a single robot. The inputs are (i) distance from the obstacle in front, (ii-iii) distance from the obstacle diagonally ahead along the two front diagonals, (iv) angular deviation of the robot from the goal, (v) distance of the robot from the goal, and (vi) preferential turn. The first five inputs are continuous inputs while the last input is a discrete (label) input which tells whether the robot should preferably turn left or right.

The preferential turn input is used only when an obstacle is directly in front and the robot needs to decide whether to overcome it from the left or right side. This input is a strategy input which decides the strategy of the robot in overcoming the obstacle ahead. Three strategies are devised for this input. (a) The first strategy is to align the robot towards the goal. If the goal is towards left, a left turn is preferential and vice versa. If the goal is directly ahead of the obstacle, a right turn is preferred. (b) The second strategy is to take the same turn which was taken at the previous time step. (c) The third strategy is to take the turn which is preferred for the avoidance of the other robots as we shall see in next section.

First strategy is suited for the most general scenarios. A problem however comes if the robot is in front of an elongated obstacle with the goal on the other side. This may happen due to a new obstacle, or a previously seen obstacle with the goal at a large distance. In such cases the robot first attempts to align towards the goal while still travelling straight. As the obstacle is still in between the robot and the goal, it goes further on the other side. Now the preferential turn directs the robot to reverse its obstacle avoidance strategy from avoiding the obstacle from left to avoiding the obstacle from right (or vice versa). These oscillations are controlled by strategy 2, which is applied only when the robot is near some obstacle in front and the strategy is applied till the obstacle is not passed. The corresponding distance is tracked to judge whether the obstacle is completely passed.

When the obstacle is far, the robot still attempts to align itself by small steering so as to overcome it. When some moving robot is far away, it also attempts to overcome the moving robot far away from small steering, which is governed by an additional input as we shall see later for case of multiple robots. In case the other robot or obstacle is near, one strategy clearly takes precedence over the other. However if these are far the two strategies may cancel each other's effect out if they differ. Hence no steering is produced. Now we either change the strategy of robot avoidance or obstacle avoidance. The strategy invoked by the lesser distance gets imposed.

The robot has a single output which is the steering that the robot should turn in order to reach the goal or avoid the obstacles. Apart from path length and clearance, one of the important objectives of a good planning algorithm is a smooth path. The fuzzy inference system is tuned to produce outputs such that the path is smooth and only in the cases where a steep turn is mandatory, the motion of the robot is forced by a not much smooth path.

Speed control of the robot is not included in the fuzzy rules but is rather kept independent [38-39]. The front and the diagonal distances are analyzed to decide the maximum speed allowable while keeping a safety distance from the obstacles as well as giving the robot enough time to slow down or stop to avoid collisions. Hence in case of a deadlock the robot would come to a standstill, rather than collide with the obstacles. In this algorithm human aid is expected in such scenarios of deadlock (people nearby may clear the way, or the other robots may move, etc.). The robot waits for some time for the obstacles to clear, and then may sound an alarm for human aid. In the future version the attempt would be to replace this with strategies which can avoid deadlocks. Examples include moving all robots back and then moving them by some priority, re-planning paths at coarser level with marked region as blocked, etc.

The key aspect of the fuzzy inference system is the rules. The rules are manually written. There are three types of rules. The first type avoids collisions of the robot as well as attempts to keep the robot at a safety distance from obstacles. The second type attempts to align and move the robot towards the goal. The rule becomes dominant when the robot is near the goal. The third type (as we see later) attempts to escape the robot from collisions with the other robots.

The relation between the coarser path and the fuzzy inference system determines the hybridization between the algorithms at the two levels [13]. The immediate goal of the fuzzy planner at all times is a vertex of the coarser planner at a distance larger than β (≥ *sizeMax*) units away. This distance needs to be large as it is possible for the immediate goal to be currently on some obstacle. The value may not be very large, which makes the planner primarily fuzzy with limitations of being struck or go wide away from guiding path making the path length sub-optimal. The concept is explained in figure 3.

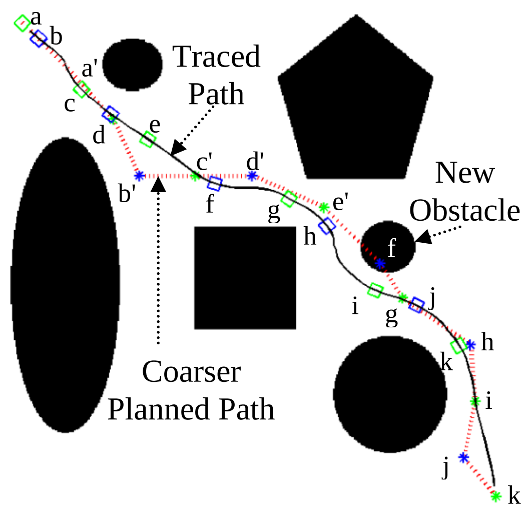


Figure 3: Robotic Guidance by Coarser Planned Path. The coarser level path (red line) is a collection of vertices. Initial position is *a* with goal *a'*. When robot comes to *b*, it is close enough to the goal and the goal is hence changed to *b'* and so on till the final goal *k'*. Traced trajectory is shown by black line. *f'* is inside obstacle as the obstacle was not present at the time the coarser map was built.

3.3 Planning with multiple robots

In the absence of communication, it is clear that multiple robots can interact in multiple ways [26, 40]. As a simple example consider two robots travelling in a straight line such that the two collide at some point of time. Now the first robot has the option of changing its path such that the other robot goes first from the point of collision and similarly modifying the path. In presence of communication the robots can cooperatively decide the order or path, but in the absence of communication it is not possible. Suppose the two robots take opposite decision regarding strategy to avoid each other, they again come into collision prone situation with a rather odd looking strategy. This could go on indefinitely with one robot blocking other robot. The situation worsens in the case of the presence of more robots.

Similar situation is seen at times with humans walking as well. Two people tend to see each other coming in each one's way, both stop, seeing other stop start at the same time, until one of them aggressively moves ahead with the other waiting. Hence in place of making complex protocols, which the other robots being different may not follow, moving the robots by simple plans is opted. In most cases such symmetric conditions do not occur when a robot is unable to devise a motion plan considering the motion of the other robots. If such scenarios do occur, there is an ample of time for the symmetry to be broken as was the case with the example of walking humans. If the robots come very close and still motion plan looks non-obvious, human assistance may be required, which is a very rare case. As per deadlock avoidance strategy, the robots would wait for some time before calling for assistance.

The task is hence to predict the intent of other robots and to plan a strategy to place oneself maintaining large distances from these robots. The robot tracks the position, orientation and speeds of all the other robots. This is in consensus with the notion of tracking of dynamic obstacles for any dynamic planning technique. The other robots are assumed to travel with the same speed and the same direction for the rest of their journeys, till the time either the other robot collides with some obstacle, or the robot being planned collides with some obstacle or another robot. In this extrapolation of motion, at all times four scenario indicators are measured which form the input to the fuzzy system. These are (vii) least distance of the robot from any robot directly in front, (viii) least distance of the robot from any robot in the diagonal right quadrant, (ix) least distance of the robot from any robot in the diagonal left quadrant, and (x) preferential turn.

The distances are measured from the current robot position to the extrapolated position of the other robot where minimum distances are recorded. In this manner the distances play a role closely similar to static obstacles. Say two robots are heading towards each other with the same speed, which means that they collide in between. This is equivalent to having a static obstacle in between the map, which is exactly what the distance from front would be returned. The input preferential turn in this case is taken as a simple strategy for which the two diagonal quadrant distance inputs are considered. If the left distance is less, the robot attempts to maximize it by moving rightward and vice versa.

In static obstacles we had measured the front diagonal distance along the diagonal line, while here we take in the entire quadrant. The reason is that the static obstacles are usually large while mobile robots are small and may escape the diagonal line in search. Further the obstacles are known as a map and searching the entire quadrant is time consuming, while the position of other robots is known from which distance and quadrant determination may be done in a unit time. This further clarifies that the modelling is not to have a mobile robot which only uses sonar sensors and maps the sensor readings to the actuator output, but a more advanced robot which can track other robots, make dynamic maps, etc.

The new inputs for coordination between the robots are broadly the same as the inputs for static case. These inputs are taken in addition to the old inputs (in place of simply taking minimum of the corresponding inputs) so as to have a better control over the relative motion of robots. Moving robots behave different than the static obstacles, and additional inputs equip us to better control the behaviour.

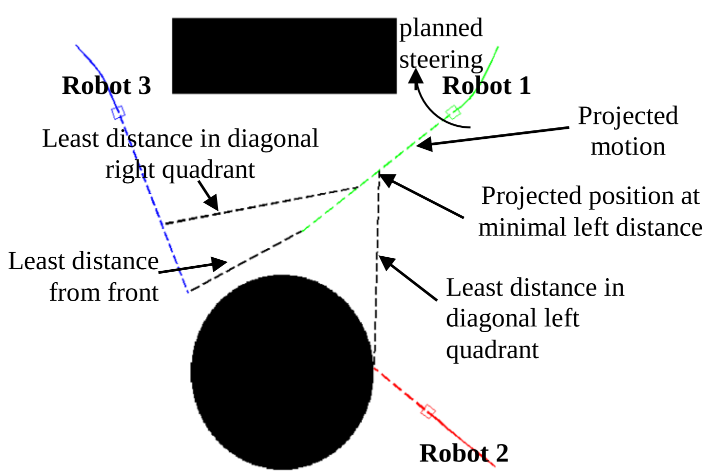


Figure 4: Behaviour of the robot with static obstacles and other robots. Robots attempt to maximize separation. Coloured dashed lines represent the projected motion. Minimum distances are shown as black dashed lines drawn at the point in projected motion where minimum distances were recorded. For robot 1, minimal distance at left quadrant is from robot 2 and right quadrant from robot 3. Robot plans to steer right to overcome obstacle, increase separation from robot 2, and overcome robot 3 from its right.

The fuzzy rules largely attempt to maximize the input distances to obstacles and align the robot to move towards the goal. Analytically it may be said that in a scenario with static obstacles and multiple robots, the planner attempts to keep as large distance as possible from each of these. If the number of robots and obstacles is very large, the behaviour is largely guided by the immediately surrounding robots, as further extrapolation would result in some collision beyond which motion is not analyzed. Very large separations from the coarser path are unlikely as in such a case the distance from the goal would be very large and the goal would attempt to strongly attract the robot. Hence robot attempts to maximize separations from the nearest obstacles and robots, while marching roughly towards the immediate goal. This is very similar to mechanism by which humans walk. Like humans turning of a robot gives a gesture whether it wants to go first. As a result potential collision is altered to no collision with decent separation distance. The other robot which earlier was likely to collide, may now see the robot at a later time lying at the other side, and may move in opposite direction to maximize separation. This is similar to the use of walking gestures with humans. Humans further use facial gestures in case of deadlocks, which is not implemented in this work. Figure 4 shows the concept of motion of the robot.

3.4 General Algorithm Structure

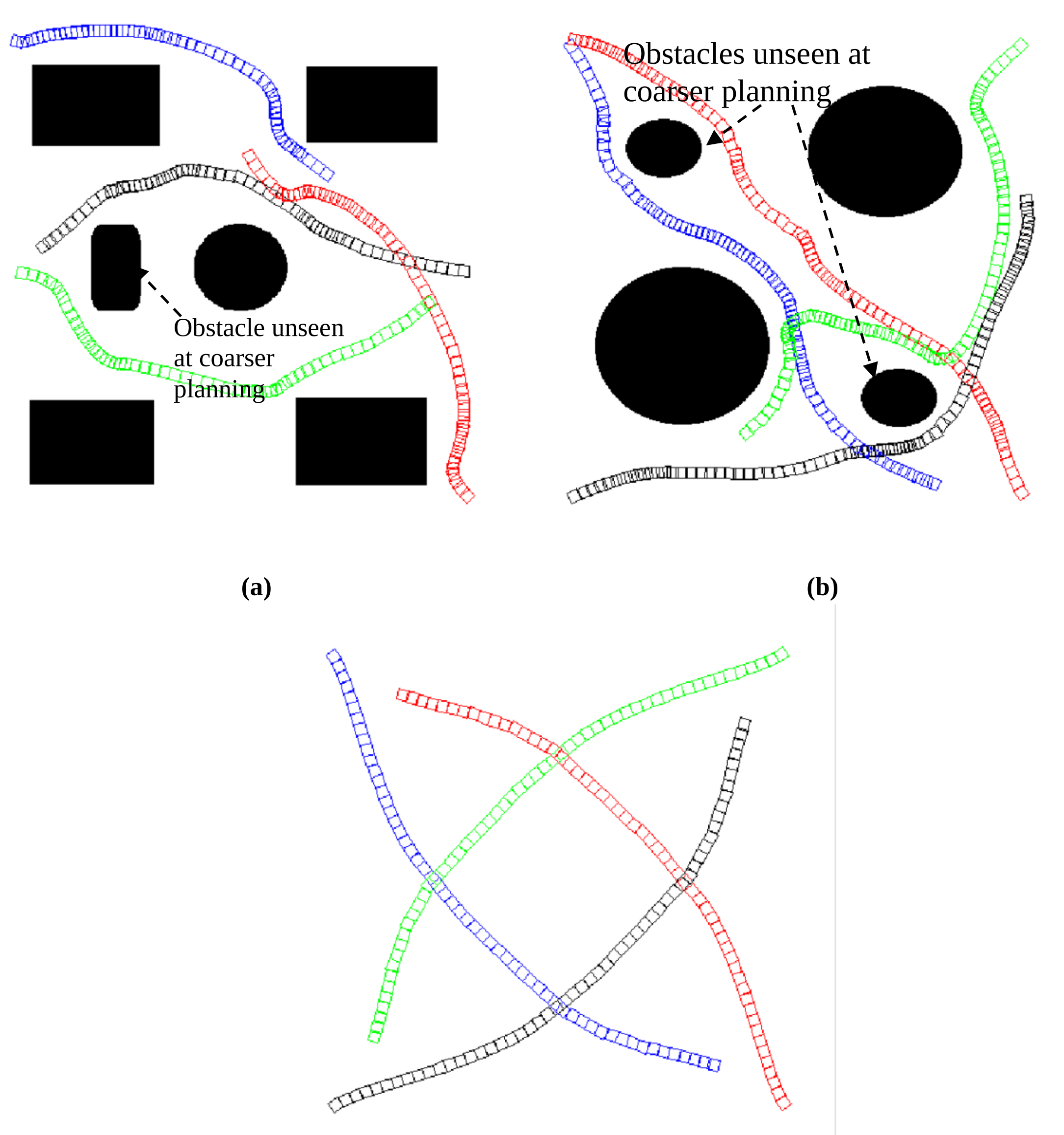
The general algorithm for the motion of a single robot is as follows.

```
Plan(source, goal, current map, coarser map)
Path ← getCoarserPath(coarser map, source, goal)
CurrentPos ← source
CurrentSpeed ← 0
waitTime = 0
while CurrentPos ≠ goal
        FuzzyGoal ← ComputeFuzzyGoal(CurrentPos, Path)
        Track other robots
        I ← Compute fuzzy inputs
        steer ← FIS(I)
        speed ← DesiredSpeed(I)
        [CurrentPos, CurrentSpeed]  ← move(CurrentPos, CurrentSpeed, speed, steer)
        if speed = 0
                if waitTime<waitThreshold, waitTime ← waitTime+1
                else, call for human help
        else waitTime = 0
        if change in coarser map
                Path ← getCoarserPath(coarser map, CurrentPos, goal)
```

The function *getCoarserPath*() uses A* algorithm on the robot's source and goal along with the coarser level map and returns the coarser level path. The robot is simulated till it reaches the goal. The function *ComputeFuzzyGoal*() updates the goal given to the fuzzy planner (*FuzzyGoal*). A change of goal takes place if the robot is very close to the current goal. In most cases immediately next vertex in coarser path becomes the next goal. Function *FIS*() evaluates the output of fuzzy inference system, which is the steering of the robot. The function *DesiredSpeed*() analyses the scenario and computes the desirable speed of the robot considering the safety distances. The function *move*() moves the robot as per acceleration limits and steer limits. If the robot is at a standstill due to some deadlock, it waits for some time and then calls for human help. This is controlled by the variable *waitTime*. In case of any change of coarser map, the coarser path is invalidated and re-computed.

4. Simulation Results

The algorithm was tested via a number of experiments covering various types of scenarios possible. In all the scenarios the various robots were expected to go through short paths with high clearance. The A* algorithm used at a coarser level plays a major role to decide the obstacle avoidance strategy for the robots. The fuzzy rules at the finer level are such that a large deviation from the coarser path is unlikely. The finer level fuzzy planner itself ensures that various robots stay away from each other as well as any new obstacle is easily avoided. While the individual testing of the components seems easy, some interesting behaviours are seen while the components mix. In terms of developed fuzzy inference systems, various robots and obstacles via rules influence the robot to move, while the suggested moves may be contradictory to each other. The robot is still able to move much the way as humans move in such environments. By experiments it is seen that while the robots are away from each other, they move easily. However having combination of robots and obstacles creates an interesting situation wherein some robots may make the motion difficult for a specific robot which has to go as directed by the coarser path, different robots may accidently block each other's way and then make out, robots may disagree with plans and later seem to come to a consensus, etc. The plans may hence be regarded non-optimal from classical sense of planning with communication. However the uncertainty of knowing the intent of the other robot makes the difference. Three scenarios are presented here.



(c)

Figure 5: Simulation results for various scenarios. The images are captured just before the end of the journey. Corner points denote the starting position. For details as to how the various robots avoid each other, please refer video 1.

The first scenario shows a common scenario, where multiple regular shaped obstacles are placed in the map. The scenario is shown in figure 5(a) and summarized in table 1. For a greater understanding of the motion please refer to video 1. The coarser level planning for all the robots does a decent job of figuring out the prospective paths of the robots. The major challenge is the new obstacle placed in the path. Consider the green robot for which the coarser path crosses the new obstacle. The robot quickly steers to avoid the new obstacle, disobeying the coarser path, and then approximately follows the coarser path. The red robot and blue robot attempt to use the same section of the map at the same time and thus block each other's path. The red robot first attempted to go close to the obstacle, and then left extra space for the blue robot. The blue robot has a hard task being surrounded by the red robot on one side and the obstacle on the other side. It slows so as to ensure it is rightly placed.

In scenario 2 four robots are made to move diagonally, while circular obstacles are placed in the centre as shown in figure 5(b). Two new obstacles are added to make it difficult for the robots to move. The attempt is to study the interactions between the robots. The red and the blue robot are diagonally apart and move leftwards to avoid collision. However, soon afterwards the green robot has a major decision to make regarding its strategy to avoid the blue robot, while also avoiding nearby obstacle. It can be seen that decision to go after blue robot results in a non-optimal path, but the decision is only on the basis of analyzed motion of the blue robot. Due to a highly packed environment with the blue robot very near, the green robot has to slow by a large amount. This is similar to a human being packed in a congested environment and hence having to slow down.

The two scenarios judged interaction between the robots and obstacles simultaneously well, however presence of obstacles simplifies interaction between multiple robots. Attempting to escape multiple robots simultaneously is harder than escaping them one by one. Hence an obstacle free environment is constructed with four robots to move diagonally as shown in figure 5(c). By optimal paths they all collide simultaneously in the centre. The robots seem to avoid each other nicely. All of them further maintain a large separation between themselves. Each robot sees the other three, and decides a direction of steering to escape collision. At later stages it steers more to increase the clearance. The attempt to increase clearance happens to a large extent, until the attempt to reach the goal becomes dominant in the fuzzy system.

5. Discussions

Table 1 shows the path lengths and clearances between the robots for different scenarios. An optimal path with an ideal tradeoff between the two objectives can only be made by an offline centralized planner. It would hence be interesting to discuss in what situations do the robots deviate from the ideal plans. A single robot has a near-optimal coarser path which is traced by the robot to a large extent with deviations due to nonholonomic constraints. This makes the traced path close to optimal. Moving robots barely affect the optimality if they go through regions far away from the robot, or through regions isolated from the robot by some obstacles. In this way obstacles play a role in making planned path close to optimal in multi-robot scenarios.

However the effect is large if the robots by their ideal paths collide or lie close. The effect can be studied in two heads, if there is no conflict of interests and if there is a conflict of interests. In case of the former case, the two robots succeed to a large extent to increase the clearance between themselves and the obstacles, and the path is close to optimal. However the latter case leads to large deviations from the optimal plan. The deviations are proportional to the time required to reach a consensus regarding the travel plan. Usually the consensus would come due to presence of obstacles, change of immediate goal, steering of robots by different magnitudes making one robot dominate, etc. In reality it is difficult to get scenarios where robots tend to collide, and consensus is not obvious too soon. This makes the algorithm near-optimal for most cases.

The case of too many robots, all tending to collide at a place is far more difficult to get, but also more interesting. In such cases it may happen that a change of plan of a robot produces likely collision with another robot, which then has to change its plan and so on. Now multiple robots need to reach a consensus. For some robots a change of plan may not be possible, which dominate and hence help to reach a consensus early. Other robots may with time get into similar situations like near an obstacle, or largely deviated from goal, where change of plan is not likely. Like the case of two robots, as time passes the consensus starts getting formed between the robots.

Overall the algorithm seemed to perform well for all scenarios presented, there however exist some limitations. One of the biggest limitations of the current work is that a robot may make the other robot deviate to a large level, which may ultimately land up at some other side of the obstacle. In such a situation the robot may either get trapped or may take a rather odd path to reach near to the current goal. Currently the preferential robot turn is guessed from the fuzzy inputs, which vaguely represent robotic map. It is possible for the preferential turn in certain cases to guide the robot in wrong direction, especially with unknown obstacles and robots. A more deliberative technique may work better, at the cost of computational time. Further the current fuzzy system was manually designed and the performance may improve on using some automated technique. The large number of variables makes it a difficult problem from an optimization perspective.

Table 1: Summary of Simulation Results

Scenario	Robot	Initial Source/ Direction	Goal/ Maximum Speed	Distance	Travel Time	Average Speed	Minimum Clearance	Average Clearance
1	**1**	(40,40)/ π/4	(460,460)/ 10	665.0495	97	6.8562	8	59.8826
	2	(460,460)/ 3π/4	(40,40)/ 10	669.6466	98	6.8331	20	60.1983
	3	(250,40)/ 0	(250,460)/ 10	474.3280	61	7.7759	13	72.4770
	4	(250,460)/ Π	(250,40)/ 10	449.6468	59	7.6211	27	78.7942
2	**1**	(40,40)/ π/4	(460,460)/ 10	626.0939	91	6.8802	23	55.3454
	2	(460,460)/ 3π/4	(40,40)/ 10	602.0465	78	7.7185	25	74.1804
	3	(40,460)/ 3π/4	(40,460)/ 10	746.7693	92	8.1171	12	60.7582
	4	(460,40)/ -π/4	(460,40)/ 10	702.4915	92	7.6358	15	61.1259
3	**1**	(40,40)/ π/4	(460,460)/ 10	609.4166	64	9.5221	43	133.8977
	2	(460,460)/ 3π/4	(40,40)/ 10	608.7718	65	9.3657	43	133.7010
	3	(40,460)/ 3π/4	(40,460)/ 10	609.0541	65	9.3701	42	134.3143
	4	(460,40)/ -π/4	(460,40)/ 10	609.5628	65	9.3779	43	134.6095

All units are arbitrary. Unit simulation distance and unit simulation time are used.

5. Conclusions

In the absence of communication different robots need to strategically decide their motion so as to not to collide with the other robots, while maintaining a short path length and a high clearance. Since the plan cannot be verified for consensus with the other robots, the task is difficult. In this paper a two layer planning was proposed. The coarser map was generated using a new cell decomposition technique and the coarser path guided the robot which moved using a fuzzy inference system. In all experimented scenarios the robots could escape from all new and previously known obstacles. Further they could coordinate with each other for motion. Deadlocks were not observed but are possible in specific scenarios, in which case human aid is suggested.

By simulations it is visible that the various robots behave similar to humans. This explains how complex coordination issues can be solved using simple rules with fuzzy inference system. The deadlock is due to the fact that humans can understand body language and gestures, while the inputs in this planning are limited. This further ensures that the robots in this manner can move amidst mixed human and robot environment. This makes a realistic step towards the use of multiple robots for different tasks in indoor environments.

References

[1] Latombe JC. Robot Motion Planning. Norwell, MA, USA: Kluwer Academic Publishers; 1991
[2] Kala R, Shukla A, Tiwari R. Robotic Path Planning using Evolutionary Momentum based Exploration. J Exp Theor Artif Intell 2011; 23(4): 469-495.
[3] Khatib O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. Int J Rob Res 1986; 5(1): 90-98.

[4] Tiwari R, Shukla A, Kala R. Intelligent Planning for Mobile Robotics: Algorithmic Approaches. Hershey, PA: IGI Global Publishers; 2013.
[5] Kambhampati S, Davis LS. Multiresolution path planning for mobile robots. IEEE J Rob Automat 1986; 2(3): 135–145.
[6] Parker LP. Current research in multirobot systems. Artif Life Rob 2003; 7: 1-5.
[7] Sánchez-Ante G, Latombe JC. Using a PRM Planner to Compare Centralized and Decoupled Planning for Multi-Robot Systems. In: Proceedings of the IEEE International Conference on Robotics and Automation. IEEE; 2002. pp 2112 – 2119.
[8] Bennewitz M, Burgard W, Thrun S. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. Rob Auton Syst 2002; 41: 89–99.
[9] Bennewitz M, Burgard W, Thrun S. Optimizing schedules for prioritized path planning of multi-robot systems. In: Proceedings of the 2001 IEEE International Conference on Robotics and Automation. IEEE; 2001. pp 271 – 276.
[10] Holland JM. Designing Autonomous Mobile Robots: Inside the Mind of an Intelligent Machine. Oxford, UK: Elsevier; 2003.
[11] Álvarez-Sánchez JR, de la Paz López F, Troncoso JMC, Sierra DS. Reactive navigation in real environments using partial center of area method. Rob Auton Syst 2010; 58(12): 1231-1237.
[12] Sgorbissa A, Zaccaria R. Planning and obstacle avoidance in mobile robotics. Rob Auton Syst 2012; 60(4): 628-638.
[13] Kala R, Shukla A, Tiwari R. Fusion of probabilistic A* algorithm and fuzzy inference system for robotic path planning. Artif Intel Rev 2010; 33(4): 275-30.
[14] Selekwa MF, Dunlap DD, Shi D, Collins Jr EG. Robot navigation in very cluttered environments by preference-based fuzzy behaviors. Rob Auton Syst 2008; 56(3): 231-246.
[15] Tu KY, Baltes J. Fuzzy potential energy for a map approach to robot navigation. Rob Auton Syst 2006; 54(7): 574-589.
[16] Parhi DR, Mohanta JC. Navigational control of several mobile robotic agents using Petri-potential-fuzzy hybrid controller. Appl Soft Comput 2011; 11(4): 3546-3557.
[17] Motlagh O, Tang SH, Ismail N, Ramli AR. An expert fuzzy cognitive map for reactive navigation of mobile robots. Fuzzy Sets Syst 2012; 201: 105–121.
[18] Jolly KG, Kumar RS, Vijayakumar R. Intelligent task planning and action selection of a mobile robot in a multi-agent system through a fuzzy neural network approach. Engg Appl Artif Intell 2011; 23: 923–933.
[19] Ng KC, Trivedi MM. A neuro-fuzzy controller for mobile robot navigation and multirobot convoying. IEEE Trans Syst Man Cybern Part B: Cybern 1998; 28(6): 829-840.
[20] Yang X, Moallem M, Patel RV. A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation. IEEE Trans Syst Man Cybern Part B: Cybern 2005; 35(6): 1214-1224.
[21] Aguirre E, González A. Fuzzy behaviors for mobile robot navigation: design, coordination and fusion. Int J Approx Reason 2000; 25(3): 255-289.
[22] Innocenti B, López B, Salvi J. A multi-agent architecture with cooperative fuzzy control for a mobile robot. Rob Auton Syst 2007; 55(12): 881-891.
[23] Saffiotti A. The uses of fuzzy logic in autonomous robot navigation. Soft Comput 1007; 1(4): 180-197.
[24] Xu WL, Tso SK. Real-time self-reaction of a mobile robot in unstructured environments using fuzzy reasoning. Engg Appl Artif Intel 1996; 9(5): 475-485.
[25] Baxter JL, Burke EK, Garibaldi JM, Norman M. Shared Potential Fields and their place in a multi-robot co-ordination taxonomy. Rob Auton Syst 2009; 57(10): 1048-1055.
[26] Snape J, van den Berg J, Guy SJ, Manocha D. The Hybrid Reciprocal Velocity Obstacle. IEEE Trans Rob 2011; 27(4): 696-706.
[27] Chand P, Carnegie DA. A two-tiered global path planning strategy for limited memory mobile robots. Rob Auton Syst 2012; 60(3): 309-321.
[28] Kala R, Shukla A, Tiwari R. Robotic path planning in static environment using hierarchical multi-neuron heuristic search and probability based fitness. Neurocomputing 2011; 74(14-15): 2314-2335.
[29] Zhang L, Kim YJ, Manocha D. A hybrid approach for complete motion planning. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE; 2007. pp 7-14.
[30] Lu Y, Huo X, Arslan O, Tsiotras P. Incremental Multi-Scale Search Algorithm for Dynamic Path Planning With Low Worst-Case Complexity. IEEE Trans Syst Man Cybern Part B: Cybern 2011; 41(6): 1556-1570.
[31] Hou ESH, Zheng D. Hierarchical path planning with hexagonal decomposition. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics. IEEE; 1991. pp 1005-1010.

[32] Takahashi O, Schilling RJ. Motion planning in a plane using generalized Voronoi diagrams. IEEE Trans Rob Automat 1989; 5(2): 143-150.
[33] Hwang JY, Kim JS, Lim SS, Park KH. A fast path planning by path graph optimization. IEEE Trans Syst Man Cybern Part A: Syst Hum 2003; 33 (1): 121–128.
[34] Urdiales C, Bantlera A, Arrebola F, Sandoval F. Multi-level path planning algorithm for autonomous robots. IEEE Elect Lett 1998; 34 (2): 223–224.
[35] Chen DZ, Szcerba RJ, Uhran JJ. A framed-quadtree approach for determining Euclidean shortest paths in a 2-D environment. IEEE Trans Rob Automat 1997; 13 (5): 668–681.
[36] Yahja A, Stentz A, Singh S, Brumitt BL. Framed-quadtree path planning for mobile robots operating in sparse environments. In: Proceedings of the IEEE International Conference on Robotics and Automation. IEEE; 1998. pp 650–655.
[37] Vendrell E, Mellado M, Crespo A. Robot planning and re-planning using decomposition, abstraction, deduction, and prediction. Engg Appl Artif Intel 2001; 14: 505–518.
[38] Kuwata Y, Karaman S, Teo J, Frazzoli E, How JP, Fiore G. Real-Time Motion Planning With Applications to Autonomous Urban Driving. IEEE Trans Control Syst Technol 2009; 17(5): 1105-1118.
[39] Montemerlo M, Becker J, Bhat S, Dahlkamp H, Dolgov D, Ettinger S, Haehnel D, Hilden T, Hoffmann G, Huhnke B, Johnston D, Klumpp S, Langer D, Levandowski A, Levinson J, Marcil J, Orenstein D, Paefgen J, Penny I, Petrovskaya A, Pflueger M, Stanek G, Stavens D, Vogt A, Thrun S. Junior: The Stanford entry in the Urban Challenge. J Field Rob 2008; 25: 569–597.
[40] Clark C, Rock S, Latombe J. Dynamic networks for motion planning in multi-robot space systems. In: International Symposium on Artificial Intelligence, Robotics and Automation in Space. ESA; 2003.