On Repelling Robotic Trajectories: Coordination in Navigation of Multiple Mobile Robots

Citation: R. Kala (2018) On repelling robotic trajectories: coordination in navigation of multiple mobile robots. Intelligent Service Robotics 11(1): 79–95.

Final Version Available At: https://link.springer.com/article/10.1007/s11370-017-0238-5

Abstract: The paper attacks the problem of motion planning of a set of mobile robots. While artificial potential fields are the simplest methods of use, they are also locally optimal and can be easily stuck in scenarios. Probabilistic roadmap, elastic roadmaps, elastic strip and similar methods have a weak modelling of coordination between the robots. An inspiration is drawn from the artificial potential field method where the potential is computed in configuration space. In this paper the notion is extended to a 'trajectory space', where the complete trajectories of robots repel each other. With the added assumption of communication between the robots and higher computational costs, the resultant approach is near optimal and does not get the robot stuck or trapped. A variant of the algorithm with no direct communication is also presented. The method is experimented by using computer simulations and found to perform better over well-known approaches in the literature.

Keywords: Robot Motion Planning, Navigation, Multi-robot systems, Artificial Potential Fields, Elastic Strip.

1. Introduction

Robots are increasingly being used for a variety of home and office tasks. It is likely that very soon multiple mobile robots will actively be found in homes and offices. The most primitive capability of the robot is to be able to navigate from the current position to a desired goal position by avoiding obstacles, known as the problem of robot motion planning [1,2]. The presence of multiple robots in the same workspace results in many robots that need to avoid each other while navigating, known as the problem of multi-robot motion planning [3,4]. The problem is challenging due to the need to plan for many robots in near-real time, while catering to the needs of completeness and optimality.

This paper proposes a new method to solve the problem of multi-robot motion planning. The problem is to move N robots from their starting position to their goal position, such that no two robots collide with each other or with any static obstacle. The problem is given in figure 1. Let us fist assume a static environment. The complete specification of the robot's position and pose is called as the configuration of the robot. All the possible set of configurations of the robot constitute the configuration space ξ_{static}. The sub-set of all configurations wherein the robot does not collide with any static obstacle is called as the free-configuration space ξ_{static}^{free}. The compliment of the free configuration space is the obstacle-prone configuration space ξ_{static}^{obs}, denoting all configurations wherein the robot collides with any static obstacle. Thus $\xi_{static}^{free} = \xi_{static} \setminus \xi_{static}^{obs}$. The workspace is the physical space wherein the robots operate. The problem of single robot motion planning in static environment is to find a trajectory τ, such that $\tau(t)$ denotes the position of the robot at any time t in the configuration space. The trajectory must start at the source $\tau(0)=S$ and end at the goal $\tau(T)=G$, where T is the time at which the robot reaches the goal. The constraint is that the trajectory should be collision-free, that is $\tau(t) \in \xi_{static}^{free} \forall t$. The trajectory τ may have additional kinematic and non-holonomic constraints as per the robot model.

For multiple robots, the problem is to find the trajectory of all robots such that no robot collides with a static obstacle or any other robot. Let τ_i be the trajectory of the robot R_i, starting from its source $\tau_i(0)=S_i$ and ending at the goal $\tau_i(T_i)=G_i$. Here T_i is the time at which the robot R_i reaches its goal. The condition of non-collision is given by equation (1).

$$\tau_i(t) \in \xi_{static}^{free} \forall t, R_i, \tau_i(t) \otimes R_i \cap \tau_j(t) \otimes R_j = \varphi \forall t, R_i, R_j, R_i \neq R_j \quad (1)$$

Here the function $\otimes$ is used to convert a point in the configuration space to a collection of all points that the robot occupies in the workspace. The condition of non-collision between all robots is simply that the intersection of all points occupied by every pair of robots in the workspace should be an empty set. Here the

assumption is that all the robots can communicate to each other and exchange information. The easiest implementation is to have a central server that collects all the requirements from the individual robots, plans each of them, and transmits the motion controls to the robots for navigation. Alternatively, one of the robots may act as a central server, the robot being elected by the robots themselves.

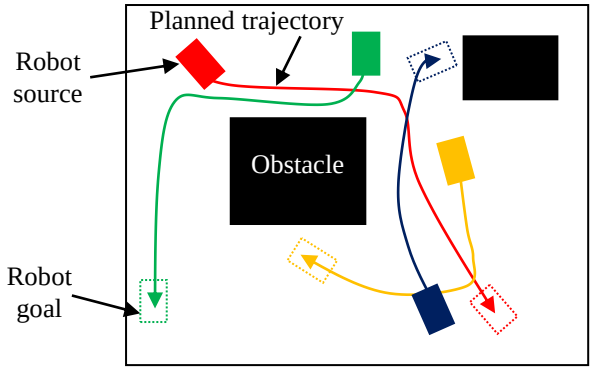


Fig. 1 Problem Description. Each robot needs to go from its current location marked as solid circle to its goal marked as a dotted circle, avoiding all other robots and obstacles on the way.

Another more practical variant of the problem is when the robots do not have any communication with each other, wherein the problem is robot motion planning in a dynamic environment. Here the robot may instead even refer to humans and other dynamic obstacles. Let $\xi_{dynamic}^{free}(t)$ denote the configuration space of a robot at time *t*, where the subscript dynamic states that the free-configuration space changes with time. The problem is to compute a collision-free trajectory τ, that is, $\tau(t) \in \xi_{dynamic}^{free}(t) \forall t$. The motion planning algorithms fall in two categories, reactive and deliberative. The reactive algorithms take the immediate action based on the current percept only and therefore avoid static and dynamic obstacles by continuously taking reflexive actions. The deliberative algorithms on the other hand attempt to find a collision-free and optimal trajectory by considering numerous options and computing ahead of time. Another mechanism is to first assume that the environment is static and plan for the same to get some trajectory. As the environment changes, the trajectory is continuously adapted. In the worst cases, the complete trajectory needs to be re-planned, when small corrections do not lead to a feasible trajectory. The more difficult mechanism is to predict the trajectory of the other robots, and use the same to construct a dynamic configuration space that is used for the actual motion of the robot. Again adaptation of the trajectory will be needed if there is a difference between the predicted and actual trajectories of the other robots. The simplest assumption is to assume constant speed profiles.

The common metrics to adjudge the optimality of the trajectory for the case of a single robot are path length, time to goal, average (or minimum) clearance and average (or minimum) smoothness. The clearance of a robot R_i at any time *t* is the distance from the nearest obstacle, and is a safety measure against sensing and actuation uncertainties. The clearance is given by equation (2), while the average clearance over the path length is given by equation (3).

$$C_i(t) = \min_{R_i, R_j R_i \neq R_j} \left(\left| \tau_i(t) \otimes R_i - W_{static}^{obs} \right|, \left| \tau_i(t) \otimes R_i - \tau_j(t) \otimes R_j \right|, C_{\max} \right) \quad (2)$$

$$C_i = \frac{\int C_i(t) \tau'_i(t) dt}{\|\tau_i\|} \quad (3)$$

The clearance is averaged over the trajectory length $\|\tau_i\|$. $C_i(t)$ is the clearance at time *t*, while C_i is the averaged clearance. $\tau'(t)$ is the robot speed. The trajectory is first mapped from the configuration space to the workspace using the function $\otimes$. Here $| a - b |$ denotes the minimum distance between any element of *a* and any element of *b*. C_{max} is the largest desired clearance. *W* is the workspace.

The metrics for multiple robots are the same as that for single robots, with the different metrics averaged with the number of robots. The metrics are hence average time to reach goal $\left(\sum_{i=1}^{N} T_i / N\right)$, average path length $\left(\sum_{i=1}^{N} \|\tau_i\| / N\right)$ and average clearance $\left(\sum_{i=1}^{N} C_i / N\right)$, where *N* is the number of robots. The cooperation metric

measures the magnitude by which one robot cooperates with other robot, sacrificing its good trajectory so as to allow another robot to go by using a near-optimal trajectory. It is natural that the optimal trajectories of all robots combined may not be a feasible plan in multi-robotics due to collisions between the robots. The robots must hence cooperate and give up their optimal trajectories. Consider $\tau^*_i(t)$ to be the optimal trajectory of the robot in the absence of any other robot, planned in static free-configuration space $\tau^*(t) \in \xi_{static}^{free} \forall t$. The cooperation is measured as the difference between the cost of the trajectory in a multi-robot setting with its optimal cost not considering any other robot, and is given by equation (4). The average cooperation $\left(\sum_{i=1}^{N} coop_i / N\right)$ and maximum cooperation ($\max(coop_i)$) becomes other indicators of optimality in the case of multiple robots.

$$coop_i = Cost(\tau_i) - Cost(\tau_i^*) \tag{4}$$

Here $cost(\tau_i)$ denotes the path cost of trajectory τ_i and can be taken as any of the metrics for the case of a single robot.

The key contributions of the paper are: (i) A potential inspired approach for speedy construction of roadmap is proposed which ensures high clearance, less number of points in the roadmap, and can handle narrow corridor problem of sampling based approaches. (ii) The concept of trajectories attracting and repelling each other is floated, which can further be extended into multiple problems and domains. (iii) The computation of overall optimal plan, given the optimal plan of individual trajectories, is computationally less expensive as compared to the standard optimization techniques. This is due to the trajectory attracting and repelling heuristics. (iv) The planner is a fundamentally new approach to coordination between robots as opposed to the other approaches in the literature. The approach is near-optimal, near-complete, with small execution times for the case of multiple robots. (v) The planning framework is extended and studied for the case with no communication between the robots, predicting the moves of the other robots.

2. Background

One of the most popular planning technique for the navigation of multiple mobile robots is Artificial Potential Field [5], wherein every robot is positively charged, every obstacle is positively charged, and the goal is strongly negatively charged. This causes a repulsion between every pair of robots, repulsion between robots and the obstacles, and attraction between the robots and their respective goals. As a result the approach can naturally avoid other robots and other dynamic obstacles, and work in partially known environments. However, the approach faces problem of equi-potential regions, wherein all forces cancel each other out, causing no motion for the robot. Random small perturbations can help escape small local minima, however the robot may get trapped in complex environments. Further, the approach causes robots to oscillate in narrow corridors. The other robots act as complex obstacles that cannot be easily surpassed. A set of robots could trap each other, while the number of equi-potential points significantly increase with the number of robots due to complex ways in which multiple robots combat each other, many times during the motion. The approach is neither complete, nor optimal. However the parameters can be tuned so as to show local-optimality [6-8], and the algorithm can be fused with a deliberative technique to add some optimality and completeness [9-10]. The fusion uses a deliberative planner to plan for a static map, while the dynamic obstacles are reactively avoided by using artificial potential field. The fused approach is still neither optimal nor complete, as the deliberative planner does not have information about the other dynamic obstacles while planning. Overall, the problem with the potential approach for multiple robots is assuming other robots as static obstacles. The approach does not benefit form communication between the robots, if available.

The Probabilistic Roadmap [11-12] is a deliberative planning technique that represents the complete configuration space as a roadmap consisting of vertices and edges in free space. The approach samples out vertices from free configuration space and attempts to connect the vertices by edges, if a collision-free trajectory is found between the neighbouring vertices. The roadmap can then be used for an online planning of a single robot, in which case the approach is probabilistically complete and probabilistically optimal, meaning that the probability of being optimal and complete tends to one as the number of samples tend to infinity. Multiple robots may be planned by either applying a centralized technique on the roadmap, or planning each robot separately using prioritization. Various methods are proposed for effective generation of the roadmap. In a previous work the author used the biased and heuristic exploration of the Rapidly-exploring Random Trees algorithm to produce a roadmap [13]. Adaptive roadmap [14-15] technique continuously adapts the roadmap for changing environment. Elastic roadmap [16-18] model each edge as an elastic strip that adapts with the changing environment as per the elastic properties to deal with dynamic obstacles.

The planning for multiple robots may be done in a centralized or a decentralized manner [19-20]. The centralized approach considers all the robot configurations and all their possible actions simultaneously, while simultaneously searching for trajectories for all the robots considering all possible robot interactions. So the configuration of the system is the joint configuration of all robots, while the configuration space is the cross product of the configuration spaces of the individual robots. The search operates in such a space and is therefore very computationally expensive and realizable for a few robots only. The decentralized approach, on the contrary, plans for every robot individually, making strategies for avoiding collisions if the plans of the individual robots are found to be collision prone. It may be difficult to have a consensus between the robots [21]. The approach can handle a large number of robots. One of the common decentralized planning approach is prioritization [22-23], wherein every robot is given a priority and a lower priority robot adjusts its plan in case of a possible collision with a higher priority robot. In prioritization, the robot with the highest priority clearly follows its optimal route and does not cooperate with the other robots that may have to adjust their plans by a large amount. Prioritization hence has a very poor cooperation between the robots. The path velocity decomposition scheme [24] breaks the problem into path planning that is done independently for all robots. Thereafter the speed of the robots are planned so as to avoid collision. The path cannot change at this phase and therefore the optimality is compromised.

Sometimes the paths may be locally-corrected to avoid collisions between the robots. The local search may be centralized, decentralized or co-evolutionary in nature [25]. Such techniques however attempt to deviate the entire trajectory for all possible robots. Hence a significant amount of computation is wasted, and very little computation is put to deviate the trajectory of colliding robots, and only the colliding regions of the trajectory. Having some heuristic means embedded into this operation can hence improve the operation significantly. This is the focus of the paper.

Another mechanism to correct the robot trajectory catering to dynamic obstacles is Elastic Strip [26]. Here the trajectory is modelled as a string made of an elastic material that has external forces applied by obstacles and other robots, causing a continuous deformity in the shape so as to always make the string away from the obstacles. Similarly the string has internal forces that causes it to contract to the shortest possible length. Together the forces cater to avoiding obstacles while being aligned at reasonable small path lengths. Again the robots and obstacles are seen as static obstacles that unnecessarily continuously deforms the shape of the strip, while consuming computation. There is no benefit from communication between the robots, if available. Sometimes, two robots may have opposing means to avoid each other for which a consensus cannot be reached without communication. This paper attempts to modify the same algorithm for better deformation of the trajectory using heuristics, while benefiting from communication or known trajectories of the robot, if available.

3 Related Works

The proposed work is inspired from the Artificial Potential Field method. That said, the proposed approach is not an incremental extension of Artificial Potential Field. The Artificial Potential Field is largely a reactive approach while the proposed algorithm is to some extent deliberative in nature. Further the Artificial Potential Field is usually applied in the workspace/configuration space, while the proposed method is applied to a new space called as the trajectory space. Hence the two methods are very different in operation. A lot of work exists in the literature for motion planning of single and multiple robots using potential fields. Baxter et al. [27] accounted for the sensing errors, attempting to correct the perception of one robot by the perception of the other robot. The perception between robots was fused, which was then used for computing the potentials. In another related work Vadakkepat et al. [28] optimized the parameters of the artificial potential field using multi-objective optimization in order to make the approach locally near-optimal in nature. The work of Tu and Baltes [29] used fuzzy arithmetic to compute the potential values to be used for the navigation of the robot. The approaches however face all general problems noted for potential fields.

Typically, the potential field approach assumes the other robots, obstacles and target to be stationary. In a different vein, Ge and Cui [30] accounted for the speeds of the obstacles and targets, and used the same to land at the target with the desired speed. Jaradat et al. [31] extended the approach to use fuzzy inference system for the computation of the potentials. Yin et al. [32] further accounted for the accelerations of the target and obstacles for the computations. Chang and Yamamoto [33] adapted the approach to be used for the problem of robot exploration, wherein a robot is used to get a Voronoi map of the environment. Masoud [34] used the harmonic potential field and made a mechanism to directly convert the gradient values to the motor control signals using a synchronization mechanism. Lee et al. [35] specifically looked into the problems of equi-

potential regions when the obstacle is in-between the robot and the goal, and solved the problem by introduction of new sources of potential. The approaches solve some of the problems associated with potential field, however do not effectively account for coordination mechanisms between robots.

Another popular approach for reactive robot navigation is the Fuzzy Inference System. To ease and automate the task of design of a fuzzy rule base, Motlagh et al. [36] used Cognitive Maps for reactive planning of a single mobile robot. The authors accounted for robot slippage as an input which meant the model could adapt to actuation uncertainties. Selekwa et al. [37] modelled multiple behaviours for navigation of a robot, where each behaviour was a fuzzy system. The resultant motion of the robot was a result of integration of different behaviours. Kala et al. [38] showed the mechanism of using a probabilistic version of A* algorithm for guiding a robot by using a fuzzy inference system. This solved the problem of robot being struck and made the path near optimal. Hank and Haddad [39] planned a reference trajectory using a Random Profile Approach method and then made the robot follow the trajectory for constraints using fusion of obstacle avoidance and goal seeking fuzzy behaviours. As a result of the approaches, the system has some optimality, however again does not explicitly account for the coordination aspects between robots. Other methods of reactive planning include case based reasoning [40], geometrical approaches [41], neural networks, velocity obstacles, etc.

Soloveyet al. [42] solved the problem of multi-robot motion planning in a centralized mechanism. In order to reduce the computational complexity, the motions were restricted to a discrete set of options represented in a graph. This significantly reduced the computation time, although the centralized nature restricted its application to a very high number of robots. An earlier work from the author [43] gave a basic notion that coordination can be achieved by iterative modification of robot trajectories. The work neither used any heuristic in the selection of the trajectories to modify, nor in the modification of the trajectories itself. In fact the initial trajectory generation of the method was a naïve PRM, without considering the potential heuristics in the placement of the samples. With the same work as a basis the proposed algorithm develops the potential heuristic at the stages of coordination. Saska et al. [44] solved the problem of multi-robot motion planning in the specific context of formation control. First an optimization based motion planning was done for a virtual leader, based on which the follower trajectories were derived. The Model Predictive Control mechanism was used for the motion of the robots. The problem was for a simpler context of formation control with relatively same trajectories.

Indelman [45] also solved for multi-robots, while projecting the robots into a belief space and using joint observations of multiple-robots for probability updates of the belief space. Again the approach is computationally expensive due to operations at belief space level, while much projection into the future is not possible. A number of approaches also use Partially Observable Markov Decision Process for the problem of navigation, for which a lot of strategies have been tried. The process is time consuming and therefore Lee and Kim [46] proposed the use of GPU for the computation. Similarly, Feyzabadi and Carpin [47] used hierarchies to solve the same problem. The states were clustered to make a coarser transition system for the same. Wu et al. [48] solved the problem in a decentralized manner for agents that have bounded communication. The decentralized nature enabled effective online learning of a policy. The agents maintained a belief state and updated based on the histories and consistency with observations. Overall, the approaches are computationally expensive, unless some heuristic or discretization is used which leads to a sub-optimal performance.

Kim et al [49] solved the problem navigation of multiple robots using potential field, whose output was later corrected by using a fuzzy controller. A hybrid of both approaches made several robots reach their goals. Das et al. [50] solved the problem of multi-robot motion planning from an optimization perspective. The authors optimized for every step's movement of all the robots, while every obstacle exerted a force on the other robots that led to robots avoiding each other. Hoy et al. [51] solved the problem of multi-robot navigation in partially known environments. The motion planning is done for the part of the known map, while the robot is also controlled to avoid the obstacles around. A Model Predictive Control Scheme is used for the same. Further, Charalampous et al. [52] showed the navigation of an autonomous vehicle in an unknown area, wherein the vehicle made its own map, localization and motion planning. Support Vector Machine was used for trajectory planning.

Based on these studies it can be observed that much thrust is yet to be put into planning of multiple robots inspired by reactive planning techniques in complex obstacle frameworks, in a manner such that the robots can interact in complex ways. Rather much thrust is on empowering the methods to be near optimal and capable of coping up with uncertainties. While deliberative techniques may be computationally too costly, the paper attempts to seek mechanisms to make the reactive planning near optimal and near complete by effective addition of some deliberation in planning.

4 Algorithm

4.1 Basic Concept

The basic inspiration behind Artificial Potential Field method is that every obstacle applies a repulsive potential and the goal applies an attractive potential to a charge, which hence when let free from the source can get to the goal. However the same motivation cannot be directly extended to multiple robots or charges. If a large set of similar charges are released, each 'somehow' attracted towards its own goal; one would expect many of these charges to come to a standstill, each repelling one other producing an equi-potential point. Even if perturbed in a random direction, the possibility of getting re-struck (maybe at a worse situation) is large. Hence a completely nature-inspired approach may not be the best course of action.

Hence a different (and slightly more abstract) analogy is used. Assume a collection of charges are onboard, each repelled by each other and the static obstacles around. Each charge has an additional preferential position which attracts it (and no other charge) by a force directly proportional to distance. Consider an initial configuration where the charges are all at their preferential positions. These charges move as per the defined dynamics and soon come to a rest at the equi-potential point in such a system. It can be argued that the system so defined results in maximum separation between the charges and obstacles with the least deviation from the preferred positions.

In a physical sense, the algorithm treats the charges as current trajectory of the robot τ_i while the preferential positions are the individual (single-robot) optimal trajectories τ^*_i. The space is hence a trajectory space, where each point corresponds to a trajectory while the static obstacles correspond to a trajectory whose positions does not change with time. The maximum separation objective is equivalent to maximizing clearance, while the least deviation objective is equivalent to least cooperation, which in turn implies close to optimal path length and time to destination. While in general this is simple to understand, the obvious questions are: What is a trajectory space? How do trajectories repel and attract? What constitutes equi-potential points of trajectories? How do the trajectories reach their equi-potential points? The general notion is given by figure 2. Figure 2 is a hypothetical space where the complete trajectory of the robot, including both the special and temporal components, is represented by a point in this space. So every point in this space should be visualized as the robot going from the source position to its goal position by some speed. Any slight deviation in the path or speed, moves the trajectory as a point in this space. Some points in this space are marked as obstacles, because the trajectory will not obey the kinematic constraints of the robot.

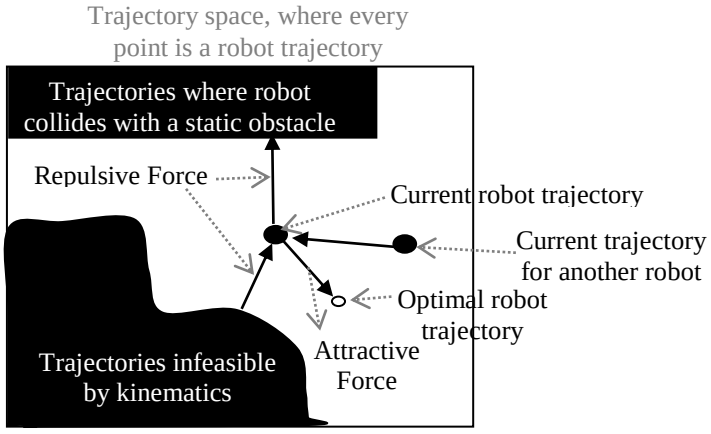


Fig. 2 Motion of robotic trajectories in a trajectory space. Every point in the space represents a trajectory. Static obstacles have static trajectories which cannot move in space. Every robotic trajectory is moved such that it is repelled by the trajectories of the other robots and obstacles, and attracted by its own optimal trajectory computed in the case of absence of other robots. The trajectory motion stops on reaching the equi-potential point, giving the optimal plan.

4.2 Overall Design

The problem of trajectory planning for multiple robots is decomposed into two independent problems. The first problem deals with the computation of optimal trajectory $\tau^*_i(t)$ of a robot R_i. The second problem is to coordinate the various robots to compute the overall optimal plan τ. A roadmap based approach is proposed for the first step, where the construction of the roadmap is inspired from the potential field approach. For the second problem the notion of attraction and repulsion of trajectories is floated, which happens iteratively till the stopping criterion is not met. Re-planning may be required if a set of robots cannot reach the goal or cannot avoid collision. This is done by altering the roadmap and re-planning the specific robots. The algorithm is shown in figure 3.

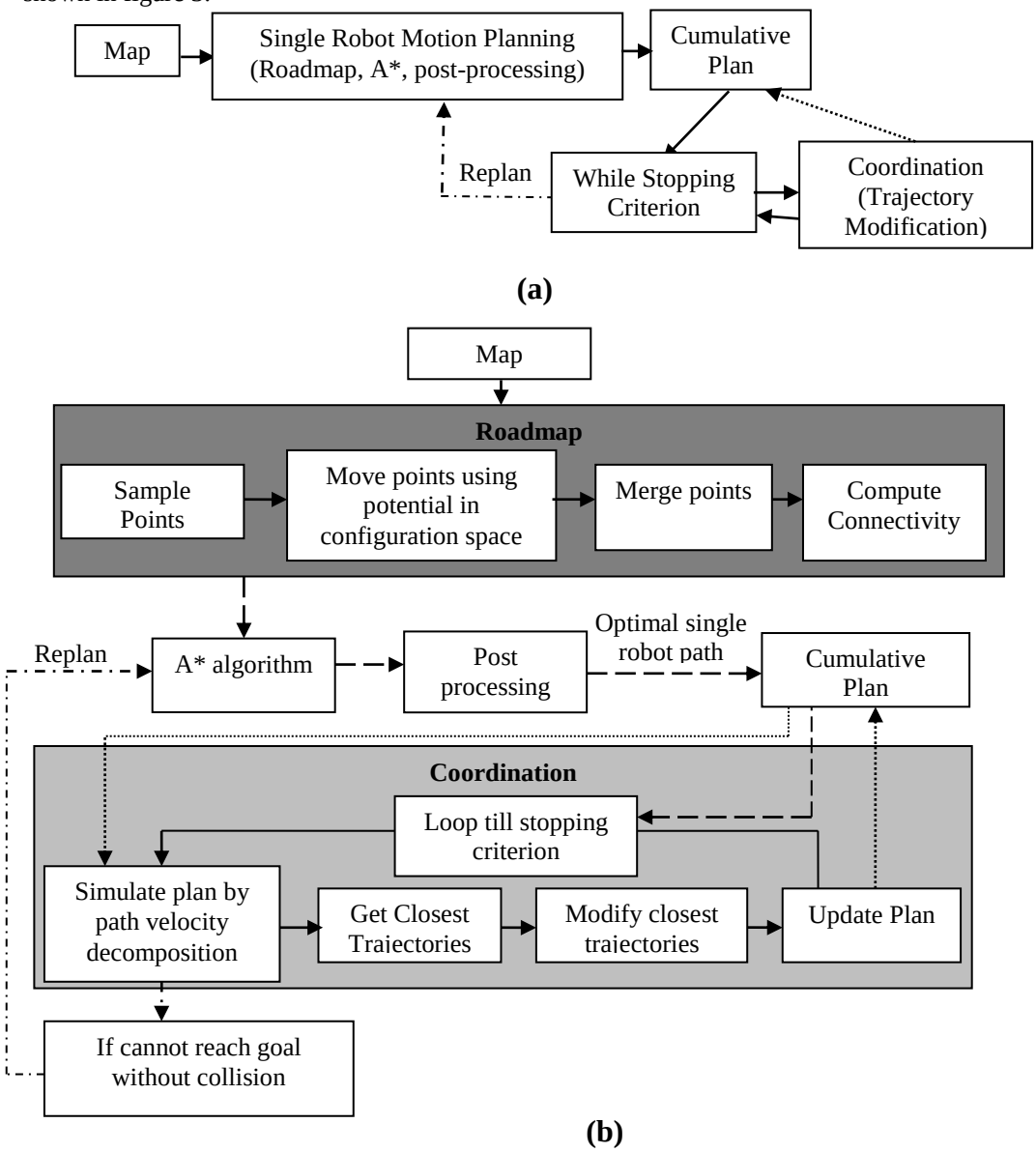


Fig. 3 General structure of the algorithm. (a) Simplified (b) Detailed. The algorithm first samples the map into a roadmap, where each sample is moved away from the obstacles using a potential function. The roadmap is used to develop an initial plan which consists of all optimal trajectories of robots which were computed without considering the other robots. Coordination phase identifies the closest trajectories in trajectory space in the plan, and modifies the same using repulsion forces by other obstacles and trajectories and attraction forces by owns optimal trajectory. Each coloured box represents a module, and the logic flow within the module is given by a solid line. The normal logic flow between modules is given by a dashed line. The ".-" line scheme is used for an

erroneous flow of the logic, which links it to the corrective step. The dotted style of line represents the database style query/update relations.

4.3 Computing single-robot optimal trajectories

The first task is to setup a roadmap considering the static configuration space ξ_{static}. $|M|$ points are randomly sampled, where M denotes the roadmap. Each of these points is a potential robot position and hence such points need to simultaneously assure high clearance and small path lengths. By analogy of human motion we know that the attempt to maximize clearance is done by a decent amount if there is a scope, otherwise the increase in clearance is done to the maximum permissible value. All points within obstacle prone static configuration space ($M_i \notin \xi_{static}^{free}$) are randomly moved by small amounts till they reach a point in free static configuration space. Hence each point M_i is in the free configuration space ($M_i \in \xi_{static}^{free}$) and is acted upon by repulsion from all neighbouring obstacles until the point reaches an equi-potential point or the resultant force acting on the point is not large. Repulsion ensures points lie far from obstacles, while taking only significant magnitude eliminates large motion of the point which could make the points in the roadmap too far from obstacles resulting in a loss of optimality.

Consider the point M_i. The net force F acting at the point is given by equation (5). Unlike the case of robot, the points are virtual agents with no inertia. Hence motion of the particle is given by equation (6).

$$F = \sum_{\theta} -\frac{1}{r^2(\theta)}.u(\theta) \tag{5}$$

$$M_i \leftarrow M_i + \alpha \max\left(\|F\|, F_{\max}\right)u(F) \tag{6}$$

$r(\theta)$ is the distance of closest obstacle from the robot in direction θ. $u(\theta)$ is the unit vector in direction θ. In this approach 8 uniformly apart values of θ are taken. α is the scaling factor which relates force to the magnitude of displacement. Sometimes an obstacle very close to the robot can cause a very large displacement which is undesirable especially due to the discritized nature of time and motion. The maximum force is hence given a threshold of F_{max}, which states the maximum distance a point may move in a unit time. $u(F)$ is a unit vector in direction F.

Large roadmaps increase planning (and re-planning) time, which makes it important to only have potentially useful points in the roadmap. Hence if any pair of points (M_i, M_j) are closer by a distance of η, the point is replaced by the mid-point of the two points given by equation (7).

$$M \leftarrow \left(M - \{M_i, M_j\}\right) \cup \frac{M_i + M_j}{2}, \|M_i - M_j\| < \eta, M_i, M_j \in M, M_i \neq M_j \tag{7}$$

Here η is the distance threshold for connection between any two points. Equation (7) when iteratively applied over the roadmap, gives the final roadmap such that equation (8) is valid.

$$M : \|M_i - M_j\| \geq \eta \forall M_i, M_j \in M, M_i \neq M_j \tag{8}$$

A* algorithm is used to compute the optimal path for every robot using this roadmap. The cost functions of the A* algorithm is given by equation (9).

$$g(b) = \begin{cases} 0 & b = S_i \\ g(a) + \dfrac{\|a-b\|}{vel_i^{\max}}.pen(C(b)) & b \neq S_i \end{cases} \tag{9}$$

$$h(b) = \frac{\|b - G_i\|}{vel_i^{\max}}$$

$$f(b) = g(b) + h(b)$$

Where (a, b) are adjacent edges of the roadmap. $g(b)$, $h(b)$ and $f(b)$ denote the historic, heuristic and total costs. vel_i^{max}, S_i and G_i denote the maximum speed, source and goal of robot R_i. $C(b)$ denotes the clearance at point b.

pen() returns the penalty for clearances lower than the maximum threshold. In the current implementation the penalty linearly decreases (minimum unity) with an increase of clearance.

For every robot R_i, A* algorithm returns a path τ_i^A which is optimal as per the given roadmap space *M*. τ_i^A is converted to optimal path τ_i^* after little post processing which includes path smoothening by the use of a cubic spline, deleting some points and adjusting positions of some points.

5 Trajectory Space

5.1 Realizing the space.

Consider a robot R_i with a trajectory τ_i . As per definition the position of R_i at any time *t* is given by $\tau_i(t)$ with $\tau_i(t)$=*undefined* for all $t \geq T_i$ (the travel time of the robot). Consider a space ς which has a dimensionality of $|\xi|.T^{\max}$. Here $|\xi|$ denotes the dimensionality of configuration space ξ and T^{max} is practically the maximum time that any robot may be in motion. Hence τ_i represents a single point in this space for $0 \leq t \leq T^{max}$. Consider τ_o to be the mapping of the obstacle *o* in configuration space to the trajectory space which may be easily given by $\tau_o(t) = o : o \in \xi_{static}^{obs} \forall t$.

In this manner all robot trajectories and obstacles represent points in the trajectory space ς. The next task is to make robot trajectories repel from each other and the obstacles, and to make them attract to the optimal trajectories. It should hence be simple to carry the attractions and repulsions. However such attractions and repulsions do not have any physical interpretation. Further, if directly applied, two trajectories such that robots collide with each other may have large distances if distances are measured as Euclidian norms in this trajectory space. Hence the space cannot be visualized by a Euclidian equivalent.

The easiest way to handle the issue is to map the problem of attraction and repulsion of trajectories in ς into an equivalent problem in ξ, apply the modification in ξ, and to re-map the effects in ς.

5.2 Path Velocity Decomposition

The problem of navigation planning of multiple robots may be easily broken into the problem of path planning and velocity planning [24]. The potential method in this case is only used for path planning, while a heuristic technique is used to compute the velocity. This decision goes from the fact that forces would be applied in ξ, in which case there is no way to specify the time of arrival of the robot at a position. It is possible to fix the speed to some value, which however can only sub-optimally solve the problems where the robots can avoid each other with some fixed speeds.

Consider a plan τ where the trajectory of robot R_i is given by τ_i. Let us assume that at time *t* the position of the robot is $\tau_i(t)$ while its current speed is v_i. To compute the immediate speed of the robot R_i, it extrapolates the motion of all robots including itself with the assumption that all the robots continue to travel as per their trajectories in plan τ (considering path only) with speed equal to the current speed. The distance to first collision is measured. Let the distance as measured by R_i on extrapolating other robots motion be d_i. The preferred speed is the maximum speed the robot may possess such that it would be able to stop with the maximum acceleration to avoid the collision intimated by the extrapolation. This form of speed assignment ensures that the robot travels with the maximum speed possible at all times, and in case the plan is collision prone, the robot slows or even stops to avoid collision. This is given by equation (10).

$$v_i^{pref} = \sqrt{2.acc_i^{\max} \max(d_i - s(v_i), 0)} \tag{10}$$

Here $acc_i^{\max}$ is the maximum acceleration and $s(v_i)$ is the safety distance which must be maintained. Equation (10) measures the preferred speed assuming a constant maximum retardation of $acc_i^{\max}$ for an available stopping distance of d_i such that the robot stops at a distance of $s(v_i)$ before the prospective collision. The safety distance to be maintained should naturally be large for large speeds to accommodate for uncertainties, while at smaller speeds the factor can be less. This is hence given by equation (11).

$$s(v_i) = k.v_i \tag{11}$$

Here *k* is the factor that scales the speed to maximum safety distance required.

The immediate speed of the robot may not be updated to v_i^{pref} due to speed (0, $vel_i^{\max}$) and acceleration ($-acc_i^{\max}$, $acc_i^{\max}$) constraints, in which case the closest possible speed is chosen. Note that with this heuristic the robots may not trace the complete trajectories but may end up waiting indefinitely with a zero velocity.

5.3 Attracting and Repelling Trajectories

Distance between any two points τ_i and τ_i in ς is defined as the least distance observed in corresponding ξ when the robots/obstacles follow their respective trajectory in presence of all other robots/obstacles constituting the overall plan τ . This is given by equation (12).

$$\left|\tau_i - \tau_j\right|_{\varsigma} = \min_t\left(\left|\tau_i(t) \otimes B_i - \tau_j(t) \otimes B_j\right|\right) \tag{12}$$

Here B_i denotes the robot R_i or the obstacle O_i, whichever the case may be. The norm $|.|$ is returns the distance between the closest points of the robots/obstacles.

The algorithm finds the robot pairs (R_i, R_j) such that $\left|\tau_i - \tau_j\right|_{\varsigma}$ is the least for all pairs of robots that constitute the plan τ . An attempt is made to modify the trajectories τ_i and τ_i in ξ such that the distance between them increases in ς . Let *c* be the instance of time when the robots record the least distance. The robots can take zero speeds and hence it is necessary to record the first instance when the least distance is observed. Let the positions of the robots at the least distance be $\tau_i(c)$ and $\tau_j(c)$. Any other robot R_k also has a position of $\tau_k(c)$ at the time instance.

The aim is to increase $\left|\tau_i - \tau_j\right|_{\varsigma}$ .The best way to do so is to modify the segment of trajectory τ_i around the point $\tau_i(c)$ and trajectory τ_j around the point $\tau_j(c)$. The trajectories cannot be made to repel in ς which means that the problem needs to be transformed into ξ. In ξ the mechanism to repel (or attract) a point from another set of points is known (as we shall also see later). This notion extended to define the operation of repulsion (or attraction) of a trajectory from a set of points in ξ. A trajectory (say τ_i) represents a continuous set of points in ξ. Let us uniformly sample the trajectory along some number of points. Each such point $\tau_i(s)$ may be made to attract or repel and thus move in ξ which gives a new position of $\tau_i(s)$. This excludes the source and the goal which are fixed. The trajectory may be reconstructed using the new values of $\tau_i(s)$. It must be noted that while *c* denotes the specific time of closest distance of the robot following the trajectory, *s* is a sample taken out of the trajectory whose movement results in modification of the trajectory.

In order to further extend the notion to repulsion (or attraction) of two trajectories (say τ_i and τ_j) one of the trajectories (say τ_j) needs to be defined as a static obstacle in a static configuration space ξ from which the other trajectory may be repelled. The task is hence to represent the entire trajectory τ_j as a point or a set of points in ξ from which the entire trajectory τ_j would be repelled (or attracted). The best manner of doing this is to have a static obstacle (representing τ_j) placed at $\tau_j(c)$ (along with the other robots acting as obstacles at time *c*), where the two trajectories record their minimal distance of $\left|\tau_i - \tau_j\right|_{\varsigma}$. The entire trajectory τ_j hence attempts to shift away from $\tau_j(c)$ which may or may not be possible depending upon the presence of other obstacles or forces.

The concept can be understood with an analogy of driving vehicles in open fields. Imagine driving a vehicle in an open field where you get close enough to a particular vehicle. If the entire travel plan could be reset, in the next run knowing that the other vehicles would follow almost similar paths, you would like to keep away or keep an additional distance to the point of closest distance as far as possible. This mostly affects the path near the minimum distance path, which is also true for the potential methods where the potential fades away with distance. The other vehicle might do the same, correcting in the opposite mechanism to still further increase the distance. Hence while deciding your motion, all you know about the other vehicle's motion is its position at the closest meet. If the travel could be further reset, you might attempt to increase the distance from some other vehicle, or further increase the distance from the earlier encountered vehicle.

The heuristic to increase $\left|\tau_i - \tau_j\right|_{\varsigma}$ by modification of τ_i and τ_j is simple. First treat R_j as obstacle placed at $\tau_j(c)$ and modify τ_i. Then treat R_i as obstacle placed at $\tau_i(c)$ and modify τ_j. Imagine the motion of the robots as per

plan τ which lead to the least distance $\left|\tau_i - \tau_j\right|_\varsigma$ between R_i and R_j. In this plan, for the motion of R_i, the robot R_j acts as an obstacle and attempts to push the robot far away, while the reverse is true for R_j. The combined effects lead to an increased least distance $\left|\tau_i - \tau_j\right|_\varsigma$, if it was possible to increase the same.

5.4 Modifying the trajectory

Consider the trajectory τ_i of the robot R_i. A static obstacle map in the configuration space is constructed which consists of all static obstacles and the robot positions at time c as additional obstacles. The obstacle-free map is given by equation (13).

$$\xi_{static}^{free}(c, R_i) = \xi_{static} / \left(\xi_{static}^{obs} \cup \left\{q : \tau_i(c) \otimes R_i \cap \tau_j(q) \otimes R_j = \varphi \forall j \neq i\right\}\right) \tag{13}$$

The trajectory τ_i is sampled along a few points. Let $\tau_i(s)$ be any such point. This point is moved in the configuration space using potential field. Every static obstacle and robot obstacles contributes repulsive force given by equation (14).

$$F_{repel} = \sum_\theta -\frac{1}{r^2(\theta)}.u(\theta) + \sum_{j \neq i} -\frac{1}{\left\|\tau_j(c) - \tau_i(s)\right\|^2}.u\left(\tau_j(c) - \tau_i(s)\right) \tag{14}$$

The first factor is a repulsive force from the sampled directions (θ) measured from the point $\tau_i(s)$, while the second factor is the repulsive force from other robots with any general robot R_j placed at $\tau_j(c)$. 8 uniformly apart values of θ are taken. $u()$ denotes a unit vector in the stated direction or in the direction of the stated vector.

The points are also subjected to an attractive force, which in trajectory space corresponds to a trajectory being attracted towards the optimal trajectory τ_i^*. The point (say $\tau_i^*(s)$) in τ_i^* which lies closest to the point $\tau_i(s)$ is computed. The point $\tau_i^*(s)$ then attracts $\tau_i(s)$ by a force which increases with the increase of separation given by equation (15).

$$F_{att} = \left(\frac{\left\|\tau_i^*(s) - \tau_i(s)\right\|}{A}\right)^2 .u\left(\tau_i^*(s) - \tau_i(s)\right) \tag{15}$$

Here A is the scaling factor of the attractive force, which also decides the relative contributions of the attractive and repulsive forces.

A common scenario in multi-robotics is to have two robots heading towards each other, in which case both of them apply repulsive forces on each other and attempt to move each other backward, until both come to a standstill at the equi-potential point. While the problem may be difficult to solve in the case of lack of communication or a common navigation law for robots, in this case another heuristic is placed in the form of an additional force. For every robot R_j a repulsive force is applied in the direction of $\tau_j(c)$-$\tau_i(s)$. An additional force is applied in the direction perpendicular to this force with a magnitude proportional to this force, only if the robot R_j lies anywhere 'ahead' of the robot R_i. 'Ahead' covers all regions which subtend an angle in the range (-π/2, π/2) from the current robot's orientation. Two perpendiculars are possible, clockwise and anti-clockwise. If the robot R_j subtends an angle in the range [0, π/2), the robot R_i should be looking to avoid R_j by drifting rightward (relative to R_i) and hence clockwise perpendicular is chosen. It may be verified that robots always choose opposing manners of avoiding each other. If the angle subtended is in the range (-π/2,0), anti-clockwise perpendicular is chosen. This force is given by equation (16).

$$F_\perp = \sum_{j \neq i} -\frac{1}{\left\|\tau_j(c) - \tau_i(s)\right\|^2}.\cos(\phi)u_\perp\left(\tau_j(c) - \tau_i(s)\right) \tag{16}$$

$u_\perp(\,)$ denotes the unit vector perpendicular to the mentioned vector. ϕ is the angle subtended by R_j from direction of orientation of R_i. cos(ϕ) measures the similarity of the applied scenario to the most unlikely scenario of two robots facing each other in which case the maximum force is applied.

The total force is given by equation (17) using which the point may be displaced using equation (18).

$$F = F_{repel} + F_{att} + F_{\perp} \quad (17)$$

$$\tau_i(s) \leftarrow \tau_i(s) + \alpha \max(\|F\|, F_{\max}) u(F) \quad (18)$$

α is the scaling factor which relates force to the magnitude of displacement. The maximum force is given a threshold of F_{max}, which states the maximum distance a point may move in a unit time. $u(F)$ is a unit vector in direction F. Smoothing over the new trajectory is applied by the use of a cubic spline.

It should be noted here that the point $\tau_i(s)$ in a single iteration cannot be moved by large amounts. This is due to the fact that the entire computation is based on the evidence of distance $|\tau_i - \tau_j|_\varsigma$. If any point in the trajectory of any robot before time c is altered, it is evident that the robot R_i would not reach the point $\tau_i(c)$ at time c as per the altered plan but at a different time (depending upon whether its trajectory before the point c was shortened or elongated and whether the other robots now come in R_i's way or clear R_i's way). In order to excessively increase the distance $|\tau_i - \tau_j|_\varsigma$ only assuming R_j as a static obstacle at $\tau_j(c)$, one may instead get the robot R_i closer to some other robot at some other location. Making small changes ensures the robot does not go too close to some other robot, and if it goes slightly the point of focus in next iteration may be shifted and points modified to increase the shortened distance.

5.5 Trajectory Motion

The algorithm is iterative, whereby in every iteration two closest trajectories move such that their distance increases (in the trajectory space). Hence the sight in the trajectory space would be the trajectories slowly drifting apart from each other, till they all enjoy a comfortable distance. In the configuration space every iteration marks a point where two robots get very close to each other and then modification of their trajectories makes the distances large. Hence points of near collisions as per optimal paths, would soon become less collision prone and later distance would further increase. Soon some other point between any other pair of robots might have the smallest distance, which then becomes the point of work and trajectories are modified to increase the distance even further. Hence as iterations proceed, trajectories start placing the robots as far apart as possible. This could be stopped after pre-specified number of iterations or when the trajectories stop getting affected signifying their positions at equi-potential regions. The equi-potential region denotes the region where all the forces applied at the snapshot recording the minimal distance between any two robots cancel each other out, which would happen once the trajectories are fairly wide apart, or as much wide apart as possible in the case of narrow regions. The concept is given in figure 4. The flowchart of the algorithm is given in the coordination module of Figure 3(b).

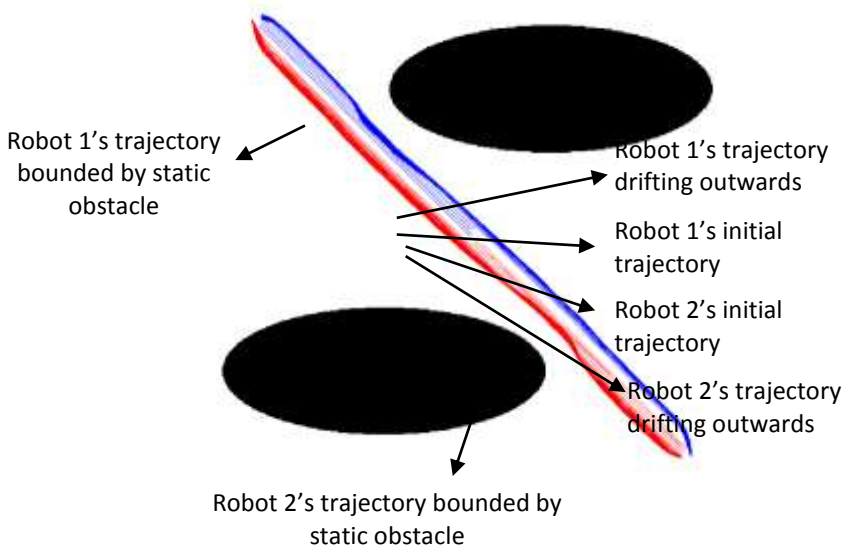


Figure 4: Robot coordination. As per initial trajectories both robots stop at the centre and hence do not trace the complete trajectories. Along with iterations, both trajectories drift towards their left by each other's repulsion. The drifting is bounded by repulsion from the static obstacle and attraction from owns optimal trajectory.

The algorithm is given below.

Coordinate(ξ_{static}, $\tau_i^* \forall i$)

$\tau \leftarrow \bigcup_i \tau_i^*$
while stopping criterion is not met
simulate τ by path velocity decomposition heuristic
compute $\min_{i,j}\left(\left|\tau_i - \tau_j\right|_\varsigma\right)$ and corresponding time c
compute $\tau_k(c) \forall c$ which act as obstacles
for all $\tau_i(s)$ in sampled τ_i, compute new $\tau_i(s)$ by equation (18)
$\tau_i \leftarrow$ reformulated trajectory from all $\tau_i(s)$
for all $\tau_j(s)$ in sampled τ_j, compute new $\tau_j(s)$ by equation (18)
$\tau_j \leftarrow$ reformulated trajectory from all $\tau_j(s)$
end while
return τ

5.6 Planning without direct communication

In case there is no direct communication, the robots do not know about each other's trajectories. The algorithm was first designed assuming that there is communication between the robots. However the scenarios of no direct communication are practical and cannot be neglected. Hence once the algorithm design was over, the approach was extended to the case of no direct communication. Here other robots may refer to any of the other robots in the scenario, humans or other dynamic obstacles. This approach may similarly be extended to the case when there is communication between a pool of robots and no communication exists with the other robots (or dynamic obstacles).

A solution would be to repel the trajectories from the current position of the robot, which converts the algorithm to a conventional potential field method. A better means would be to attempt to predict the intent of the robot and assume it as the trajectory which it follows. Much like in natural human walking or driving, the robots tend to travel much of their distances without making much turns. Hence here the robot R_i assumes other robots R_j would travel in their current direction with the current speed, till they collide with a static obstacle. The collision criterion comes from the fact that one robot's behaviour by another robot should not be affected if they are separated by a wall or obstacle. The untraced trajectory of R_i is made to repel from these extrapolated trajectories of other robots.

At any time *t*, let the positions of the robot R_i be $\tau_i^{traced}(t)$, speed be $v_i(t)$ and orientation be $\theta_i(t)$. The robot R_i traces all the other robots to get their speeds, positions and orientations. The assumed trajectory of any robot R_j is given by equation (19).

$$\tau_j(u) = \tau_j^{traced}(t) + v_j(t)(u-t), u \geq t, \tau_j(u_1) \in \xi_{static}^{free} \forall t \leq u_1 < u \tag{19}$$

The closest distance is measured and accordingly the obstacles are defined which are used to affect the sampled trajectory from current position $\tau_i^{traced}(t)$, which gives the modified trajectory τ_i. The immediate speed of the robot is as per equation (10). The robot is made to trace the trajectory with this speed till the next iteration of planning is called.

6 Results

The approach was experimented using computer simulations. The algorithm was developed in MATLAB. The scenario parameters could be specified in a separate file while the map could be given as an image file to the simulation tool. The results were displayed as a graphic, while the performance metrics associated were displayed at the console. The assumption is that the robot knows the complete map. The static component of the map will be typically produced by a SLAM which typically requires a robot equipped with a lidar/3D image

sensor and an IMU. Then there are two variants of the algorithm. The first variant assumes communication between the robots. Here all robots should have an IMU/vision sensor so as to localize themselves and a communication device so as to transmit and receive the positions. The second version, without communication, just needs all robots to know their position. Typically one can also employ an external vision system that makes the map, tracks and localizes the robot; and is connected with a server where the algorithm runs. This server is connected wirelessly to all the other robots and transmits the actions that the robots need to take. The virtual sensor used assumes a 360 degrees field of view and an infinite sensing capability. This sensor is virtual and can be assumed knowing the complete map a priori.

6.1 Simulation Results

The algorithm was tested via simulations on a number of scenarios. The results of 3 scenarios are discussed. In all the scenarios four robots are generated at one corner and are needed to reach the other corner of the map which is a grid of 500×500 blocks. This typically represents a big office space whose orthographic map is resized into an image of size 500×500 by a reduction of resolution. The first scenario consists of a number of obstacles placed within the map. The major task for the robots is hence to first compute their optimal paths and then to avoid each other. The obstacles to some extent screen a robot from the other, making it impossible for the robots to collide. Hence the second scenario consists of a single obstacle. The robots may now be found close to each other, and hence the coordination algorithm becomes important. In the third scenario no obstacle is placed. The optimal paths of the robots are hence diagonal lines, such that all robots collide at the centre. The paths traced by the robots are shown in figure 5. For a detailed insight of the results please refer video 1.

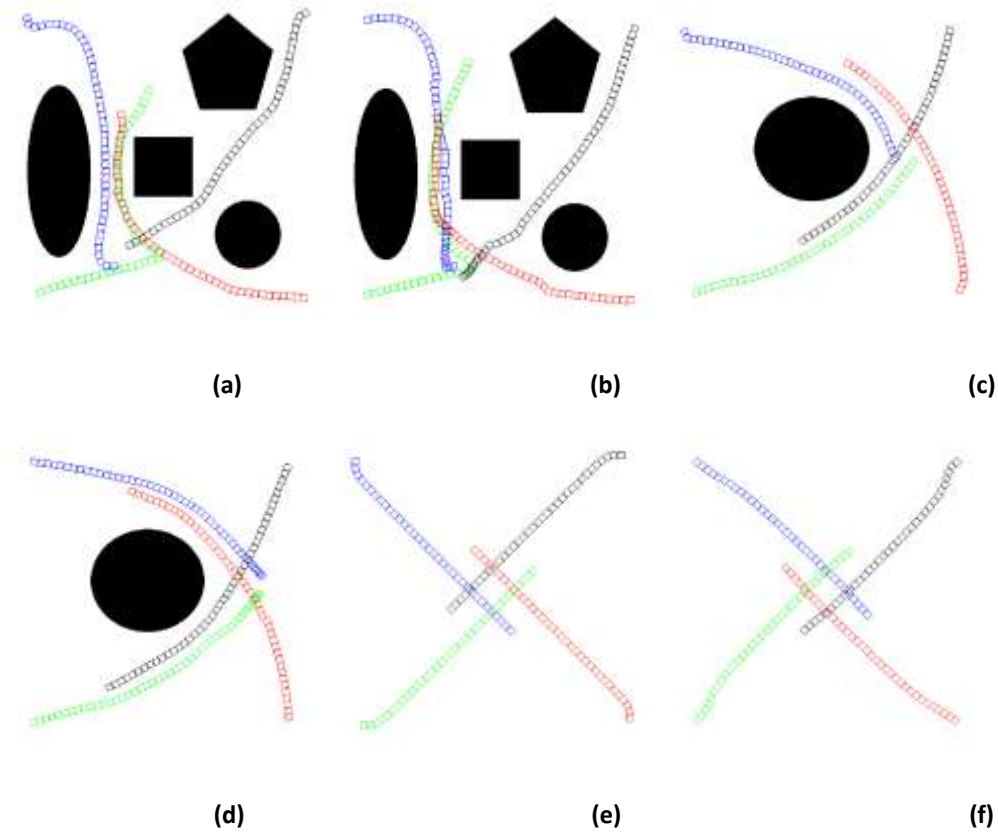


Fig. 5 Experimental Results. (a), (c) and (e) cover scenario 1, 2 and 3 respectively with communication between the robots. (b), (d) and (f) cover the corresponding scenarios without communication between the robots. For detailed insight please refer video 1.

Figure 5(a) shows the path traced by the robot with the proposed algorithm for scenario 1. The sampling based optimal path computation algorithm placed the different robots such that there was a potential collision between the different pairs of robots at different places. Hence the coordination algorithm had to place the robots away from each other, or to time the robots such that no collision happens. It may be seen that enough separation was made available and hence the path was neither too long and nor clearance too small. The corresponding path in the absence of communication between the robots is shown in figure 5(b). Lack of communication makes it impossible for one robot to align as per the requirements of the other robot. The figure shows two robots unable to judge each other's intensions, which hence had to slow and take corrective actions when they were quite close to each other.

Figure 5(c) shows the results to the second scenario. Although in reality the optimal paths of different robots should not have collided with each robot encircling the obstacle in the same clockwise/anti-clockwise direction, the sampling based optimal path planner instead scheduled different robots at the same place. It may be noted that the individual optimal path planner of robots do not have communication, and both clockwise and anti-clockwise encircling have the same path cost. Hence additional coordination was required, which moved some robot far apart to have enough clearance between the robots. The path without communication is shown in figure 5(d). Again two robots were found to be too close and could not judge each other's intensions until they reached very close to each other. Both robots wanted to avoid by each other's right side, which meant opposing plans getting the robots on the verge of collision. In the communication case such plans could be communicated and optimized until a consensus is reached.

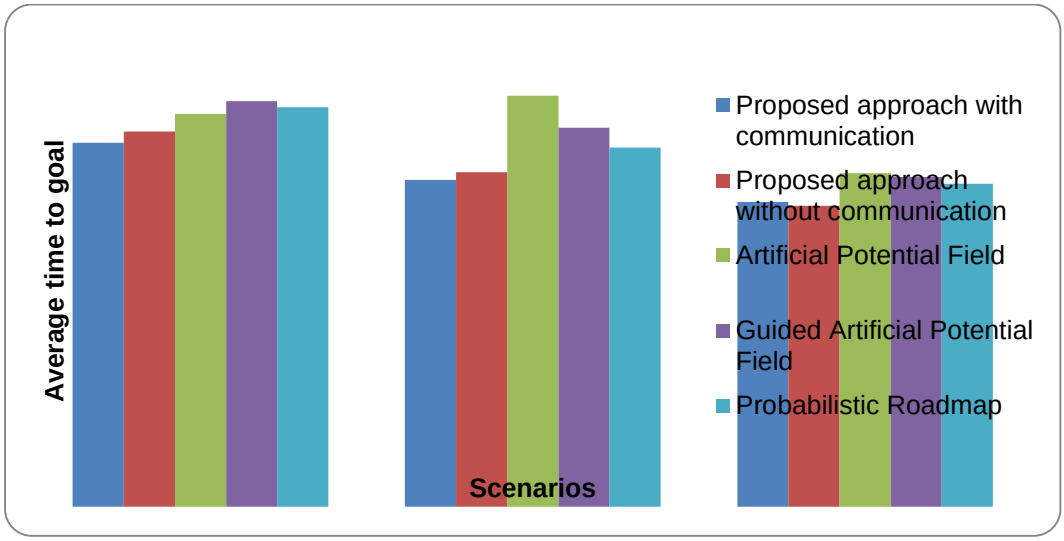
Figure 5(e) shows the case of no obstacles. All the robots easily avoided each other by turning leftward. While the final motion was simple, it was not easy to get such a plan of action. As per initial plan, all robots would have met at the centre and hence each robot has to move leftward or rightward. Due to symmetry it was impossible to guess whether turn should be leftward or rightward. Even though the avoidance heuristic in such a case preferred a left direction, algorithmically the scenario may not be perfectly symmetrical, which meant two robots took opposing plans such that they collide a little leftward or rightward. After a few iterations of coordination, the robots reach to a consensus of the direction of avoidance, while the potential function looked after the magnitude. Though the benefit of having a perpendicular force was also visible in the first scenario, the use is best visible in this scenario. The case of absence of communication is shown in figure 5(f), where the robots seem to have done well in judging each other's intensions, which by their very initial trajectories are reasonably clear.

6.2 Comparisons

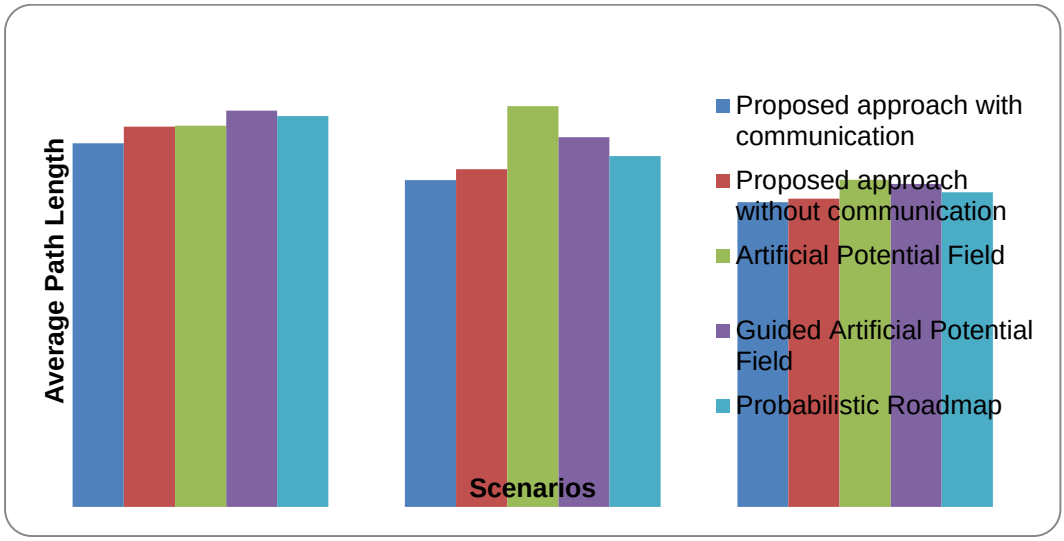
The algorithm is compared against 3 commonly used and related approaches. The first is a typical implementation of artificial potential field [5]. The attractive potential of goal was made decreasing with distance up to a limit beyond which it was made uniformly small to eliminate the attractive potential being too small far off from the goal. The robot was disallowed to make sharp turns or move backwards. The second approach was a guided artificial potential field which eliminates the problem of robot taking globally sub-optimal path or getting struck. The optimal single robot path was used to guide the artificial potential field. A point at a distant in the optimal path acted as the immediate goal of the robot planned using potential field [36]. The third approach was probabilistic roadmap [12] with the planning of individual robots done using prioritized A* algorithm. Each of the paths obtained was subjected to a post processing and local optimization.

For comparisons the main metrics used are average time of travel and the average path length. The results are given in figure 6. Figure 6(a) and 6(b) show the performance using the metrics of average time and travel and average path length, which show a difference between the algorithms. Figure 6(c) shows the metric of average clearance. All methods were tuned to maintain nearly the same clearance. Experiments hence revealed no significant difference in clearance. The cooperation metrics behave similar to the presented metrics. It can be seen that the proposed approach with communication was the best followed by proposed approach without communication. For measuring the relative performance over path length metric, effective path length is used as a metric. The effective path length is defined as the path length over the shortest distance path length between the source and goal. Since the optimal value of the path length is not known, the Euclidian distance between the source and goal is used as an estimate. Using this formula, the plots on effective path length are given in Figure 6(d). It can be seen that there is a significant improvement as a result of the use of the proposed algorithm. The relative performance is also plotted as Figure 6(e). For scenario with no obstacle, algorithm without

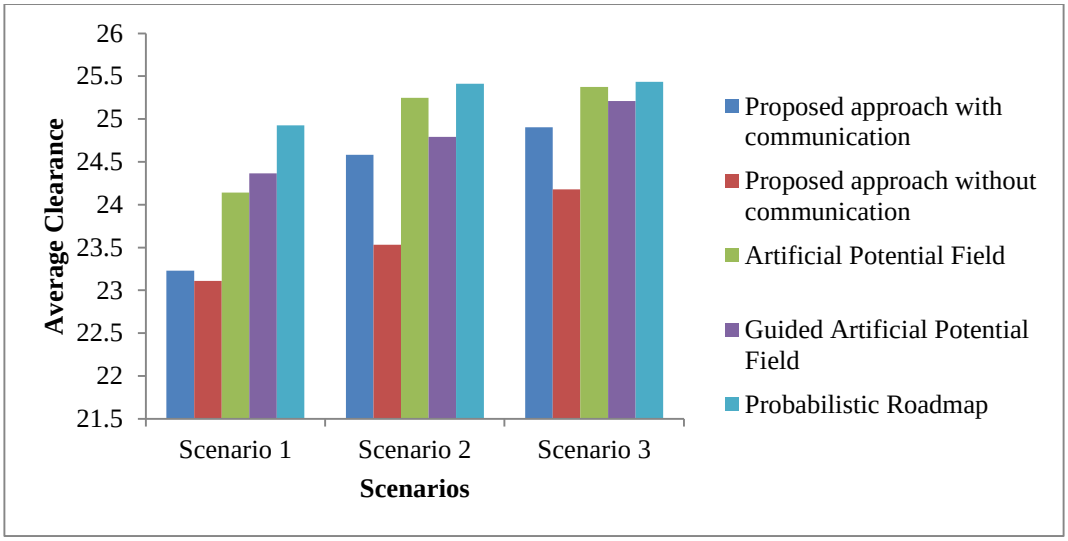
communication became better as the intensions were clear and there was no incentive of communication. As maps and parameters were chosen such that the robots are not trapped and all paths to the goal have nearly same path cost, there is no direct incentive of guidance to potential field, though it does lead to a better cost. Cost largely depends upon the manner in which robots meet, and hence for scenario 1 path cost with guidance is high due to a worse position of meet as compared to the case without communication. Probabilistic roadmap approach is next best after the proposed approaches.



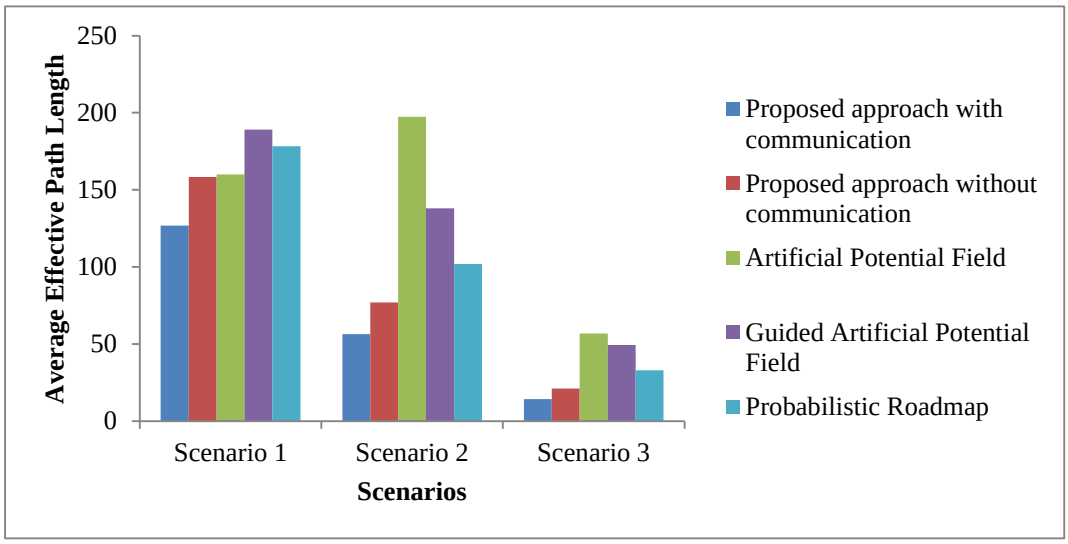
(a)



(b)


26
25.5
25
24.5
24
23.5
23
22.5
22
21.5
Average Clearance
Scenario 1
Scenario 2
Scenario 3
Scenarios
Proposed approach with communication
Proposed approach without communication
Artificial Potential Field
Guided Artificial Potential Field
Probabilistic Roadmap

(c)


250
200
150
100
50
0
Average Effective Path Length
Scenario 1
Scenario 2
Scenario 3
Scenarios
Proposed approach with communication
Proposed approach without communication
Artificial Potential Field
Guided Artificial Potential Field
Probabilistic Roadmap

(d)

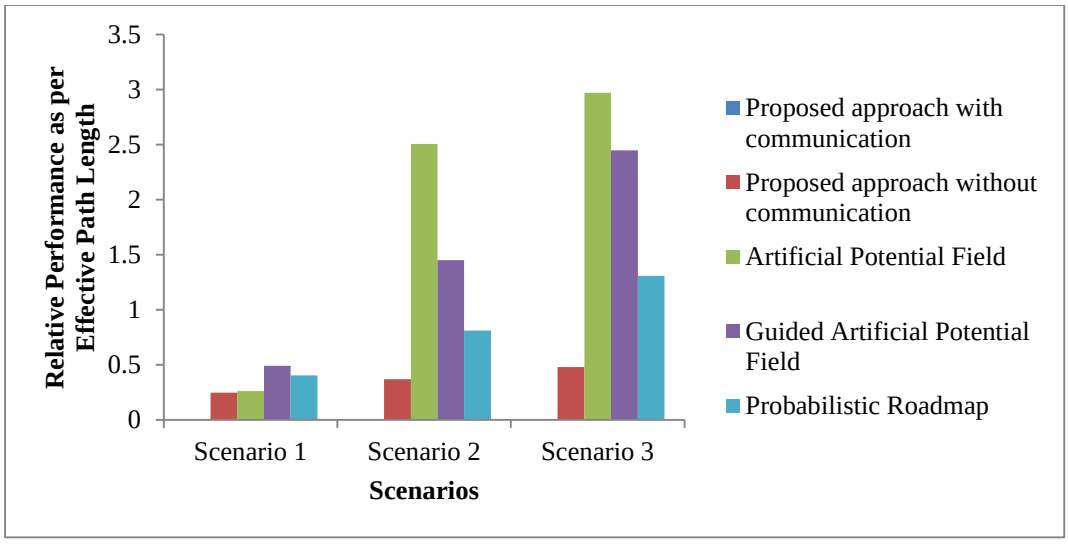

3.5
3
2.5
2
1.5
1
0.5
0
Relative Performance as per Effective Path Length
Scenario 1
Scenario 2
Scenario 3
Scenarios
Proposed approach with communication
Proposed approach without communication
Artificial Potential Field
Guided Artificial Potential Field
Probabilistic Roadmap

Fig. 6 Comparative Analysis. The algorithms (with and without communication) are compared with Artificial Potential Field [5], Probabilistic Roadmap [12], and a fusion of the two called as guided artificial potential field using fusion similar to [36].

The general problem associated with potential field approach was that at times it got trapped, especially while some other robot was coming. The parameters were adjusted so that this does not happen in the studied experimental cases, while it may be the case in general. The other problem was while some other robot (say robot *B*) came and applied a strong repulsive force, the robot (say robot *A*) had to quickly move back. Due to steering limits a turn was made in the opposing direction which made the robot *A* somewhat far. After robot *B* passed, the normal route for robot *A* continued which appeared as robot *A* taking an unnecessary circular turn to avoid robot *B*. Due to acceleration limits and constantly changing potential field due to other robot's motion, the robots at times got reasonably close to each other. For the no obstacle scenario the problem was symmetry, wherein the forces applied attracted and repelled the robot in the same line, which meant that they should have stopped close to each other. In simulations slight in-symmetries became dominant when the robots were very close, wherein they all turned to their right for avoidance.

The guided potential field approach gives some optimality and completeness to the problem, but does suffer from the above mentioned limitations. An additional limitation is that if a robot changes its path to avoid another robot, which may be by a large amount, another path may now be better from a guiding perspective. But the robot would have to change its course to make the final path resemble the guided path. No significant loss due to this phenomenon was however observed.

The major problem with the probabilistic roadmap was that the higher priority robots took the best suited paths centrally located in case of narrow passages. This made the other robots need to deviate largely from their optimal paths and thus increasing their path lengths by a large amount. A general expectation was that this method would have the robots spread out in the map. Two paths (say *I* and *II*) may be nearly optimal at a global level for a couple of robots (say *A* and *B*). Planning individual robot's paths may make one path (say *I*) likely for both robots. However in case robot *B* is planned after robot *A*, robot *B* knowing the presence of robot *A* at path *I*, may now chose path *II*. In case of proposed approach both would however use path *I*. This was however not experimentally observed. The reason is that the approach works on a graph sampled from the configuration space and not the actual graph. Hence the path costs are only approximately known, to which the presence of a robot does not carry a significant difference.

6.3 Convergence

While it was intuitively said that all trajectories merge to some equi-potential point, which is the optimal travel plan, it is important to experimentally study the same. A path cost metric was devised which was taken as the average time of journey for robots, and if the robots did not reach their goal, a penalty was added equal to tracing the rest of the trajectory by $1/4^{th}$ of the maximum speed. The path cost for different number of iterations is shown in figure 7(a), which shows that the trajectories indeed converge. The oscillatory nature at the start is due to conflicting intentions. A robot in attempt to avoid one robot, may in turn find another one, which can only be resolved once a consensus is reached regarding the manner of avoiding each other.

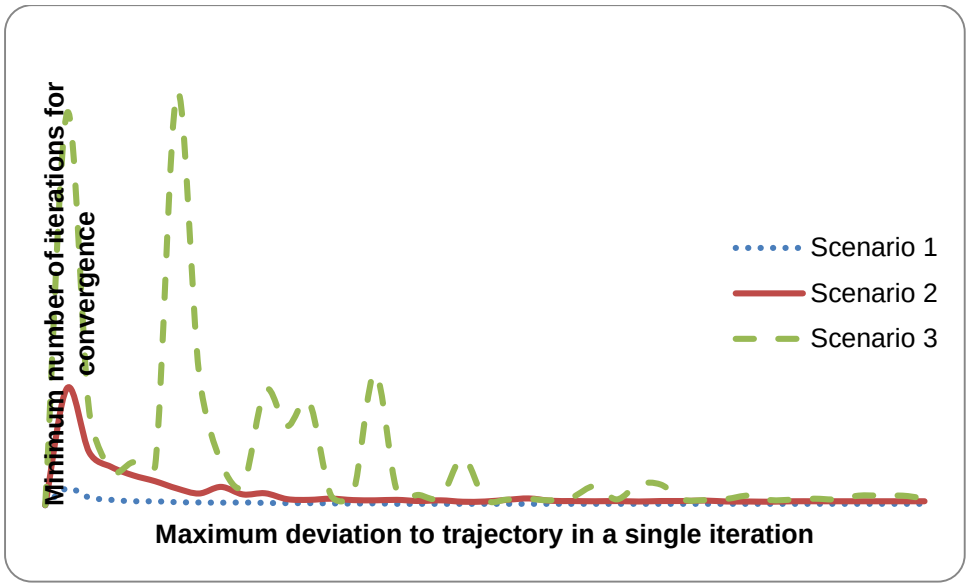
The only important parameter of the algorithm is the maximum deviation that can be made in a trajectory per iteration of the algorithm. Lower values mean slow motion of trajectories and slow convergence. A higher value has the risk of overstepping the optima. The minimum number of iterations needed for convergence is shown in figure 7(b). For experiments the value was specified as 10, which is not very large from an algorithm perspective. The minimum number of iterations was of the order of 10 to 30. The highly oscillatory nature in no obstacle scenario is due to the kind of stochastic nature of the algorithm in different runs. In symmetrical scenario different robots may assume different ways to avoid each other. If all the robots are in consensus initially, very little iterations may be required; which is better than the case of not having any 2 robots in consensus for which larger number of iterations is required; which is still better than having no pair of robots in consensus.

6.4 Scalability

It is worthy to note the effect of increasing the number and robots on the algorithm performance. The planning happens at two levels, the single robot motion planning to get the initial trajectory, and the multi-robot coordination. For single robot motion planning, the complexity is O(*n*) for *n* robots, since all robots are planned independently one after the other. In terms of the number of robots, the complexity of the coordination step is

O(n^2), although the algorithm is anytime in nature, wherein the solution keeps improving with time. So the number of robots cannot be increased to the order of hundreds due to this complexity. However, it is widely accepted that in case of a very large number of robots, only the neighbouring 3-6 robots account for the immediate motion. Hence the variant with communication may be used with only neighbouring few robots, in which case the complexity reduces to O(n log n) with a drop in optimality and completeness. In this paradigm the algorithm can be used like any other reactive navigation algorithm scalable to a very large level.

(a)



(b)

Fig. 7 Algorithm Analysis. (a) Convergence of the algorithm along with iterations. (b) Effect of increasing maximum deviation to trajectory in a single iteration to the number of iterations required for convergence. Increasing the factor decreases convergence time, but increases the risk of overstepping optima.

Similarly the effect of increasing the number of obstacles is assessed. The single robot motion planning has the same complexity with respect to the number of obstacles as a standard Probabilistic Roadmap approach. Large number of obstacles and obstacles with narrow corridors affect the performance. For coordination, the increased number of obstacles partition the environment and do not let the trajectories of multiple robots affect each other. This greatly simplifies the problem. So the more be the number of obstacles, the easier is the problem. There is still a problem when multiple robots get scheduled through the same narrow corridor, having not enough space for the robots to pass by from different directions. In such a case significant trajectory modifications and re-planning at the single-robot level will be required.

7 Conclusions

If all the robots choose to take their optimal paths, there is likely to be a collision. The aim of a good coordination strategy is to direct the robots so that no collision happens while the robots take a near-optimal trajectory. This paper proposed a very strong coordination strategy that gives near-optimal results, while taking a small computation time. The problem was converted into a new space, called as the trajectory space, wherein every robot trajectory is a point. The problem of multi-robot coordination was hence converted into the problem of finding the equi-potential point in this space that can be done very quickly using heuristics. This is a new concept in the literature of motion planning and gives a new fundamental look at the problem. When operating without communication, the robots could guess the intent of the robots to judiciously make a move, which imitates the natural human walking wherein humans plan their moves in anticipation of the moves of the people around. Experimental results confirm that the robots could coordinate with each other while taking near-optimal trajectories and driving with high speeds. The approach was adjudged to be better than the popular approaches of artificial potential field, guided artificial potential field and probabilistic roadmap.

One of the wonderful attributes of human walking is the ability to predict regions that may have high density, and to avoid the regions when deciding the route. The robots, being connected to each other, should be able to display the same skills in a much better manner. For the case of vehicles in a traffic scenario, the same concept was used in [53], wherein vehicles could judiciously schedule themselves based on road usage known before time due to communication. The same capability needs to be given to the robots in the experimented scenarios. For the case of no communication, anticipative techniques may be made. Instead of starting from a single trajectory, multiple trajectories from different homotopic groups could be considered. Better anticipative techniques to guess the robot's intent can be very helpful to improve the performance for the case of no communication. Since there are multiple robots, a distributed version of the algorithm is of a lot of benefit.

References

[1] Latombe, J. C. (1991) Robot Motion Planning, Kluwer Academic Press, MA.
[2] Tiwari, R., Shukla, A., Kala, R. (2013) Intelligent Planning for Mobile Robotics: Algorithmic Approaches, IGI Global Publishers, Hershey, PA.
[3] Arai, T., Pagello, E., Parker, L. E. (2002) Editorial: Advances in Multi-Robot Systems. IEEE Trans. Rob. Autom. 18(5), 655-661.
[4] Parker, L. E., Schneider, F. E., Schultz, A. C. (2005) Multi-robot systems: From swarms to intelligent automata vol. 3, Springer-Verlag, New York.
[5] Khatib, O. (1985) Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In: Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, Missouri, pp. 500-505.
[6] Hwang, Y. K., Ahuja, N. (1992) A potential field approach to path planning. IEEE Trans. Rob. Autom. 8(1), 23-32.
[7] Barraquand, J., Langlois, B., Latombe, J. C. (1992) Numerical potential field techniques for robot path planning. IEEE Trans. Syst. Man Cybern. 22(2), 224-241.
[8] Chang, H. (1996) A new technique to handle local minimum for imperfect potential field based motion planning. In: Proceedings of the 1996 IEEE International Conference on Robotics and Automation, pp. 108-112.
[9] Kala, R., Shukla, A., Tiwari, R. (2010) Dynamic Environment Robot Path Planning using Hierarchical Evolutionary Algorithms. Cybern. Syst. 41(6), 435-454.
[10] Sgorbissa, A., Zaccaria, R. (2012) Planning and obstacle avoidance in mobile robotics. Rob. Auton. Syst. 60(4), 628-638.
[11] Kavraki, L. E., Kolountzakis, M. N., Latombe, J. C. (1998) Analysis of probabilistic roadmaps for path planning. IEEE Trans. Rob. Autom. 14(1), 166-171.
[12] Kavraki, L. E., Svestka, P., Latombe, J. C., Overmars, M. H. (1996) Probabilistic roadmaps for path planning in highdimensional configuration spaces. IEEE Trans. Rob. Autom. 12(4), 566-580.

[13] Kala, R. (2013) Rapidly-exploring Random Graphs: Motion Planning of Multiple Mobile Robots. Adv. Rob. 27(14), 1113-1122.
[14] Gayle, R., Sud, A., Andersen, E., Guy, S. J., Lin, M. C., Manocha, D. (2009) Interactive Navigation of Heterogeneous Agents Using Adaptive Roadmaps. IEEE Trans. Visual. Comput. Graph. 15(1), 34-48.
[15] Gayle, R., Sud, A., Lin, M. C., Manocha, D. (2007) Reactive Deformation Roadmaps: Motion Planning of Multiple Robots in Dynamic Environments. In: Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, pp. 3777-3783.
[16] Hilgert, J., Hirsch, K., Bertram, T., Hiller, M. (2003) Emergency path planning for autonomous vehicles using elastic band theory. In: Proceedings of the 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics vol. 2, pp. 1390- 1395.
[17] Quinlan, S., Khatib, O. (1993) Elastic bands: connecting path planning and control. In: Proceedings of the 1993 IEEE International Conference on Robotics and Automation, pp. 802-807.
[18] Yang, Y., Brock, O. (2010) Elastic roadmaps - motion generation for autonomous mobile manipulation. Auton. Rob. 28(1), 113-130.
[19] Lumelsky, V. J., Harinarayan, K. R. (1997) Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model. Auton. Rob. 4, 121–135.
[20] Sánchez-Ante, G., Latombe, J. C. (2002) 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, Washington DC, pp. 2112 – 2119.
[21] Desaraju, V. R., How, J. P. (2012) Decentralized path planning for multi-agent teams with complex constraints. Auton. Rob. 32, 385–403.
[22] Bennewitz, M., Burgard, W., Thrun, S. (2001) Optimizing schedules for prioritized path planning of multi-robot systems. In: Proceedings of the 2001 IEEE international conference on robotics and automation, pp. 271–276.
[23] Bennewitz, M., Burgard, W., Thrun, S. (2002) Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. Rob. Auton. Syst. 41(2-3), 89– 99.
[24] Kant, K., Zucker, S. W. (1986) Toward Efficient Trajectory Planning: The Path-Velocity Decomposition. Int. J. Rob. Res. 5(3), 72-89.
[25] Kala, R. (2012) Multi-Robot Path Planning using Co-Evolutionary Genetic Programming. Expert Syst. Appl. 39(3), 3817-3831.
[26] Brock, O., Khatib, O. (2000) Elastic strips: A framework for integrated planning and execution. In: Experimental Robotics VI, Lecture Notes in Control and Information Sciences Volume 250, pp. 329-338.
[27] Baxter, J. L., Burke, E. K., Garibald, J. M., Normanb, M. (2009) Shared Potential Fields and their place in a multi-robot co-ordination taxonomy. Rob. Auton. Syst. 57(10), 1048-1055.
[28] Vadakkepat, P., Tan, K. C., Ming-Liang, W. (2000) Evolutionary artificial potential fields and their application in real time robot path planning. In: Proceedings of the 2000 Congress on Evolutionary Computation, pp. 256-263.
[29] Tu, K. Y., Baltes, J. (2006) Fuzzy potential energy for a map approach to robot navigation. Rob. Auton. Syst. 54(7), 574-589.
[30] Ge, S. S., Cui, Y. J. (2002) Dynamic Motion Planning for Mobile Robots Using Potential Field Method. Auton. Rob. 13, 207–222.
[31] Jaradat, M. A. K., Garibeh, M. H., Feilat, E. A. (2012) Autonomous mobile robot dynamic motion planning using hybrid fuzzy potential field. Soft Comput. 16, 153–164.
[32] Yin, L., Yin, Y., Lin, C. J. (2009) A new potential field method for mobile robot path planning in the dynamic environments. Asian J. Control 11(2), 214-225.
[33] Chang, Y.C., Yamamoto Y. (2009) Path planning of wheeled mobile robot with simultaneous free space locating capability. Intell. Serv. Rob. 2(1): 9–22.
[34] Masoud, A.A. (2013) A harmonic potential field approach for joint planning and control of a rigid, separable nonholonomic, mobile robot. Rob. Auton. Syst. 61: 593–615.
[35] Lee, J., Nam Y., Hong, S., Cho, W. (2012) New Potential Functions with Random Force Algorithms Using Potential Field Method. J. Intell. Robot. Syst. 66:303–319
[36] Motlagh, O., Tang, S. H., Ismail, N., Ramli, A. R. (2012) An expert fuzzy cognitive map for reactive navigation of mobile robots. Fuzzy Sets Syst. 201, 105–121.
[37] Selekwa, M. F., Dunlap, D. D., Shi, D., Collins Jr., E. G. (2008) Robot navigation in very cluttered environments by preference-based fuzzy behaviours. Rob. Auton. Syst. 56(3), 231-246.
[38] Kala, R., Shukla, A., Tiwari, R. (2010) Fusion of probabilistic A* algorithm and fuzzy inference system for robotic path planning. Artif. Intel. Rev. 33(4), 275-306.
[39] Hank, M., Haddad, M. (2016) A hybrid approach for autonomous navigation of mobile robots in partially-known environments. Rob. Auton. Syst. 86: 113–127

[40] Urdiales, C., Perez, E. J., Vázquez-Salceda, J., Sànchez-Marrè, M., Sandoval, F. (2006) A purely reactive navigation scheme for dynamic environments using Case-Based Reasoning. Auton. Rob. 21(1), 65-78.
[41] Kim, Y., Kwon, S.J. (2015) A heuristic obstacle avoidance algorithm using vanishing point and obstacle angle. Intell. Serv. Rob. 8(3): 175–183.
[42] Solovey, K., Salzman, O., Halperin, D. (2015) Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning. In: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics, Algorithmic Foundations of Robotics XI, vol. 107, Springer, pp. 591-607.
[43] Kala, R. (2014) Coordination in navigation of multiple mobile robots. Cybern. Syst. 45(1): 1-24.
[44] Saska, M., Spurný, V., Vonásek, V. (2016) Predictive control and stabilization of nonholonomic formations with integrated spline-path planning. Rob. Auton. Syst. 75: 379–397.
[45] Indelman, V. (2017) Cooperative multi-robot belief space planning for autonomous navigation in unknown environments. Auton. Robot. DOI 10.1007/s10514-017-9620-6
[46] Lee, T., Kim, Y.J. (2013) GPU-based motion planning under uncertainties using POMDP. In: Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, pp. 4576-4581.
[47] Feyzabadi, S., Carpin, S. (2017) Planning using hierarchical constrained Markov decision processes. Auton. Rob. doi:10.1007/s10514-017-9630-4.
[48] Wu, F., Zilberstein, S., Chen, X. (2011). Online planning for multi-agent systems with bounded communication. Artif. Intell. 175: 487–511.
[49] Kim, D.W., Lasky, T.A., Velinsky, S.A. (2013) Autonomous Multi-mobile Robot System: Simulation and Implementation using Fuzzy Logic. Int. J. Control, Autom. Syst. 11(3):545-554.
[50] Das, P.K., Behera, H.S., Jena, P.K., Panigrahi, B.K. (2017) An Intelligent Multi-robot Path Planning in a Dynamic Environment Using Improved Gravitational Search Algorithm. Int. J. Autom. Comput. DOI: 10.1007/s11633-016-1019-x.
[51] Hoy, M., Matveev, A.S., Savkin, A.V. (2012) Collision free cooperative navigation of multiple wheeled robots in unknown cluttered environments. Rob. Auton. Syst. 60: 1253–1266.
[52] Charalampous, K., Kostavelis, I., Gasteratos, A. (2015) Thorough robot navigation based on SVM local planning. Rob. Auton. Syst. 70: 166–180.
[53] Kala, R., Warwick, K. (2013) Multi-Level Planning for Semi-Autonomous Vehicles in Traffic Scenarios based on Separation Maximization. J. Int. Rob. Syst. 72(3-4), 559-590.