Reactive Collision Avoidance of Multiple Moving Agents by Cooperation and Conflict Propagation

K. Madhava Krishna and Henry Hexmoor
CSCE Dept., University of Arkansas
Fayetteville AR 72701

Abstract: *A strategy for collision avoidance between several moving robots that are not in possession of each other's plans is presented here. A robot's awareness of other robots is limited to the knowledge of their current states represented by their present and impending velocities and their motion direction. A robot is aware of the presence of other robots when they fall within its field of vision. Collision avoidance is attempted at three levels namely at individual, cooperative and propagation levels through velocity control. At individual level it suffices that one of the robots involved in a forthcoming collision modifies its velocity. The cooperative level is characterized by the requirement that all the robots involved in collision modify their velocities in a synchronized fashion. In the third level robots not involved in a collision are entailed to participate by altering their velocities in a manner that resolves collision conflicts between the robots involved. The third level is termed as the propagation level since the collision conflict is propagated to robots not a part of the conflict and their assistance sought in avoiding conflicts. The strategy is implemented in a distributed fashion across all robots in the system. Simulation results are presented to authenticate the efficacy of the proposed method.*

I. INTRODUCTION

The shift in research community towards multi-robotic systems has entailed probe into issues in navigation involving multiple robots. Literature abounds in work relating to multi robot path planning and it is beyond the scope of this effort to review all of them. Multi-robot motion planning algorithms are traditionally classified as centralized [4,7,12,16] or decentralized [1,3,5]. In centralized approaches a single processor computes the plans for all the robots and the robots are controlled from a unified command. In the decentralized approach each robot computes its own plan and coordination between robots occur when conflicts are detected through exchange or broadcast of their plans. The tradeoffs between the two approaches are well documented. In case of centralized approach that computes all possible conflicts over entire trajectories the number of collision checks to be performed and the planning time tends to increase exponentially as the number of robots in the system increases. Also the requirement that all the world knowledge be localized at a single place often turns out to be not practical. Also complete recalculation of paths is required even if one of the robot's plans is altered or environment changes. However centralized approaches can guarantee completeness and optimality of the method. Decentralized approaches on the other hand are less computationally intensive as the computational burden is distributed across the agents and in principle the computational complexity of the system can be made independent of the number of agents in it. It is more tolerant to changes in the environment or alterations in objectives of the agents. However they are intrinsically incapable of satisfying optimality and completeness criterion.

A number of recent approaches try to provide algorithms that combine the advantages of both the approaches. Li and Chou [13] present a grouping strategy based on the hierarchical sphere tree that groups robots dynamically. Though the approach is purely a centralized one, the grouping strategy reduces planning time greatly for a large number of robots. The method is especially suitable in cases where the robots are crowded at their starting and goal configurations. Guo and Parker [9] present a distributed algorithm which provides for optimality. The algorithm is distributed in that each robot computes its own plan and the computations for optimal collision free motion in the form of the modified velocity profile is done on each robot. A performance index based on the velocity profile is also computed for each robot and is broadcast to all other robots along with the velocity profile. All the robots adopt the profile corresponding to the minimum performance index as the optimal profile. Since each robot ends up calculating the velocity profiles for every other robot along their entire trajectories the computational feasibility of the proposed method when the number of robot increases is in question. The complexity of the search space is also exponential in the number of robots in the system. In [6] a methodology that is centralized within a network and distributed across networks is proposed. Networks get formed when robots are within a distance where communication is possible between them. A plan merging protocol (PMP) is presented in [2] as a solution for the deadlock problem that occurs in distributed approaches.

This paper presents a novel approach for resolving collision conflicts between multiple robots. The approach does not require the exchange or broadcast of complete plans as is the case with the typical decentralized approaches [1, 3, 9] nor does it rely on assigning priorities to robots such as in [3, 7]. The approach is based on changing velocities of the robots involved in a conflict in a synchronized fashion that is termed as cooperative resolution. The term cooperative is not a misnomer for it helps in achieving the following capabilities:

1 Avoid collision conflicts in a manner that conflicting agents do not come too near while avoiding one and another where and whenever possible. Thus agents take action in a fashion that benefits one another apart from avoiding collisions.

2 Provides a means of avoiding conflicts in situations where a single agent is unable to resolve the conflict individually.
3 Serves as a pointer to areas in the possible space of solutions where a search for solution is likely to be most fruitful

The present work is novel and different from others as the resolution of collision conflicts is attempted at three levels, namely the individual, cooperative, and propagation levels. Functionally cooperation is a methodology for pinning down velocities in the joint solution space of velocities of the robots involved in conflict when there exists no further solution in the individual solution spaces of those robots. When joint actions in the cooperative phase are not sufficient for conflict resolution assistance of other robots that are in a conflict free state at that instant is sought by the robots in conflict by propagating descriptions of the conflicts to them. When such free robots are also unable to resolve the conflict collision is deemed inevitable. The concept of propagating conflict to robots not directly involved in a conflict is not found in robotic literature. Such kind of transmission of requests to robots though not invoked frequently is however helpful in resolving a class of conflicts which otherwise would not be possible as our simulation results reveal.

The method presented here is more akin to a real-time reactive setting where each robot is unaware of the complete plans of the other robots and the model of the environment. The work closest to the present is a scheme for cooperative collision avoidance by Fujimora's group [8] and a distributed fuzzy logic approach as reported in [13]. Their work is based on devising collision avoidance for two robots based on orientation and velocity control and extend this strategy for the multi robot case based on the usual technique of priority based averaging (PBA). However we have proved in an earlier effort of ours [11] that such PBA techniques fail when individual actions that get weighted and averaged in the PBA are conflicting in nature. The work of Lumelsky [14] is of relation here in that it does not entail broadcast of plans to all other robots. It describes an extension of one of the *Bug* algorithms to a multi robotic setting. There is not much mention of cooperation or collaborative efforts between the robots except in the limited sense of "reasonable behavior" that enables shrinking the size of collision front of a robot that is sensed by another one.

Conflict resolution through cooperation and conflict propagation is pertinent to many applications that entail a number of robots that crisscross each other in quick succession or in situations where robots find themselves coming together to get across an intersection from various directions which would otherwise result in a logjam. In many such situations it is not reasonable to expect that the information about all such robots be maintained and their actions controlled from a central command nor does exchange or broadcast of their entire plans to one another at the time of eye contact appear intuitive.

II PROBLEM FORMULATION AND PREMISES

The following premises have been made for algorithm development and simulations:

a. Each robot *Ri* is assigned a start and goal locations and it has access to its current state and its current and aspiring velocities.
b. All robots are circular and described by their radius
c. The current state of *Ri* is represented as $S_i = \{vc_i, vn_i, \theta_i\}$ where vc, vn represent its current and aspiring velocities and θ its current motion direction.
d. Each robot can see any other robot within its field of vision and occlusion relations between robots are not considered.
e. Robots are capable of broadcasting their current states to each other. They do so only to those robots that are within its field of vision or sensing range.
f. Robots accelerate and decelerate at constant rates that is same for all robots. Hence a robot *Ri* can predict, when another robot *Rj* would attain its aspiring velocity vn from its current velocity vc.

A. Formalizing a Collision Conflict (CC)

Since robots are not point objects a collision is not merely a space-time collision, i.e. two or more robots reaching the same point at same time. Rather a CC is one that is spread over an interval of time and hence robots are prohibited from moving over a range of velocities for avoiding it. The CC is formalized here for the simple case of two robots moving at constant velocities.

Shown in figure 1, two robots *R1* and *R2* of radii r1 and r2 and whose states are (vc_1, vn_1, θ_1) and (vc_2, vn_2, θ_2) respectively, where vc_1, vc_2 are the current velocities while vn_1, vn_2 are the aspiring velocities for *R1* and *R2* respectively. Point C in the figure represents the intersection of the future paths traced by their centers. For purpose of collision detection one of the robots *R1* is shrunk to a point and the other *R2* is grown by the radius of the shrunken robot.

The points of interest are the centers C21 and C22 of *R2* where the path traced by the point robot *R1* becomes tangential to *R2*. At all points between C21 and C22 *R2* can have a potential collision with *R1*. C21 and C22 are at distances $(r1 + r2)\cos ec(|\theta_1 - \theta_2|)$ on either side of C. The time taken by *R2* to reach C21 and C22 given its current state (vc_2, vn_2, θ_2) is denoted by t_{21} and t_{22}. Similar computations are made for R1 with respect to R2 by making R2 a point and growing R1 by r2. Locations C11 and C12 and the time taken by R1 to reach them t_{11} and t_{12} are thus computed. A collision or CC is said to be averted between R1

and R2 if and only if $[t_{11}, t_{12}] \cap [t_{21}, t_{22}] \in \Phi$. The locations C11, C12, C21 and C22 are marked in figure1.

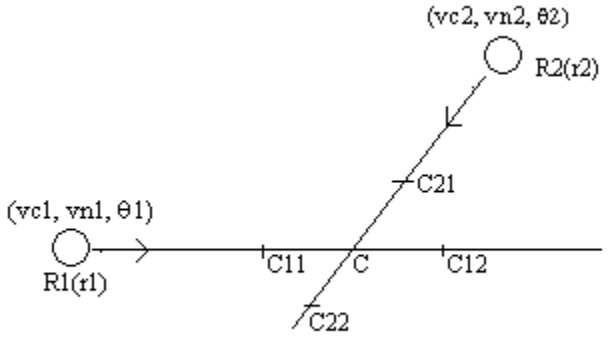

Figure 1: Two robots *R1* and *R2* with radii r1 and r2 along with their current states are shown. When *R1* is shrunk to a point and *R2* grown by radius of R1, C21 and C22 are centers of *R2* where the path traced by *R1* becomes tangential to *R2*.

A direct collision conflict (*DC*) between robots *R1* and *R2* is said to occur if *R1* occupies a space between C11 and C12 when the center of *R2* lies between C21 and C22 at some time t.

A robot is concerned only about its time nearest CC with another robot within a given reaction time t_r that is same for all robots. A uniform t_r across all robots facilitates commutativity in collision relations, i.e., if *R1* has a CC with *R2* in t_r so does *R2* with respect to *R1*

III THREE PHASES OF RESOLUTION

A. Characterizing the Individual Phase

A pair of robots *R1* and *R2*, which have a DC between them are said to be in individual phase of navigation if the conflict is resolved by either of the following two means:

(i) *R1* controls its velocity to v_{12} such that it is able to get past C12 before *R2* reaches C21 or *R1* controls its velocity to v_{11} such that it does not reach C11 before *R2* reaches C22.

(ii) *R2* controls its velocity to v_{22} such that it is able to get past C22 before *R1* reaches C11 or *R2* controls its velocity to v_{21} such that it does not reach C21 before *R1* reaches C12.

In both cases it would suffice that only one of the two robots controls its velocity. This indeed is the crux of the individual phase where at-least one of the two robots is able to individually avoid the conflict without requiring the other to take action. Thus the range of velocities that permit individual resolution of conflict by *R1* is given by: $v \in [0, v_{11}] \cup [v_{12}, v_{1M}]$, where v_{1M} represents the maximum permissible velocity for *R1*. They are given by: $v_{11} = vc_1 + a_{-m}t_{22} \pm \sqrt{(vc_1 + a_{-m}t_{22})^2 + (vc_1^2 + 2a_{-m}s)}$

Here s denotes the distance from *R1's* current location to C11. In the same vein the velocity that causes *R1* to be ahead of C12 when *R2* reaches C21 under maximum acceleration, a_m, is given by:

$v_{12} = vc_1 + a_m t_{21} \pm \sqrt{(vc_1 + a_m t_{21})^2 + (vc_1^2 + 2a_m s')}$,

where, s' the distance from R1's current location to C12 can also be written as $s' = s + (r1 + r2)\cos ec(|\theta_1 - \theta_2|)$. In a similar fashion velocities v_{21} and v_{22} are computed. For any robot the lower velocity is denoted as v_1 and the higher velocity by v_2 with the robot index dropped for notational simplicity. In other words the lower and upper velocities for *R2* is denoted as v_1 and v_2 instead of v_{21} and v_{22}.

B. The Cooperative Phase

A pair of robots *R1* and *R2* are said to be in cooperative phase of navigation if and only if they are able to resolve the collision conflict between the two through either of the following rules:

(i) *R1* is able to get past C12 under maximum acceleration before *R2* can get to C21 under maximum deceleration.

(ii) *R2* is able to get past C22 under maximum acceleration before *R1* can get to C11 under maximum deceleration.

Both robots *R1* and *R2* engage in velocity control in complementary fashion to circumvent the collision since individual efforts have failed. Essentially the cooperative resolution is a search in the joint space of velocities of both the robots that would resolve the conflict. Figure 2 depicts the range of possible velocities for two robots *R1*and *R2* along the abcissa and the ordinate. The inner rectangle represents the region where a solution can be found if and only if *both* robots alter their velocities. It also represents the area of possible velocities for which a collision is bound to occur and is termed as conflict area (CA). The figure is for a particular case of the two robots approaching each other along orthogonal directions with velocities 2.5 units each. The horizontal and vertical lines in the center of the figure indicate the velocity of the two robots as 2.5 units. For $v1 = 2.5$ corresponding to the robot's (*R1*) velocity on the abcissa, robot *R2* must possess a velocity, which lies either above or below the segments AB and CD of the rectangle when projected onto the ordinate. Similarly for $v2 = 2.5$ on the ordinate, robot *R1* must possess a velocity either to the right or left of the segments BC and AD when projected onto the abcissa to avert collision. Figure 3 depicts the evolution of CA over time when it fills up the entire space of possible velocities. This represents the instant when cooperation between the robots becomes inevitable. Cooperation as mentioned in section I serves as a pointer by

guiding the search in quadrants 2 and 4 of the rectangle, where robot actions are complementary. In these regions search for a collision free pair of velocities is likely to be more fruitful. Since a search is nonetheless time intensive the rules (i) and (ii) mentioned above where robots resort to maximum acceleration and deceleration in a complementary fashion offer the lower and upper bound solutions. A failure of the solutions at the bounds implies a failure anywhere inside and a pointer to resort to conflict propagation as the last resort.

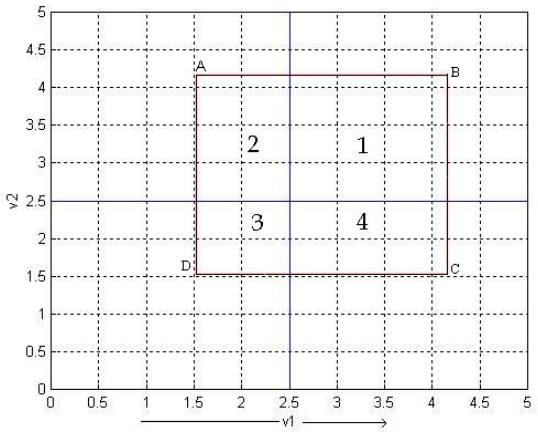

Figure 2: The range of possible velocities for either of the robots R1 and R2 shown along the x and y axis. The inner rectangle represents the area where the robots need to cooperatively find a solution Search is limited to quadrants 2 and 4 where robot actions are complementary

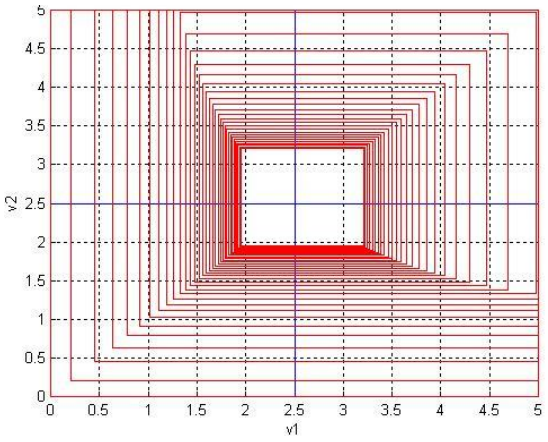

Figure 3: The evolution of conflict area with time. When it expands to occupy the entire space of possible velocities it marks the advent of the cooperative phase.

C The Conflict Propagation Phase

At times robots *R1* and *R2* are unable to resolve conflicts between them either individually or cooperatively because velocities that resolve the conflict between the two result in conflicts with other robots say *R3* and *R4*. In such a scenario *R3* and *R4* are requested to aid in solving the conflict between *R1* and *R2* and are said to be in an Indirect Conflict (*IDC*) with *R1* and *R2*. Conflict propagation can be a recursive process resulting in a generalized multiple tree like structure (figure 4a) where each node of the tree represents a robot and the links the direction of flow of conflicts. In figure 4a robots A and B have a direct conflict between them whose resolution leads to further conflicts with other robots. When A attempts the lower velocity v_1 it results in conflicts with C and D, while attempts at higher velocity v_2 results in conflict with E. Similarly B's attempt to resolve its conflict with A by adopting the lower velocity results in conflict with E while its attempt at v_2 results in conflict with F and G. Each node is divided along its center by a dashed line. Conflicts propagated to the left of the dashed line (the left sub trees of that node) are those that arise due to v_1 and those propagated to the right of the dashed line arise due to v_2.

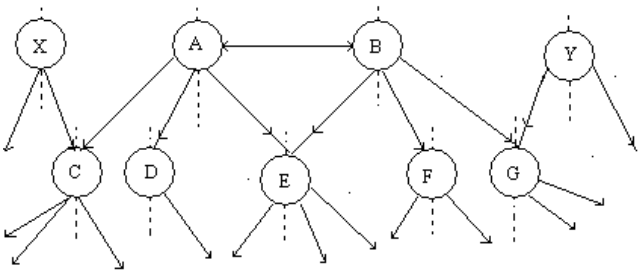

Figure 4a: Conflict propagation can result in a generalized multiple tree structure whose links represent the flow of conflicts between robots

The robot node C's attempt to resolve the indirect conflict can lead to further conflicts with other robots and hence a nested tree structure develops. Further a robot can receive requests to resolve conflicts from more than one robot such as the node C which receives requests from X and A. For real-time considerations the following restrictions are enforced for curtailing the growth of the tree. The node at the top level propagates a conflict maximally to two nodes one on its left and the other on its right. In the generalized case a top level node which has propagated its conflict to more than one node on its left or right would need confirmation from all the nodes on its left or right that they have been able to solve the requests before the top level node can adopt the corresponding velocity. Propagating only those conflicts that do not need more than one robot on left or right prevents longer waiting time. To prevent exponential growth of the tree an intermediate node propagates the conflict if and only if it requires the assistance of only one node below it. If an attempt to solve the conflict by the intermediate node results in further conflict with more than one robot then the node returns a failure to its parent node instead of further propagating the conflict. All conflicts are propagated only to two levels of hierarchy below the starting node. The restricted tree structure is shown in figure 4b.

In figure 4b robots R1, R2 and R3 at the first level of a tree like structure are the ones that are unable to resolve their *DCs* cooperatively and propagate it to R4, R5, R6 and R7 at the second level via the links shown.

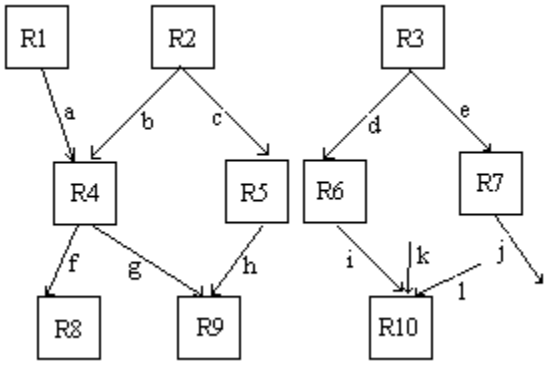

Figure 4b: The restricted tree structure.

The robots at the second level that receive requests can solve it or propagate to maximum of one level below if there are robots willing to accept or return a message to the transmitting robot indicating failure. A robot that receives requests from more than one robot to participate in its conflict such as *R9* receives requests from *R4* and *R5*, such as *R9* prioritizes the requests in order of time to collision of *R4* and *R5* with the robots with which *R4* and *R5* are in conflict. If any of the requests could be resolved, no further requests are entertained and no pending requests are attempted to be resolved. If a request does not get resolved it could be further propagated to robots that are willing to accept at the third level. For example in figure 4, *R4* propagates *R1*'s request to *R9* via link g and *R2*'s request to *R8* via link f. When a robot resolves a request all the links connecting to the robot that sourced the request is removed and all other pending requests with their links to the source are also removed. For example if *R9* manages to solve the request from *R4*, all links till source *R1*, namely 'g' and 'a' are removed as well as the links of the other pending request from *R5* sourced from *R2*, namely 'h' and 'c' are also removed. Robots at the third level of hierarchy do not propagate requests.

IV THE ALGORITHM

A robot can find itself involved in the following kinds of conflicts:

Mutual Direct Conflict (MDC): A pair of robots *R1*, *R2* are said to be in MDC with one another if *R1's* first direct conflict (DC) in reaction time t_r is with *R2* and *R2's* first DC in reaction time t_r is with *R1*.

Non-mutual Direct Conflict (NMDC): A robot R1 is said to be in NMDC with R2 when R1's first DC in t_r is with R2 while R2's first DC in t_r is however not with R1.

Indirect Conflict (IDC): A robot *R1* is said to be in IDC with *R2* if *R1* has no MDC or NMDC at that instant and has received a request for resolving *R2's* conflict with some other robot *R3*.

The broad steps of the overall algorithm are delineated below. Each step of the algorithm itself requires further decomposition into several modules and subtasks that are dealt very briefly here for brevity of space.

For any robot *Ri* do the following steps until *Ri* reaches its target:

1. If *Ri* has time nearest conflict within t_r with another robot *Rj* then do steps 1a to 1c.
   1a. If *Ri*'s conflict with *Rj* is of type MDC then resolve conflict through *ResMDC* module
   1b. If *Ri's* conflict with *Rj* is of type NMDC then resolve conflict through *ResNMDC* module.
   1c. If either of the steps 1a or 1b leads to further conflicts with other robots propagate conflicts to those robots
2. If *Ri* has received a request from *Rj* to solve *Rj*'s *DC* with *Rk* and *Ri* itself has no *DC* with any other robot execute step 2a
   2a. Resolve *Ri's* IDC with *Rj* through *ResIDC* module
3. Move *Ri* on its current direction with its current collision free velocity vi

ResMDC module: Let *R1,R2* be a pair of robots that have a *DC* between them. Let t_{1c}, t_{2c} be the time taken by them respectively to reach the point C shown in figure 1. The *ResMDC* module operates according to the steps mentioned in section IIIA and IIIB, where the individual and cooperative phases were described. First steps (i) and (ii) of section IIIA are tried and if they fail step (i) of IIIB is resorted if $t_{1c} < t_{2c}$ else step (ii) of IIIB is adopted.

ResNMDC module: This module is very similar to the *ResMDC* module except with a delay where the robot *R1* that has a *NMDC* with *R2* waits for *R2* to resolve its *MDC* before modifying its velocity.

ResIDC module: The robot *R1* that is in *IDC* with *R2* is requested modification of its velocity by *R2*. This modification by *R1* is such that it permits *R2* to adopt a velocity that resolves *R2's DC* with some other *R3*. *R1's* response to this request and requests fare along the lines mentioned in section IIIC earlier. Multiple requests to *R1* are considered according to their priorities based on the prioritization scheme stated in IIIC.

V SIMULATION EXPERIMENTS

In simulation snapshots shown in this section robots are modeled homogenously mainly for simplicity with maximum acceleration and deceleration of $2m/s^2$, maximum velocity of $5m/s$ and reaction time of $12s$. The simulations were developed on the Borland JBuilder IDE for Java.

Figure 5a shows an instant during the navigation of a system of five robots where robots 1 and 3 are unable to resolve their conflicts between them individually as well as cooperatively as cooperative solutions lead to indirect conflict with robot 4. Hence 1 and 3 propagate a request to resolve their

conflict to 4 thereby embarking on the conflict propagation phase as the last attempt to resolve their conflicts. Robot 4 accepts requests from 1 and 3 and is able to solve the request of 1 by modifying its current velocity such that 1 and 3 are able to avoid their mutual direct conflicts. This scenario is depicted in figure 5b where 4 moves faster in such a way 1 and 3 are able to avoid their mutual direct conflict.

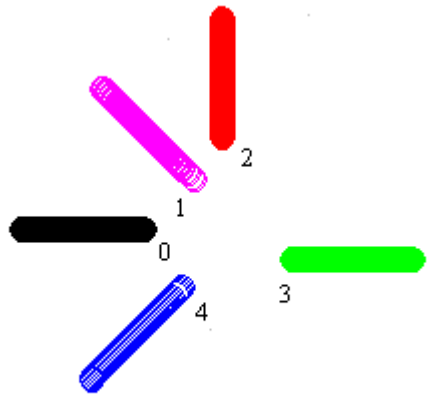

Figure 5a: A snapshot of a system of five robots

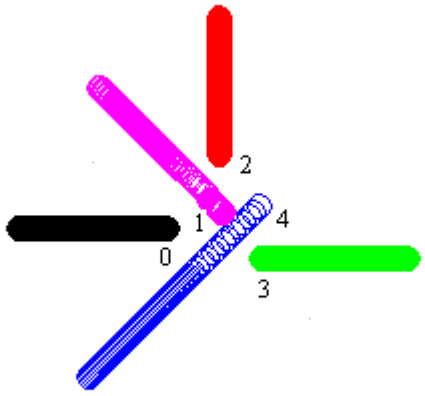

Figure 5b: Robots 1 and 3 propagate requests to resolve their conflicts to robot 4, which accepts the request and moves faster such that 1 and 3 are able to avoid their mutual direct conflict

Figure 6 shows the space-time evolution of trajectories for the robots of figure 5. The x and y axes indicate the regions in the x-y plane occupied by a robot every time it samples the environment. Robot samples of the environment in time are shown along the z axis as sampling instants. The five solid lines of the figure correspond to the trajectories of the five robots. The figure shows that the robot trajectories do not overlap in spacetime confirming that all collision conflicts were resolved by the algorithm.

Figures 7 and 8 show snapshots during navigation of a system of eight and thirty robots. The traces of the trajectories have not been depicted in figure eight. The number of resolutions that involved cooperation and propagation of conflicts were four and two for the eight bodied system. In case of the system with thirty robots the number of attempts to resolve by cooperation and propagation rose to eight and four respectively. Elsewhere we have reported [10] the effects of varying the parameters such as reaction time, maximum acceleration and velocity and the number of robots on the need to cooperate and propagate conflicts.

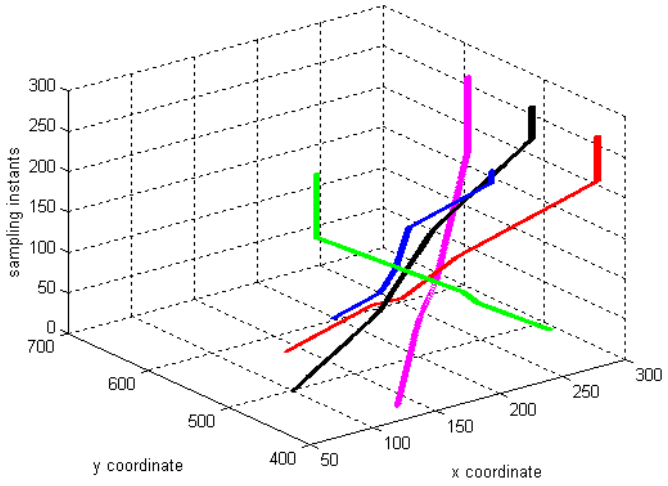

Figure 6: Spacetime evolution of trajectories for the five robot system.

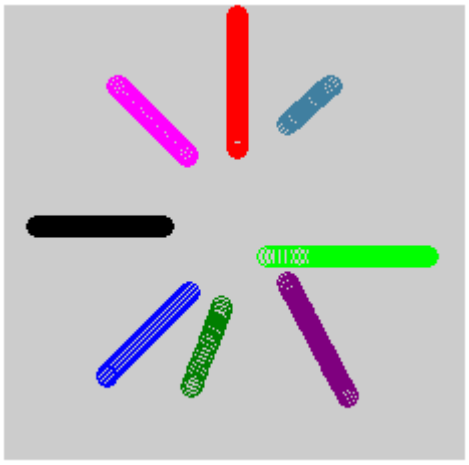

Figure 7: A snapshot during the navigation of a system of eight robots coming towards a common junction

VI CONCLUSIONS

A novel distributed three-tiered approach for coordinated cooperative collision avoidance for a multi robot system from a reactive navigation standpoint has been presented and the simulation results confirm the efficacy of the proposed model. Robots resolve conflicts at three levels namely, individual, cooperative and conflict propagation phases. The approach is particularly suitable for a large number of robots moving about in shop floors, factories, airports and the like where a-priori knowledge of the plans of all other robots in the system is not made available for every robot in lieu of computational complexity. Future areas of work include incorporating a

cooperative orientation control scheme and the investigation of various social cues such as benevolence and deception in conflict resolution in a multi-robot system.

ACKNOWLEDGEMENTS

This work is supported by AFOSR grant F49620-00-1-0302.

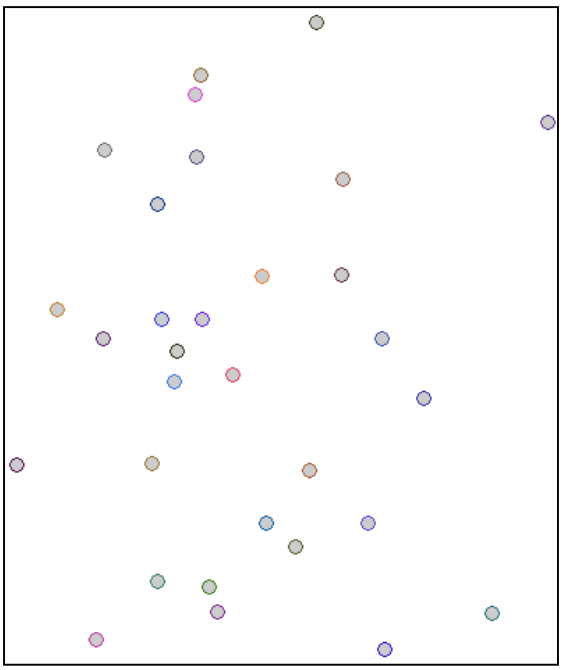

Figure 8: A snapshot during navigation of thirty robots

REFERENCES

[1] R. Alami, F. Ingrand, and S. Qutub, "A Scheme for Coordinating Multi-robot Planning Activities and Plans Execution", *Proc 13th European Conference on AI,* 1998

[2] R Alami, F Robert, F Ingrand & S Suzuki, "Multi-Robot Cooperation through Incremental Plan Merging", *Proc. IEEE Int. Conf. on Robotics and Automation,* pp. 2573-2578, 1995

[3] K. Azarm and G. Schmidt. "Conflict-Free Motion of Multiple Mobile Robots Based on Decentralized Motion Planning and Negotiation", *Proc. IEEE Int. Conf. on Robotics and Automation*, p. 3526-3533, 1997.

[4] J Barraquand, B Langlois & J.C. Latombe, "Numerical Potential Techniques for Robot Path Planning", *IEEE Trans on Syst., Man, and Cyb.,* 22(2):224-241, 1992.

[5] S Carpin and E Pagello, "A Distributed Algorithm for Multi-robot Motion Planning", *In Proceedings of the fourth European Conference on Advanced Mobile Robotics*, 2001

[6] C.M. Clark, S.M. Rock and J.C. Latombe, "Motion Planning for Mobile Robots using Dynamic Networks", *Proc. IEEE Int. Conf. on Robotics and Automation*, 2003

[7] M Erdmann & T. Lozano-Perez, "On Multiple Moving Objects", *Proc. IEEE Int. Conf. on Robotics and Automation,* 1986

[8] A. Fujimoru, M. Teramoto, P.N. Nikiforuk and M M Gupta, "Cooperative Collision Avoidance between Multiple Mobile Robots", *Journal of Robotic Systems 17(7),* 347-363, 2000

[9] Y. Guo and L. Parker, "A Distributed and Optimal Motion Planning Approach for Multiple Mobile Robots", *Proc. IEEE Int. Conf. on Robotics and Automation,* pp: 2612-2619, 2002.

[10]. K. Madhava Krishna and Henry Hexmoor, "Towards Quantification of the need to Cooperate between Robots", *Proceedings of PerMIS, 03* (*Performance Metrics in Intelligent Systems)*, Sep. 16-18, 2003

[11] K.M. Krishna and P.K Kalra, "Detection Tracking and Avoidance of Multiple Dynamic Objects", *Journal of Intelligent and Robotic Systems*, 33(3), 371-408 Apr 2002, Kluwer Academic

[12] S.M. LaValle & S.A. Hutchinson, "Optimal Motion Planning for Multiple Mobile Robots having Independent Goals", *IEEE Trans. On Robotics and Automation*, 14:912-925, 1998

[13] T. Li & H. Chou, "Motion Planning for a Crowd of Robots", *Proc. IEEE Int. Conf. on Robotics and Automation,* 2003

[14] V.J. Lumelsky & K.R. Harinarayan, "Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model", *Autonomous Robots*, 4:121-135, 1998.

[15] P. Srivastava, S. Satish and P. Mitra.: A distributed fuzzy logic based *n*-body collision avoidance system, in: *Proc. of the 4th Internaionalt. Symposium on Intelligent Robotic Systems*, January 1998, pp. 166–172, Bangalore.

[16] C.W. Warren, "Multiple Path Coordination using Artificial Potential Fields", *Proc. IEEE Int. Conf. on Robotics and Automation,* pp 500-505, 1990