ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

A Review: Priority Based Motion Control of Multiple Robot Systems

Abhay .A. Deshpande[1], Dr. Nataraj.K.R[2]

Asst.Prof, Dept of ECE, RV College of Engineering, Bangalore, Karnataka, India[1]

Prof &HOD, Dept of ECE, SJB Institute of Technology, Bangalore, Karnataka, India[2]

Abstract: Networks formation between multiple robots and establishing a communication link is more important. Using its own motion planner and based on the current world model, robot constructs a collision free trajectory. The technique employed for trajectory planning is kinodynamic randomised motion planner techniques. The motion planning is made more efficient and feasible by using priority systems. Employing priority systems helps the robots to act spontaneously when a problem is encountered. Based on the priorities, a robot plans its path. A robot with low priority has to re-plan its path allowing the robot with high priority to continue along its path. There are two types of priority systems discussed in this paper i.e. user-defined and robot-determined priority system. Among the two priority systems discussed, robot-determined priority system gives better results which are later showed using a test platform. This paper discusses a new motion planning algorithm for multiple robots which can plan trajectories in real time and can be implemented in the physical world.

Keywords: kinodynamic randomised motion, priority systems, user-defined priority and robot-determined priority.

I. INTRODUCTION

When we think of mobile robots, first thing that strikes our mind is the robot like *Mars Exploration Rover Mission (MER)* used in space programs. It involves two rovers – *Spirit and Opportunity*, which explores the planet Mars. These kind of complicated robots require precise motion planning as they are going to be deployed in unknown environments. A recent space program, *Chandrayaan – 1*, also involved sophisticated robots like Moon Mineralogy Mapper (M3), which characterizes and maps lunar surface mineralogy in the context of lunar geologic evolution [1].

Another example can be RMS (Remote Manipulator System) which is employed in industries. They are usually stationary and hence it is very necessary to provide limbs to them and thus providing an increased degree of autonomy.

These groups of robots are built for a particular goal with many operators working on it (Remote Operated Vehicle). The motions of these robots require complex planning in order to accomplish the given tasks autonomously. Their goals should be achieved without colliding among themselves or with the static obstacles or with the dynamic obstacles. The system of sensors fails to give an overall knowledge of the unknown or partially known environment. Also, another option which is continuous inter-robot communication also fails and is unfeasible due to many practical reasons. Only few robots which are in the communication range of each other can share information like their goals and world models [2].

This paper concentrates on motion planning dynamic robots, the formation of networks among them. These networks or dynamic links are able to: 1, establish the communication link between the two robots 2, after establishing the communication link, sharing of information on its environment, destination; and finally, 3, after forming the links, it is time to plan the collision free path for all the robots.

In the following sections, an overview of the above mentioned approach is discussed.

II. PLANNING IN DYNAMIC NETWORKS

A. Network Formation and Establishing the Communication Link

If two robots say, *a* and *b*, are well within the range of communication of the robots, a communication link is formed between them. When this link is formed between two or more robots a *network of robots* is formed. These robots can then share

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

information about their environment, destination and the information required to plan the path through direct (figure 1) or indirect communication links. Indirect communication links can be through other robots within the network (Figure 2). It is important to note that robots can communicate only with the robots of the same network and not with the robots of other networks. The figure1 shown below depicts the formation of links and networks. [3].

As the robots are dynamic, they may leave a network and form a new network. The figure2 below depicts the dynamic behaviour of the robots.

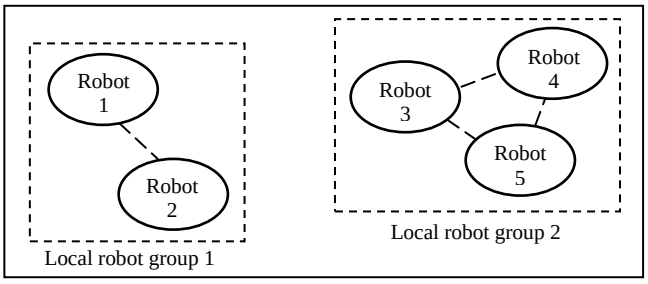

Figure 1: Shows two local robot groups, where local group 1 has two robots – 1 & 2 and local group 2 has three robots – 3, 4 & 5. Dashed lines indicate that the two robots are well within the communication range of each other and there exists a communication link. There are four links. Networks are nothing but local groups. There are two networks in the above figure.

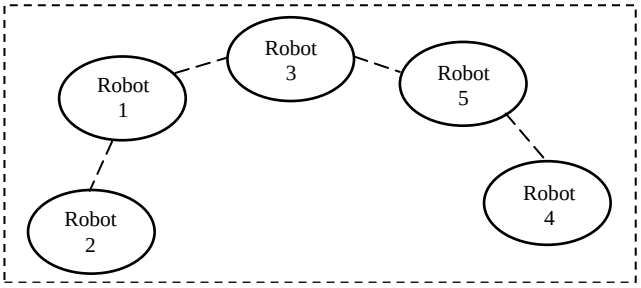

Figure 2: The robot 3 in figure 1 moved into the communication range of robot1 forming a link. Robot 4 moved out of communication range of robot 3 disconnecting the link between them. As all robots are linked indirectly with each other, they together form a single local group or network.

As the robots move in and out of a network, their path cannot be predefined. Hence path should be planned simultaneously as the robot moves and form new networks. The path should be planned in such a way that it should not collide with the robots which are already in the network. To enable such a kind of planning, the robot entering a network should be provided with the local information of the world model and local robots. This is discussed in the next section.

B. Planning Process

To precisely plan a collision free path, each robot in the network should know about all other robot's goal, priority number and about the local world model. To enable this, a protocol is required which can share this information and control the process of planning. This makes parallel motion planning for each and every robot in a network possible.

It is important to note that a priority system is employed in this paper in which, each robot is given a priority number. This priority order determines how robots plan around each other in the network which is discussed later.

The motion planning process can be triggered if any one of the following situation occurs:

- When a new network is formed due to the robots from different networks entering into the communication range of the other.
- When there is a recognizable change in the environment or world model. For example, if there is a new obstacle in the previously planned path.
- When the destination of few or all robots are changed.

If there is a new obstacle in a robots path, the motion planner in that robot is triggered. The robot selects a new milestone.

If a new destination is requested for a robot, the motion planner in each robot plans a collision free path towards the new destination.

If a new robot enters the network, the motion planner of each robot starts exchanging the information about each robot i.e. its goal, priority number, milestones of its collision free path etc is shared with all other robots. Once this information is shared, each robot has updated world model [4].

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

Each robot then compares its priority number with others. If a robot has lower priority number, its centralized motion planner uses milestones from higher priority robots and plans its path which does not intersect with the same of the higher priority robots. While the lower priority robot plans its new collision free path, higher priority robot will continue along its old path as low priority robot will avoid it.
The figures below describes how exactly the planning with priority system is executed. Figure has the paths of the robots in the left and the robots communication range and links in the right.

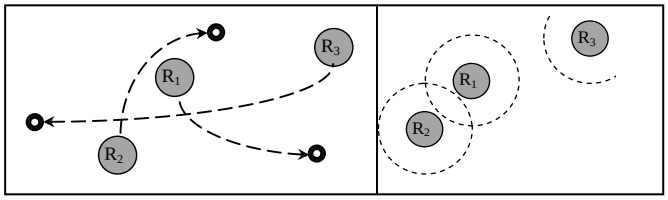

Figure 3a: There are three robots R_1, R_2, R_3 with priority order $R_1>R_2>R_3$. Initially, only R_1 and R_2 are in a network with communication links established between them. R_2 and R_1 having already planned the collision free paths to their goals, move in their respective paths until any other robot (R_3 in the above figure) enters the communication network of R_1 and R_2.

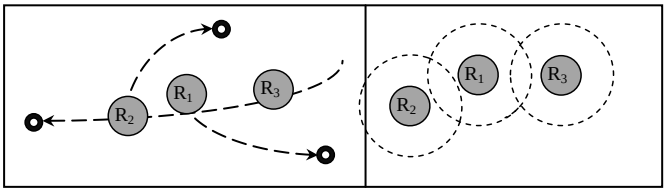

Figure 3b: Here R_3 enters the communication range of R_1 and R_2. As soon as it enters, all robots compare its priority with the other robots. One with the highest priority (R_1) moves along the same path. Robot with next highest priority (R_2) will plan a new path if its path is overlapping with the one having higher priority than it (In this case, path of R_2 is not overlapping with that of R_1. Hence no new path is planned). This process repeats until all robots plan a collision free path. R_3, having the least priority and its path overlapping with that of R_1 and R_2, plans a collision free path (shown in figure 3c) which doesn't disturb the motion of R_1 and R_2.

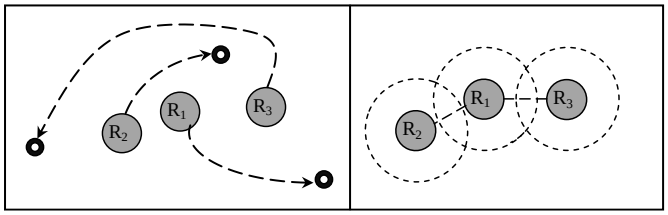

Figure 3c: Shows the new path of R_3. As explained above, R_1 is following its original path and R_2 having its path away from that of R_1 also follows the same path.

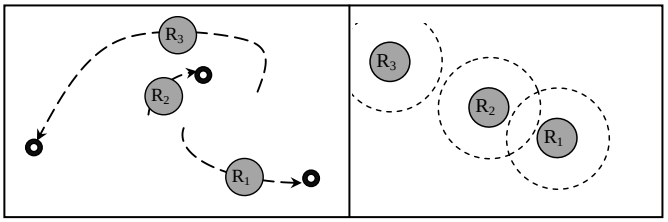

Figure 3d: Shows the robots moving towards its goal locations in its path planned based on the priority system. R_3, moves out of communication range of R_1 and R_2 and thus breaking the communication link.

C. Priority Systems

Basically there are two types of priority systems concentrated in this paper.

- User-defined priority system
- Robot determined priority system

1) User-defined Priority System

From the discussions in the previous section, it is clear that each robot have a unique priority number. To practically implement priority system, each robot should know all the robots in its network and also its priority order. More importantly, it should have a priority list of all the robots in the network.

In this priority system, as the name itself suggest, the priority number is given by the user. It can be given randomly or based on its importance initially while designing the system.

In this priority system, if a robot encounters a problem and re-plans its path, then it must broadcast its new path to all the local robots having less priority so that they can re-plan. To keep track of all robots, we make use of two types of list. One, the list of robots having higher priority number i.e. HIGH list and other, the list of robots having lower priority number i.e. LOW list. The algorithm for the motion planner of each robot can be as discussed below.

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

Step 1: Whether there is any new robot in the communication range of a robot? Also check whether any robot has moved out of the communication range of the robot.

Step 2: If there is a new robot, then check its priority number. Else, continue checking for the new robot.

If any robot has moved out, remove it from the corresponding list.

Step 3: After checking the priority number of the new robot, add the robot to the corresponding list depending on the priority number. In other words, if the priority number is high, then add the robot to the high list else, add it to the low list.

Step 4: Refresh the two lists and arrange the robots in the decreasing order of the priority number for easier access.

Step 5: If the robot is added to the LOW list, there is no need for planning as the robot encountered will plan its own collision free path considering this robot. If the robot is added to the HIGH list, then a new collision free path should be planned considering all the robots in the new HIGH list.

Step 6: After re-planning, the robot has to broadcast its plan to the robots in the LOW list so that they can re-plan their paths avoiding the robots in their HIGH list.

The summary of above discussed algorithm is given below.

If any new robot is found in the communication range of the robot (say A), it first checks the new robot's priority. If the robot has less priority, it is added to a list called L list (low list) and if it has higher priority, it is added to H list (High list).

If the new robot has higher priority, robot A re-plans a new path to avoid collision with those in H list. Then, it broadcasts its new path to all robots in the L list so that they can re-plan. [5].

If the new robot has less priority, robot A just adds it to L list.

2) Robot-determined Priority System

The limitation of user-defined priority system is that robots with high priority number doesn't have to plan much and as the priority number decreases, the complexity of the milestones and hence the collision free path also increases. This makes the planning process slow and less efficient.

To overcome the above limitation, we use robot-determined priority system where the priority of the robot is decided based on its local area and thus ensuring equal distribution of planning responsibilities among them.

Here, unlike user-defined priority system, the priority of the robot is not determined by the user initially. It is prioritised based on the number of obstacles (static or dynamic) in its local area (communication range) and the robot with more number of obstacles in its local area is given the higher priority. [6].

The priority number of each robot is decided by the number of obstacles in its range and is given by,

$$\text{Priority number, } P = \alpha(i) + \beta(i) + 10^{-3}\,\gamma\ ; \qquad i = 1, 2, 3 \ldots, \text{total number of robots}$$

Here, α (i) is the number of robots in the local area of the i^{th} robot, β (i) is the number of robots in the local area of i^{th} robot. This makes the priority of the robots having busy environments high. Hence, the robots with most free environment re-plan.

γ is the unique identification number of the robot. 10^{-3} times γ is used to make sure that robots having same number of obstacles in their local area get slightly different priority number.

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

The basic algorithm of the robot-determined priority system is as discussed below. As in case of user-defined priority system, even robot-determined priority system makes use of two lists to keep track to all robots.

Step 1: Whether there is any new robot in the communication range of a robot? Also check whether any robot has moved out of the communication range of the robot.

Step 2: If there is a new robot, then, *calculate its priority number by determining the number of robots in its local group, number of obstacles (static and dynamic)*. Else, continue checking for the new robot.
If any robot has moved out, remove it from the corresponding list.

Step 3: After calculating the priority number of the new robot, add the robot to the corresponding list depending on the priority number. In other words, if the priority number is high, then add the robot to the high list else, add it to the low list.

Step 4: Refresh the two lists and arrange the robots in the decreasing order of the priority number for easier access.

Step 5: If the robot is added to the LOW list, there is no need for planning as the robot encountered will plan its own collision free path considering this robot. If the robot is added to the HIGH list, then a new collision free path should be planned considering all the robots in the new HIGH list.

Step 6: After re-planning, the robot has to broadcast its plan to the robots in the LOW list so that they can re-plan their paths avoiding the robots in their HIGH list.

The algorithm is almost similar to user-defined priority system except the 2nd step. It can be summarised as below.

When a new robot is encountered, the robot A calculates the priority of it by the method discussed above and adds the new robot to the corresponding list. The trajectories are then planned as discussed in user-defined priority system.

III. PLANNING EXPERIMENTS

The main objective of this discussion is to create a planning algorithm which can plan the trajectories for the robots in real time. Simulated results and experiments are shown below which shows the ability to build collision free trajectories for many robots in the complex environments efficiently. Initially, the micro – autonomous rovers test is simulated and later implemented.

A. Test platform

To test/simulate the rovers in two dimensional works space, Micro – Autonomous Rovers (MARS) test platform is used. A rectangular (3m x 2m) flat granite table is used as the platform.

The geometry of the robot is cylindrical in shape having two wheels allowing them to rotate. For planning process, an off – board motion planner is provided for each robot. After planning, the control signals are processed by off – board processor and are transmitted to the respective robots through remote control signal.

B. Vision System of the Test Platform

To determine the position of the obstacles, providing position sensor for each robot is not feasible. Hence, to keep track of the positions of the obstacles and the robots, a vision system having an overhead camera with infrared filters is adopted. They capture information on the position of each robot by detecting LEDs present on top of each robot. These cameras are integrated to a vision system processor which determines the position and velocity of each robot by using the information provided to it. The vision system is updated at the rate of 30Hz.

C. User Interface

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

A Graphical User Interface (GUI) designed in Java is used in this test platform. The advantage of this interface is that it provides a simple, global view of the world model (top view). The user interface is user friendly as it incorporates drag and drop system to set a robot's goal. Once the goals are set, they are updated to the motion planner to plan the trajectories. [7].

D. Communication

As all the processing work is done off – board, communicating with robots and processor plays an important role. It requires many computers with local area network (LAN) and the communication is established by Real Time's Network Data Delivery Service (NDDS) software.

NDDS employs subscribes/publish architecture. To start the communication, the sender will publish a request. The robots that subscribe to it will be added to its communication range and thus establishing a communication link.

E. Communication between Nodes

From the above discussion, we know that NDDS employs publish/subscribe architecture.

As vision system is the source of information for all other *nodes (GUI, motion planner and controller)*, they need to subscribe to it to develop a communication link with it. As all nodes are linked to vision system, an indirect communication link is created between the nodes.

GUI subscribes to the vision system as it needs information on the position, velocity of the robots and also the goal locations of them to display the same graphically to the user. It also needs to send information to motion planners regarding the new goal location requested by the user via vision system.

Motion planner subscribes to the vision system as it needs to construct new trajectories if a problem is encountered or if a new goal location is requested by the user via GUI and needs to publish the same to the controllers. Motion planners only publish the milestones of the paths as it is more efficient than publishing the whole trajectory.

Controllers subscribes to the vision system as they need the data published by the motion planner and send the corresponding control signal to the robot through remote control signals.

F. Synchronizing the Clock

For the real time parallel planning to happen, all robot's processor should be synchronised i.e. the clocks of all processors should start at the same time. To make this possible, an enable signal is sent from the GUI to the other nodes indirectly connected to GUI via NDDS. When the signal is received, the clocks of the nodes will be reset (start from time zero) ensuring the synchronization.

G. Implementation using GUI

A simulator built in java is used to demonstrate how the motion planning is executed in the physical word.

To start with, a simple world model with only one robot and two obstacles (a moving and a static) is considered. In the figure 6, smaller gray dot represents robot, cross – hairs represents the goal and black dots denotes the obstacles. Robot first plans an obstacle free path to its goal considering only those obstacles in its communication range (figure 6(a)) and starts traversing the same (figure 6(b)). As it moves, it encounters the moving obstacle (figure 6(c)) and re-plans its path avoiding the obstacle (figure 6 (d)). As it approaches the goal location, it encounters the static obstacle (figure 6(e)) and again re-plans its path avoiding the obstacle (figure 6(f))

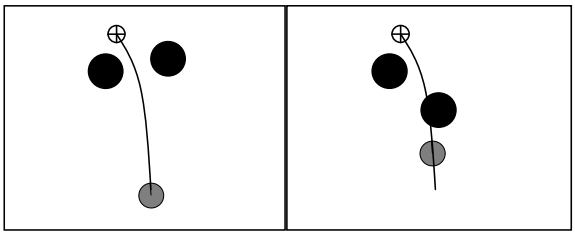

Figure 6(a)

Figure 6(b)

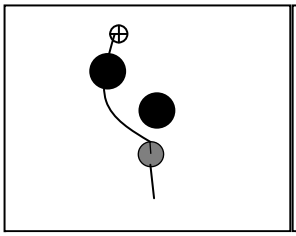

Figure 6 (c)

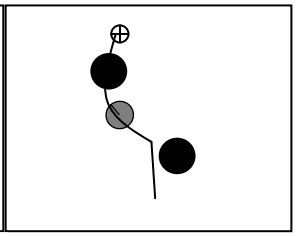

Figure 6 (d)

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

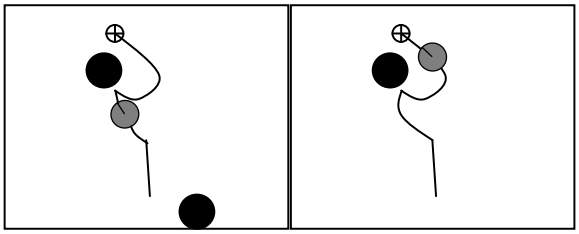

Figure 6 (e) Figure 6 (f)

To test the planning process in mor e position of the obstacles are selected randomly. The velocities of t periment is repeated by varying the number of robots and obstacles. The

TABLE I
SIMULATION RESULTS

Trail no.	1	2	3
No. of robots	5	10	15
No. of static obstacles	5	5	0
No. of dynamic obstacles	5	0	0
Average no. of plans	55	128	215
Average plan time (ms)	38.55	96.3	4.62
Average maximum plan time (ms)	102.55	276.5	44.44
Average 1st plan time (ms)	59	44.6	44.6
Average re-plan time (ms)	8.6	5.6	15.3

Table – 1 depicts a world model having 15 bodies (robots & obstacles) in three different combinations. It is also evident that average maximum plan time is very much larger than average re-plan times. This is because, when a robot needs to re-plan, it first checks whether the current path is collision free or not. If it is, then, the robot continues with the same path and new path is created and thus consuming less time.

H. Comparison of User-defined and Robot-determined Priority Systems

Table – 2 compares user-defined and robot-determined priority system with a world model having 15 robots. The time values have a significant difference. Average plan time of robot-determined priority system is 3.63ms less than that of user-defined priority system. Also, average maximum plan time in RDPS is 29.38ms less than that in UDPS. Hence, clearly, planning is faster in RDPS compared to UDPS ensuring real time planning.

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

TABLE 2
COMPARISON OF UDPS AND RDPS

Trail type	User-defined	Robot-determined
No. of robots	15	15
No. of static	0	0
No. of dynamic	0	0
Average plan time	4.6	1
Average	44.4	15.1

I. Simulation No. 2

This world model consists of 10 rovers and 5 static obstacles. The figure 7 below shows the rovers and its motion as viewed in GUI. Here, smaller black dot represents rovers, cross – hairs represents goals and big black circles represent obstacles. The lines denote the path of the particular robot. The path changes as the robot encounters problems. [8].

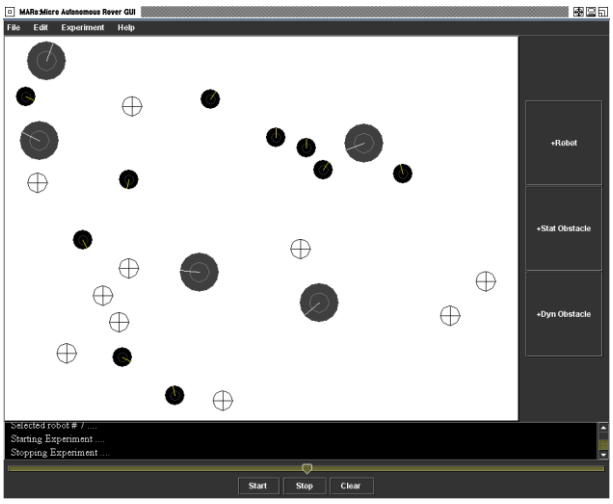

Figure 8 (a): Shows the goal location, initial position of robots and obstacles that user requested

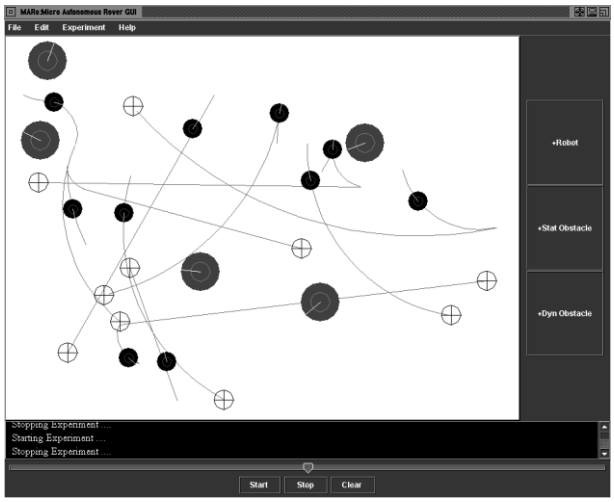

Figure 8 (b): Shows the first planned trajectory of the robots towards the goal location

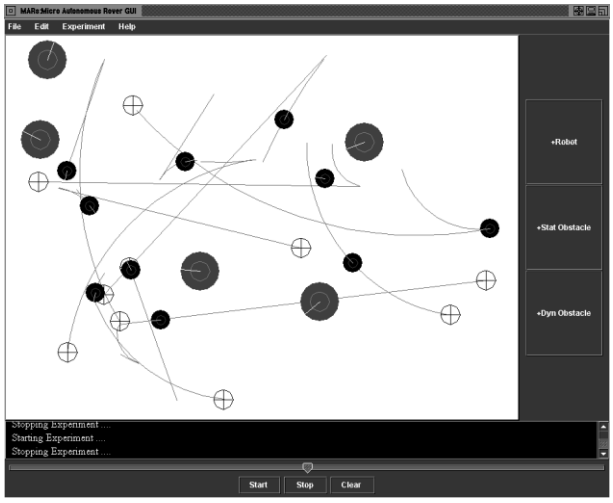

Figure 8 (c): Robots that have encountered problems re-planning a new collision free path

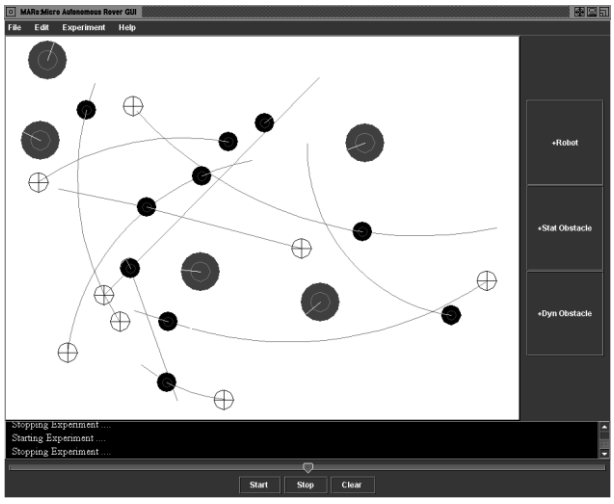

Figure 8 (d): Robots moving in their new collision free path

ISSN: 2319-8753

International Journal of Innovative Research in Science, Engineering and Technology
Vol. 2, Issue 1, Janauary 2013

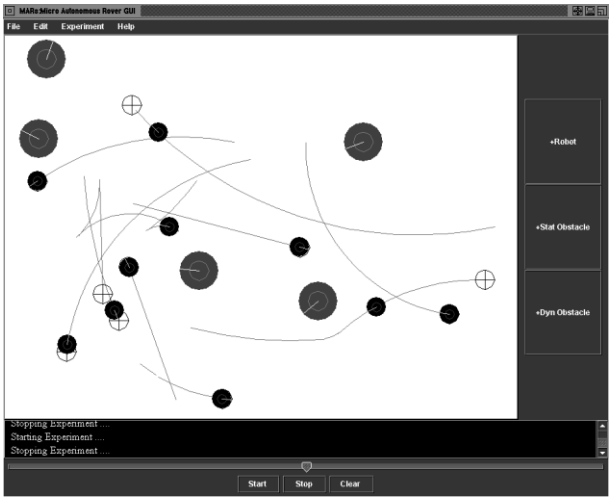

Figure 8 (e): Robots re-planning again if necessary approaching their respective goals

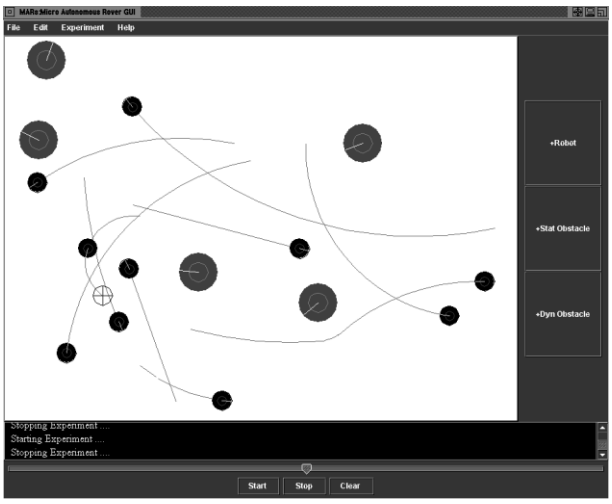

Figure 8 (f): Robots at their goal locations

IV. CONCLUSIONS

It has been observed from the above experimental results that the motion planning algorithm discussed in this paper has executed its objective successfully. The algorithm could even handle complex randomized world models which had around 15 bodies including robots, static and dynamic obstacles. The average plan time was in milliseconds and the robots could re-plan almost spontaneously if any obstacle was encountered indicating the ability of the algorithm to plan in physical environment. The algorithm will even work for three dimensional test platforms.

The success of the priority system is even more highlighted when there was a large difference in planning time between user-defined priority and robot-determined priority. Clearly, robot-determined priority had the edge as it was faster by around 70% than user-defined priority in planning. It also equalised the planning responsibilities on each robot.

V. FUTURE WORKS

In future, the robot-determined priority system will be made more efficient by introducing a new parameter i.e. velocity. In this paper velocity of each robot was made constant but, that's not the case in physical environment. Hence, an effort will be made to calculate a robots priority number based on its velocity. Also, a serious effort will be made to increase the communication range of the robots which will make it more feasible for real world implementation. Each robot should be giving a separate vision system as an overhead camera limit the area of operation.

REFERENCES

[1] C. Clark & S. Rock, "Randomized Motion Planning for Groups of Nonholonomic Robots," *International Symposium of Artificial Intelligence, Robotics and Automation in Space*, 2001.

[2] D. Hsu, R. Kindel, J. C. Latombe, and S. Rock. "Randomized Kinodynamic Motion Planning with Moving Obstacles," in *Workshop on the Algorithmic Foundations of Robotics*, 2000.

[3] Christopher M. Clark, Stephen M. Rock, Jean-Claude Latombe, *Motion Planning for Multiple Mobile Robot Systems using Dynamic Networks.*

[4] D. Hsu, J. C. Latombe, & R. Motwani, "Path planning in expansive configuration spaces," *Proceedings of the IEEE International Conference on Robotics and Automation*, pg 2719-2726, 1997.

[5] M. Bennewitz, W. Burgard & S. Thrun. Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems, *Proc. Int. Conf. on Robotics and Automation*, 2001.

[6] C. Clark, T. Bretl & S. Rock, "Kinodynamic Randomized Motion Planning for Multi-Robot Space Systems," *Proceedings of IEEE Aerospace Conference*, 2002.

[7] D. Hsu, R. Kindel, J.C. Latombe, & S. Rock. Randomized Kinodynamic Motion Planning with Moving Obstacles, *Int. J. of Robotics Research*, 21(3):233-255, March 2002.

[8] Lee, Lee, & Park. Trajectory Generation and Motion Tracking for the Robot Soccer Game, *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, p. 1149-1154, 1999.

Biography

Abhay .A. Deshpande is completed B.E, M.Tech and currently pursuing Doctor of Philosophy in Visvesvaraya Technological University. His interests includes Robotics Control, Wireless Communications, Sensor devices.