Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 133 (2018) 424–431

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Robotics and Smart Manufacturing (RoSMa2018)

UAV-Robot Relationship for Coordination of Robots on a Collision Free Path

Ashish Upadhyay[a]*, Kushashwa Ravi Shrimali[a]†, Anupam Shukla[b]

[a]*DSPM International Institute of Information Technology, Naya Raipur, India*
[b]*ABV Indian Institute of Information Technology and Management, Gwalior, India*

Abstract

The use of UAVs and robots in the commercial purposes has increased significantly in last few years. Along with the research being conducted for both UAVs and mobile robots individually throughout the research community, there has been a significant amount of work going in the field of the relationship of UAVs and mobile robots. In this paper, we present an algorithm to establish a relationship between a UAV and n robots. The target is to successfully lead n robots to n different goal positions. None of the goals is shared by more than one robots. UAV will be acting as the leader of robots and will have the bird-eye view of the environment. The UAV will calculate an efficient path plan for every robot that will help the robot to reach the goal by avoiding the static obstacles as well as the inter-robot collision during the movement of robots. We have simulated our algorithm in software using Python as well as MATLAB and also have compared two pathfinding algorithms wiz. A* and PRM for finding the path for a robot from its source to the given destination. The result establishes the efficient coordination between UAVs and robots for targeting the goal.

© 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the scientific committee of the International Conference on Robotics and Smart Materials.

Keywords: UAV; Mobile Robots; A*; PRM; UAV-Robot Relationship;

* Corresponding author. Tel.: +91-79-79-067989.
E-mail address: ashish15100@iiitnr.edu.in
† Corresponding author. Tel.: +91-82-09-956918.
E-mail address: kushashwa16100@iiitnr.edu.in

1877-0509 © 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Peer-review under responsibility of the scientific committee of the International Conference on Robotics and Smart Materials.
10.1016/j.procs.2018.07.052

1. Introduction

We are aware of the importance of autonomous UAVs and robots, and also the drastic increase in their popularity in the last few years [1-6]. Instead of performing a whole task by using only single robot or UAV, the research community is being attracted towards the idea of using a group of autonomous vehicles to achieve the same goal by dividing it into subsets [1]. The groups can consist of UAVs and robots and can perform the tasks that seem to be incapable of a human being. UAVs and robots are being extensively used in military purposes, offensive as well as defensive activities. Some of the popular works are the use of robots for searching mines and also for search and rescue operations [3]. UAVs are being used by militaries in war fields to perform surveillance operations that can be very difficult for a human being even risking their own life.

There have been several works to demonstrate the relationship between UAVs and robots. Ulun et al. used a team of some robots and a UAV to mark a dynamic target which will be identified by the UAV and then asking the robots to surround the target with a circle of pre-defined radius. The robots will be guided by a virtual leader, i.e., the projection of the UAV [1]. Chaimowicz *et al.* showed how UAVs and robots can be used in urban areas to exchange information in a fast manner [4]. Tanner gave an idea of making a team of UAVs and robots to detect the moving target by using the decentralized algorithms combined with obstacle avoidance algorithms [5]. Luo *et al.* presented another idea of teamwork of UAV and robots in the case of a hazardous chemical accident. In this work, UAV searches for the damaged area and communicate that target to robots to carry out the rescue operation [6].

In this paper, we have proposed a novel method to accomplish a goal by using a team of n robots and a UAV where UAV will be acting as a leader of the flock of robots. In our case, the goal will be to coordinate n robots to n goals by avoiding the obstacles present in the environment as well as the inter-robot collision during the motion of robots. Results are successfully verified by simulating the proposed algorithm in MATLAB. Also, we have used two different state-of-the-art pathfinding algorithms and compared them on the basis of time taken. GA has been also widely used for Robot path planning. Mathematical analysis of GA has been described in [7-10].

In section 2 the problem definition is given. We have presented every single detail of the formulated problem. In section 3 our proposed methodology is explained to solve the defined problem. The challenges faced in solving the problem is also discussed there. Two algorithms which are used for individual path planning is also discussed in this section. In section 4, the experimental results are presented in order to verify the effectiveness of proposed idea. In section 5, our work is concluded on some remarks and possible future works.

2. Problem Definition

Consider a UAV in a 3D environment, it is having the bird-eye view of the whole environment. The environment consists of n robots and corresponding n goals. The requirement is to develop a path plan for all the n robots to reach one identical goal each. The decision will have to take by UAV that which robot will move towards which goal like, R_i should move towards G_j, where R_i represents i^{th} robot and G_j represents j^{th} goal. One robot will occupy only one goal and no goal is occupied by two robots. Then UAV will calculate a collision-free path for these robots to reach the goal.

The obstacles are pre-defined and are static in nature. The plan should be as good that it can avoid the collision with these obstacles and also avoid the inter-robot collision during their movement. The goal is to minimize the total cost c taken by each robot to reach its goal. c is defined in equation (1).

$$c = \sum_{i=0}^{n-1} d_i \tag{1}$$

$$d_i = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \quad (2)$$

Here, n is the number of robots/goals. (x_1, y_1) is the position of i^{th} robot and (x_2, y_2) is the position of i^{th} goal.

The path plan will be a queue of points of the environment which will represent that either a robot will move in next timestamp or will wait. If the next point in the queue is same as of the previous one, it means that the robot will have to wait at its current position in next timestamp for 1 unit.

In fig. 1 we can see that how the UAV is having the aerial view of the whole environment. It can detect that where robots are located and where the goals are located.

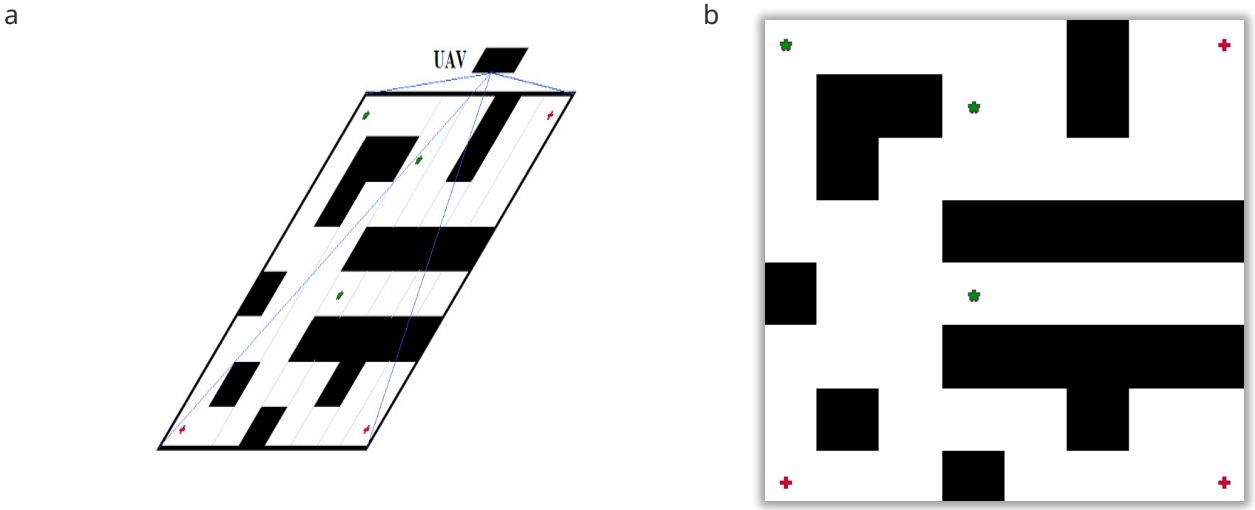

Fig.1. UAVs bird -eye view of an example environment. Red dots represent the robots and green dots represent the goals, (a) 3-D view; (b) 2-D view.

3. Proposed Methodology

In this section, we describe the working of our proposed algorithm. The working of the whole algorithm can be divided into three phases. Each phase is independent of other if previous one is completed successfully. We would consider the obstacles as static in the environment.

The UAV can have the bird-eye view of the environment as shown in the Fig. 1 and based on that a 2D map is generated like Fig. 2. The camera mounted on UAV can give the whole aerial view by which the map can be generated for robot path planning. The UAV can recognize the position of each robot and each goal.

This part can be divided into 3 phases. Working of each phase is explained here.

3.1. UAV calculates the Robot-Goal combination

In this phase, the UAV will have the aerial view of the whole environment. The UAV will first divide the environment into a rectangular grid. Now, UAV will identify the positions of the robots and goals in that grid environment. After identifying the positions, the UAV will create a $n \times n$ 2D-matrix (where n is the number of robots and goals) which will represent the straight line distance of every robot from every goal. For example, the

2D-matrix for the environment given in Fig. 2 is given in Table 1. Here, there are 3 robots and 3 goals. Goals are represented by green boxes and robots are represented by red boxes.

Table 1. Distance Matrix (The environment is converted into 8x8 grid)

	R_0	R_1	R_2
G_0	6	7	8
G_1	7	14	7
G_2	9	10	5

Every entry a_{ij}, in the given matrix, is the Euclidean distance calculated between R_i and G_i.

$$a_{ij} = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \tag{3}$$

Here, (x_1, y_1) is the position of i^{th} robot and (x_2, y_2) is the position of j^{th} goal.

The decision of robot-goal combination will be taken on the basis of the cost function that will be the sum of the Euclidean distance of all the robots from the goal. The cost function is same as defined earlier.

Now, the UAV will generate all the possible permutations to robots and goals and will check that which combination is having the least cost. For example, here the combination of $(R_0 \rightarrow G_1;\ R_1 \rightarrow G_0;\ R_2 \rightarrow G_2)$ will be the best combination having the least cost of $19\ (7 + 7 + 5)$.

3.2. UAV calculates the path plan for every robot

In our algorithm, we are using the coupled approach for path planning. That means UAV will plan the path for multiple robots by avoiding the collision between them. The plan of a robot will be a queue of points in the environment which will either movement in next timestamp (if the next point is different) or wait in next timestamp (if the next point is same). After having the plan of their respective path, robots can move accordingly.

This part can also be divided into 2 sub-phases

3.2.1. Individual path planning

First, individual paths will be calculated for every robot from its source to the destination. At this point of time assumption will be taken that no other robot exists in the environment. Likewise, optimal path for every robot will be calculated from its respective position to the respective goal. In our algorithm, we used two path planning algorithms to calculate the individual path for every robot [11, 12]. The two algorithms are:

- *A-star search (A*)*

A* is most popular best first search which is widely used to find the shortest and optimal path between the source and destination. It solves the problem by searching all the possible path to find the goal. It expands the nodes by using cost function $f(n) = h(n) + g(n)$, where n is a node present in the path, $g(n)$ is the actual cost to reach that node from the source and $h(n)$ is the heuristic function, the estimated cost from that node to the goal. Implementation of A* algorithm is done by use of priority queue or heap. A* algorithm always searches optimal

path. The A* algorithm takes a graph as an input and explores, one by one, all the regions, finding the shortest path from the source to all states in the explored regions. The A* algorithm does that such that all the near regions are explored before the further ones, while the exploration is also biased towards the regions closer to the goal (indicated by the heuristic function). Since A* algorithm only works in discrete spaces, it is necessary to create a discrete space of the map. In A* search, more the pixel of the map, more will be the complexity.

Fig. 2 shows the path planned for a robot using A* in an environment.

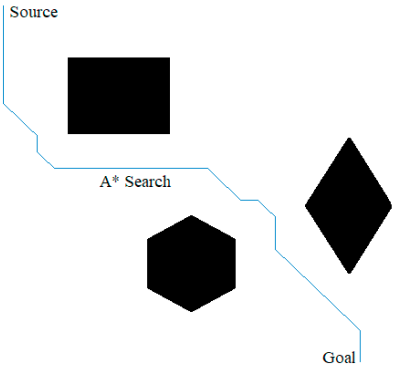

Fig. 2. Individual path planning using A*.

- *Probabilistic Roadmap (PRM)*

The algorithm has two stages: an offline roadmap (graph) building stage and an online planning/query stage. The aim of the offline roadmap (graph) building stage is to randomly draw a small graph across the workspace. All vertices and edges of the graph should be collision-free so that a robot may use the same graph for its motion planning. The PRM selects a number of random points (states) in the workspace as the vertices. In order to qualify being a vertex, a randomly selected point (state) must not be inside some obstacle. Let there be k number of states which is an algorithm parameter. Higher are the number of vertices or k, better would be the results with a loss of computational time. The algorithm then attempts to connect all pairs of randomly selected vertices. If any two vertices can be connected by a straight line, the straight line is added as an edge.

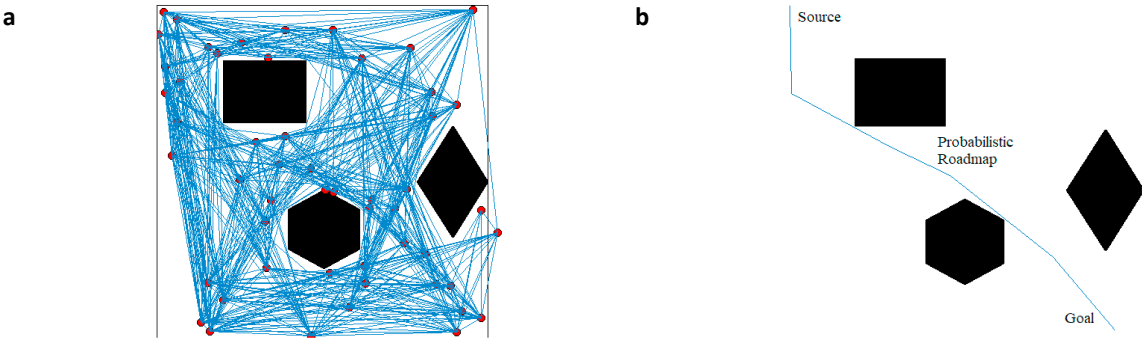

Fig. 3. Individual path planning using PRM, (a) Generation of random graphs in the environment; (b) best path selected from the graph generated

The online planning/query stage aims to use the roadmap (graph) developed earlier for planning the path of a robot. Since a graph is already known, any graph search algorithm can be used. In this case A* is used. The weights of the edges are taken as the Euclidean distance between the connecting points, and the heuristic function (denoting the nearness of the point to the goal) is taken as the Euclidean distance to the goal. Fig. 3 depicts the path planning done by PRM.

3.2.2. *Collision avoidance between two robots on the basis of priority*

After the individual path planning, the collision avoidance between two mobile robots will be done on the basis of priority. A robot with lower priority will wait for some timestamps for the robot with higher priority to pass from the collision point.

The priority to the robots will be assigned on the basis of their respective path length. Greater the path length, higher the priority. In case of collision at a point, a robot with lower priority will wait for the passing of robot with higher priority.

Here, a collision can be of two types:

- *Single point collision*

As shown in fig. 4 (a), $(1, 1)$ is the point of collision (block marked with *). Suppose, robot 1 (R_1) is having the higher priority than robot 2 (R_2), then the path plan of both the robots will be like this:

$$R_1 = [(0, 0);\ (1, 1);\ (2, 2)]$$

$$R_2 = [(2, 0);\ (2, 0);\ (1, 1);\ (0, 2)]$$

Robot 2 (R_2) will wait for 1 timestamp at $(2, 0)$.

- *Multiple point collision*

As shown in fig. 4 (b), $[(1, 2); (2, 2); (3, 2)]$ are the points of collision (blocks marked with *). Suppose, robot 1 (R_1) is having the higher priority than robot 2 (R_2), then the path plan of both the robots will be like this:

$$R_1 = [(0, 0);\ (1, 0);\ (1, 1);\ (1, 2);\ (2, 2);\ (3, 2);\ (4, 2)]$$

$$R_2 = [(4, 0);\ (3, 0);\ (3, 1);\ (3, 1);\ (3, 1);\ (3, 1);\ (3, 2);\ (2, 2);\ (1, 2);\ (0, 2)]$$

Robot 2 (R_2) will have to wait for 3 timestamps at $(3, 1)$.

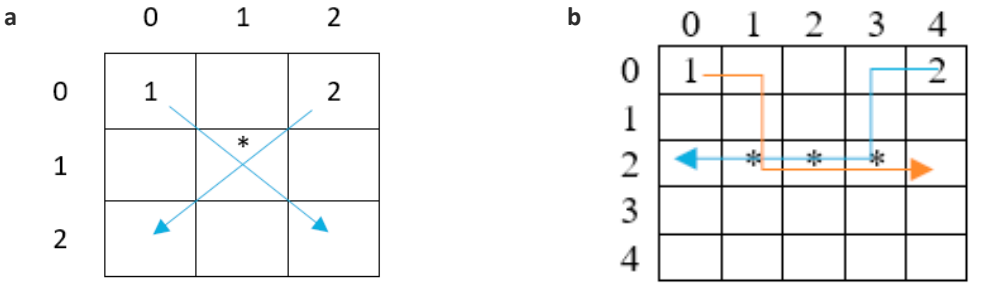

Fig. 4. (a) Single point collision scenario; (b) multiple point collision scenario.

3.3. *Plan Communication*

In this phase, UAV will communicate the respective path plan to every robot. After getting the path plan robots will move towards goal accordingly. Once all the robots achieve their goal position they will raise the flag and the task is completed.

4. Experiment

We have implemented the proposed algorithm in Python 3.6.4 using Enthought's Canopy IDE. The simulation is done on a workstation with 2GB RAM and Intel Core i5 – 2.30 GHz processor. The individual path planning algorithms were also implemented in MATLAB (R2017b version) on the same workstation and results were quite promising. But due to Python having vast varieties of data structures like Lists and Dictionaries, we found Python easier for the implementation of this particular problem.

We have simulated our algorithm in 450 × 450 grid size warehouse domain. Some blocks containing obstacles are marked black. One robot can occupy one block at a time only if that block is not occupied by another robot or obstacle. Every block is either a blank space (represented by white) or an obstacle (represented by black). Fig. 2 represents the environment taken for simulation. It is the 2D representation of environment where robots will move. The 3D results are also shown below to make clear the view of UAV in the environment. Green symbols represent the position of goals and the red symbols represent the initial position of robots. The comparison between two algorithms in terms of time taken to calculate the path for different number of agents is given in Table 2.

Table 2. Comparison of two algorithms for individual path planning in terms of time taken.

Number of Agents	A* Search (sec.)	PRM (sec.)
3	9.549	5.424
5	16.814	7.982
10	35.458	15.136
15	65.271	20.568
25	135.569	37.435
30	178.326	44.283
50	-	75.209

A time bound of 3 minutes was kept to complete the task. After a point of time, when agents were kept more than 30 A* failed to obtain the solution of given task in stipulated time.

The simulation results for the environment shown in Fig. 1 is provided here. The robots on *X* and *Y* axes, whereas the UAV can move on *X, Y* and *Z* axes as well. Fig. 5 represents the path planned for three agents using both the algorithms.

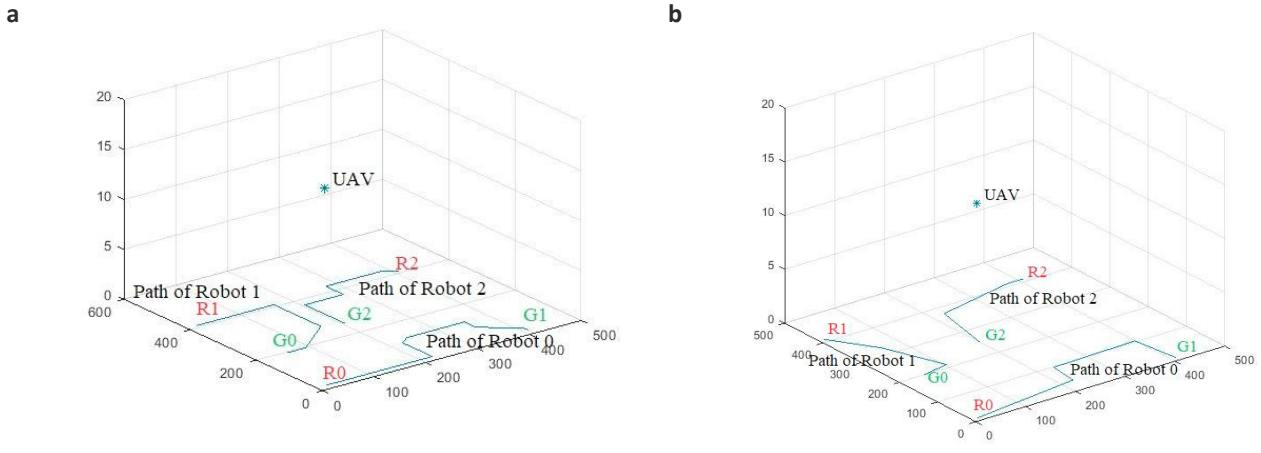

Fig.5. (a) Pathfinding for robots using A* search, in a 3D environment; (b) pathfinding for robots using PRM, in a 3D environment.

The result of both the algorithms are very similar in term of path length and are either optimal or very close to optimality. But according to the data shown in Table 2, we can observe that PRM is much more efficient in terms of time complexity than that of A* search. It is clear that due to the exponential behaviour of A* in time complexity, it cannot be used as a good problem-solving algorithm in today's real-life scenario where every second is costlier than the last one.

5. Conclusion

In this paper, a method is proposed to establish a relationship between a UAV and n robots to achieve a goal, i.e., to coordinate all the robots to different goals. A UAV can have the bird-eye view, which a robot cannot. A UAV operates in 3-dimensions whereas a robot operates only in 2-dimensions. The idea proposed in this paper can be very useful in many areas like surveillance system or in warehouse automation. For example in Kiva-like domain [13] where a UAV can act as a leader of some automated robots running on the ground. After calculating the most efficient combination of robots and goals, UAV will communicate the path to each of the robot in a centralized manner. The simulations are done in Python and MATLAB to verify the effectiveness of proposed method. For optimal path planning, two algorithms, A-star search (A*) and Probabilistic Roadmap (PRM) are used and compared for the different number of agents. The simulation results justify the use of PRM over A* in every case. While path length in both the cases is remains almost same, time taken to calculate the path increases exponentially in case of A*. That is why A* cannot be taken in the picture for practical implementation. One of the possible challenges for future work can be the presence of dynamic obstacles in the environment.

References

[1] Ulun S, Unel M. Coordinated motion of UGVs and a UAV. InIndustrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE 2013 Nov 10 (pp. 4079-4084). IEEE.

[2] Esin YH, Ünel M. Formation control of nonholonomic mobile robots using implicit polynomials and elliptic Fourier descriptors. Turkish Journal of Electrical Engineering & Computer Sciences. 2010 Oct 25;18(5):765-80.

[3] Merino L, Caballero F, Martinez-de Dios JR, Ollero A. Cooperative fire detection using unmanned aerial vehicles. InRobotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on 2005 Apr 18 (pp. 1884-1889). IEEE.

[4] Chaimowicz L, Cowley A, Gomez-Ibanez D, Grocholsky B, Hsieh MA, Hsu H, Keller JF, Kumar V, Swaminathan R, Taylor CJ. Deploying air-ground multi-robot teams in urban environments. InMulti-Robot Systems. From Swarms to Intelligent Automata Volume III 2005 (pp. 223-234). Springer, Dordrecht.

[5] Tanner HG. Switched UAV-UGV cooperation scheme for target detection. InRobotics and Automation, 2007 IEEE International Conference on 2007 Apr 10 (pp. 3457-3462). IEEE.

[6] Luo C, Espinosa AP, De Gloria A, Sgherri R. Air-ground multi-agent robot team coordination. InRobotics and Automation (ICRA), 2011 IEEE International Conference on 2011 May 9 (pp. 6588-6591). IEEE.

[7] Mishra A, Shukla A. Mathematical analysis of the cumulative effect of novel ternary crossover operator and mutation on probability of survival of a schema. Theoretical Computer Science. 2017 Mar 1;666:1-1.

[8] Mishra A, Shukla A. Mathematical analysis of schema survival for genetic algorithms having dual mutation. Soft Computing. 2017:1-9.

[9] Mishra A, Shukla A. Analysis of the Effect of Defining Length and Order of Schemata on Probability of Survival of a Group of Schemata. InAnnual international conference on intelligent computing, computer science and information systems (ICCSIS 16), Pattaya, Thailand. https://doi. org/10.15242/IAE. IAE0416 2016 Apr.

[10] Mishra A, Shukla A. A new insight into the schema survival after crossover and mutation for genetic algorithms having distributed population set. International Journal of Information Technology.:1-4.

[11] R. Kala (2014) Code for Robot Path Planning using A* algorithm, Indian Institute of Information Technology Allahabad, Available at: http://rkala.in/codes.html, Retrieved December 14, 2017.

[12] R. Kala (2014) Code for Robot Path Planning using PRM algorithm, Indian Institute of Information Technology Allahabad, Available at: http://rkala.in/codes.html, Retrieved December 14, 2017.

[13] Wurman PR, D'Andrea R, Mountz M. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. AI magazine. 2008 Mar 20;29(1):9.