Annals of University of Craiova, Math. Comp. Sci. Ser.
Volume 36(1), 2009, Pages 37–48
ISSN: 1223-6934

An Optimal Path Algorithm for Autonomous Searching Robots

Claudiu Popirlan and Mihai Dupac

Abstract. Constructing architecture and generating optimal paths for autonomous robots are some of the heavily studied subjects in mobile agents applications. The aim of this paper is to find and analyze a optimal path algorithm for a group of autonomous robots using agent-based architecture in a virtual reality environment. The optimal path is determined by a heuristic approach, A-Star algorithm. The master-slave architecture control the communication between robots (slave agents) and the mobile agent (master). The robots communicate with the mobile agents to generate an optimal path (time and collision-free optimal path) by presenting all the subsequent trajectories while navigating among various obstacles. The results are conducted to show the effectiveness of the proposed architecture.

2000 Mathematics Subject Classification. Primary 68T99; Secondary 65D17.
Key words and phrases. Mobile Agents, Dynamic Modeling, Analysis, Virtual Reality, Visualization, Optimal Path.

1. Introduction

Applications in the field of autonomous robots are generally based on navigating from a starting point to goal point in a known or unknown environment. In daily life autonomous robots are used in many missions like planetary exploration and space applications. In these kinds of applications the points that should be visited are unknown. But, for an autonomous robot, consuming less energy is very important and it is obvious that the more autonomous robot travels more energy and time is consumed. To fulfill this constraints, a shorter path is preferred rather than a longer path. Therefore an intelligent path planning algorithm is always required. Since the robot works in a real and dynamic environment, the path planning algorithm should construct the path in real time.

Agent-based approaches receive considerable attention in the literature. The agent-based approaches applications are distributed among several agents. Zavlanos and Pappas proposed a method to solve multi-agent assignment problems, in the nature of mobile robots ([1]). Rushan et-al applied agent approach to a heterogeneous mobile robot team to find a solution for localization problem ([2]). In their work some agents are qualified as localizers. These agents localize themselves and the other mobile robots in the environment. This localization information is passed to all members of the team.

Mobile agent is now a popular abstraction mechanism to construct adaptable distributed systems in convenient way. To develop a system using mobile agents, one has to overcome several new complexities (as well as security/ safety problems) in his/her code arise from the mixed use of mobility primitives. Over the last few years, numerous mobile agent systems [26, 24, 25, 22, 23] have been developed because this

Received: 11 June 2009.

technology is promising for building customizable, adaptable and robust distributed systems.

The concept of mobile agents is studied and defined by [8] and [4]. They are processes (i.e., executing programs) that can migrate from one machine of a system to another machine (usually in the same system) in order to satisfy requests made by their clients [27]. They implement a computational metaphor that is analogous to how most people conduct business in their daily lives: visit a place, use a service, and then move on [27].

Research on robots has attracted attention in the last years since [17]. Most of the research have been directed to the use of kinematic models of the mobile robots to achieve and accomplished the motion control [12, 11, 14]. Later on, the research has been focused on robots with additional sensory system to develop autonomous guidance path planning systems [15]. Sophisticated sensory systems has been used in [16], helping the software to learn about the operating environment and to evaluate path constraints for a good path planning programming.

Path planning consists of determining a route joining two input configurations, i.e., from one coordinate location to another location. Path planning algorithms are determinant for the mobile agent behavior, including collision avoidance. Modeling and simulation of a group of mobile robots, the controllability issue and path planning was studied in [18]. Different other approaches including rigid formations, potential field methods, and neural have been studied in [17]. The motion planning for mobile robots when using a dynamic environment and moving obstacles was studied in [19].

In this paper, an agent-based approach is proposed for optimal path planning for mobile robot. The approach consists of two agents types: the slave agent is responsible for the path construction, and the master one is responsible for low-level control of the mobile robot. Finding the optimal path solution is time-consuming. For this reason a heuristic approach A-star is used. These agents communicate via a communication module which is based on open agent architecture and is describe in Section 3.1. The equations of motion (kinematics and dynamics) are presented, and the motion planning with obstacle avoidance (collision free) analyzed.

2. Mathematical Model of the Autonomous Car Like Robot

An autonomous mobile car-like robot has to move on an optimal path (time and collision-free optimal path) while navigating among various obstacles, and satisfying the kinematic and dynamic constraints. In the next two sub-sections, the kinematic and dynamic constraints of an autonomous car-like robot are explained in detail.

2.1. Kinematics. The autonomous car-like robot is modeled in this study as a rigid body with four wheels. The robot motion is defined on a plane surface, with the 4 wheels making contact point with the planar ground surface.

For the study of the robot kinematics one can define the configuration of the moving autonomous robot at every time instant by a triple $(x(t), y(t), \alpha(t))$, with $x(t)$ and $y(t)$ the x-coordinate and y-coordinate of the robot center of the mass (relative to the origin of a Cartesian reference frame xOy) at the time frame t , and $\alpha(t)$ the robot orientation, i.e., the angle between the robot direction (the main axis of the robot) and the Ox axis at the same time frame t. The longitudinal speed of the autonomous robot (velocity in the x direction) is given by its first derivative $u = \dot{x}$, while the lateral speed of the autonomous robot (velocity in the y direction) is given by $v = \dot{y}$. The angular speed ω of the autonomous robot is given by $\dot{\alpha}$.

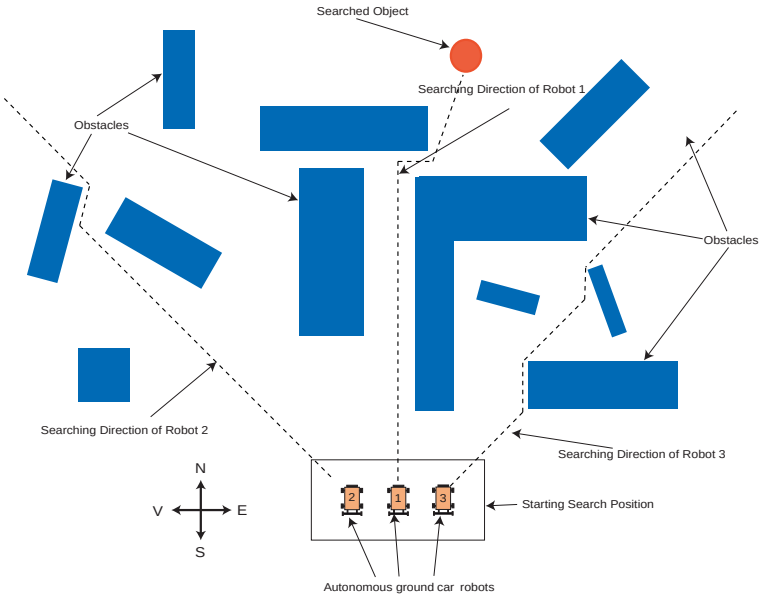

FIGURE 1. The Virtual Reality Defined Scene with Three Searching Robots

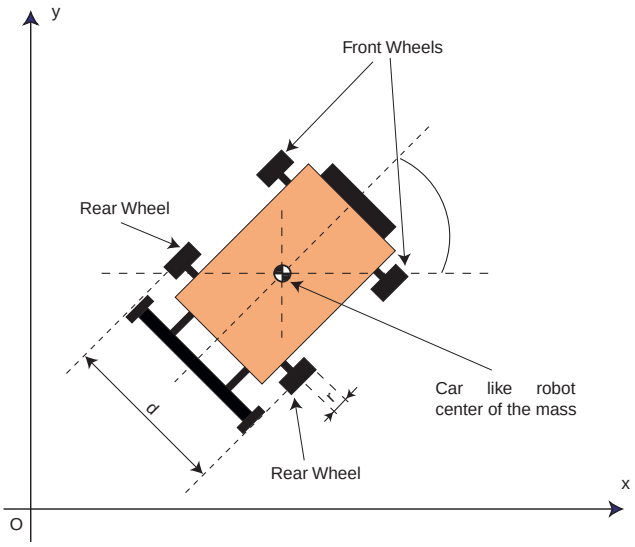

FIGURE 2. The Autonomous Ground Car like Searching Robot

The longitudinal, lateral and the angular speed of the autonomous robot can be determined using the angular speeds on the left and right drive wheels of the robot. For simplicity it was considered that the left and the right angular speed of the wheels are the same. Considering ω_{left} the angular speed of the left drive wheel and ω_{right} the angular speed of the right drive wheel, one can write,

$$w = \frac{r(\omega_{left} + \omega_{right})}{2}, \quad \dot{\alpha} = \frac{r(\omega_{left} - \omega_{right})}{d} \tag{1}$$

where d is the distance between the wells and r is the wheel radius as shown in Fig 2. The longitudinal speed $u = \dot{x}$ and the lateral speed $v = \dot{y}$, are calculated using

$$\dot{x} = w\cos(\alpha), \quad \dot{y} = w\sin(\alpha) \tag{2}$$

Using Eqs. (1) and (2) one can write

$$\begin{aligned} \dot{x} &= \frac{r\cos(\alpha)(\omega_{left} + \omega_{right})}{2} \\ \dot{y} &= \frac{r\sin(\alpha)(\omega_{left} + \omega_{right})}{2} \\ \dot{\alpha} &= \frac{r(\omega_{left} - \omega_{right})}{d} \end{aligned}$$

or equivalent

$$\begin{aligned} u &= \frac{r\cos(\alpha)(\omega_{left} + \omega_{right})}{2} \\ v &= \frac{r\sin(\alpha)(\omega_{left} + \omega_{right})}{2} \\ \omega &= \frac{r(\omega_{left} - \omega_{right})}{d} \end{aligned}$$

In order to prevent slippage, the frictional force should be greater than the acting resultant force, where the resultant force is composed of the normal and tangential forces. For a nonholonomic motion, the non slip condition can be written as,

$$\dot{x}\sin(\alpha) = \dot{y}\cos(\alpha).$$

2.2. Dynamics. While in motion, the autonomous robot is subjected to various dynamic constraints (such as sliding or torque constraint), constraints that can be described in terms of the related robot velocity and acceleration. The nonholonomic equations of motion of the autonomous mobile robot can be written in a matrix form based on the Euler Lagrange formulation as,

$$\begin{aligned} \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\alpha} \end{bmatrix} &= \frac{1}{r} \begin{bmatrix} \cos(\alpha) & \cos(\alpha) \\ \sin(\alpha) & \sin(\alpha) \\ -d & d \end{bmatrix} \begin{bmatrix} \tau_l \\ \tau_r \end{bmatrix} \\ &\quad + \begin{bmatrix} \sin(\alpha) \\ \cos(\alpha) \\ 0 \end{bmatrix} \lambda \end{aligned} \tag{3}$$

where τ_l is the torque of the left wheel, τ_r is the torque of the right wheel, m is the mass of the motor, I is the robot mass and inertia, and λ the Lagrange multipliers of constrained forces. The friction force acting on a wheel is F. Two DC motors are assumed to propel the robot. Each DC motor is connected to the drive wheel on each side and generate torque. For the sake of briefness, the discussion of the DC motor model is thus omitted here, eventhough the model is included in the simulation. Considering are τ_{linear} and $\tau_{angular}$ the linear and angular torques, one can write,

$$\begin{aligned} \tau_{linear} &= \frac{\tau_l + \tau_r}{r} \\ \tau_{angular} &= \frac{d(\tau_l + \tau_r)}{r} \end{aligned}$$

From the previous two equations one can deduce

$$\begin{aligned} \ddot{x} &= \frac{\tau_{linear}}{m}\cos(\alpha) + \frac{\lambda}{m}\sin(\alpha) \\ \ddot{y} &= \frac{\tau_{linear}}{m}\sin(\alpha) - \frac{\lambda}{m}\cos(\alpha) \\ \ddot{\alpha} &= \frac{\tau_{angular}}{I} \end{aligned} \tag{4}$$

Applying the second derivative to the Eq. (2) one can obtain

$$\begin{aligned} \ddot{x} &= -w\dot{\alpha}\sin(\alpha) + \dot{w}\cos(\alpha) \\ \ddot{y} &= -w\dot{\alpha}\cos(\alpha) + w\sin(\alpha) \\ \ddot{\alpha} &= \dot{\omega} \end{aligned} \tag{5}$$

Finally, from Eqs. (4) and (5) one can deduce

$$\begin{aligned} \frac{\tau_{linear}}{m}\cos(\alpha) + \frac{\lambda}{m}\sin(\alpha) &= -w\dot{\alpha}\sin(\alpha) + \dot{w}\cos(\alpha) \\ \frac{\tau_{linear}}{m}\sin(\alpha) - \frac{\lambda}{m}\cos(\alpha) &= -w\dot{\alpha}\cos(\alpha) + w\sin(\alpha) \\ \frac{\tau_{angular}}{I} &= \dot{\omega} \end{aligned}$$

3. The Proposed Mobile Agents System

For generating an optimal path two kinds of agents are presented. A slave agent is used to construct the path using well known heuristic A-star algorithm. The other agent is the mobile agent responsible for low-level control of the mobile robot. Slave agent and mobile agent communicate with each other through the Communication Module as shown in Fig. 3.

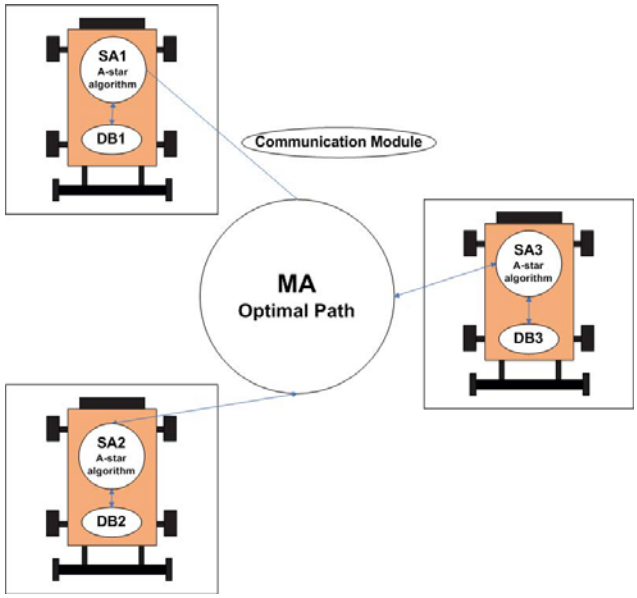

FIGURE 3. The agent-based system structure

The agent-based system consists of:
- 1 (n>0) Mobile Agent(MA),
- 3 (n>0) number of Slave Agents(SA),
- distributed data bases(DB1, DB2, DB3)

As seen from Fig. 3, slave agent sends the present coordinates and the goal coordinates to mobile agent through the communication channel and asks a planned path from the mobile agent. The mobile agent plans direction based on the partial generated map of the environment. While constructing the path, slave agent uses the A-star algorithm, which is explained below. A-star search is one of the widely used informed search strategies ([3]). It is used to find a path from the starting node to the goal node in a graph. In this study, the environment is divided into $1m.$ by $1m.$ square grids. The center point of each square is considered as a node of the graph. Then the A-star algorithm uses this graph to construct the path. The cost for each grid is calculated as the sum of two costs: $f(n) = h(n) + g(n)$ where $g(n)$ is the associated cost of the starting node to the node n path, and $h(n)$ is the associated cost of the node n to the goal node path. Since $g(n)$ gives the path cost from the start node to node n and $h(n)$ is the heuristic distance which is the estimated cost of the closest path from n to goal, $f(n)$ becomes the estimated cost of the shortest solution through node n. The A-star algorithm is an optimal search strategy if $h(n)$ is an admissible heuristic provided such as $h(n)$ never overestimates the exact cost of the goal. The input of the A-star search algorithm is the graph and the output is a back-pointer path which is a sequence of nodes starting from the goal and back to the start. If O is the open set priority queue and C is the closed set containing all processed nodes, the A-star search algorithm can be expressed as below (1).

Algorithm 1 A-star search algorithm

while O is not empty **do**
  Find $nbest$ from O such that $f(nbest) \leq f(n)$ for all n in O;
  Remouve $nbest$ from O;
  Add $nbest$ to C;
  if $nbest = ngoal$ **then**
    $EXIT$;
  end if
  Expand $nbest$ for all x, neighour of $nbest$ and not in C;
  if x is not in O **then**
    Add x to O;
  end if
  if $g(nbest) + dist(nbestx) < g(x)$ **then**
    Update x's back-pointer to point the $nbest$;
  end if
end while

The neighborhood of a grid cell uses eight-point connectivity relation and the heuristic distance ($h(n)$) is calculated by using Euclidean distance which is always smaller than or equal to the actual distance.

3.1. The Communication Module. The system must be able to make possible the agents communication, including the communication protocols. The communication protocols should allow message understanding and wireless transmission. Message understanding implies decision acceptance or rejection based on the collected sensors

information and other mobile agents information. Mobile agents communication is based on sending (action) and receiving (perception) messages.

The mobile agents will communicate to each other to achieve the defined goals. The degree of coherence and coordination comes from the extent to which the system avoid redundant actions, competition on resources, bottlenecks and the unsafe operating conditions. The goal is to maintain an overall coherence, without having always a global control in place. The coordination between mobile agents not entering the competition is based on cooperation. For the agents entering in the competition, or those having reciprocal dependence, the coordination is based on negotiation.

In our system, the communication language has two levels:

- Level 1: communication between agents and sensors;
- Level 2: communication between agents.

Communication between agents and sensors (level 1) is in one sense: from sensor to agent. The agent receive informations from sensor and make decision.

Communication between agents (MA and SA), level 2, is done via messages. An agent can communicate with other agent by message passing. An agents that wants to communicate with another agent first has to create a message object, then send it to the target agent. A message object has a kind and an optional argument object. The receiver agent determine what to do by checking the kind of received message and get parameters as the argument object. For system implementation one can use a subset of a standard indicators ([21]) of the Knowledge Query and Manipulation Language (KQML):

$tell$
$: content < expression >$
$: language < word >$
$: ontology < word >$
$: in - reply - to < expression >$
$: force < word >$
$: sender < word >$
$: receiver < word >$

This indicators can be used when an mobile agent found an object and announce its finding. The message agents of our system act somewhat like the active packets in the capsule tradition of active networking research; they are minimal programs that carry a payload of communications data across the network. They make their decisions about where they need to go based on information that routing agents, with which they share the network, accumulate and cache.

3.2. Autonomous Robots Obstacle Avoidance and Sensors. Obstacle avoidance was integrated on the mobile agent software. The software adjust the direction of the robot based on any obstacles in its path. While the robot executes the searching, if any obstacles is found on the robot path, the direction is adjusted to run tangent to the obstacle, and so, the old path is replaced with a new path to avoid the obstacle. Since the mobile agent continuously update the path based on the sensors information, the obstacle avoidance analysis and the direction path is constantly replanning. This continuous replanning allows the searching robot to handle a highly dynamic environment and to take advantage of any path free opportunity (the visibility graph).

For landmark navigation, the sensors installed on the autonomous robot, such as stereo vision sensors (two CCD cameras fixed on the left and right side of the searching robot), and a laser scanning system in connection with a GPS-receiver

(with the GPS signals available every second) allows the robot to navigate and to have sufficiently conditions for landmark detection with obstacle avoidance (collision free). The collision free is implemented for the dynamic motion. The laser scanning system and GPS-receiver allows the agent to plan the path and trajectory based on the 3D created map of the environment (the map is updated every second based on the mobile agents communication). The two CCD cameras synchronously capture the image and combine it with the 3D created map for the searching object recognition purpose. For the present described system (car like robot) the obstacles are assumed to be convex polygons.

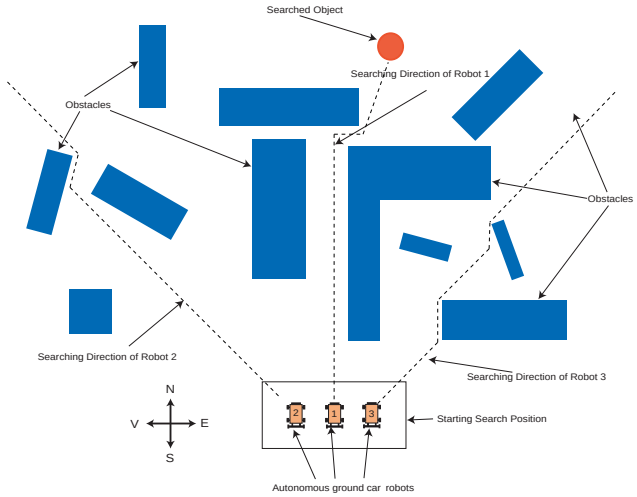

FIGURE 4. Initial Trajectories of 3 Autonomous Searching Robots

4. Autonomous Robots Path Analysis

The aim of this study is to generate an optimal path (time and collision-free optimal path) and to present all the subsequent trajectories of a group of autonomous robots (3 autonomous robots are used in this analysis) navigating among various obstacles. The robots are defined as an autonomous ground car like vehicles and programmed to perform objects searching in a defined environment. The robots are equipped with software (mobile agent) and sensors, and moves based on the described equations of motion while satisfying the kinematic and dynamic constraints. While in motion the robots detect the position of all obstacles (based on the attached sensors) while looking for the searched object. The searching directions of the 3 autonomous robots are defined as: north (N), north-east (N-E) and north-vest (N-V). If a robot have to deviate from its current direction of movement, i.e., N, N-E or N-V, (for obstacle avoidance purpose), it starts decelerating in order to achieve the turning velocity. After a turn, the robot will continue moving in the predefined direction (following a straight path).

For the analysis the subsequent trajectories of the 3 autonomous robots and the optimal path are shown Fig. 4 to Fig. 8.

The initial trajectories of the searching robots are provided in Fig. 4. The trajectories of the robots are predefined as follows: The robot number 1 will move to the

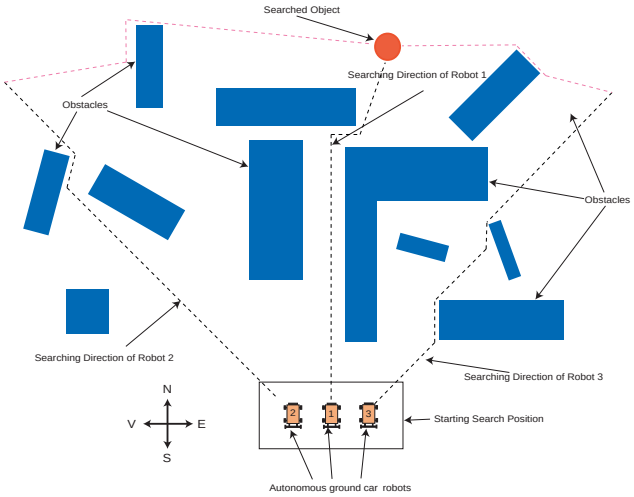

FIGURE 5. Autonomous robots behavior after one of the robots find the searched object - trajectories are marked with red

north (N), the robot number 2 is moving to north-vest (N-V) and the robot number 3 is moving to north-east (N-E). In order to maintain the path, the robots avoid obstacles taking the shortest path. The trajectories of the robots are marked on Fig. 4 with a black dashed line.

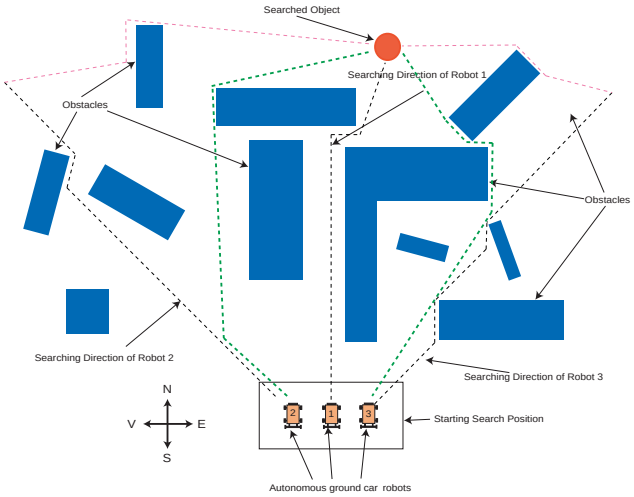

FIGURE 6. Initial Optimization of the Trajectories - marked with green

Figure 5 shows the autonomous robots behavior after one of the robots (robot 1) find the searched object. Based on the described algorithm 1, the robots 2 and 3 change the direction. The new direction of each robot is a new straight line to the searched object. The new trajectories are marked on the Fig. 5 with a red dashed line. Both robots are able to reach the searched object as shown in Fig. 5.

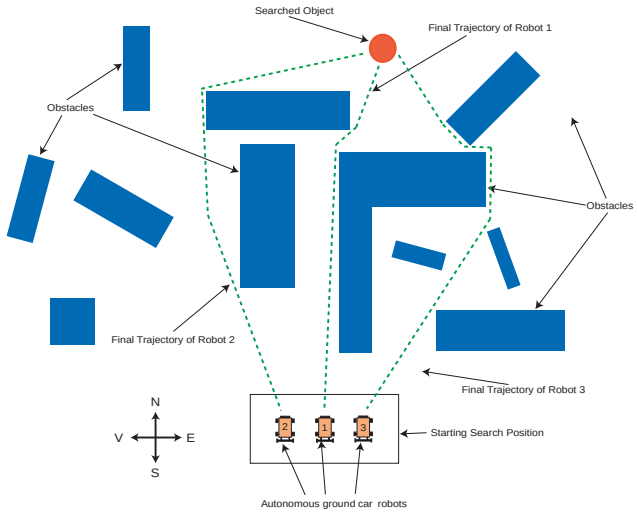

FIGURE 7. Final Optimization of the Trajectories - marked with green

Figure 6 shows behavior of robots 2 and 3 after reaching the searched object. The autonomous robots optimized the previous path using the A-star algorithm as shown in Fig. 6. The new trajectories are marked on the Fig. 6 with a green dashed line.

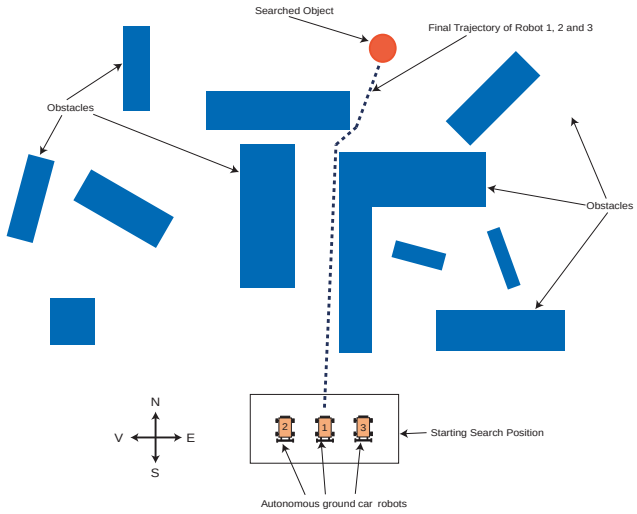

FIGURE 8. Final Path - marked with black

Since there are not other optimal trajectories between the starting base and the searched object, the autonomous robots optimized the trajectories shown in Fig. 6 as resulting from Fig. 7

Finally, all the robots take the shortest path (the trajectory discovered by robot 1) as this is the optimal path from the base to the searched object. The obtained final optimal path is shown in Fig. 8.

5. Conclusions and future work

In this paper, an agent-based real-time path planning algorithm is proposed for a group of autonomous searching robots. The path is constructed by A-star heuristic algorithm. The optimal path analysis of a group of autonomous robots is provided. Anyway, the present study need to be extend for a dynamically changed environments. A increase on the grid size resolution so that the robots can travel in more complicated environments is also desirable. Simulations and experimental results are also needed in order to totally validate the model.

References

[1] M.M Zavlanos and G.J. Pappas, *IEEE Transactions on Robotics*, Volume 24, Issue 1, 232 - 242, (2008).

[2] S.M. Rushan, M. Mehrandezh, R.Paranjape, *Canadian Conference on Electrical and Computer Engineering CCECE '06*, 1522 1525, (2006).

[3] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun, *Principles of Robot Motion*, M.I.T. Press, 527-532, (2005).

[4] P. Braun and W. Rossak, *Mobile Agents: Concepts, Mobility Models, & the Tracy Toolkit*, Elsevier Inc.(USA) and dpunkt.verlag(Germany), 2005.

[5] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 1995.

[6] Claudiu Popîrlan, Cristina Popîrlan, Algorithms for Mobile Agents in Network using Tracy(Mobile Agent Toolkit), *Research Notes in Artificial Intelligence and Digital Communications*, Vol.106, Thessaloniki, Greece, p.31-38, (2006).

[7] C.I. Popîrlan, & C. Popîrlan, Mobile Agents communication for knowledge representation, *11-th World Multiconference on Systemics, Cybernetics and Informatics (WMSCI 2007)*, Orlando, USA, July 8-11, *1*, 92-96, (2007).

[8] J. Baumann, *Mobile Agents: Control Algorithms*, Lecture Notes in Computer Science, Springer, 2000.

[9] B. Eckel, *Thinking in Java*, Prentice Hall (4-th Edition), 2006.

[10] *The Sun Developer Network (SDN) Web page*, (`http://java.sun.com/`).

[11] G. Mester, Motion Control of Wheeled Mobile Robots, *4th Serbian-Hungarian Joint Symposium on Intelligent Systems, SISY*, 119-130, (2006).

[12] W. Dong, Y. Guo, Dynamic tracking control of uncertain nonholonomic mobile robots, *Intelligent Robots and Systems, (IROS 2005), IEEE/RSJ*, 2774 - 2779, (2005).

[13] P. Vleugels, Exact motion planning for tractor-trailer robots Svestka, *J. Robotics and Automation, Proceedings of IEEE International Conference*, Volume 3, 2445 - 2450, (1995).

[14] A. De Luca, G. Oriolo, C. Samson, *Robot Motion Planning and Control: Feedback control of a nonholonomic car-like robot*, Springer Verlag, 2006.

[15] X. Xiang, J. Zhang, C. He, An Efficient System for Nonholonomic Mobile Robot-Path Planning, *Advances in Intelligent Systems Research, International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2007)*, (2007).

[16] T. R. Wan, H. Chen, R.A. Earnshaw, A Motion Constrained Dynamic Path Planning Algorithm for Multi-Agent Simulations, *The 13th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2005*, 211-218, (2005).

[17] R. Gayle, A. Sud, M. C. Lin, D. Manocha, Reactive Deformation Roadmaps: Motion Planning of Multiple Robots in Dynamic Environments, *Intelligent Robots and Systems, IROS 2007. IEEE/RSJ International Conference*, 3777-3783, (2007).

[18] G. Klancar, B. Zupancic, R. Karba, Modelling and simulation of a group of mobile robots, *Simulation Modelling Practice and Theory*, Volume 15, 647658, (2007).

[19] T.-J. Pan, R.C. Luo, Motion planning for mobile robots in a dynamic environment with moving obstacles, *Robotics and Automation, IEEE International Conference Proceedings*, Volume 1, 578 - 583, (1990).

[20] F. Preparata, M. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.

[21] T. Finin, Y. Labrou, J. Mayfield, *Software Agents: KQML as an agent communication language, invited chapter in Jeff Bradshaw (Ed.)*, MIT Press, Cambridge, 1997.

[22] D. Kotz, R.S. Gray, Mobile Agents and the Future of the Internet, *ACM Operating Systems Review*, Volume 33(3), 7-13, 1999.
[23] Q. Wenyu, S. Hong, X. Defago, A Survey of Mobile Agent-Based Fault-Tolerant Technology, *Parallel and Distributed Computing, Applications and Technologies*, Volume 5, 446-450, 2005.
[24] H. Peine, An Introduction to Mobile Agent Programming And The Ara System, *Technical report ZRI-Report 1/97, Department of Computer Science, University of Kaiserslautern*, Germany, 1997.
[25] R.S. Gray, D. Kotz, G. Cybenko, D. Rus,D'Agents: Security in a multiple-language, *In Giovanni Vigna (Ed.) Mobile Agents and Security, Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 1998.
[26] D. Wong, N. Paciorek, T. Walsh, J. DiCelie, M. Young, B. Peet, Concordia: An infrastructure for collaborating mobile agents, *Lecture Notes in Computer Science*, Volume 1219, 86-97, 1997.
[27] D. Johansen, R. van Renesse, F.B. Schneider, Operating System Support for Mobile Agents, *Technical report, Department of Computer Science, Cornell University*, USA, 1994

(Claudiu Popirlan) University of Craiova, Faculty of Mathematics and Computer Science, Department of Computer Science, 13 Alexandru Ioan Cuza Street, Craiova, 200585, Romania
E-mail address: popirlan@inf.ucv.ro

(Mihai Dupac) University of Craiova, Faculty of Mathematics and Computer Science, Department of Computer Science, 13 Alexandru Ioan Cuza Street, Craiova, 200585, Romania
E-mail address: mihai.dupac@inf.ucv.ro