Chapter 9
Cellular Robotic Ants Synergy Coordination for Path Planning

Konstantinos Ioannidis, Georgios Ch. Sirakoulis and Ioannis Andreadis

Abstract In this chapter, a unified architecture is proposed for a robot team in order to accomplish several tasks based on the application of an enhanced Cellular Automata (CA) path planner. The presented path planner can produce adequate collision-free pathways with minimum hardware resources and low complexity levels. During the course of a robot team to its final destination, dynamic obstacles are detected and avoided in real time as well as coordinated movements are executed by applying cooperations in order to maintain the team's initial formation. The inherit parallelism and simplicity of CA result in a path planner that requires low computational resources and thus, its implementation in miniature robots is straightforward. Cooperations are limited to a minimum so that further resource reduction can be achieved. For this purpose, the basic fundamentals of another artificial intelligence method, namely Ant Colonies Optimization (ACO) technique, were applied. The entire robot team is divided into equally numbered subgroups and an ACO algorithm is applied to reduce the complexity. As each robot moves towards to its final position, it creates a trail of an evaporated substance, called "pheromone". The "pheromone" and its quantity are detected by the following robots and thus, every robot is absolved by the necessity of continuous communication with its neighbors. The total complexity of the presented architecture results to a possible implementation using a team of miniature robots where all available resources are exploited.

K. Ioannidis (✉) · G.Ch. Sirakoulis · I. Andreadis
Department of Electrical and Computer Engineering, Democritus University of Thrace, 67100 Xanthi, GR, Greece
e-mail: kioannid@ee.duth.gr

G.Ch. Sirakoulis
e-mail: gsirak@ee.duth.gr

I. Andreadis
e-mail: iandread@ee.duth.gr

G.Ch. Sirakoulis and A. Adamatzky (eds.), *Robots and Lattice Automata*,
Emergence, Complexity and Computation 13, DOI 10.1007/978-3-319-10924-4_9

9.1 Introduction

Robot navigation is perhaps the most significant and active research field on the area of mobile robotics due to its applicability to a wide variety of tasks such as industry and human-supported works. The development of a rapid and efficient procedure which deals with the path planning problem is considered to be a key step for the motion of a mobile robot [68]. Path planning typically refers to the design of geometric specifications of the positions and orientations of robots in the presence of obstacles. An efficient path planner aims for the creation of a continuous motion that connects a starting point and a goal point in the configuration area of a mobile robot with the presence of obstacles (Fig. 9.1). The complexity of the problem is significantly increased when the developed framework is required to create collision free trajectories for every robot of a cooperative team. Cooperative robotic teams are extensively used in systems for accomplishing tasks where single robots fail to complete successfully their goals [46]. A variety of practical and potential applications can benefit from the use of a cooperative robot team, such as exploration of unknown areas [10], search and rescue [53] and formation control [25].

In general, the most widely known classification for path planning solution algorithms relies on the discrimination of the environments between static and dynamic. In the static case, all necessary information relative to the position of obstacles is known a priori while in dynamic planning, robots display a complete unawareness of their configuration area. For example, Tzionas et al. proposed in [63] an approach based on a retraction of free space onto the Voronoi diagram, which is constructed through the time evolution of Cellular Automata (CA) after an initial phase, during which the boundaries of obstacles are identified and coded with respect to their orientation. Despite its rapid execution, the approach suffers from a constraint of the

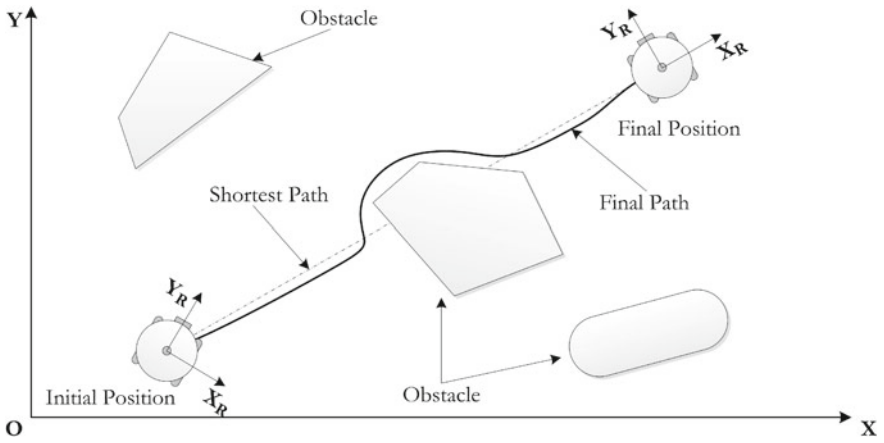

Fig. 9.1 An example of path planning that connects a starting point and a goal point in the configuration area of a mobile robot with the presence of obstacles

workspace, which is restricted in the sense that a generally shaped robot is enclosed within a diamond-shaped figure. A similar CA-based technique was proposed by Marchese in [47] where an anisotropic propagation of attracting potential values in a 4-D space-time using a multilayered cellular automaton (MCA) architecture was exploited. The algorithm executes a search for all the optimal collision-free trajectories following the minimum valley of a potential hypersurface embedded in a five dimensional space. However, efficiency cannot be obtained due to the uncompleted priority planning and thus, collisions between robots may occur. A cell based method was also proposed in [70] where a known environment was assumed. The configuration area was represented by occupancy grids and it was separated into a grid of equally spaced cells. The problem is converted to a graph-search problem and so, an A-star algorithm was applied [17]. Due to its features, the A-star approach increases the complexity of the system and thus, more processing resources are required in a real system. Dubins' theorem was also exploited for increased efficiency in [66] in order to deal with static environments. A genetic algorithm as well as a hierarchical structure of chromosomes to denote a possible path were used to identify the most optimal route. The method requires significant amounts of time rendering its implementation in real-time systems constrained.

Despite their efficiency, static environment approaches restrict their applicability in non-real world scenarios since most of the environments include dynamic obstacles. Path planners for dynamic environments may properly modify the path of a robot in cases of unexpected changes of its immediate environment. For example, the approach in [5] extracts a path from a static environment for a robot which is equipped with proximity sensors. Based on the sensors' readings, the robot bypasses possible obstacles that unexpectedly may block its route. The method requires part of information related to the status of the environment in order to be functional. A similar approach was proposed in [55] where a mobile robot modifies its path in the presence of semi-dynamic obstacles. The navigational planning is achieved with the application of a genetic algorithm until the robot reaches its final destination. In addition, a fuzzy-logic sensor fusion system was developed in [61] for target recognition, wherein the proposed path planning solver is based on a grid-map-oriented system that permits path revision through interactions with dynamical environments. An efficient dynamic system algorithm, using a neural dynamic model and a distance transform model was proposed in [67]. The approach develops efficient collision-free trajectories for a robot, and although dynamic programming is employed, computational cost remains in high levels. The A-star approach was also used in [24] as a global path planner to produce a series of sub-goal points to the target point. A potential field method was embedded as a local path planner in order to smooth the path between the preplanned sub-goal points. Thus, the method fails to cope with a real-time robot systems since global information of the entire configuration area is required. In order to produce smooth paths and reduce the complexity of the solution, a series of waypoints are interpolated in [2] with the use of a B-spline which is altered only in the area local to the obstacle. B-splines are piecewise series of polynomials and, therefore, the solution remains complex despite their low order. Finally, in [71], an improved Hopfield type neural network model was applied in

order to propagate the target activity among neurons, in the manner of physical heat conduction. The motion of a robot is defined through the dynamic neural network activity. In general, the method creates efficient paths for a robot, nonetheless, the exploitation of the neural network leads to high computational burden.

The discrimination of the path planning algorithms into either static or dynamic provides a base categorization without, nonetheless, any reference to the solution itself. A more proper differentiation is relied on the nature of the problem and the manner that an algorithm approaches its solution. Thus, research in path planning could also be classified into four basic categories: visibility graphs, potential fields, cell decomposition and heuristic algorithms [63]. Visibility graphs include the determination of a line collection in free space such that a connection of features of an object to those of another is accomplished [42]. A visibility graph can produce $O(n^2)$ edges for (n) features and so it is characterized by high complexity. On the contrary, potential field approaches exploit potential functions that are developed for obstacle avoidance in static environments. These approaches treat a robot's configuration as a point in a potential field that combines attraction to the goal and repulsion from obstacles. For example, the method proposed in [3] relies on the generation of a potential field by sigmoid and normal functions which eventually create the vector fields that control the velocity and heading of robot swarms. Similarly, relative headings to the goal and to obstacles, the distance to the goal and the angular width of obstacles were used in [36] to compute a potential field over the robot heading. The computed potential field controls the angular acceleration of the robot, steering towards the goal and far from the obstacles. In general, potential field approaches produce efficient trajectories however; higher accuracy can only be achieved by using higher order potentials, increasing the computational complexity. Heuristic approaches have been proposed as alternative solutions in order to simplify the problem. Ferguson proposed in [20] an extended version of a D-star algorithm [60] by using linear interpolation during each vertex expansion. Heuristic methods are characterized by their high time complexity which depends on the applied heuristic. On the contrary, cell decomposition methods were proposed to reduce the total complexity of the problem. Their main concept includes partitioning the free space into regions and identifying possible contacts between a single robot and obstacles in each region. For example, in [16], a variant of the A-star algorithm, namely Theta-star, propagates information along grid edges without constraining the paths to grid edges. As A-star based algorithms, the method requires a complete knowledge of the configuration area to produce the desired paths, thus, its implementation to a real system would be intricate. Nevertheless, Charalampous et al. in [12] combined the A-star with CA, and tested successfully the resulting method in real world planar environments. More specifically, the finite properties of the A-star algorithm were amalgamated with the CA rules to built up a substantial search strategy [11]. The corresponding algorithm's main attribute is that it expands the map state space with respect to time using adaptive time intervals to predict the potential expansion of obstacles.

Additionally, during the past few years, many artificial intelligence algorithms were employed as possible solutions to the problem so that the main drawbacks of the above solutions could be eliminated. In [43], a fuzzy based approach was proposed

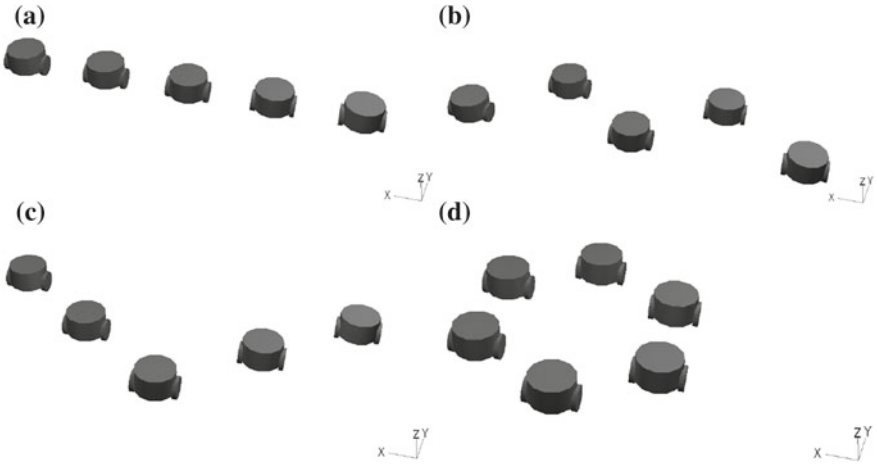

Fig. 9.2 Several formations of cooperative robotic teams

to navigate a mobile robot in an unknown dynamic environment filled with obstacles. The method requires for the robot to be equipped with a variety and highly accurate sensors to produce the desired pathways increasing the total cost of the system. More efficient paths were produced by the neural network in [69] nonetheless; in static environments and with high computational cost. A cell decomposition method along with an artificial intelligence method was used as a possible solution to the problem in [48]. Multilayered Cellular Automata (MCA) were applied where the configuration area is presented as a lattice of cells and four layers of identical grids are exploited for solving the path planning problem.

The majority of the above approaches deals with systems which include one robot and so only a single path should be extracted. The complexity of the solution is significantly increased in systems of multiple robots and is proportional to the total number of its members (Fig. 9.2). Such systems must complete further goals beyond the coverage of the desired distance. In case of a cooperative robot team, tracing paths for every robot becomes even more complex. The coordinated motion of a robotic team comprises one of the most widely studied research field in cooperative robotics and is known as formation control. Fredslund et al. in [26] achieved a collective behavior in a group of distributed robots using local sensing and minimal communication. Each robot references itself locally to one neighboring robot and keeps a certain bearing and distance by using an appropriate sensor. Moreover, a novel approach was proposed in [30] where formation structures are represented in terms of queues and formation vertices and formation control is accomplished using artificial potential trenches. A feedback law using Lyapunov type analysis was also exploited for a single robot and so collision avoidance and tracking of a reference trajectory was achieved [50]. Then, the method extends the resulted pathway to the case of multiple non-holonomic robots. Finally, a coordinated control scheme based

on a leader-follower approach was proposed in [18] so that formation maneuvers could be achieved. First and second order sliding mode controllers were used for asymptotically stabilizing the vehicles to a time varying desired formation.

The aforementioned methods for formation control and collision avoidance display high levels of computational complexity cost, despite their efficiency in both trajectory accuracy and formation immutability. Thus, their implementation in real systems could be either exclusionary or restricted. A minority of these methods could be potential for possible real system implementation nonetheless; robots must be equipped with farfetched sensors and high technical specifications which eventually increases the total cost of the system. On the other hand, their implementation in low cost robots restrict the number of tasks that could be executed leading to single purpose systems. For example, most of the commercial robots are equipped with digital cameras which are unable to be exploited since all the computational resources are occupied by the path planner. A low computational cost path planning approach could provide spare resources that are beneficial for exploiting the digital cameras to accomplish further tasks. However, these cameras capture low resolution images since higher resolution cameras require more processing power to analyze and manipulate a digital image. Instead, low resolution images are frequently inappropriate to achieve specific goals. Low resolution images can lead to false results when they are used to accomplish tasks like panoramic images or simultaneous localization and mapping (SLAM). Consequently, spare processing amounts could be used to increase the captured images' resolution with the application of image resizing methods. Several commonly used interpolation methods have been used for resizing images, such as nearest neighbor [38], bilinear [38] and bicubic interpolation [39]. In addition, a quadratic image interpolation method was proposed in [52] with adequate visual results nonetheless; its high computational cost prohibits its implementation. The preservation of the edges of a low resolution image comprises a crucial task and thus, a fuzzy decision system was developed in [45] to classify all the pixels of the input image into sensitive and non-sensitive class. Bilinear interpolation was applied to the non-sensitive regions while a neural network was used to interpolate the sensitive regions along the edges of the images. In order to decrease the computational complexity and preserve the satisfying image results, an edge-oriented method was proposed in [13] where the processed image is discriminated in non-edge and edge areas and each region is processed by a different type of interpolation method. The method achieves real-time image enlargement, despite the two stages of processing, nevertheless, the classification of the areas depends on a predefined threshold. Finally, the method in [44] initially estimates local covariance coefficients from the low resolution image which are sequentially used to adapt the interpolation at a higher resolution based on the geometric duality between the low-resolution and the high-resolution covariance. Despite its visually accurate resulted images, the method displays high levels of computational cost and thus, its application in real-time systems is restricted.

In this chapter, a unified architecture is presented for a robot team in order to accomplish several tasks and includes the application of an enhanced CA path planner combined with Ant Colonies Optimization (ACO) technique resulting to "Cellular

Robotic Ants". The presented path planner can produce adequate collision-free pathways with minimum hardware resources and low complexity levels. During the course of a robot team to its final destination, dynamic obstacles are detected and avoided in real time as well as coordinated movements are executed by applying cooperations in order to maintain the team's initial formation. The inherit parallelism and simplicity of CA result in a path planner that requires low computational resources and thus, its implementation in miniature robots is straightforward. The significant cooperations are limited to a minimum so that further resource reduction can be achieved. For this purpose, the basic fundamentals of ACO technique were applied. The entire robot team is divided into equally numbered subgroups and an ACO algorithm is applied to reduce the complexity. As each robot moves towards to its final position, it creates a trail of an evaporated substance, called "pheromone" . The "pheromone" and its quantity are detected by the following robots and thus, every robot is absolved by the necessity of continuous communication with its neighbors. The total complexity of the presented architecture results to a possible implementation using a team of miniature robots where all available resources are exploited. More specifically, the method was implemented in a cooperative robot team using a 3-D simulator, called Webots [51]. For testing purposes, the under study robot team was constituted of two subgroups with five robots each. All essential sensors, that all robots must be equipped, and their direct relations with the cell length and pheromone were introduced. The accuracy of the method was tested by using two different types of objects, rectangular and circular shaped. In both cases, the method created successfully collision free paths and the corresponding results exhibit the effectiveness and the robustness of the method.

The rest of the chapter is organized as follows. All the theoretical background for the CA and ACO algorithms is presented in Sect. 9.2 while the path planner, derived from the combination of CA and ACO, is presented in Sect. 9.3. Simulation results and the implementation of the architecture are presented in Sect. 9.4. Finally, conclusions are drawn in Sect. 9.5.

9.2 Cellular Automata and Ant Colony Optimization Principles

Cellular Automata (CA) were originally introduced by John Von Neumann [65] and his colleague Stanislaw Ulam [64]. CA can be considered as dynamical systems in which space and time are discrete and interactions are local. In general, a CA is consisted of a large number of identical entities with local connectivity arranged on a regular array. A finite Cellular Automaton could be defined by the quadruple:

$$\{d, q, N, F\} \tag{9.1}$$

From Eq. 9.1, variable d is a vector of two elements, m and n, denoting the vertical and horizontal CA dimensions, respectively. Both of these variables are expressed in number of cells. At each time step, the state of each cell is updated using a value

from the set $Q = 1, 2, \ldots, q-1$, called set of states. The neighborhood of each cell is defined by the variable N. For a 2-D CA, two neighborhoods are often considered, Von Neumann and Moore neighborhood. Von Neumann neighborhood is a diamond shaped neighborhood and can be used to define a set of cells surrounding a given cell $(x0, y0)$. Equation 9.2 defines the Von Neumann neighborhood of range r.

$$N^{v}_{(x0,y0)} = (x, y) : |x - x0| + |y - y0| \leq r \tag{9.2}$$

For a given cell $(x0, y0)$ and range r, Moore neighborhood can be defined by the following equation:

$$N^{M}_{(x0,y0)} = (x, y) : |x - x0| \leq r, |y - y0| \leq r \tag{9.3}$$

The transition rule f determines the way in which each cell of the CA is updated. The state of each cell is affected by the cell values in its neighborhood and its value on the previous time step, according to the transition rule or a set of rules. The state of every cell is updated simultaneously in the CA, thus, providing an inherent parallel system.

Consider for example a 2-D CA with two possible states, "0" and "1", as it is presented in Fig. 9.3. For this example, von Neumann neighborhood was used. In Fig. 9.3a, the central cell is stated as "1" at time t (black cell) while in Fig. 9.1b it is stated as "0" (white cell). The central cell updates its state to the next time step according to a simple XOR transition rule. If a central cell or any of its adjacent cells have the "1" state, at the next time step its state will be stayed into "1", as it is presented in Fig. 9.3, while its adjacent cells will update their state as "1".

CA have sufficient expressive dynamics to represent phenomena of arbitrary complexity and at the same time can be simulated exactly by digital computers, because of their intrinsic discreteness, i.e. the topology of the simulated object is reproduced in the simulating device. The CA approach is consistent with the modern notion of

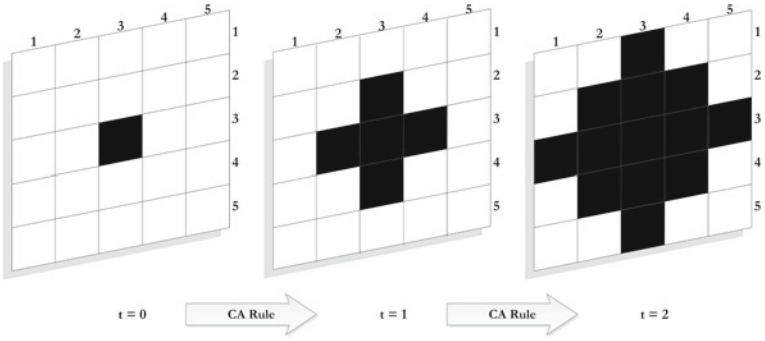

Fig. 9.3 2-D CA example with two possible states, "0" and "1". *Black cells* represent cells with state "1" while *white cells* represent "0" stated cells

unified space time. In computer science, space corresponds to memory and time to processing unit. In CA, memory (CA cell state) and processing unit (CA local rule) are inseparably related to a CA cell. Furthermore, CA are an alternative to partial differential equations [54, 62] and they can easily handle complicated boundary and initial conditions, inhomogeneities and anisotropies [31, 56].

The basic element of ACO algorithms is "ants" that is, agents with very simple capabilities which, to some extent, mimic the behavior of real ants [22]. Real ants are in some ways much unsophisticated insects. Their memory is very limited and they exhibit individual behavior that appears to have a large random component. However, acting as a collective, ants collaborate to achieve a variety of complicated tasks with great reliability and consistency [19], such as defining the shortest pathway, among a set of alternative paths, from their nests to a food source [4]. This type of social behavior is based on a common feature with CA, called self-organization, a set of dynamical mechanisms ensuring that the global aim of the system could be achieved through low level interactions between its elements [32]. The most vital feature of this interaction is that only local information is required. There are two ways of information transfer between ants: a direct communication (mandibular, antennation, chemical or visual contact, etc.) and an indirect communication, which is called stigmergy (as defined by Grassé [33]) and is biologically realized through pheromones, a special secretory chemical that is deposited, in many ant species, as trail by individual ants when they move [6]. More specifically, due to the fact that ants can detect pheromone, when choosing their way, they tend to choose paths marked by strong pheromone concentrations. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the nest. During the return trip, the quantity of pheromone that an ant leaves on the ground may depend on the quantity and quality of the food. The pheromone trails will guide other ants to the food source. This behavior is known as "auto catalytic" behaviour or the positive feedback mechanism in which reinforcement of the previously most followed route, is more desirable for future search. In ACO algorithms, an ant will move from point i to point j with probability:

$$\rho_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})} \tag{9.4}$$

where, $\tau_{i,j}^{\alpha}$ and $\eta_{i,j}^{\beta}$ are the pheromone value and the heuristic value associated with an available solution route, respectively. Furthermore, α and β are positive real parameters whose values determine the relative importance of pheromone versus heuristic information.

During their search for food, all ants deposit on the ground a small quantity of specific pheromone type. As soon as an ant discovers a food source, it evaluates the quantity and the quality of the food and carries some to the nest on their back. During the return trip, every ant with food leaves on the ground a different type of pheromone of specific quantity, according to the quality and quantity of the food.

In ACO algorithms, pheromone is updated according to the equation:

$$\tau_{i,j} = (1 - \rho)\tau_{i,j} + \Delta\tau_{i,j} \tag{9.5}$$

where, $\tau_{i,j}$ is the amount of pheromone on a given position (i, j), ρ is the rate of the pheromone evaporation and $\Delta\tau_{i,j}$ is the amount of pheromone deposited, typically given by:

$$\Delta\tau_{i,j} = \begin{cases} 1/L_k, & \text{if ant } k \text{ travels on edge } i, j \\ 0, & \text{otherwise} \end{cases} \tag{9.6}$$

where L_k is the cost of the kth tour of an ant (typically is measured as length). Finally, the created pheromone trails will guide other ants to the food source.

Consider for example the experimental setting shown in Fig. 9.4. The ants move along the path from food source F to the nest N. At point B, all ants walking to the nest must decide whether to continue their path from point C or from point H (Fig. 9.2a). A higher quantity of pheromone on the path through point C provides an ant a stronger motivation and thus a higher probability to follow this path. As no pheromone was deposited previously at point B, the first ant reaching point B has the same probability to go through either point C or point H. The first ant following the path BCD will reach point D earlier than the first ant which followed path BHD, due to its shorter length. The result is that an ant returning from N to D will trace a stronger trail on path DCB, caused by the half of all the ants that by chance followed path DCBF and by the already arrived ones coming via BCD. Therefore, they will prefer path DCB to path DHB. Consequently, the number of ants following this path

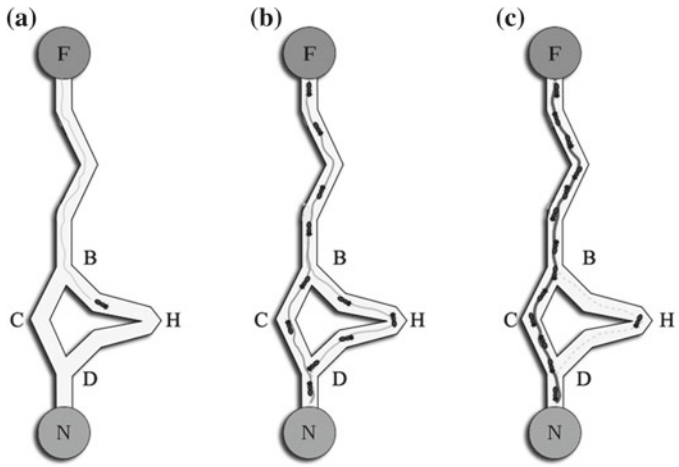

Fig. 9.4 An example of real ants colony: **a** An ant follows BHD path by chance, **b** Both paths are followed with same probability and **c** Larger number of ants follow the shorter path

will be increased during time than the number of ants following BHD. This causes the quantity of pheromone on the shorter path to grow faster than the corresponding longer one. Consequently, the probability with which any single ant chooses the path to follow is quickly biased towards the shorter one.

The ACO algorithms are basically a colony of artificial ants or cooperative agents, designed to solve combinatorial optimization problems. These algorithms are probabilistic in nature because they avoid the local minima entrapment and provide very good solutions close to the natural solution [7]. ACO algorithms are extensively used to a variety of applications such as the travel salesman problem [23], image retrieval [40, 41], classification [49], electrical load pattern grouping [14], video games [57], seismic methods [15], communications networks [21], etc. Moreover, as it is proposed in this paper, ACO algorithms were also used for solving the path planning in a team of robots and most of the effort was to implement the algorithm in real systems. More specifically, in [58] heat applicators and sensors were used as virtual pheromone and detectors, respectively, so that ACO algorithm could be functional. Furthermore, Garnier et al. in [29] used a visual system, a computer and a projector to track robots, process the data and project the light, respectively. Moreover, Garnier et al. in [27] proposed a group of ant-like robots that had to establish a route between a starting area and a target area in a network of corridors, mimicking the experiments we performed with ants in authors' previous studies [28]. For technical convenience pheromone trails were replaced by light trails projected along the paths followed by the robots by a video projector (as proposed in [58] and implemented in [29]). Robots could detect and follow these light trails thanks to two photoreceptors that mimic the antennae of the ants as also described in the corresponding review article of Akst [1]. Finally, in [34, 35], RFID stickers were used as data carriers, which were a priori placed in the area, and they stored data representing the pheromone levels. At each robot of a team, an RFID reader/writer was mounted in order to read/write the data stored in the RFID stickers.

9.3 Cellular Ants: A Combination of CA and ACO Algorithms for Path Planning

One of the main goals of the proposed method is to create collision free trajectories for every robot of a cooperative team. No a priori knowledge of the configuration area is required. Obstacle avoidance must be achieved in real time. Knowing their final position, that is the end of a straight line path, robots can move randomly in the configuration area, according to ACO algorithm. To prevent a scattered formation due to either an obstacle or a complete absence of pheromone, cooperations between the members of the team are applied so that their formation could be regained or retained immutable, respectively. According to the ACO algorithm, every single ant is governed by a set of simple behavior rules, leading to an uncomplicated approach of the path planning problem. Due to CAs, these behavior rules are applied simultaneously

to all ants, in a discrete and iterative way. A concurrent evolution of the entire system ensures the rapid formation of all possible trajectories. Furthermore, the proposed method covers the need for self-organization, since the utilized artificial intelligence algorithms embody this particular attribute. In the following subsections, the proposed method is described in detail. Simulation results are provided as well.

9.3.1 Proposed Method

First, an appropriate CA must be defined, meaning that the variables in Eq. 9.1 must be specified. The configuration area, in which the robot team operates, is considered to be a 2-D lattice and so is divided into a simple rectangular grid of identical square cells. The dimensions of the CA (variable d) are selected based on the dimensions of the configuration area and the cell length. Let $m \times n$ be the dimensions of the CA, therefore $d = m, n$, and w the length of each cell.

Moreover, each cell can take a finite number of possible states, named set of states $Q = 0, 1, 2, \ldots, q$. Variable q is proportional to the number of robots that comprise the team and the predefined pheromone levels. Firstly, vector $F = 0$ represents the state of a cell unoccupied by a robot, an obstacle or pheromone, called a free cell. In addition, vector $R = 1, 2, \ldots, r$ represents all possible states of a cell occupied by a robot. Due to the duality of their nature as a CA cell and as an artificial ant, a cell occupied by a robot is labeled as a cellular ant. For reasons of complexity, the entire robot team is divided into smaller subgroups, equally numbered and forming the same pattern. Furthermore, $P = r + 1, r + 2, \ldots, \rho$ is a vector of all possible pheromone states, where ρ denotes a cell with the maximum pheromone level. Finally, variable $W = \rho + 1$ indicates a cell occupied by an obstacle. These vectors were created to avoid overlapping between possible states according to the following equations:

$$F \cup W \cup R \cup P = Q \;\&\; F \cap W \cap R \cap P = Q \tag{9.7}$$

Summarizing, every CA cell could obtain one possible state at each time step:

- Free cell: $cell(x, y) = 0$
- Cellular ant: $cell(x, y) = r$ where $v \in N : r_v \in R$
- Pheromone cell: $cell(x, y) = \rho$ where $v \in N : \rho_v \in R$
- Obstacle cell: $cell(x, y) = \rho + 1$ where ρ the state of maximum pheromone level

where x and y are the Cartesian coordinates of a cell in the CA grid.

Every CA cell evolves its state according to its current state and the state of each neighboring cell. For the proposed method, Moore neighborhood was used with range $r = 1$ (Eq. 9.3). Moreover, cells located on the frontier of the grid evolve their state using null boundary conditions, meaning that all cells of the firstlast row and those of the first/last column are enduringly in the 0 state.

At every time step, all cells update their state simultaneously by applying the appropriate local transition rule of the CA, $F : Q \leftrightarrow Q$, so that:

$$cell_{i,j}^{t+1} = F cell_{i,j}^{t}, cell_{i-1,j-1}^{t}, cell_{i-1,j}^{t}, \ldots, cell_{i+1,j+1}^{t} \tag{9.8}$$

Three different sets of CA rules were created for achieving different operations: collision avoidance, pheromone update and formation control.

9.3.1.1 CA Rules for Collision Avoidance

Appropriate CA rules were created so that objects could be avoided. As a cellular ant moves towards to its final position, it scans its Moore neighborhood at every time step in order to detect a potential obstacle. Due to this attribute, both static and dynamic objects can be detected. Depending on the status of its adjacent cells and its direction, the appropriate CA rule is selected for the evolution of the state of the corresponding cellular ant. Considering an obstacle detected by a cellular ant, at first, the applied CA rule is chosen according to the position of the cellular ant in the formation. If it is positioned on the right of the central cellular ant of the formation, the right pathway is selected to bypass the obstacle. On the contrary, the left pathway is selected in case of a left positioned cellular ant. Subsequently, the rest of the avoidance process is akin with a wall following behavior. If the obstacle is avoided and the cellular ant regain its position over the desired shortest path, suitable CA rules are applied in order to continue its way to its final destination. Table 9.1 illustrates a small subset of eight collision avoidance CA rules out of a total of forty possible CA rules. Figure 9.3 represents the application of CA rules for obstacle avoidance as a schema. More specifically, in Fig. 9.5a, a cellular ant detects that its NW and its N adjacent cells are occupied by obstacles, its NE cell as a free cell, "detects" that its SW is a cellular ant cell and its W is an obstacle cell. By applying the appropriate CA rules, at time $t + 1$, the under study central cellular ant cell is stated as a pheromone cell with maximum pheromone quantity and its NE free cell becomes a cellular ant

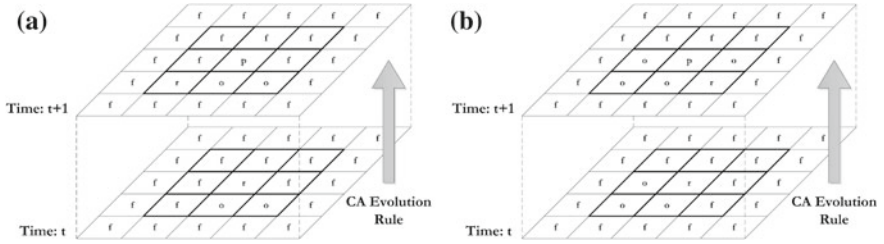

Fig. 9.5 Schematic presentation of two CA rules for collision avoidance: f denotes a free cell, r denotes a cellular ant cell, o an obstacle cell and ρ a pheromone cell with the highest possible pheromone state. **a** NW and N adjacent cells are obstacle cells and **b** N, NE and E adjacent cells are obstacle cells

Table 9.1 A subset of CA rules for collision avoidance

Cell state									
At time t									At time $t+1$
$S_{i,j}$	$S_{i-1,j-1}$	$S_{i-1,j}$	$S_{i,j+1}$	$S_{i,j+1}$	$S_{i+1,j+1}$	$S_{i+1,j}$	$S_{i+1,j-1}$	$S_{i,j-1}$	$S_{i,j}$
r	f	f	f	f	f	f	o	f	f
f	f	r	f	f	f	f	f	o	r
r	f	f	f	f	o	f	f	f	f
f	f	r	f	o	f	f	f	f	r
r	f	f	f	f	f	o	o	f	f
f	r	f	f	f	f	f	f	o	r
r	f	f	f	f	o	o	o	f	f
f	f	f	f	r	o	o	f	f	r

S denotes cell state, (i, j) Cartesian coordinates of cell on the CA, r a cellular ant cell, o an obstacle cell and f a free cell

cell. Essentially, a state transition is occurred between the central cell and its NE cell. Figure 9.5b illustrates a similar situation with different states of its adjacent cells.

9.3.1.2 CA Rules for Simulating the Pheromone Functions

To simulate all relative functions of pheromone, appropriate CA rules were also created. There is an immediate correlation between the state of a pheromone cell and the corresponding pheromone value. The state of a pheromone cell is relevant to the detected pheromone quantity (value $\tau^{a}_{i,j}$). Every cellular ant scans its front adjacent cells in order to be stated with the appropriate pheromone state. If a cell is marked with the highest pheromone state, it will be more likely for the cellular ant to follow this trail, according to Eq. 9.4. Moreover, supplementary CA rules were created in order to simulate the pheromone evaporation process. A pheromone cell surrounded by either free cells or pheromone cells updates its state according to Eqs. 9.5 and 9.6 by decreasing its value. Furthermore, the evaporation rate is expressed as state per time steps due to the quantized states of the CA. Depending on the pheromone levels and the evaporation rate, numerous CA rules could be used to simulate both the stigmergic behavior of a cellular ant and the evaporation of the pheromone. Table 9.2 illustrates a small subset of eight pheromone update CA rules, while Fig. 9.4 represents a schematic example of their application. In Fig. 9.6a, a cellular ant, namely r, detects two of its adjacent cell to be stated as pheromone cells, $\rho 5$ and $\rho 1$, respectively. The $\rho 5$ cell has a higher pheromone state meaning that it is occupied by a higher amount of pheromone. According to Eq. 9.4, it is more likely that the cellular ant will follow this path and thus, the appropriate CA rule is applied in order this pathway to be followed. Additionally, in Fig. 9.6b, a pheromone cell with state $\rho 5$ at time t updates its state by decreasing its state in order to simulate the pheromone evaporation.

Table 9.2 A subset of CA rules for collision avoidance

Cell state									
At time t									At time $t+1$
$S_{i,j}$	$S_{i-1,j-1}$	$S_{i-1,j}$	$S_{i,j+1}$	$S_{i,j+1}$	$S_{i+1,j+1}$	$S_{i+1,j}$	$S_{i+1,j-1}$	$S_{i,j-1}$	$S_{i,j}$
r	f	f	f	f	f	f	o	f	ρ
f	f	f	r	f	f	f	f	o	r
r	f	f	f	f	f	f	f	f	f
f	f	r	f	o	f	f	f	f	r
r	f	f	f	f	f	$\rho 2$	$\rho 5$	f	ρ
$\rho 2$	f	r	f	f	f	f	f	$\rho 5$	r
ρ	f	f	f	f	f	f	f	f	ρ
$\rho 1$	f	f	f	f	f	f	f	f	$\rho 2$

S denotes cell state, (i, j) Cartesian coordinates of cell on the CA, r a cellular ant cell, o an obstacle cell and f a free cell and ρ pheromone cells (increased pointers mean low levels of pheromone)

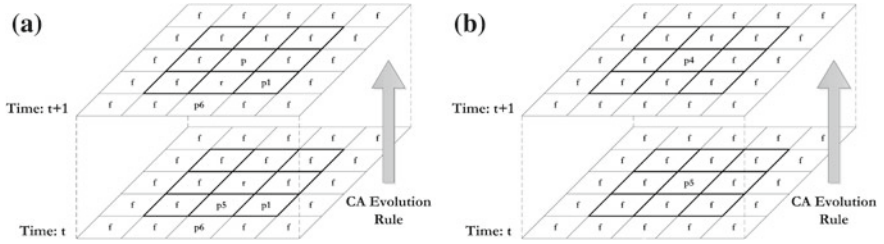

Fig. 9.6 Schematic presentations of two CA rules for simulating the pheromone functions: f denotes a free cell, r denotes a cellular ant cell and ρ a pheromone cell (indices represent pheromone level; higher index value corresponds to higher pheromone level, state p without any index represents a pheromone cell with the highest possible pheromone state). **a** Following the strongest pheromone trail and **b** Simulating the pheromone evaporation process

9.3.1.3 CA Rules for Formation Control

Each subgroup of cellular ants must retain and regain their initial formation during the entire route. If an obstacle is detected by a cellular ant of a subgroup, the formation of its subgroup will be scattered. Moreover, in case of complete absence of pheromone, according to Eq. 9.4, a cellular ant could move towards to any of its adjacent free cells with the same probability leading to a scattered formation as well. In order to prevent such behaviors, suitable CA rules were created based on cooperations between the cellular ants of every subgroup. It is assumed that every cellular ant has a substandard ability to communicate with its immediate neighbors. The coordinates of every cellular ant in the grid are required in order to complete the formation control process. Therefore, these coordinates are handled as the exchanged data. Depending on these data, all the necessary decisions are taken by each cellular ant meaning, whether to swap positions in the formation with its neighbors, move one cell to their destination or just stay still until the formation is regained by the other

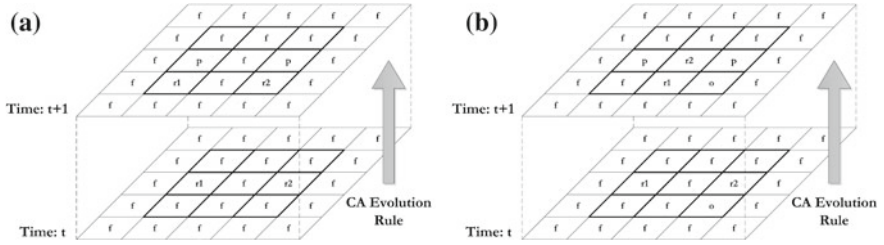

Fig. 9.7 Schematic presentations of two CA rules for formation control: f denotes a free cell, $r1$, $r2$ denote cellular ant cells, o an obstacle cell and ρ a pheromone cell with the highest pheromone state. **a** Moving to their goal and **b** Exchanging positions in formation

cellular ants of the subgroup. These decisions are expressed as simple CA rules in order to evolve their state to the next time step. Figure 9.7 illustrates two examples of the CA rules set for formation control. In Fig. 9.7a, no pheromone is detected by the $r1$ and $r2$ cellular ants and therefore, they communicate in order to proceed to data exchange. Since $r2$ finds its neighbor $r1$ to its vertical position and vice-versa, both cellular ants commonly decide to update their state in order to move one cell towards their destination. On the contrary, in Fig. 9.5b, cellular ant $r2$ detects an obstacle and therefore, it will try to bypass it leaving its position in the formation. At the same time, the $r1$ cellular ant discovers that it has smaller vertical distance than the predefined with $r2$ (after communication). Thus, they commonly decide to exchange their position in the formation in order to regain the formation as soon as possible.

For example, consider a cellular ant cell with state r and coordinates (i, j) with no obstacle found in its neighborhood at time t. At the given time, let us assume that all cells of its Moore neighborhood have the following states: $cell^t_{i,j} = a0 = r$, $cell^t_{i-1,j-1} = a1 = f$, $cell^t_{i-1,j} = a2 = f$, $cell^t_{i-1,j+1} = a3 = f$, $cell^t_{i,j+1} = a4 = f$, $cell^t_{i+1,j+1} = a5 = f$, $cell^t_{i+1,j} = a6 = f$, $cell^t_{i+1,j-1} = a7 = f$, $cell^t_{i,j-1} = a8 = f$. When applied to Eq. 9.8, these values result to $cell^{t+1}_{i,j} = a0^{t+1} = F(a0^t, a1^t, a2^t, a3^t, a4^t, a5^t, a6^t, a7^t, a8^t) = F(r, f, f, f, f, f, f, f, f) = r$. Furthermore, at time step t, the free cell with coordinates $(i + 1, j)$ will have the following values $a0 = f$, $a1 = f$, $a2 = r$, $a3 = f$, $a4 = f$, $a5 = f$, $a6 = f$, $a7 = f$, and $a8 = f$. Applying simultaneously the appropriate CA rule, the corresponding results are: $cell^{t+1}_{i,j} = a0^{t+1} = F(a0^t, a1^t, a2^t, a3^t, a4^t, a5^t, a6^t, a7^t, a8^t) = F(f, f, r, f, f, f, f, f, f) = r$. In other words, with the use of simple rules, the cellular ant cell manages to cover the distance of a cell to reach the goal position.

The main aims of the method are both the creation of collision free trajectories in dynamic environments and a fixed formation for every subgroup of the team. In case of a scattered formation, the members of the subgroup must recover the formation as soon as possible with the minimum requirements. For reasons of simplicity and for the given example, a straight line formation was applied. All cellular ants of the first

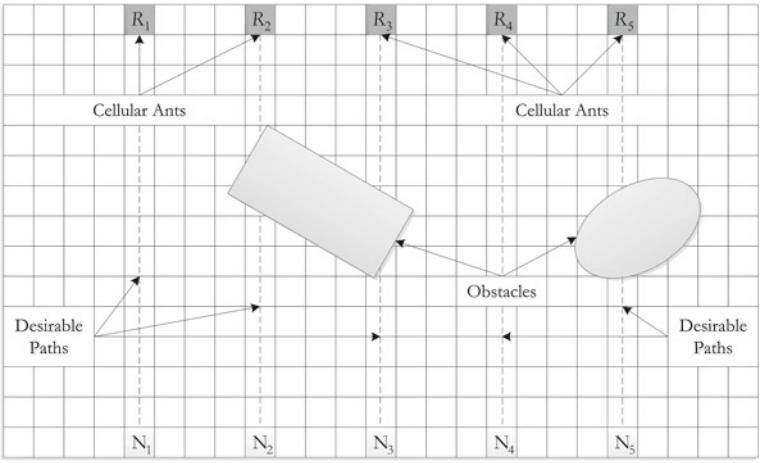

Fig. 9.8 Initial state of the CA grid, where *R* denotes cellular ant cells and *N* final positions (nests)

subgroup are deployed in either the first row or the first column of the CA having a specific vertical or horizontal distance, respectively. The forthcoming subgroups are subsequently introduced at the same positions in the grid after a user defined time period. Figure 9.8 illustrates the initial state of the CA grid for the aforementioned example using a first row deployment of the cellular ants.

For this particular example and for testing purposes, this specific type of formation was selected due to the nature of the pattern that cellular ants form [37]. Consider a real system consisting of multiple robots, each equipped with a digital camera. Images taken from each camera have a horizontal difference presenting the same scenery. Many image processing algorithms exploit this difference for a variety of applications, such as resolution enhancement [59] and panoramic images [9].

The method assumes that the initial and the final position of every cellular ant are its food source and its nest, respectively. In case of a position exchange in the formation, the corresponding cellular ants will exchange their final positions-nests, as well. At every time step, each cellular ant scans its Moore neighborhood in order to detect any possible obstacle. If an object is detected, the appropriate CA rule for obstacle avoidance is selected and applied without requiring any further process, e.g. pheromone detection. Otherwise, every cellular ant will scan its front adjacent cells in order to detect any pheromone quantity. The probability for every front cell is calculated according to Eq. 9.4 and a quantization is applied. Each quantized probability is mapped to a pheromone state and every cell is marked with the corresponding pheromone state. A higher pheromone quantity means that the specific path is more likely to be followed. Moreover, the formation of the subgroup is scattered either when an obstacle is detected or in cases of complete pheromone absence. In order to keep the formation of the subgroup immutable (lack of pheromone) or regain it (obstacle detection), formation control principles are applied where all decisions are

expressed as CA rules. These functions are achieved through cooperations between the cellular ants of a subgroup, meaning that data relative their position in the grid are exchanged. Every cellular ant knows about the presence of its two nearest cellular ants with which data exchange is achieved. In case of a deviated cellular ant from its desired shortest path due to an obstacle, its absence can be detected by its neighbors using its coordinates. If a cellular ant bypasses an obstacle from the left pathway, the horizontal distance with its left neighbor will be decreased while the distance with its right neighboring cellular ant will be increased. Thus, a position exchange between the cellular ant and its left neighbor will occur while the right neighbor will wait until the exchange is completed. When swapping is completed, neighboring cellular ants synchronize their actions by exchanging all necessary data, until all vertical coordinates are equalized. All formation control functions are continuously applied till the formation is regained, searching only for obstacle cells since the method is applicable in dynamic environments. When the straight line formation is regained, the entire subgroup will move towards to its final position as a team, reaching their targets synchronized.

As a cellular ant moves to a free or a pheromone cell, its state will be updated with the corresponding CA rule and thus, at the next time step, it will be stated as a pheromone cell with the maximum defined pheromone state. The cellular ants of the next subgroup detect these pheromone trails and consequently, no formation control principles are required in order to retain or regain the formation of the next subgroup. This advantage reduces the complexity that the formation control process introduces. Additionally, in a real system, the formation control process requires the members of the team to be able to communicate and thus, more time and resources are needed in order to keep the formation fixed. By using simple sensors for pheromone detection, if some pheromone is detected, cooperations and data exchange are no longer necessary. Finally, according to ACO algorithm, past events or situations could be expressed as modifications or "unusual" status of pheromone, acting as a local memory, since it is deposited by the anterior ants. In the proposed method,

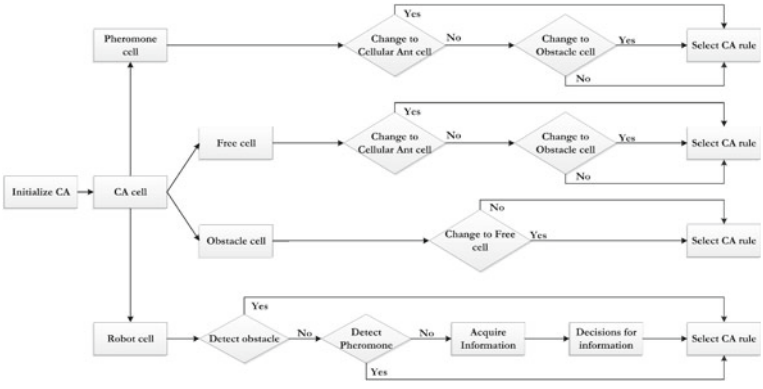

Fig. 9.9 Flowchart of the proposed method

these "unusual" pheromone allocations are expressed as a position exchange in the formation and are detected by the forthcoming cellular ants in order to prevent a false pheromone trail pathway. The method in a form of a flowchart is provided in Fig. 9.9.

9.4 Simulation Results

For testing purposes of the proposed method, a simulation environment was created and some results are illustrated in Fig. 9.10. The under study environment operates under Microsoft Windows OS and was created using the C# program language and a graphical user interface. The created simulator includes multiple subgroups constituted of numerous cellular ants. For testing purposes, each subgroup is comprised by five cellular ants and a straight line formation was applied. The program provides the ability to the user to define the dimensions of the CA, the number of cellular ants for all subgroups, the number of iterations-time steps and to draw rectangular or elliptical shapes as possible objects. Additionally, the pheromone levels and the evaporation ratio are also user defined. For the examples illustrated in Fig. 9.10, the dimensions of the CA are 21×25 cells with a number of time steps equal to 100. Every subgroup comprises by 5 cellular ants and is inserted in the grid after 25 time steps. Also, 10 pheromone levels were selected, meaning that 10 possible states can be used for the pheromone cells and an evaporation rate equal to 1 state per 10 time steps. Cellular ants are denoted with the letter R and corresponding numbers representing their position in the formation; object cells are depictured with darkened green squares, nests (final positions) with green and denoted with the letter N and finally, pheromone cells are illustrated with tones of blue depending on the quantity of the pheromone. As Fig. 9.10 shows, initially, the CA path planner creates collision free paths for every cellular ant using the formation control rules since no pheromone is deposited on the grid. At each time step, every cellular ant moves towards its final destination avoiding all possible obstacles found on its desired shortest route and leaving a pheromone trail on the grid. After a short period, the second subgroup leaves from their food source and thus, amplifying the existing pheromone trails and so on. Figure 9.10a illustrates the simulation results in case of rectangular objects while Fig. 9.10b presents the case of elliptical objects. For comparison reasons, all simulation parameters are kept immutable for both cases.

Simulation results indicate that the proposed method can produce accurate collision free trajectories for every cellular ant with low complexity since ACO principles are applied. Due to the fact that the proposed method mainly uses a CA, time complexity is found to be approximately $O(m \times n)$ as the majority of the CA algorithms, where $m \times n$ are the dimensions of the rectangular grid, meaning that time complexity is proportional to the CA dimensions. Moreover, time complexity is kept low in any environment, both static and dynamic. The procedure retains the principles of self organization of ACO algorithms and concurrently uses CA to achieve our goal. Furthermore, the whole process is accomplished without any interference of a central

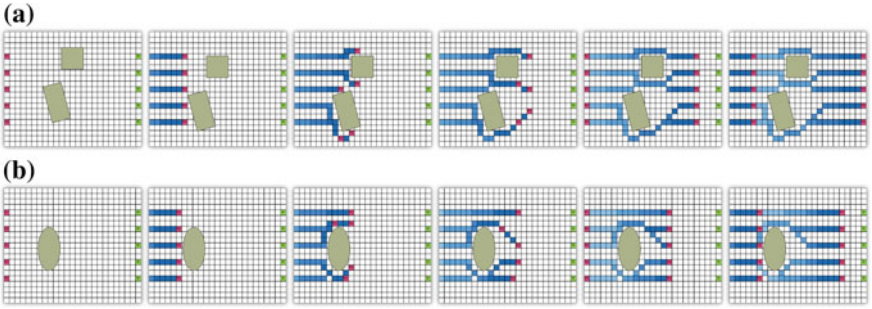

Fig. 9.10 Simulation results for **a** rectangular objects and **b** *elliptical* objects, where cellular ants are denoted with the letter *R* and a number (formation position), final positions are represented by *pale gray cells*, object cells are illustrated by *darkened cells* and pheromone cells are illustrated with tones of gray

control, making the system autonomous, and is completely parallel. Finally, due to the fact that each subgroup consists of multiple robots, a cellular ant could wrongly follow a pheromone trail leading to a scatter formation. Using the advantages of the CA path planner, this problem could be resolved as depicted in Fig. 9.10.

9.4.1 Implementation of the Method in a Simulated Cooperative Robot Team

The proposed method was implemented in a simulated cooperative robot team using a three dimensional simulator, called Webots [51]. All robots and instances simulated in Webots are actually real robots that either can be purchased or used for educational reasons. For our experiments, multiple miniature robots were used, called e-puck [8]. All robot controllers needed for the implemented method were created using the language C. E-puck robot is equipped with a variety of sensors which can be used for the implementation and its hardware structure is considered to be simple. To be fully functional, the method requires that every robot of the team must be equipped with specific types of sensors such as distance sensors, differential wheels, communication modules for data transmission/receipt and appropriate sensors for pheromone detection. Real e-puck robot is equipped with all the above sensors as well as the corresponding e-puck robot in Webots. A more detailed description about the sensors that were used for the implementation is presented below.

The proposed method requires every robot to be equipped with specific types of sensors so that it can have a completely awareness of its environment. Considering that, every robot could avoid possible obstacles, detect pheromone trails and

communicate with its neighboring robots during its route to its final destination. Consequently, for every action different type of sensors is needed.

9.4.1.1 Sensors for Obstacle Avoidance

E-puck robot uses infrareds as distance sensors to detect obstacles in the configuration area. For reasons of correspondence, infrared distance sensors were also used for the implementation in Webots. Unfortunately, as Fig. 9.11 illustrates, infrareds on the real e-puck robot are misplaced. Due to the quantization of the configuration area, obstacle detection cannot easily be achieved, leading to a less accurate system. For example, if an obstacle with small size is present between two infrared sensors, due to the distance between them, the robot will not be able to detect the object. By dividing the configuration area into smaller cells, the system can be more accurate at the expense of memory resources. However, in systems with miniature robots, memory restrictions are of great importance.

An obstacle can be detected only if it is in the range of a distance sensor so that the corresponding cell can be stated as an obstacle cell. Consequently, the cell length must be at least equal to the distance sensor range. Small memory amounts are required for large cells but less accurate readings from the sensors are achieved. Conversely, using small cell length, more accurate readings can be processed from the sensors but memory demands exceed the limit. To overpass this problem, the best solution is to define an average cell length, and use numerous sensors. For that reason, some modifications are required to be done on the e-puck robot. Different number of proximity sensors was attached on the robot to define the appropriate cell length so that maximum accuracy could be achieved. All distance sensors were

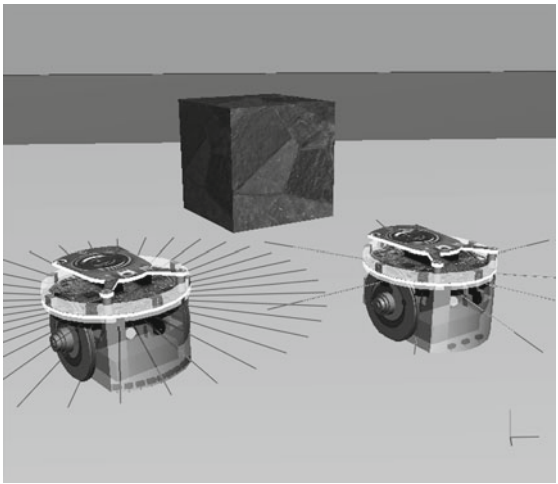

Fig. 9.11 From *left* to *right*: modified and real e-puck robot. The continuous lines represent the range of the infrared sensors

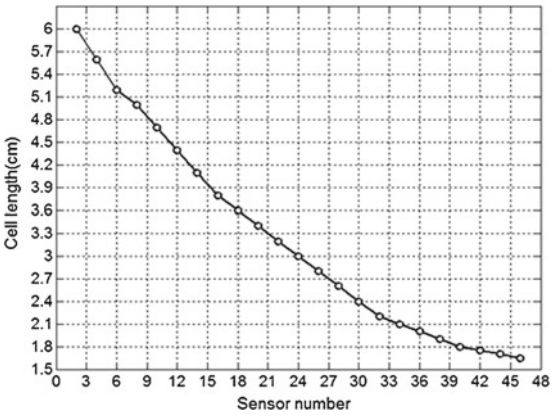

Fig. 9.12 Appropriate cell length for different number of proximity sensors

placed around the e-puck robot every time with different angle. Figure 9.12 illustrates the cell length of the system in regard to the required number of distance sensors so that an object could be detected at any angle.

In case of minimum sensor number, big cell length is required and therefore less memory is needed. Nevertheless, less accuracy is achieved and thus, some objects may not be detected. As the cell length is decreased, more sensors are required so that small obstacles can be detected. High accuracy is achieved but more memory is required. Therefore, to achieve maximum accuracy, multiple sensors must be used without using inefficient number of memory. For that reason, thirty six infrared sensors for obstacle avoidance were used instead of eight that real e-puck robot has. Each sensor was placed on the circumference of each robot having a distance of ten degrees with its neighboring sensors. Figure 9.11 illustrates the position of each sensor on a robot while Figs. 9.11 and 9.14 present the modified e-puck robot. Moreover, to state a free cell as an obstacle cell, the readings from the sensors must be compared with a threshold, relative to cell length. For accuracy reasons, this number is equal to the weighted sum of the sensors readings which are physically included in the correspondence cell. A 1×5 Gaussian convolution operator was used as coefficient. For example, in Fig. 9.13, assume that the value which describes the state of the north cell is equal to:

$$\begin{aligned} value = {} & 0.0545 \times a0 + 0.2242 \times a1 + 0.4026 \times a2 + 0.2242 \times a3 \\ & + 0.0545 \times a4 \end{aligned} \tag{9.9}$$

where $a0$, $a1$, $a2$, $a3$ and $a4$ are the readings from the sensors which are found in the north cell, from left to right. This value is compared with a threshold and if it is greater, the cell will be stated as obstacle cell.

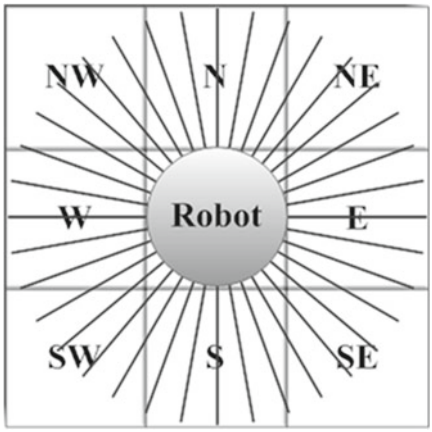

Fig. 9.13 Sensor Positions on a Robot

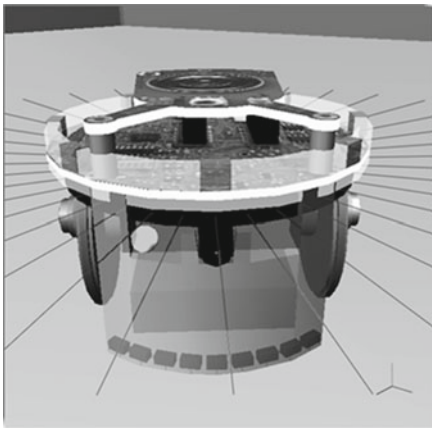

Fig. 9.14 Modified e-puck robot with infrared ground sensors

9.4.1.2 Sensors for Pheromone Detection

A special component, given by Webots and called pen, was attached to every robot. The pen component models a pen attached to a mobile robot, typically used to present the trajectory of the robot. The created trails can be considered as pheromone paths which can be detected by every robot. Essentially, the quantity of pheromone can be simulated by tones of gray created by the pen on the ground. A tone near to black represents a high accumulation of pheromone while near to white tones correspond to low levels of pheromone. A transition from black to white tones simulates the evaporation process of pheromone. Moreover, appropriate sensors must be selected so that the tones of gray can be detected. For this purpose, multiple infrared sensors were attached to every robot looking down to the ground, as illustrated in Fig. 9.14. According to their structure, infrareds can detect the amount of

the reflected beam which is transformed to electrical signal. If a sensor measures a trail with a black tone, a high value will be resulted. In case of closely white trails, small readings from the sensors will occur. As in obstacle avoidance, these readings are combined using a Gaussian convolution kernel. Finally, the resulted values are compared with thresholds to produce different pheromone levels for the pheromone cells of the CA.

At this point, it should be mentioned that in real e-puck robots, electrical circuits using infrared sensors can be attached on their front for the same purpose. Three infrared sensors are attached under the camera of the robot, facing the ground, and are used as ground sensors to detect possible chromatic trails.

9.4.1.3 Communication Modules

In case of absence of pheromone or a scattered formation, cooperations between the robots of each subgroup must be achieved so that their formation can be either retained or regained, respectively. Cooperation tasks are accomplished using data exchange through a communication link. Each robot communicates with its neighbors and swaps information about its position in the grid. Depending on these data, robots decide whether to exchange their position in the formation, move towards their final destination or remain to their current position. Real e-puck robot is equipped with a Bluetooth module for serial communications. In Webots, the modified e-puck robot uses a transmitter and a receiver component to achieve interactions. Essentially, these two components simulate the operations that a real serial communication module can complete. Each robot must scan its neighborhood to avoid obstacles, detect possible pheromone trails and communicate with other robots, if necessary, to update its state to the next time step. Thus, the required time to create its path is proportional to the execution time of each above operation. The total execution time is equal to:

$$\tau_{total} = \tau_{prox} + \tau_{ph} + \tau_{com} + \tau_{ex} \tag{9.10}$$

where τ_{total} is the total execution time, τ_{prox} is the required time for handling the proximity sensors and τ_{ph} is the necessary time for handling the sensors for pheromone detection. Moreover, τ_{com} is the required time for transmittingreceiving data for cooperation tasks and τ_{ex} is the necessary time for each robot to process all data. Each of the above parameters are strictly connected with the specifications of the available hardware modules that every robot is equipped. At this point it must be mentioned that the processor of every robot is far faster than the other sensors and thus, τ_{ex} is considered to be insignificant. Moreover, in case of a detected pheromone trail, communications between robots are not required making the creation of the paths even faster. Depending on the used sensors in either a system or a simulated system, the

method could create collision free paths at real time and regain a scattered formation after a small time period.

9.4.2 Simulator Results

For testing purposes, the proposed method was implemented in a cooperative robot team consisted of ten modified e-puck robots, using Webots. The entire team was divided into two equally numbered subgroups of robots. For this specific system, the cell length was decided to be equal to 2 cm for accuracy reasons. Moreover, every robot must cover a distance of 140 cm meaning that the configuration area is divided into a lattice with dimensions 70×70 cells. Finally, every robot moves to its final destination following the desired shortest path, that is a straight line trail, and each subgroup is forming a straight line pattern.

At the first time step, the first subgroup leaves from its initial positions, or else food sources. Each robot enables its distance sensors so that possible obstacles can be detected. Depending on the received readings, every robot decides if an obstacle is present or not. In case of a detected obstacle, the correspondent cell will be stated as an obstacle cell and by applying the appropriate CA rule, collision avoidance is achieved. If no object is present, every robot enables its ground sensors to search if any pheromone is deposited on the ground. Depending on the sensor readings, the robot decides if any of its neighboring free cells must be stated as a pheromone cell with state that represents the detected pheromone level. If a pheromone cell is present, the appropriate pheromone CA rule is applied. Obviously, no robot of the first subgroup detects any pheromone trail so cooperation tasks are applied. Every robot communicates with its neighbors and exchanges data related to its position in the grid. According to the response, each robot selects the appropriate CA rule to evolve its state to the next time step, meaning that the robot will adjust its movements. The same procedure is repeated until every robot covers the desired distance. After a period of time, the second subgroup begins its route to their final points in the configuration area following the exact same process.

Figures 9.15, 9.16 and 9.17 illustrate screenshots obtained during the evolution process. In particular, screenshots of the robot team avoiding a rectangular shaped object are shown in Fig. 9.15 while Fig. 9.16 demonstrates the avoidance of circular object. Moreover, Fig. 9.17 illustrates how robots react in a case of a dynamical environment. The presented results prove that the proposed method can produce accurate collision free paths by using simple and cheap sensors. Furthermore, the robustness and the effectiveness of the method are established.

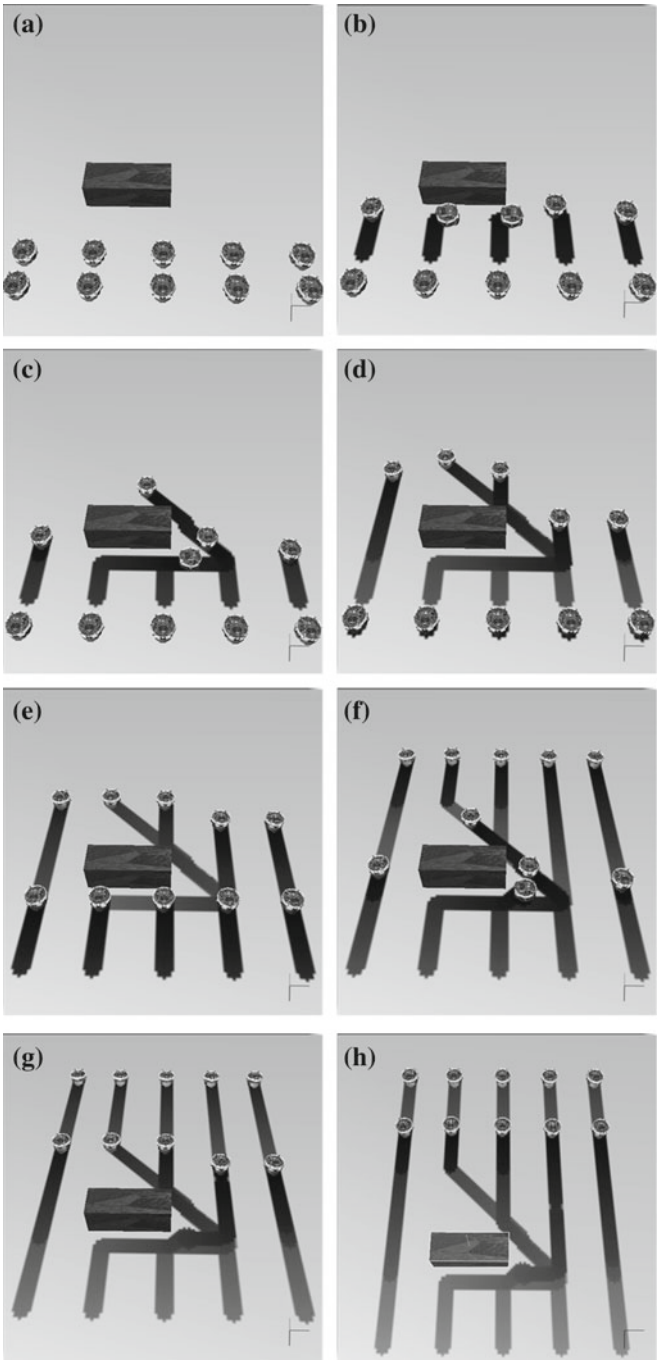

Fig. 9.15 Modified e-puck robot team avoiding a *rectangular shaped* object (from *left* to *right* and from *upper* to *lower* images), where green spots are the robots and "pheromone" trails are illustrated with lines of tones of *black*

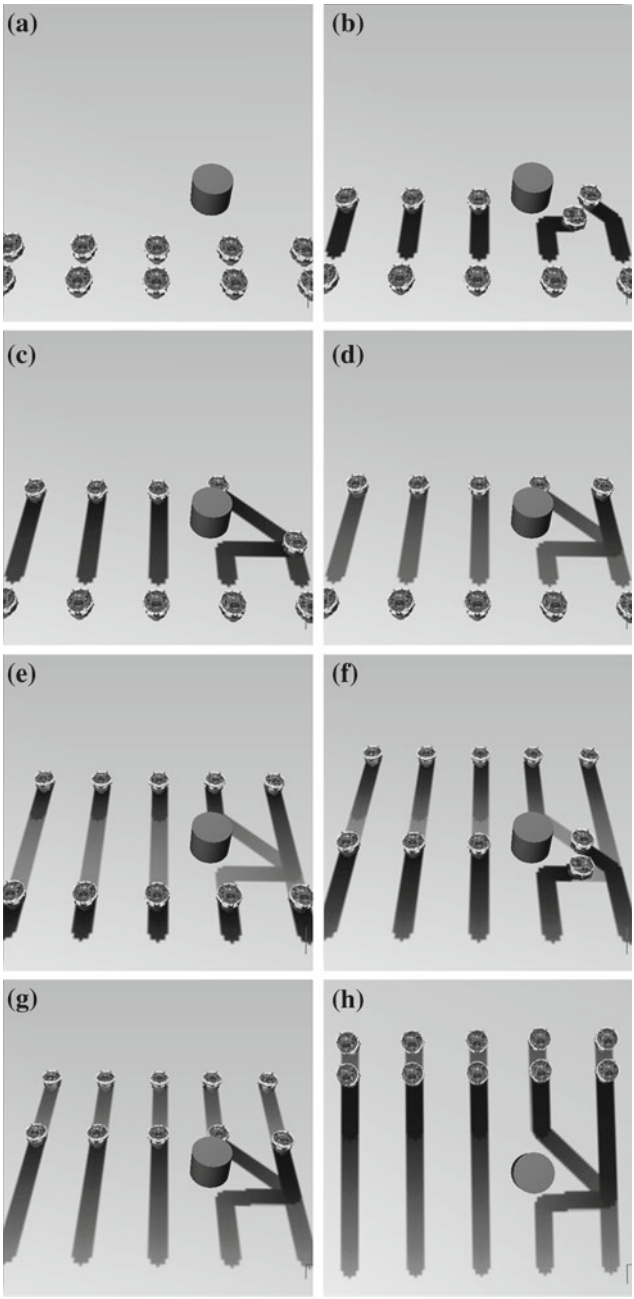

Fig. 9.16 Modified e-puck robot team avoiding a *circular shaped* object, (from *left* to *right* and from *upper* to *lower* images), where green spots are the robots and "pheromone" trails are illustrated with lines of tones of *black*

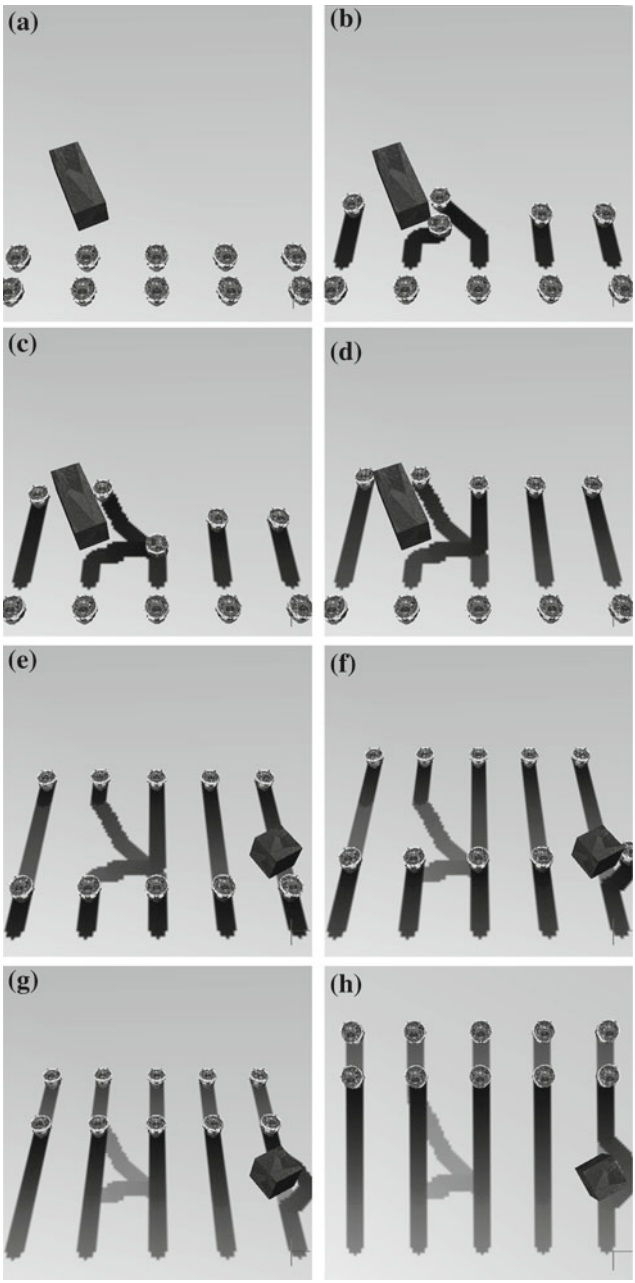

Fig. 9.17 Modified e-puck robot team avoiding dynamic moving objects, (from *left* to *right* and from *upper* to *lower* images), where *green spots* are the robots and "pheromone" trails are illustrated with lines of tones of *black*

9.5 Conclusions

In this chapter, a method for solving the path planning problem in a cooperative robot team was presented. The method is the result of a combination between CA and ACO algorithms resulting to the so called "Cellular Robotic Ants". To test the effectiveness of the method, a 2-D environment was created. The complexity of the proposed method is proportional to the CA dimensions, according to CA algorithms, and is considered to be low compared to other relative methods. Moreover, the method was implemented in a cooperative robot team using a three dimensional simulator, called Webots. For testing purposes, the under study robot team was constituted of two subgroups with five robots each. All essential sensors, that all robots must be equipped, and their direct relations with the cell length and pheromone were introduced. By some modifications on the real e-puck robot, the total amount of the required memory was reduced, thus causing the computational cost to be decreased. Therefore, the method is applicable to a real system consisting of e-puck robots but with some restrictions, as aforementioned. The accuracy of the method was tested by using two different types of objects, rectangular and circular shaped. In both cases, the method created successfully collision free paths. Presented results exhibit the effectiveness and the robustness of the method. Finally, the proposed "Cellular Robotic Ants" architecture covers the needs of self organization and autonomy of the system since no central control interferes.

References

1. Akst, J.: Send in the bots. Scientist **27**(10), 45 (2013) (Cited By since (1996))
2. Arney, T.: Dynamic path planning and execution using b-splines. In: Third International Conference on Information and Automation for Sustainability, ICIAFS 2007. pp. 1–6 (2007)
3. Barnes, L., Garcia, R., Fields, M.A., Valavanis, K.: Swarm formation control utilizing ground and aerial unmanned systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2008, pp. 4205–4205 (2008)
4. Beckers, R., Deneubourg, J.L., Goss, S.: Trails and u-turns in the selection of a path by the ant lasius niger. J. Theor. Biol. **159**, 397–415 (1992)
5. Belkhous, S., Azzouz, A., Saad, M., Nerguizian, C., Nerguizian, V.: A novel approach for mobile robot navigation with dynamic obstacles avoidance. J. Intell. Robotics Syst. **44**(3), 187–201 (2005)
6. Blum, C.: Ant colony optimization: Introduction and recent trends. Physics of Life Reviews **2**(4), 353–373 (2005)
7. Bonabeau, E., Dorigo, M., Theraulaz, G.: Inspiration for optimization from social insect behaviour. Nature **406**, 39–42 (2000)
8. Bonani, M., Raemy, X., Pugh, J., Mondana, F., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., Martinoli, A.: The e-puck, a Robot Designed for Education in Engineering. In: Proceedings of the 9th Conference on Autnomous Robot Systems and Competitions, vol. 1, pp. 59–65 (2009)
9. Brown, M., Lowe, D.: Automatic panoramic image stitching using invariant features. International Journal of Computer Vision **74**(1), 59–73 (2007)
10. Burgard, W., Moors, M., Stachniss, C., Schneider, F.: Coordinated multi-robot exploration. IEEE Trans. Robot. **21**(3), 376–386 (2005)

11. Charalampous, K., Amanatiadis, A., Gasteratos, A.: Efficient robot path planning in the presence of dynamically expanding obstacles. In: Sirakoulis, G., Bandini, S. (eds.) Cellular Automata. Lecture Notes in Computer Science, vol. 7495, pp. 330–339. Springer, Berlin Heidelberg (2012)
12. Charalampous, K., Kostavelis, I., Amanatiadis, A., Gasteratos, A.: Real-time robot path planning for dynamic obstacle avoidance. J. Cell. Automata Appear (2014)
13. Chen, M.J., Huang, C.H., Lee, W.L.: A fast edge-oriented algorithm for image interpolation. Image Vis. Comput. **23**(9), 791–798 (2005)
14. Chicco, G., Ionel, O.M., Porumb, R.: Electrical load pattern grouping based on centroid model with ant colony clustering. IEEE Trans. Power Syst. **28**(2), 1706–1715 (2013)
15. Conti, C., Roisenberg, M., Neto, G., Porsani, M.: Fast seismic inversion methods using ant colony optimization algorithm. IEEE Geosci. Remote Sens. Lett. **10**(5), 1119–1123 (2013)
16. Daniel, K., Nash, A., Koenig, S., Felner, A.: Theta*: any-angle path planning on grids. J. Artif. Intell. Res. **39**, 533–579 (2010)
17. Dechter, R., Pearl, J.: Generalized best-first search strategies and the optimality of a*. J. ACM **32**(3), 505–536 (1985)
18. Defoort, M., Floquet, T., Kokosy, A., Perruquetti, W.: Sliding-mode formation control for cooperative autonomous mobile robots. IEEE Trans. Ind. Electron. **55**(11), 3944–3953 (2008)
19. Deneubourg, J., Goss, S.: Collective patterns and decision-making. Ethol. Ecol. Evol. **1**(4), 295–311 (1989)
20. Dhiman, N.K., Deodhare, D., Khemani, D.: A review of path planning and mapping technologies for autonomous mobile robot systems. In: Proceedings of the 5th ACM COMPUTE Conference: Intelligent and Scalable System Technologies, COMPUTE '12, pp. 3:1–3:8. ACM, New York, NY, USA (2012)
21. Di Caro, G., Dorigo, M.: Antnet: Distributed stigmergetic control for communications networks. J. Artif. Int. Res. **9**(1), 317–365 (1998)
22. Dorigo, M.: Optimization, learning and natural algorithms. Ph.D. thesis, Politecnico di Milano, Italy (1992)
23. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. **1**(1), 53–66 (1997)
24. Du, Z., Qu, D., Xu, F., Xu, D.: A hybrid approach for mobile robot path planning in dynamic environments. In: IEEE International Conference on Robotics and Biomimetics, ROBIO 2007, pp. 1058–1063 (2007)
25. Fax, J., Murray, R.: Information flow and cooperative control of vehicle formations. IEEE Trans. Autom. Control **49**(9), 1465–1476 (2004)
26. Fredslund, J., Mataric, M.: A general algorithm for robot formations using local sensing and minimal communication. IEEE Trans. Robot. Autom. **18**(5), 837–846 (2002)
27. Garnier, S., Combe, M., Jost, C., Theraulaz, G.: Do Ants Need to estimate the geometrical properties of trail bifurcations to find an efficient route? A swarm robotics test Bed. PLoS Comput. Biol. **9**(3), e1002,903+ (2013)
28. Garnier, S., Gurcheau, A., Combe, M., Fourcassi, V., Theraulaz, G.: Path selection and foraging efficiency in argentine ant transport networks. Behav. Ecol. Sociobiol. **63**(8), 1167–1179 (2009)
29. Garnier, S., Tache, F., Combe, M., Grimal, A., Theraulaz, G.: Alice in pheromone land: An experimental setup for the study of ant-like robots. In: IEEE Swarm Intelligence Symposium, SIS 2007, pp. 37–44 (2007)
30. Ge, S.S., Fua, C.H.: Queues and artificial potential trenches for multirobot formations. IEEE Trans. Robot. **21**(4), 646–656 (2005)
31. Georgoudas, I., Sirakoulis, G., Scordilis, E., Andreadis, I.: A cellular automaton simulation tool for modelling seismicity in the region of xanthi. Environ. Model. Softw. **22**(10), 1455–1464 (2007)
32. Goss, S., Beckers, R., Deneubourg, J., Aron, S., Pasteels, J.: How trail laying and trail following can solve foraging problems for ant colonies. In: Hughes, R. (ed.) Behav.ural Mechanisms of Food Selection, NATO ASI Series, vol. 20, pp. 661–678. Springer, Berlin (1990)

33. Grassé, P.P.: La reconstruction du nid et les coordinations interindividuelles chezBellicositermes natalensis etCubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. Insectes Soc. **6**(1), 41–80 (1959)
34. Herianto Kurabayashi, D.: Realization of an artificial pheromone system in random data carriers using rfid tags for autonomous navigation. In: IEEE International Conference on Robotics and Automation, ICRA '09, pp. 2288–2293 (2009)
35. Herianto Sakakibara: T., Kurabayashi, D.: Artificial pheromone system using RFID for navigation of autonomous robots. J. Bionic Eng. **4**(4), 245–253 (2007)
36. Huang, W.H., Fajen, B.R., Fink, J.R., Warren, W.H.: Visual navigation and obstacle avoidance using a steering potential function. Robot. Auton. Syst. **54**, 288–299 (2006)
37. Ioannidis, K., Sirakoulis, G.C., Andreadis, I.: Cellular automata-based architecture for cooperative miniature robots. J Cell. Automata **8**(1–2), 91–111 (2013)
38. Jain, A.K.: Fundamentals of Digital Image Processing. Prentice-Hall Inc, Upper Saddle River (1989)
39. Keys, R.: Cubic convolution interpolation for digital image processing. IEEE Trans. Acoust. Speech Signal Process. **29**(6), 1153–1160 (1981)
40. Konstantinidis K., Andreadis I., Sirakoulis G.C.: Chapter 3—application of artificial intelligence methods to content-based image retrieval. In: P.W. Hawkes (ed.) Advances in Imaging and Electron Physics, Advances in Imaging and Electron Physics, vol. 169, pp. 99–145. Elsevier, Amsterdam (2011)
41. Konstantinidis, K., Sirakoulis, G., Andreadis, I.: Design and implementation of a fuzzy-modified ant colony hardware structure for image retrieval. IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. **39**(5), 520–533 (2009)
42. Latombe, J.C.: Robot Motion Plann. Kluwer Academic Publishers, Norwell (1991)
43. Lee, T.L., Wu, C.J.: Fuzzy motion planning of mobile robots in unknown environments. J. Intell. Robotics Syst. **37**(2), 177–191 (2003)
44. Li, X., Orchard, M.: New edge directed interpolation. In: Proceedings of the International Conference on Image Processing, vol. 2, pp. 311–314 (2000)
45. Lin, C.T., Fan, K.W., Pu, H.C., Lu, S.M., Liang, S.F.: An hvs-directed neural-network-based image resolution enhancement scheme for image resizing. IEEE Trans. Fuzzy Syst. **15**(4), 605–615 (2007)
46. Liu, J., Wu, J.: Multi-Agent Robotic Systems. CRC Press, Boca Raton (2001)
47. Marchese, F.: Multiple mobile robots path-planning with MCA. In: International Conference on Autonomic and Autonomous Systems, ICAS '06, pp. 56–56 (2006)
48. Marchese, F.M.: A directional diffusion algorithm on cellular automata for robot path-planning. Future Gener. Comput. Syst. 18(7), 983–994 (2002). Selected papers from CA2000 (6th International Workshop on Cellular Automata of IFIP working group 1.5, Osaka, Japan, 21–22 Aug 2000) and ACRI2000 (4th International Conference on Cellular Automata in Research and Industry, Karlsruhe, Germany, 4–6 Oct
49. Martens, D., De Backer, M., Haesen, R., Vanthienen, J., Snoeck, M., Baesens, B.: Classification with ant colony optimization. IEEE Trans. Evol. Comput. **11**(5), 651–665 (2007)
50. Mastellone, S., Stipanovic, D, Spong, M.: Remote formation control and collision avoidance for multi-agent nonholonomic systems. In: IEEE International Conference on Robotics and Automation, pp. 1062–1067 (2007)
51. Michel, O.: Webots: Professional mobile robot simulation. J. Adv. Robot. Syst. **1**(1), 39–42 (2004)
52. Muresan, D., Parks, T.: Adaptively quadratic (aqua) image interpolation. IEEE Trans. Image Process. **13**(5), 690–698 (2004)
53. Murphy, R.: Human-robot interaction in rescue robotics. IEEE Trans. Syst. Man Cyber. Part C Appl. Rev. **34**(2), 138–153 (2004)
54. Omohundro, S.: Modelling cellular automata with partial differential equations. Physica D: Nonlinear Phenomena **10**(1–2), 128–134 (1984)
55. Patnaik, S., Karibasappa, K.: Motion planning of an intelligent robot using ga motivated temporal associative memory. Appl. Artif. Intell. **19**(5), 515–534 (2005)

56. Progias, P., Sirakoulis, G.C.: An fpga processor for modelling wildfire spreading. Math. Comput. Modell. **57**(5-6), 1436–1452 (2013)
57. Recio, G., Martin, E., Estebanez, C., Saez, Y.: Antbot: Ant colonies for video games. IEEE Trans. Comput. Intell. AI Game. **4**(4), 295–308 (2012)
58. Russell, R.A.: Heat trails as short-lived navigational markers for mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 4, pp. 3534–3539 (1997)
59. Shen, H., Zhang, L., Huang, B., Li, P.: A map approach for joint motion estimation, segmentation, and super resolution. IEEE Trans. Image Process. **16**(2), 479–490 (2007)
60. Stentz, A.: The focussed d* algorithm for real-time replanning. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI'95, vol. 2, pp. 1652–1659. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1995)
61. Tan, K.C., Tan, K., Lee, T., Zhao, S., Chen, Y.J.: Autonomous robot navigation based on fuzzy sensor fusion and reinforcement learning. In: Proceedings of the IEEE International Symposium on Intelligent Control, pp. 182–187 (2002)
62. Toffoli, T.: Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. Physica D: Nonlinear Phenomena **10**(1–2), 117–127 (1984)
63. Tzionas, P., Thanailakis, A., Tsalides, P.: Collision-free path planning for a diamond-shaped robot using two-dimensional cellular automata. IEEE Trans. Robot. Autom. **13**(2), 237–250 (1997)
64. Ulam, S.: Random processes and transformations. Int. Congr. Math. **2**, 264–275 (1952)
65. Von Neumann, J., Burks, A. et al.: Theory of Self-Reproducing Automata. University of Illinois Press, Urbana (1966)
66. Wang, C., Soh, Y., Wang, H., Wang, H.: A hierarchical genetic algorithm for path planning in a static environment with obstacles. In: IEEE CCECE2002 Canadian Conference on Electrical and Computer Engineering, vol. 3, pp. 1652–1657 (2002)
67. Willms, A., Yang, S.X.: An efficient dynamic system for real-time robot-path planning. IEEE Trans. Syst. Man Cybern. Part B Cybern. **36**(4), 755–766 (2006)
68. Willms, A.R., Yang, S.X.: An efficient dynamic system for real-time robot-path planning. IEEE Trans. Syst. Man Cybern. Part B **36**(4), 755–766 (2006)
69. Yang, S.X., Luo, C.: A neural network approach to complete coverage path planning. IEEE Trans. Syst. Man Cybern. Part B Cybern. **34**(1), 718–724 (2004)
70. Zheng, T., Zhao, X.: Research on optimized multiple robots path planning and task allocation approach. In: IEEE International Conference on Robotics and Biomimetics, ROBIO '06, pp. 1408–1413 (2006)
71. Zhong, Y., Shirinzadeh, B., Tian, Y.: A new neural network for robot path planning. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 1361–1366 (2008)