Available online at www.sciencedirect.com
ScienceDirect
Information Sciences 177 (2007) 755–770
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Strategy creation, decomposition and distribution in particle navigation ☆

Ulaş Beldek [a,c,*], Kemal Leblebicioğlu [b,c]

[a] Electronic and Communication Engineering Department, Çankaya University, Yüzüncüyıl, Balgat, 06530 Ankara, Turkey
[b] Electrical and Electronic Engineering Department, METU, Ankara, Turkey
[c] Computer Vision and Intelligent Systems Research Lab., METU, Ankara, Turkey

Received 28 October 2005; received in revised form 28 June 2006; accepted 3 July 2006

Abstract

Strategy planning is crucial to control a group to achieve a number of tasks in a closed area full of obstacles. In this study, genetic programming has been used to evolve rule-based hierarchical structures to move the particles in a grid region to accomplish navigation tasks. Communications operations such as receiving and sending commands between particles are also provided to develop improved strategies. In order to produce more capable strategies, a task decomposition procedure is proposed. In addition, a conflict module is constructed to handle the challenging situations and conflicts such as blockage of a particle's pathway to destination by other particles.

Keywords: Genetic programming; Rule–base; Strategy planning; Genetic algorithms; Robot navigation; Maze solving; Optimization; Multi-agent systems

1. Introduction

Constructing a hierarchical and systematic approach for solving a problem or a group of problems needs strategic planning. Genetic programming (GP) [10] is one of the few existing methods that can be directly used for strategic planning. It is possible to implement strategy planning and task distribution jobs in various ways using GP. For instance, the use of evolving structures, which support and interact with each other, is the main concept discussed in [6]. In [7], different interaction types between agents are evolved by GP. Mathematical rule-based structures capable of producing a group of paths and strategy planning jobs are also discussed in [7].

Two approaches can be proposed for the control of the particles: homogeneous and heterogeneous breeding [7]. Means of interaction (communication) between the particles can also be provided by some special

☆ This work is supported by Middle East Technical University, Scientific Research Project, BAP-2002-07-04-05, "Robot Navigation by Genetic Programming".
* Corresponding author. Address: Electronic and Communication Engineering Department, Çankaya University, Yüzüncüyıl, Balgat, 06530 Ankara, Turkey. Tel.: +90 312 2844500x302; fax: +90 312 2848043.
E-mail addresses: u.beldek@cankaya.edu.tr (U. Beldek), kleb@metu.edu.tr (K. Leblebicioğlu).

0020-0255/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2006.07.008

communications operations [7]. The results in [7] show that the communication yields improved performance of the rule-based, tree-like hierarchical structures. Also, the guidance of each particle by different and independently interacting strategies (heterogeneous breeding) has an advantage over using a single evolved strategy for all the particles (homogeneous breeding).

We have presented a comparison of our study via some recent and important applications about the robot navigation problem [1–5,7–9,11–26]. Four topics are chosen as a basis of our performance analysis. These topics are learning (intelligence), behavior, task planning, and sensing mechanisms, which include the applicability and placement of the sensors and communication means of the robot.

In most robot applications, intelligent mechanisms are used [14] to equip the robot with artificial intelligence (AI). Three models concerning AI applications for robotics are discussed in [14], namely, the Hierarchical, Reactive and Hybrid Deliberative/Reactive paradigms. Multi-agent structures and path-planning concepts are further discussed in [14]. Generally, learning in robotics seems to have three different goals in the literature. In some studies, the topology of the medium is learned [22–24]. In several other applications, some parameters about the learning method are updated to adjust a number of navigation variables like related angles, speeds and directions. Lastly, learning mechanisms are used in order to obtain special structures (agents, task representations, behaviors, navigation plans) for performing important jobs in the applications [3,5,7,9,12,15,19]. We have used GP in order to construct some intelligent agents (hierarchical strategies), which carry on these navigation tasks. One of the important aspects of our approach is that the agents are produced as a result of a complete evolutionary process. They are completely rule-based and hierarchical structures, which react in response to signals they receive in their range of vision.

Our study is partially based on the work of Iba [7], with some differences in the rule–base structure. In our study, there are complete and state dependent if–then relations, whereas in [7], the operations and terminals are mathematical relations and vectors, which yield mathematical functions. Another important difference is that we propose a modular approach to produce an explicit task decomposition scheme so that some special agents can be developed and used in order to control the robots when they face specific problematic situations.

In [19], a genetic algorithm (GA) is used in order to train robots to perform some navigation tasks. In this application, three basic reactive behaviors (make a random movement, go to the goal and avoid the obstacles) are parameterized by an array containing five variables. Each of these parameter sets can be considered as strategies, which are called ecological niches. In this study, reactive behaviors are coded as constant length parameters and these parameters are used to generate actions in a scenario. More specifically, what we have achieved in our study is that some actions and basic reactive behaviors have been coded in terms of terminals and operations to create hierarchical rule–base structures at the end of the learning period where the GP is used.

Some important facets of behavior-based robots have been presented in [2]. For the robot navigation applications, behaviors are usually common, structurally-defined response strategies for special conditions in the navigation scenarios [1,3,8,12,18–23,25]. In our study, behaviors are the strategies that evolve by using the GP. We have employed communications operations to provide a certain kind of coordination between the robots. Communicated messages are also rule-based structures containing information about what should be done according to the environmental situation. Finally, a conflict module is developed using the same principles, as part of the behavioral-reactive strategy structure.

In most of the applications in the literature, an explicit or implicit task planning is performed [1,3,4,7–9,11,13,19,21–23,25,26]. In terms of task planning, our study could be considered somewhere in between the implicit and the explicit approaches. In pure homogeneous and heterogeneous breeding applications, there is implicit task planning accomplished via the use of different agents. The modular approach can be considered to correspond to the explicit task planning, where the global task is decomposed into problematic parts, and the agents' response to stimuli received are obtained separately.

Two articles [16,17] have a special place in task planning applications of multi-robot systems. They propose architectures about the coordination of the multi-robot teams for the evaluation of multi-task objectives. This architecture is called as L-ALLIANCE. The L-ALLIANCE architecture depends on the decomposition of the tasks for a robot team. Our conflict module is different from the L-ALLIANCE architecture since a single agent is produced for performing the necessary actions to save two robots in conflict.

For most of the robot applications, a sensing mechanism must be supplied in order to provide the robot with an awareness of its environment. We have proposed a limited sensing mechanism, which enables the

robot to detect only the cells surrounding it. The relative position of the robot and its goal are also sensed. Another sensing mechanism is provided through the use of communications operations. These operations equip some robots with control over the other robots.

In our first application, we have evolved logical rule-based structures by GP to solve the problem of constructing a strategy to find a path towards a destination for a particle in a grid area containing obstacles. The logical rule-based structures are generated from the operations of if–then statements and the terminals of movement styles. Means of communication between the particles are also provided. The type of communications used in the first application is based on the commands sent/received between the particles. It has been observed that although this communications style is somehow effective in training, it may produce poor test performance.

In the second application, the problem of conflict resolution is considered only for the case of two particles when they are in a narrow corridor. The main objective in this new approach is to decompose the complexity of the strategy construction problem into simpler cases and handle each case separately. So, a compact and wise way of solving the complete problem is to attack only a particular aspect of the problem at each phase. Later, the results and experiences obtained in this fashion are combined to create a complete solution. This approach necessitates the design of some training scenarios including two particles in a special conflict situation.

The paper is organized as follows: In Section 2, the problem description is given. Presentation of the GP is included in Section 3. Section 4 is devoted to the simulation studies. The conflict module is described and simulation studies associated with the GP-conflict module are presented in Section 5. Conclusions and possible future work are discussed in the final section.

2. Robot navigation problem

Our strategy development problem takes place in a 10×10 planar grid area as shown in Fig. 1. Each object (particles and obstacles) in the area can occupy a single grid point at a time. The particles are labeled as 1, 2, 3, and 4 whereas the black regions represent the obstacles. Sometimes the obstacles are replaced by gateways where passage for only a single particle is allowed. In our simulations, two or four particles were used.

The proposed scenarios were simulated in a maze (10×10 grid areas). In this maze, particles were assigned navigation tasks to carry out. The particles are equipped with the ability to sense and detect neighboring cells in the main directions (north, east, west, and south). It is also possible for the particles to sense the direction of their destinations and the existence of other particles so that they can react accordingly.

The vision range of a particle is designated as one unit (measured from its current location) in all of the main directions. So the robot (i.e., the particle) is able to detect only the neighboring cells from its current position. The particles do not have a map of the topologies associated with the scenarios.

The aim is to make the particles move towards their targets. Usually targets are chosen in such a way that the particles will encounter complex and conflicting situations while they are moving towards their destinations.

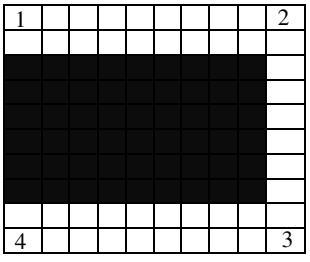

Fig. 1. A typical planar grid area.

3. Genetic programming

3.1. Terminals and operations

In order to apply GP, some suitable terminals and operations have to be chosen. We chose the terminals out of the movement types, which are applicable to the particles in the grid area, and the operations were chosen out of the if–then statements, which relate logical correspondences between the particles and their surroundings. The operations and terminals are combined to generate hierarchical tree structures as shown in Fig. 2. These trees are the basic structures that control and move the particles. The following symbols are used to represent the terminals and operations.

Terminals:

a: go one step in the direction of your orientation and keep your orientation the same.
b: go one step backward but do not change your orientation.
c: turn 90° clockwise and go one step.
d: turn 90° counterclockwise and go one step.
e: do not move and do not change your orientation.
f: make a random movement and change your orientation in the movement direction.
g: move in the direction of your destination.

Operations:

- + (Takes two arguments): It returns the first argument if there is an obstacle in front of the particle being controlled, otherwise it returns the second argument.
- **A** (Takes two arguments): It returns the first argument if there is another moving particle in front of the particle being controlled, otherwise it returns the second argument.
- **B** (Takes two arguments): It returns the first argument if the controlled particle is in a region where two main opposite sides of the particle are occupied by obstacles and other two opposite sides are free cells, otherwise it returns the second argument.
- **F** (Takes two arguments): It returns the first argument if the controlled particle and the destination of that particle lie in the same perpendicular or horizontal alignment, otherwise it returns the second argument.
- **K** (Takes two arguments): It returns the first argument if the cell in front of the controlled particle is obstacle-free, otherwise it returns its second argument.

Among the terminals, the one symbolized by 'g' moves the controlled particle in the direction of its destination but this does not mean that the particle will eventually reach its destination by continuous application of this movement since the obstacle positions may prevent the particle from reaching to its destination. This type of movement also enables the particle to move diagonally when positional differences between the particle and its destination gives both non-zero x and non-zero y components. The tree structure can be encoded as a rule-based strategy yielding only one kind of movement for a particle at every step. A particle is not allowed to go through obstacles even though the resulting action for the particle declares a movement towards an obstacle. In that case the action will not be realized and the particle will remain in its last position at that step.

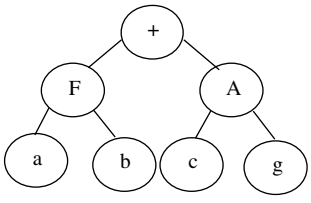

Fig. 2. A typical tree structure.

3.2. Coding and encoding of a tree structure

A typical tree structure is shown in Fig. 2. The strategy encoded in the figure can be explained as follows: First of all, it checks to see if there is an obstacle in front of the particle. If this is true, then the controller checks if the particle and its destination lay in the same horizontal or vertical alignment. If the second statement is true, the particle moves forward. If the statement is false, the particle moves backward but still facing the original direction. In case the first statement is false, the controller checks if the particle has encountered another particle in the forward direction. If this statement is true, it turns clockwise and goes one step. In the other case, the particle moves towards its destination. The movement of the particle depends on its location in the planar area, its position with respect to its destination and the other particles, and the distribution of the obstacles.

This tree structure can be represented as a string in MATLAB simulations. The representation depends on the distribution of the hierarchy of the operations, and the terminals from the top to the bottom and from the left to the right as in Lisp programming. In particular, the tree structure in Fig. 2 can be represented as the string '**+FabAcg**' in our simulations.

3.3. Simple communications operations

In addition to the primary 'if–then' operations, some other operations are also defined in order to provide interaction and communication between the particles. They are the '**send**' '**S**' and '**receive**' '**R**' operations. The operation '**S**' takes two arguments and performs two jobs. A 'send' command carries out its first argument as an input to the operations at the next higher hierarchy. Secondly, it sends the second argument to the nearest particle. If, in a tree structure, there is more than one 'send' command, the message, which has the highest priority (operated last) in evaluation of the tree structure, is sent. Similarly, operation '**R**' takes only one argument and does two jobs. When a tree has a 'receive' operation in its structure, it transfers the argument to the next higher hierarchy if the particle does not receive any message from the other particles. If the particle receives a message from some other particles, this time this operation causes the particle to evaluate the sent message.

3.4. Genetic programming application

Three different genetic operations were applied for all the simulations. These were crossover, reproduction and regeneration operations. An individual (i.e., a chromosome) is the tree structure that is responsible for the control of the particles. Depending on the breeding strategies, it consists of a single or a multiple sub-tree portions. In simulations, population size may change from 20 to 100 individuals. Eight percent of the new individuals in the proceeding generations are produced by the crossover operation, 10% are reproduced and 10% are regenerated. So the total number of individuals is kept constant at every generation. The elitist method is also applied. In all the simulations, roulette-wheel-selection is applied for crossover and reproduction operations.

In a scenario the fitness of an individual consists of different parts and is determined by considering three different criteria.

1. First of all, if there are N particles and only M of them have reached their destinations, then the first part of the fitness value to be assigned to such an individual is $(M) * C_1$.
2. If all of the particles reach their destinations, bonus fitness value is added in addition to the contribution coming from item 1. This bonus fitness value is proportional to the number of steps remaining (SL) from the reserved number of steps (RS) (i.e., bonus $=$ SL $* C_2$).
3. When an individual is not able to carry all the particles to their destinations, a suitable fitness value (which is related to the distance between the particles and their destinations) must also be assigned for that individual ($C_3 * \sum_{p\in AG}\{d(st(p), gl(p)) - d(cr(p), gl(p))\}$), where AG is the number of particles unable to reach their destinations.

The expression $d(x, y)$ in item 3 gives the Euclidean distance between the vectors x and y. In this expression, $st(p)$ gives the starting position of a particle, $gl(p)$ gives the destination of the particle, and $cr(p)$ gives the last

position of the particle. This expression rewards the movement towards the destination. According to these three different criteria, the fitness of an individual which have brought all the particles to their destinations in a scenario is calculated by the formula:

$$\text{Fitness} = N * C_1 + \text{SL} * C_2 \tag{1}$$

In the same way, the fitness of an individual that is not able to bring some of the particles to their destinations is calculated by the formula:

$$\text{Fitness} = M * C_1 + C_3 * \sum_{p \in AG} \{d(st(p), gl(p)) - d(cr(p), gl(p))\} \tag{2}$$

The parameters C_1, C_2, and C_3 are chosen heuristically as 3000, 50 and 80, respectively. On the other hand, even though the absolute values of these parameters are unimportant, the fitness value obtained in Eq. (1) must always be greater than the fitness value obtained in Eq. (2). In order to provide this inequality, C_1 must be greater than the fitness value obtained in item 3 when the number of particles that could not manage to accomplish their navigation task is 1 (meaning $\text{AG} = 1$). When $\text{AG} = 1$, the maximum probable fitness value obtained using item 3 will be equal to 1051. This value is obtained when a particle lying in one of the corners of the grid region as the starting position reaches the closest cell to its destination, which is situated at the opposite corner at the end of the simulation. This guarantees that an individual moving all the particles to their destinations has always a better fitness value than an individual, which manages to carry all the particles to their destinations except for a single particle. In this way, success of the genetic search is guaranteed since more successful individuals are assigned a better fitness value than the less successful ones. The bonus term declared in item 2 also has significance; the fitness value of a successful individual is exaggerated by its speed while it completes all the navigation tasks. By using this term, the chances of the fast and successful individuals to be chosen for genetic operations (crossover and reproduction) are enhanced.

If there is more than one scenario in a simulation, the standard fitness of an individual is determined by summing the fitness of the individual over all of the scenarios and dividing the cumulative by the total number of scenarios. The standard fitness shows the ability of the individual to solve different kinds of scenarios.

4. Simulation studies

4.1. Effect of communication

Heterogeneous breeding is used to control the robots and two implementations are performed. In the first implementation, communications operations ‘send’ and ‘receive’ are not inserted into the operations set while in the second implementation these operations are used. A population of 100 individuals is generated initially. In every implementation, four planar areas having four particles, as in Fig. 1, are used. The starting positions and the destinations for the particles are chosen to be the same in every planar area. The only difference is in the obstacle distribution. Fig. 3 shows a training scenario. In all of the scenarios, particles 1 and 3, and particles 2 and 4 have to exchange their places. The resultant strategies are tested in 12 different scenarios having different wall distributions and alignments. A test scenario is shown in Fig. 4. In all the scenarios, fitness evaluation has been done after the particles are allowed to move 50 steps. The evolved individual can be simplified

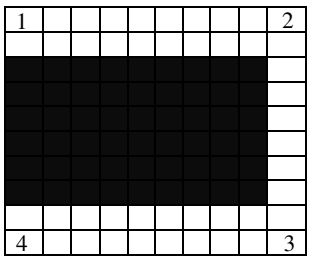

Fig. 3. Scenario 1.

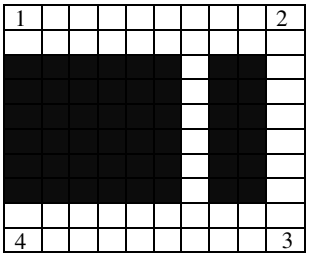

Fig. 4. Test scenario 1.

to produce a simpler tree structure by using the relations between if–then statements. Therefore, the unused branches and portions generated during the search process can be eliminated. The new individual structure is supposed to perform the same actions as the best individual. Accordingly, the simplified structure of the best heterogeneous individual without communications operations is as follows:

- The tree structure controlling the first particle: '+fAgKaf'.
- The tree structure controlling the second particle: 'Ab+ga'.
- The tree structure controlling the third particle: 'Kag'.
- The tree structure controlling the fourth particle: 'Aa+Fgba'.

This individual is able to bring three particles to their destinations in the first and the second training scenarios and it also brings two particles to their destinations in the third and the fourth scenarios. Among 12 test scenarios, the individual is able to accomplish the navigation task totally in only 2 scenarios. In four scenarios, three of the particles are navigated successfully to their destinations. In two scenarios, only two particles reach their goals and in the remaining 4 test scenarios none of the particles reach their destinations. The success rate in these training scenarios was 62.5% where 10 of the total 16 transitions were attained. In the same way, the success rate in test scenarios is 50% where 24 of 48 transitions are successful.

The simplified structure of the best heterogeneous individual with communications operations is as follows:

- The tree structure controlling the first particle: '+gSfa'.
- The tree structure controlling the second particle: 'KgSgRf'.
- The tree structure controlling the third particle: 'KaSKbfg'.
- The tree structure controlling the fourth particle: 'SgRg'.

This individual is able to bring all four particles to their destinations in the first and the second training scenarios and it also brings 3 particles to their destinations in the third and the fourth scenarios. Among 12 test scenarios, the individual is able to accomplish the navigation tasks totally in only 7 scenarios. In two test scenarios, three of the particles are navigated successfully to their destinations. In the remaining three scenarios only one particle reaches its goal in each case. Comparing the transitions made, we can say that the success rate in these training scenarios was 87.5% where 14 of the 16 transitions are managed. For the test scenarios the success rate is somewhat smaller but still satisfactory; 37 transitions were completed out of a total of 48.

The comparison of the results shows that providing the means of communications plays an important role in successfully navigating the particles to their destinations. The communications provided a success rate over 85% for training scenarios. The success in the test scenarios was smaller (over 75%), but still showing the impact of communications operations.

It has been observed that providing communications between agents produces better results as expected [6]. It is emphasized that GP can be used to provide communications between agents (the individuals) [7]. Another important issue is the ability of the terminals and operations to overcome the difficulties encountered. Since the terminals and operations allow the particles to sense only neighboring cells, other particles and the direction of their destinations, the capacity of the strategies to control the particles is limited. One can suggest new

and complicated operations by increasing the vision range of the particles. An approach to create more sophisticated strategies to increase the navigation capacity of the particles is suggested in Section 5.

4.2. Comparison of homogeneous and heterogeneous strategies

In this part, four training scenarios were added to the set of training scenarios used in Section 4.1. In the new scenario set, the starting positions of particles 1 and 2 and starting positions of particles 3 and 4 were changed and the new destinations were determined so that the particles 1 and 4 will exchange their places and particles 2 and 3 will exchange their places. The influence of homogenous and heterogeneous breeding on the success rate was investigated. Second, the effect of changing the initial positions of the particles was tracked and monitored. For each particle, 50 steps of movement were allowed. This part included three implementations. The number of individuals in the population was limited to 20 in all the implementations. The first implementation was carried out using heterogeneous breeding with communications operations, second implementation was carried out using homogenous breeding with communications operations and the last implementation was done using homogenous breeding solely without communication. The results are as follows:

- The simplified structure of the best homogeneous individual without communication is 'Kaf'.
- The simplified structure of the best homogeneous individual with communication is 'FRgSKfgg'.

The simplified structure of the best heterogeneous individual with communication:

- The minimal structure controlling the first particle: '+Sfga'.
- The minimal structure controlling the second particle: 'ARgSgf'.
- The minimal structure controlling the third particle: 'SKbfKag'.
- The minimal structure controlling the fourth particle: 'SRgg'.

When these three implementations are compared, it can be observed that the heterogeneous breeding with communication produces the best results. Among the eight scenarios, four of them were fulfilled completely, in two scenarios; three particles were carried to their destinations successfully and in the remaining two scenarios two particles managed to reach their destinations. Since there are 8 scenarios and in a scenario four transitions have to be made, a 100% success rate necessitates 32 successful transitions. In our simulation, the number of particles carried to their destination successfully was 26. This means a success rate of 81.25%. The next powerful implementation is the one with homogeneous breeding using communications operations. This implementation showed complete success in that two scenarios managed to transfer all their particles to their destinations. In the remaining four scenarios, two particles overcame the tasks per scenario. For the last two scenarios, only one particle is moved to the final location. This produced a success rate of 56.25%. The worst implementation is seen with homogeneous breeding without any communications ability. In this implementation, in only two scenarios full success is observed. For one scenario, three particles managed to attain their goals. In another scenario, the number of particles, which reached their destinations, was two. In the remaining four scenarios, only one particle reached its destination; a success rate of 40.62%.

The implementations showed that the best method is heterogeneous breeding with communication. This is as expected since all the particles are controlled by a different tree structure and this increases the adaptability of the particles to various environments. However, a strategy must be robust and grid independent, but the heterogeneous breeding lacks this property. It is case sensitive and the trees obtained are only applicable to a single predetermined particle (not for all of the particles). This is the first drawback of heterogeneous breeding. Secondly, the success rates of the other approaches may be improved by introducing new ideas. Although the homogeneous breeding idea is much closer to strategy planning, the observed results were not as good as for the heterogeneous breeding. This is partially because of the undesired effects of communications operation. If the best homogeneous strategy with communication is inspected, it will be seen that it sends and receives messages at every time. This will mean that all the particles will be in the control of other particles (since a particle is always closest to another one). Therefore, sending and receiving messages simultaneously between every particle is somewhat illogical and conflicting. So, what must be done to improve the responses of homo-

geneous strategies is to decompose the problem into simpler parts, determine a new way of communication and investigate the strategy evolution in this new framework. So, modular structures have to be constructed. Interestingly, modular strategies can be employed in the heterogeneous breeding case as well. In the next section, the idea of modular strategies will be elaborated over a standard conflict situation.

5. Modular approach: conflict module

Consider Figs. 5 and 6. Fig. 5 shows a scenario containing four particles. In this scenario, particles 1 and 3, and particles 2 and 4 have to exchange their places. An obvious and expected result is shown in Fig. 6. What is observed according to the results is that movements of particles 1 and 4 were prevented by the wall structures. More interestingly, particles 2 and 3 have blocked the other's path to its destination. A conflict module had better be designed in order to solve this and similar problems.

The theme explained in Figs. 5 and 6 brings a problem, which can be observed easily. No matter how skillful the strategies obtained are, they may be inefficient in some scenarios such as passing a narrow corridor where only the passage of one at a time is possible (because the corridor may be blocked by the particles from opposite sides). At most of the scenarios the strategies obtained are successful but some scenarios may need further development.

Although the communications employed between the particles become very efficient for most of the scenarios to overcome the obstructions, the procedure for communications operation is not perfect since communications actually is needed when some difficulties appear.

Also heterogeneous strategies are sometimes inconsistent and inapplicable although they give better results. This is because of the fact that the multiple-strategies do not have any significance when the position and orientation of the particles or the particles themselves are changed. A strategy designed for a specific particle usually proves to be useless when it is used by some other particle in heterogeneous case.

Adding up these reasons, a new way of interaction has to be substituted for the old communications operations. This new interaction type must be a single strategy and it must be able to deal with the blockage problem. We have proposed a new interaction called the conflict module. The conflict module is a single strategy concerning the conflicts between the particles in a narrow region where only one particle can pass at a time.

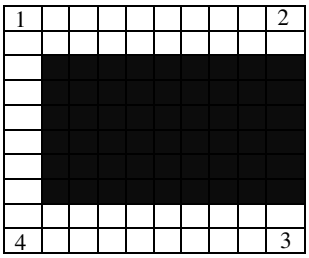

Fig. 5. A scenario.

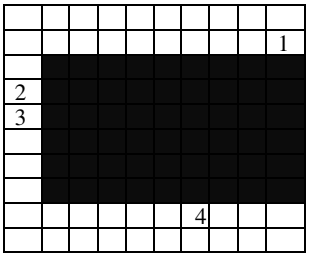

Fig. 6. A possible final instance of the scenario in Fig. 5.

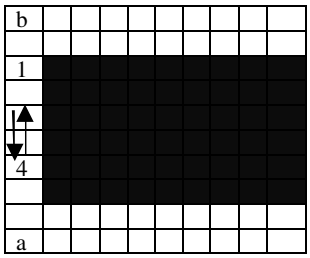

Fig. 7. Two particles in conflict.

A definition about conflict can be made as well: two particles are in conflict whenever they encounter in a narrow region and approach from opposite sides. Fig. 7 shows two particles in conflict.

As seen from Fig. 7, these particles will have difficulty when they are crossing the narrow passage while they are going in the opposite directions. When two particles come to a position requiring the use of the conflict module, the strategies controlling the particles transfer the control to the conflict module and these two particles are moved with the help of the conflict module until the conflict is resolved. Denoting the coordinates and directions of the particles in conflict by (x_1, y_1), (x_2, y_2), and d_1 and d_2, respectively, the conditions below should hold for the control to switch to the conflict module

$$(x_1 = x_2,\ y_1 \neq y_2,\ d_1 = (-1)d_2)$$

or

$$(x_1 \neq x_2,\ y_1 = y_2,\ d_1 = (-1)d_2)$$

In order for the particles return to their usual strategies, they have to solve the conflict with each other. New goals are determined for these particles. These new goals are chosen so that while they are trying to reach these new locations, no conflict exists between them. For example, in Fig. 7 the location labeled as 'a' can be determined to be the new goal of particle 1 whereas the location labeled as 'b' can be assigned as the new goal of the particle 4.

5.1. Conflict module simulations

The conflict module is planned as a strategy obtained from the operations and terminals of GP used in the first part, excluding the operations of communication. Sixteen different scenarios where two particles are in conflict are designed to train a single conflict agent. In all of these scenarios, 'a' is denoted as the goal of particle one and 'b' is denoted as the goal of particle two. Two of the scenarios are shown in Figs. 8 and 9.

The fitness assignment procedure resembles the one used in the first part of the simulations. One hundred individuals are inserted initially into the population to start the GP search. The best strategy obtained after 30 generations is shown in Fig. 10. This strategy was able to resolve the conflicts in 12 of the 16 scenarios. In two of the scenarios, the strategy was able to take both of the particles close to their goals and in the remaining two scenarios they were not able to give satisfactory results. One of the scenarios where an unsatisfactory result

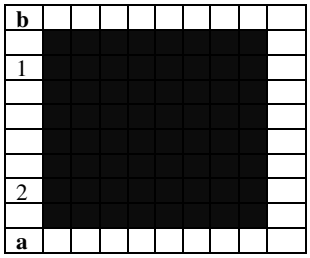

Fig. 8. Scenario 1.

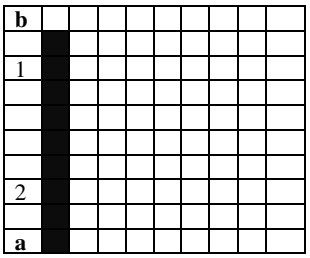

Fig. 9. Scenario 2.

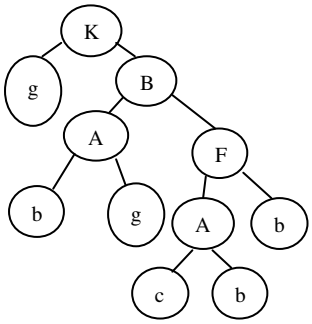

Fig. 10. The best conflict solving strategy evolved after 30 generations.

was obtained is shown in Fig. 8. As this scenario is investigated, one can easily see that the particles have to do a very difficult harmonious movement in order to reach their goals. Although the strategy obtained by GP does not show its power on this scenario it performs near full-convergence for the other scenarios.

The conflict module is only applicable when there is a conflict between particles. It cannot be used alone as a strategy for navigation. It is just an assistant to the overall navigator.

5.2. Strategies with the conflict module

Eight scenarios were initiated in order to obtain a new strategy (standard navigation agent). The scenarios were the ones used in Section 4.2. Same kind of fitness evaluation procedure as in the previous simulations was used and 100 individuals were chosen randomly as the initial population. The particles were also given a chance to make a random movement when they are unable to move for three time steps. In this way, the stable orientations of particles during motionless periods were changed. Whenever any two particles are in conflict, the evolving strategies leave the control of the conflicting particles to the conflict module created in Section 5.1. The evolving strategies assume the control again whenever the conflict is removed. The best control strategy evolved at the end of 20 generations is shown in Fig. 11 (standard navigation agent).

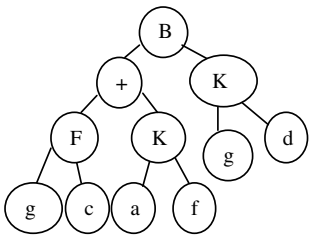

Fig. 11. The best strategy used in standard navigation situations.

This strategy, with the help of the conflict module was able to overcome the conflicting situations completely in two scenarios. In the other six scenarios, two of the particles were able to do their tasks perfectly (out of 32 navigation tasks, 20 of them were accomplished). This does not seem to be completely satisfactory. However, when the final positions of the particles are investigated, no conflict between the particles was observed.

5.3. Using more complicated scenarios in the development of a new module

A shortcoming of the present approach is evident by investigating the initial and final instants of the first scenario shown in Figs. 12 and 13, respectively. In this scenario, particles 2 and 3, and particles 1 and 4 are supposed to exchange their places. In the final instant, it is seen that particle 1 and particle 4 were unable to go beyond the solid wall structure on their way to their goals. This means that the standard navigation agent is unable to solve the problem (even with the help of the conflict module) when there are solid walls blocking and trapping the particles. A new module must be designed to give a chance for the particles to leave the locations where they are captured for a long time. This requires the particles to be equipped with some kind of "memory" which is beyond the present study.

Another way to deal with the problems of the present approach is to try to develop more powerful "agents" that can cope with more complicated scenarios. For this purpose, 8 new scenarios each, including 2 particles, were introduced in addition to the scenarios used in Section 5.1 (total number of scenarios used to evolve a new agent is increased to 24). These new 8 scenarios are chosen similar to the ones where the original conflict agent was not able to accomplish the assigned tasks completely (similar to the scenario in Fig. 8). Two of these scenarios are presented in Figs. 14 and 15, respectively. The remaining 6 scenarios are similar, but the starting positions of the particles are chosen to be different corners and the wall configurations are at different angles.

The standard fitness of a single chromosome is calculated by taking the average of fitnesses for that chromosome for each scenario. However, the weights for new scenarios are taken to be 3 times greater than the weights of the original scenarios used in Section 5.1; it is desired to develop a powerful agent to be generally successful in these new scenarios.

The factors that effect the training time are the size of the population, the number of generations that the simulation is supposed to continue, the number of steps reserved for the particles to move in a scenario and the number of scenarios used to carry out the simulations. The GP parameters for this problem were chosen as follows. An initial population of 300 chromosomes was generated randomly and the maximum tree length

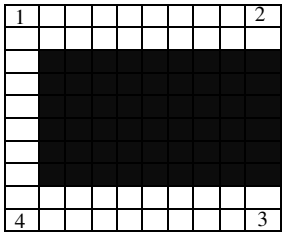

Fig. 12. Scenario 1.

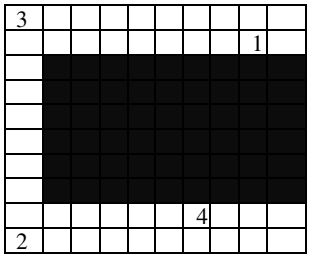

Fig. 13. The final instance for the scenario in Fig. 12.

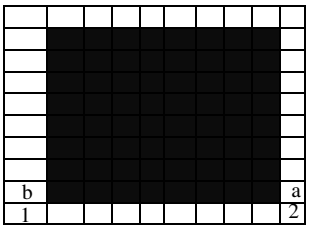

Fig. 14. A new scenario used in development of a more powerful agent (a shows the goal of first particle b shows the goal of second particle).

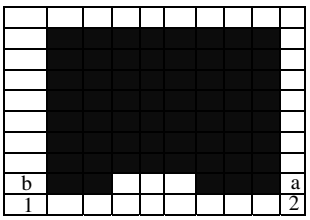

Fig. 15. A new scenario used in development of a more powerful agent (a shows the goal of first particle b shows the goal of second particle).

in the initial population was set as 6. The crossover and the reproduction rates are 90% and 10%, respectively. Two percent of the chromosomes in each generation were mutated. The probability of crossover point to be an operation was assigned to be 80%. The same operation and terminal sets used in the original GP application without communications operations were used to train the agent. The search was run for 66 generations, until no further improvement was observed. Each tree-structure allows 50 steps for the particles to move. The simulations are carried out in the MATLAB environment by a 1.6 GHz computer with a 512 kilobyte RAM capacity. The iterations for the first population take 21 min. As the simulation continues, since the tree structures become much more complicated, the iterations begin to take longer. The whole simulation (66 generations) takes two days and 2 h. The best agent obtained at the end of the simulations is shown below:

$$\text{`}{+}{+}gKfdABAKA{+}{+}gBfdaABAKABfd{+}{+}gKfdabKa{+}cdf{+}fabgf{+}AaaABAKdbKa{+}cdf{+}AKABfd$$
$${+}{+}gKfdaBAKABfd{+}{+}gKfdabKa{+}cdfKa{+}cda\text{'}$$

This agent has a standard fitness of 6806.9. Since the agent has the random movement "*f*" command in its structure, this standard fitness may change from run to run but it is observed that the variations in the fitness were not greater than 200. Figs. 16 and 17 show the standard fitness of the best agent in each generation and average summation of standard fitnesses of every chromosome in each generation.

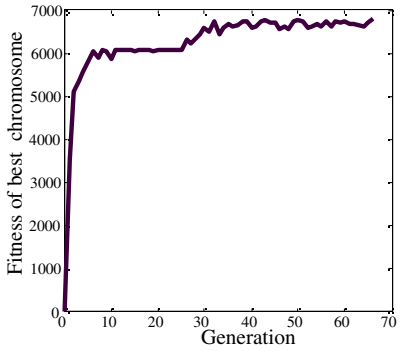

Fig. 16. Standard fitness of best agent at each generation.

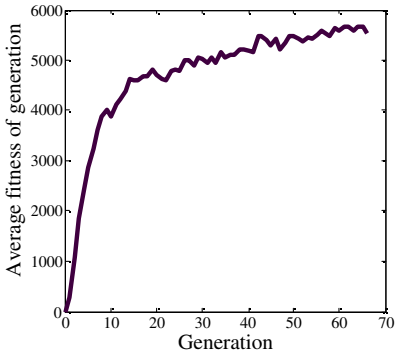

Fig. 17. Average of summation of standard fitnesses of every chromosome in each generation.

This new agent used more computation time compared to the conflict agent evolved in Section 5.1 (The population size was 3 times larger and the number of generations was more than twice the previous application.). Also, the number of training scenarios was increased by a factor of 1.5. The application in Section 5.1 used 10 h of computation time with a slower computer.

When the success rate in the scenarios was considered, the agent produced in this section outperformed the agent produced in Section 5.1. The success rate in Section 5.1 was 81.25% (26 of 32 transitions were accomplished) while the success rate of this new agent is 85.4% (41 of 48 transitions were accomplished). However, the main advantage of this new agent came from its ability to solve the new 8 scenarios. The new agent can solve the conflicts and problem of blockage in these new scenarios. On the other hand, this new agent is not completely successful in a few of the original conflict scenarios used in Section 5.1. However, it can bring at least one of the particles to its destination at each scenario (It was completely successful in 17 scenarios and at least one of the particles are navigated successfully to its goal in the remaining 7 scenarios). Even in the scenarios where the new agent is not completely successful, the unsuccessful particle navigates through all the available cells in the region where it was trapped (or enclosed) to find a way to get out of the region it is enclosed. Figs. 18 and 19 show the initial and final instants of a scenario where our agent was partially successful. In four of the similar training scenarios, the similar kinds of difficulties were encountered.

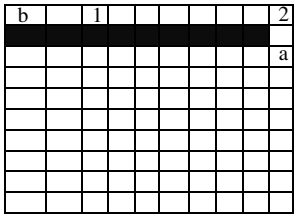

Fig. 18. Initial instance of a scenario where the new conflict agent is not totally successful (a shows the goal of first particle b shows the goal of second particle).

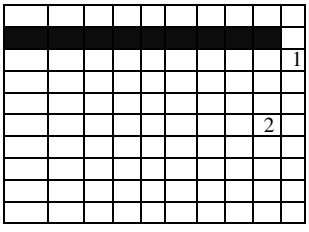

Fig. 19. Final instance of the scenario in Fig. 18.

This agent can also be partially used to overcome the blockage problem discussed in this section. It is capable of moving particles out of local regions, which is a crucial step in escaping from traps. However, it is better to generate a memory agent or new improved standard navigation agent to solve this problem since, in scenarios having more than two particles, it is possible to observe different blockage problems.

6. Conclusions and future work

The aim of this paper is to develop strategies for robots or particles to move towards their targets in grid regions containing obstacles. The developed strategies have been checked via simulations. The first set of simulations was performed to see the ability of GP to solve the particle navigation tasks and to obtain a strategy for this purpose. The effect of different control methods (homogeneous and heterogeneous control) was compared. The result was in favor of heterogeneous control as expected. Although heterogeneous control produces better results, it is a combination of different strategies for different particles. So, environmental variations such as the change of starting positions for the particles, the goals of the particles, utilization of a specific strategy developed for a particular particle for another particle, will result in poor performance. As expected, providing communications brings more success to the simulations. However, the original communication mode may sometimes produce undesired situations such as continuous control of a particle by another one. In order to deal with these problems, a new method of interaction between particles when they are in conflict (especially in a narrow corridor) was proposed. A new strategy considering different conflict situations was constructed by GP. This conflict strategy was used in the evolution of a global strategy controlling the particles in standard cases. The conflict module worked well but it was not able to solve the problem of blockage of the particles in their path to their goals by solid wall structures (i.e., traps), which the standard strategy could not handle because its ability to sense was restricted.

The particles could be considered to be reactive robots, which respond to their environment. However, the agents' ability to sense and foresee their surrounding was limited; the actions and operations proposed in GP are simple ones (Our robots' range of vision was restricted to the functionality of the proposed operations in GP). Exploring and scanning the scenarios with these operations and action set produced inadequate efficiency. In order to reach more efficient results, either the vision range or sensor range must be increased for the particles or the robots must be able to construct a map of the environment, which requires memory. This situation necessitated the construction of a "memory module" for the particles to cope with blockage of the continuous wall structures, which will be the topic studied in future work.

In this study, our objective was to construct the framework of some hierarchical or procedural control structures to implement basic navigation problems. For example, in the conflict module, a single conflict situation was proposed; two robots encounter in a narrow region. In fact, conflicts could arise in more complicated situations (conflict between three robots or more). Our conflict module is just a simple framework by whose principles a general-purpose conflict module could be developed. In our next study, another basic module (memory module) will be constructed, which will be used to help the robots leave local trap regions. Then, these basic modules (conflict module, memory module, etc.,) will be the basis for the construction of new advanced strategies, which will succeed in more complicated tasks.

References

[1] E. Aguirre, A. Gonzalez, Fuzzy behaviors for mobile robot navigation: design, coordination and fusion, International Journal of Approximate Reasoning 25 (2000) 255–289.
[2] R.C. Arkin, Behavior Based Robotics, MIT Press, Cambridge, 1998.
[3] T. Belker, M. Beetz, A.B. Cremers, Learning action models for the improved execution of navigation plans, Robotics and Autonomous Systems 38 (2002) 137–148.
[4] M. Bennewitz, W. Burgard, S. Thrun, Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots, Robotics and Autonomous Systems 41 (2–3) (2002) 89–99.
[5] R.J. Duro, J.A. Becerra, J. Santos, Behavior reuse and virtual sensors in the evolution of complex behavior architectures, Theory in Biosciences 120 (2001) 188–206.
[6] T.C. Fogarty, L. Bull, B. Carse, Evolving multi-agent systems, in: G. Winter, J. Periaux, M. Galan, P. Cueta (Eds.), Genetic Algorithms in Engineering and Computer Sciences, John Wiley and Sons, Chichester, 1995, pp. 3–22.
[7] H. Iba, Evolutionary learning of communicating agents, Journal of Information Sciences 108 (1998) 181–205.

[8] K.-Y. Im, S.-Y. Oh, S.-J. Han, Evolving a modular neural network-based behavioral fusion using extended VFF and environment classification for mobile robot navigation, IEEE Transactions on Evolutionary Computation 6 (4) (2002) 413–419.
[9] S. Koenig, Minimax real-time heuristic search, Artificial Intelligence 129 (2001) 165–197.
[10] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, 1992.
[11] M. Kruusmaa, J. Willemson, Covering the path space: a casebase analysis for mobile robot path planning, Knowledge-Based Systems 46 (5–6) (2003) 235–242.
[12] K.-H. Lee, J.-H. Kim, Multi-robot cooperation-based mobile printer system, Robotics and Autonomous Systems 54 (3) (2006) 193–204.
[13] W. Li, C. Ma, F.M. Wahl, A neuro-fuzzy system architecture for behavior-based control of a mobile robot in unknown environments, Fuzzy Sets and Systems 87 (1997) 133–140.
[14] R.R. Murphy, Introduction to AI Robotics, MIT Press, Cambridge, 2000.
[15] M.N. Nicolescu, M.J. Mataric, Learning and interacting in human–robot domains, IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans 31 (5) (2001) 419–430.
[16] L.E. Parker, L-ALLIANCE: task-oriented multi-robot learning in behavior-based systems, Advanced Robotics 11 (4) (1997) 305–322 (special issue on selected papers from IROS '96).
[17] L.E. Parker, L-ALLIANCE: a mechanism for adaptive action selection in heterogeneous multi-robot teams, Technical Report, Oak Ridge National Laboratory, Tennessee, 1995.
[18] D. Payton, R. Estkowski, M. Howard, Compound behaviors in pheromone robotics, Robotics and Autonomous Systems 44 (3–4) (2003) 229–240.
[19] A. Ram, R. Arkin, G. Boone, M. Pearce, Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation, Adaptive Behavior 2 (3) (1994) 277–305.
[20] P. Rusu, E.M. Petriu, T.E. Whalen, A. Cornell, H.J.W. Spoelder, Behavior-based neuro-fuzzy controller for mobile robot navigation, IEEE Transactions on Instrumentation and Measurement 52 (4) (2003) 1335–1340.
[21] H. Seraji, A. Howard, Behavior-based robot navigation on challenging terrain: a fuzzy logic approach, IEEE Transactions on Robotics and Automation 18 (3) (2002) 308–321.
[22] B. Ster, An integrated learning approach to environment modeling in mobile robot navigation, Neurocomputing 57 (2004) 215–238.
[23] J. Tani, Model-based learning for mobile robot navigation from the dynamical systems perspective, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 26 (3) (1996) 421–436.
[24] B. Yamauchi, R. Beer, Spatial learning for navigation in dynamic environments, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 26 (3) (1996) 496–505.
[25] J. Yen, N. Pflunger, A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation, IEEE Transactions on Systems, Man and Cybernetics 25 (6) (1995) 971–978.
[26] C. Zheng, M. Ding, C. Zhou, L. Li, Coevolving and cooperating path planner for multiple unmanned air vehicles, Engineering Applications of Artificial Intelligence 44 (8) (2004) 887–896.