GENETIC ALGORITHM IN MOBILE ROBOT CONTROL

Fadi Halal, Ioan Dumitrache

University "POLITEHNICA" Bucharest, dept of automatic control
313- splail independentei Bucharest Romania,
E mail idumitrache@ics.pub.ro

Abstract: *GA is one of the intelligent techniques that are used in the robotic filed. It is efficient to design behavioural controllers for mobile robots, such as a GA control simulated mobile robot called Khepera. In this paper we describe three cases of GA which have different chromosome attitudes, structures and objective functions (fitness). These attitudes define the basic robot behaviour and fitness teaches robots to determine his performance's best path with respect to time and distance. The results obtained demonstrate the controlled robot in different obstacle area, and we get performance robots (best path) in developed case genetic algorithm. GA has many problems depending on his condition area but finally he will succeed in obtaining it target.*

Keywords: *genetic algorithm, (khepera), KiKS simulator, fitness, chromosome,*

1. INTRODUCTION

This paper discusses the genetic algorithm ability to control a mobile robot called khepera in it simulator KiKS without violating any physical constraints of the robot, environment or task. We present three cases with different chromosomes and different fitness. From these experiments we can develop and improve the robot's performance in it environment. The robot changed it behaviour depending on the GA model. When the robot is controlled by a GA developed model control, its behaviour became more intelligent. The first section is the fundamentals of GA algorithms, in the second section we see GA for robot mobile control, the third section is the experiments and the forth section is the developments and simulations of the experiments.

1.1 Why to use GA:

GA is easy to use without a deep mathematical background; GA's mimic models of natural evolution and have the ability to adaptively search large spaces in near-optimal ways. [8] . GA can solve hard problems: GA is one of the best ways to solve a problem for which is little known ; GA is easy to extend: the user has a high degree of freedom with the possibility of implementing ; GA is easy to interface with exciting simulations and models. [1]

We used GA for optimum mobile robot path planning, that it fined a suitable collision-free path for a mobile robot to move from a start location to a target location. This path may be optimal or near-optimal with respect to time, distance and the ability to obtain the target.

When the robot is put in unstructured complex environment, and the obstacles have different sophisticated shape. In this paper we studied the effect of fitness function and chromosome on the control ability and the robot path. And we used GA for it navigation ability; we do not need to train the robot in new areas since the GA can solve the problem without any advance knowledge about it environment. Thus this control could solve the robot tasks in different and new obstacles area.

2. GA FOR ROBOT MOBILE CONTROL

2.1 GA algorithm

GA operates on the coding of decision variables instead of adapting the parameters themselves. Variables are coded into a fixed length of string called chromosomes. These chromosomes contain many genes and each gene has variables. In this experiment there are two variables: left and right motor speed. This chromosome is coded as a vector of floating point. Where the input variables (sensor values) are the proximity sensor (IR) data and robot's angle and it coordinate (x, y)

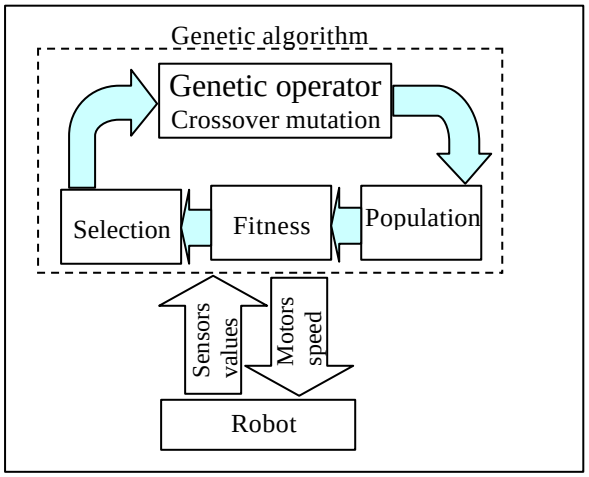

Fig. 1. GA Control and his environment

GA algorithm

The basic operation of a genetic algorithm can be summarized as the following:

1. Randomly initialize a population of chromosomes (In this study there are 50 individuals).
2. While the terminating criteria have not been satisfied.
   a. Evaluate the fitness of each chromosome: by evolution program to simulated Khepera robot in KiKS and measure his tasks abilities
   b. Remove chromosomes with low fitness.
   c. Generate new chromosomes generation, using certain selection schemas and genetic operators. [4]

Fig: 1. show the GA algorithm and his environment

2.2 Khepera implementation

Khepera is a miniature mobile robot with a diameter of 55 mm and a weight of 70 g. The Khepera robot has been designed on the basis of the following criteria: miniaturization, modular open architecture, expandability, interface, and compatibility with larger robots. [2]

The robot is supported by two lateral wheels that can rotate in both directions and two rigid points in the front and in the back. By spinning the wheels in opposite directions at the same speed, the robot can rotate without lateral displacement. [3] The sensory system employs eight sensors distributed around the body, six on one side and two on the other side. These infrared sensors are called "proximity or distance sensors". They return an integer value of 1023 when the object is very close while the value 0 means that there is no closer object (far than 10 cm). Intermediate numbers give an approximate idea of the distance between the sensor and the object. [6]

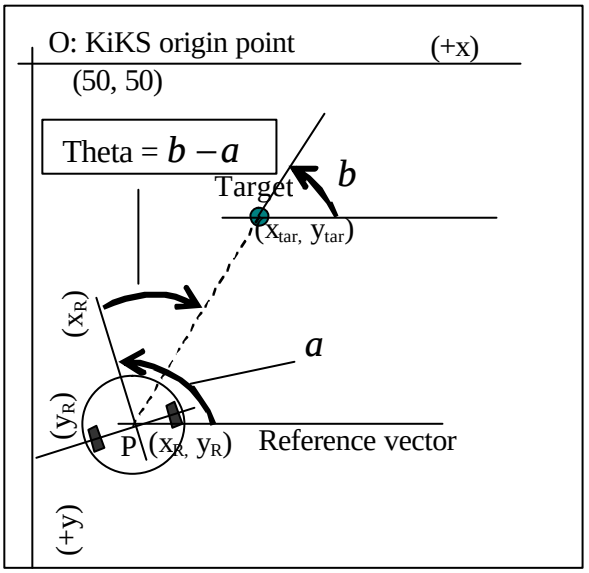

Fig. 2. Robot in his kiss's environment.

The fig 2 shows the KiKS coordinate. That is the global reference frame to specify the position of the robot choose a point P on the robot chassis as its reference position. Thus the robot local frame P :{ x_R, y_R}. [2] The target coordinate is (x_{tar}, y_{tar}).

The angle theta is the robot's target direction so we can obtain the following relation :

Theta= $b - a$

Where: Theta: robot's target direction

b : The angle between the vector from the robot to goal and the reference vector

a : Robot's angle .We can obtain b from the relation

$$b = \tan^{-1}\sqrt{(ytar - yR)/(xtar - xR)}$$

2.3 Robot's target direction models

We design two models. The first one has eight areas and the second model has ten areas as shown in Fig. 3 and 4.

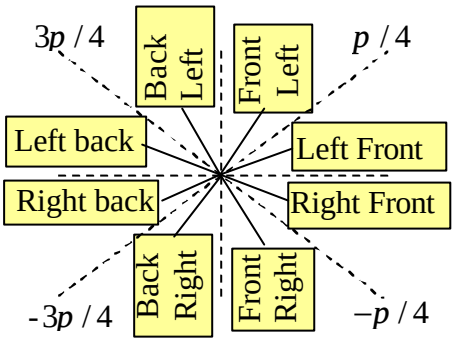

Fig: 3. Target direction of khepera movement first model: eight possible targets in keeper's direction. In chromosome with 11 genes.

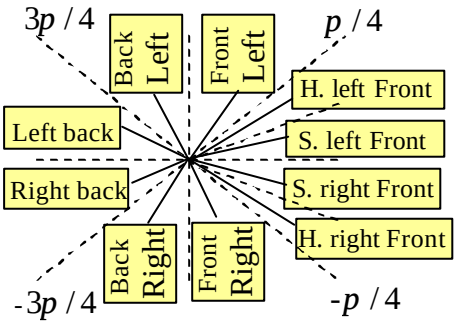

Fig: 4. Target direction of khepera movement. Second direction model: ten possible khepera's target directions. In chromosome 13 genes.

All the areas in fig 3 are the same as fig 4 with out left front is $p/4 > \text{theta} > 0$ where is divided in two areas, H. left Front $p/4 > \text{theta} > p/8$, S. left Front $p/8 > \text{theta} > 0$, and right front is $0 > \text{theta} > -p/4$,where is divided in two areas S. right Front $p/8 > \text{theta} > 0$ and h. right front $0 > \text{theta} > -p/8$. When $p/8 < \text{theta} > 0$ it means that the goal point is in the Area S. left front and the robot should go ahead and to the left side to get his target.

H. Left front	$p/4 > \text{theta} > p/8$
S. Left front	$p/8 > \text{theta} > 0$
S. Right front	$0 > \text{theta} > -p/8$
H. Right front	$-p/8 > \text{theta} > -p/4$
Front right	$-p/4 > \text{theta} > -p/2$
Back right	$-p/2 > \text{theta} > -3p/4$
Right back	$-3p/4 > \text{theta} > -p$
Left back	$p > \text{theta} > 3p/4$
Back left	$3p/4 > \text{theta} > p/2$
Front left	$p/2 > \text{theta} > p/4$

Fig: 5. robot target direction (ten areas)

2.4 Khepera differential robot

Khepera is a differential robot and he has left and right wheels that have the same zero motion line we can observe (ICR) laid along this line. [2]

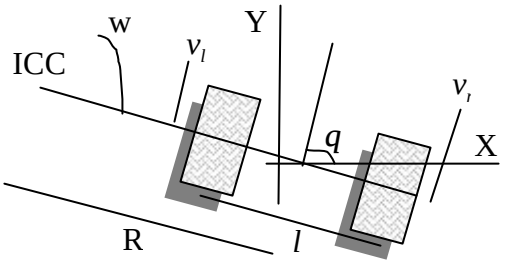

Fig: 6. Khepera model.

$$R = \frac{l(v_l + v_r)}{2(v_r - v_l)}$$

Where

l : is the distance between the two wheels

R: is the radius of the (ICC)

v_r : is the velocity of the right wheels

v_l : is the velocity of the left wheels.

If $v_r = v_l$ then the radius is infinite and the robot moves in a straight line ; If $v_l = -v_r$ the radius is zero and the robot rotates about a point midway between two wheels for other values of the $v_r and v_l$ robot moves in a curved trajectory. We can observe that the robot is controlled easily by adjusting the motor speed, and then if the robot encounters a left obstacle, it can avoid it by accelerating the left motor speed at the same time decelerating the right motor speed. If the left and right motor speeds are the same values then the robot goes ahead and the GA task adjusts these variables (genes) to get good a path, steering to it target, going straight and avoiding obstacles.

3. EXPERIMENTS

3.1 Sensor simplification

We simplify the sensor reading in three categories, without affecting Khepera's performance as is shown in Fig. 7.

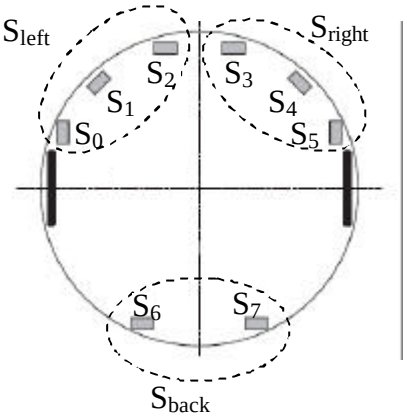

Fig: 7. Sensors reading simplification.

- $S_{left} = (S_0 + S_1 + S_2) / 3$
- $S_{right} = (S_3 + S_4 + S_5) / 3$
- $S_{back} = (S_6 + S_7) / 2$

Where S_{left}, S_{right} and S_{back} input sensor could detect respectively left, right and back obstacles. From the sensors reading and the robot's direction inputs, the Khepera's attitude can be defined according to the following states rule:

IF ((S_{left} > L) or (S_{right} > L) or (S_{back} > L))
THEN
Obstacle detected
S_t= max (S_{left}, S_{right}, S_{back})
ELSE
Target direction = $b - a$ (collision free)
END

The constant L represents a collision threshold and from the experiments the best value is between [100, 250], if l > 500 the robot cannot avoid the detected obstacle, it manoeuvres unsuccessfully.

3.2 Chromosome

Each possible attitude defined by the states rule corresponds to a gene of the chromosome, fig: 8, 10. Each gene is composed of the pair of Khepera's motor speeds (M1, M2), (Left and right motor speed). Fig. 8 represents the first chromosome model, it contains 11 genes divided in two parts; the obstacle detected states and robot target direction movement. Khepera is programmed to avoid obstacles when he encounters a collision area thus it forgets it target location, being able to move away from its arrival point in such cases. [6] That means if the target is in the front left and there is an obstacle in this area, the robot avoids it without considering the target direction.
Fig. 10 shows Chromosome model with 13 genes, 10 genes represent robot target direction and 3 genes represent obstacle detected in this model we divide the front areas in four areas

3.3 Fitness function

In general each problem to be solved requires a unique fitness function that is to evaluate a suitability of a solution with respect to the overall goal. The fitness function retunes a numerical value corresponding to the given chromosome.
In this paper we applied tree fitness function. Fig. 9. is shown two fitness functions which we applied in the first and the second experiment.; F that is calculated and summed over 1000 robot's step, is composed of the components: V, D and A. [6] where V improves the speed and D improves the straight way and A improves collision avoidance.
The third fitness function F_N has 6 components, S, ST and A. Where S improves the robot's speed, ST improves the straight walking and A improves obstacle avoiding and wall following these components is calculated and summed each robot's step. Where the most important upgrade in this function that the time is not fixed, it means when the robot gets his target the count comes down. Then TI improves time task. TR and DIS teach the robot to get it target point at short path, Thus if the robot doesn't get it goal within 1000 steps, it is punished by omitting the grade of these components TI, TR and DIS.

4. EXPERIMENTS DEVELOPMENT AND SIMULATION

4.1 Experiments development

For applying genetic algorithm we used GAOT toolbox. (The Genetic Algorithm Optimization Toolbox (GAOT) for Matlab 5) and the main function is called ga. [7] Input system contains the proximity sensor (IR) data and robot's angle and his coordinate (x, y). Output system is the motors speed is the left and right motor speed.

Target direction								obstacle detected		
M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2
Left front	Right front	Front right	Behind right	Right back	left back	Behind left	front left	Left obstacle	Right obstacle	Back obstacle
1	2	3	4	5	6	7	8	9	10	11

Fig: 8. Eleven genes chromosome representation.

$$F = \sum_{I=1}^{\#gene} (V_I * D_I * A_I); 0 \le V_I \le 1; 0 \le D_I \le 1; 0 \le E_I \le 1;$$

$V_I = \frac{\lvert M1 \rvert + \lvert M2 \rvert}{2 * M\,\max}$	**11 gene chromosome**	**13 gene chromosome**
$D_I = \begin{cases} 1 + \frac{(M1 + M2)}{2 * M\,\max} \\ \frac{2}{1} \end{cases}$	I=[1,2,9,10,11] I=[3,4,5,6,7,8]	I=[1,2,3,4,11,12,13] I=[5,6,7,8,9,10]
$A_I = \begin{cases} \sum_{t=1}^{TP_I} AAi_t\, 0 / TP_I \\ 0 \end{cases}$	$TP_I \neq 0$ $TP_I = 0$	$TP_I \neq 0$ $TP_I = 0$
$AAi_t = \begin{cases} \Delta di / d\,\max \\ \Delta gi / g\,\max \\ 1 - S_t / S\,\max \end{cases}$	If I=[1,2]and $\Delta di \le o$ Else $AAi_t = 0$ If I=[3,4,5,6,7,8]and($\Delta gi \le o$) Else $AAi_t = 0$ If I=[9,10,11]	If I=[1,2,3,4]and $\Delta di \le o$ Else $AAi_t = 0$ If I=[5,6,7,8,9,10]and($\Delta gi \le o$) Else $AAi_t = 0$ If I=[11,12,13]

Fig. 9. The fitness function for 11-genes and 13 -genes chromosome.

Where:
- i: attitude's index;
- V_i: attitude's speed;
- D_i: attitude's direction motion;
- A_i: attitude's avoid obstacles efficiency;
- TP_i: total of steps executed by attitude i;
- S_t: proximity-sensor (S_{left}, S_{right}, S_{back}) highest activity at step t;
- Mmax: maximum robot's speed (equal to 10);
- Smax: maximum sensor's reading (equal to 1023).
- dmax: maximum possible distance travelled by robot in only one step
- gmax: maximum possible robot's angle rotation in only one step;
- Δdi : difference of goal's distance between two consecutive steps;
- Δgi : difference of target's direction between two consecutive steps;
- AAi : Action's fitness;

Target direction							obstacle detected					
M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2	M1,M2
H. left front	S. left Front	S. Right Front	H. right front	Front right	Behind right	Right back	left back	Behind left	front left	left obstacle	Right obstacle	Back obstacle
1	2	3	4	5	6	7	8	9	10	11	12	13

Fig: 10. Thirteen genes chromosome representation

$F_N = S + ST + A + TR + DIS + TI \;, 0 \le$ each component ≤ 1	
$S = \sum_{i=1}^{t} \frac{\lvert M_L + M_R \rvert}{2M_{MAX}} / t$	
$ST = \left[1 - \sum_{i=1}^{t} \frac{q_t}{g_{\max}} / t \right]$ $q_t = \lvert Thetq - Thetq_{-1} \rvert$ $q_t = 0$	$S_t = 0$ $S_t \neq 0$
$A = \left[1 - \sum_{i=0}^{t} \frac{S_t}{S_{MAX}} / t \right]$	
TR = 1 TR = 0	If the robot gets the target If the robot doesn't get the target
$TI = \left[1 - \frac{t}{1000} \right]$ $TI = 0$	If the robot gets the target If the robot doesn't get the target
$DIS = \left[1 - \frac{X}{3\sqrt{(y_{tar} - y_{ini})^2 + x_{tar} - x_{ini})^2}} \right]$ $X = \sum_{i=1}^{t} \sqrt{(y_t - y_{t-1})^2 + (x_t - x_{t-1})^2}$ Distant = 0	If the robot gets the target If the robot doesn't get the target

Fig: 11. The fitness function for 13-genes chromosome (novel function)

Where
- Mmax: maximum robot's speed (equal to 10);
- (M_L, M_R): left motor and right motor speed
- Smax: maximum sensor's reading (equal to 1023).
- St: proximity-sensor (S_{left}, S_{right}, S_{back}) highest activity at step t;
-X: the distant travelled by robot in only one step
-gmax: maximum possible robot's angle rotation in only one step
-Theta: robot target's direction
- (y_{tar}, x_{tar}) (y_{ini}, x_{ini}) robot's target point and start point (coordinate)

First experiment: we used chromosome with 11-genes and F fitness function, this model is the same in [6] with the following parameter: Population size 50 individual, crossover rate 80%; mutation rate 5%; generation number 100; collision threshold l= 150; time 1000 steps. Fig 12 and 13 show the best chromosome fitness in the each generation and the average of the all the chromosome in each generation.

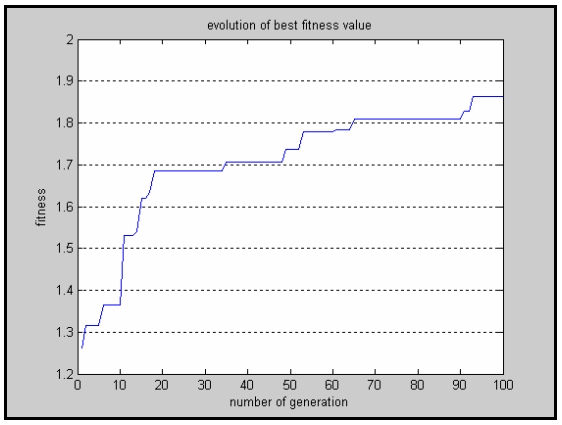

Fig: 12. Evolution of best chromosome (11 genes) in 100 generation.

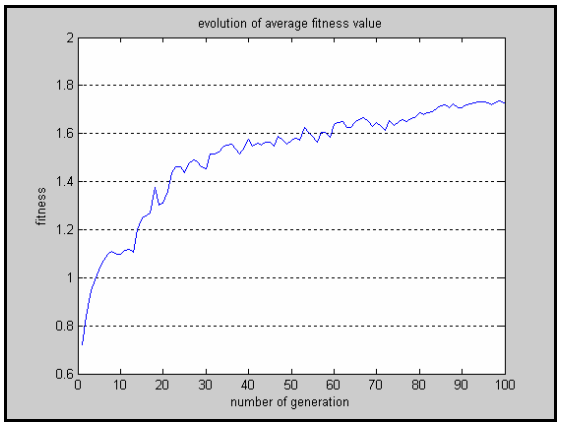

Fig: 13. Evolution of average fitness chromosome (11 genes) in 100 generation.

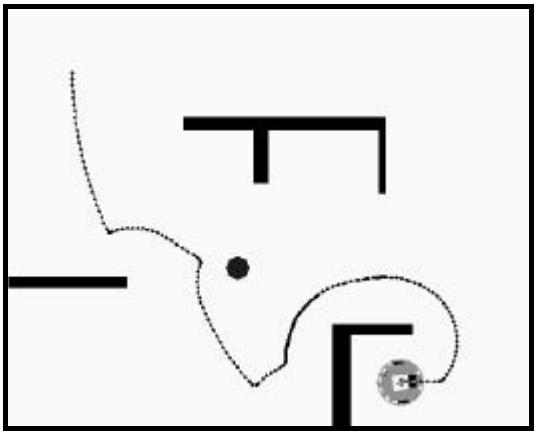

Fig: 14. The best solution 11-genes. (first experiment)

Second experiment: we used a modified model, that it has chromosome with 13-genes and F fitness function and the following parameter. Population size 50 individual, crossover rate 80%; mutation rate 5%; generation number 500; collision threshold l= 150; time 1000 steps. Fig 15 and 16 show the best chromosome fitness in each generation and the average of the all the chromosomes in each generation.

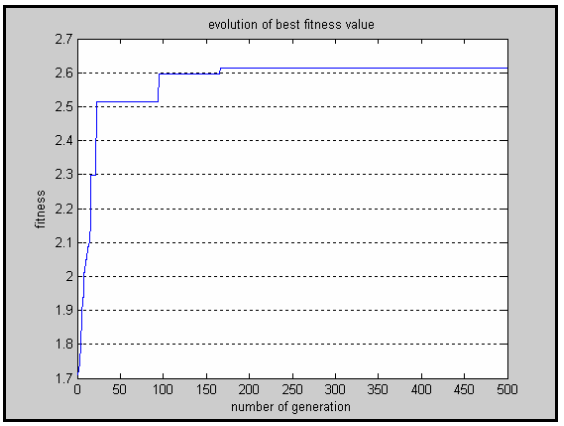

Fig: 15. Evolution of best chromosome (13 genes) in 500 generation second experiment.

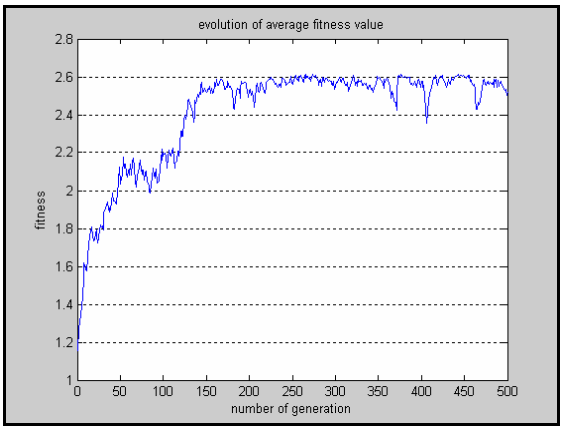

Fig: 16. Evolution of average fitness chromosome (13 genes) in 500 generation second experiment.

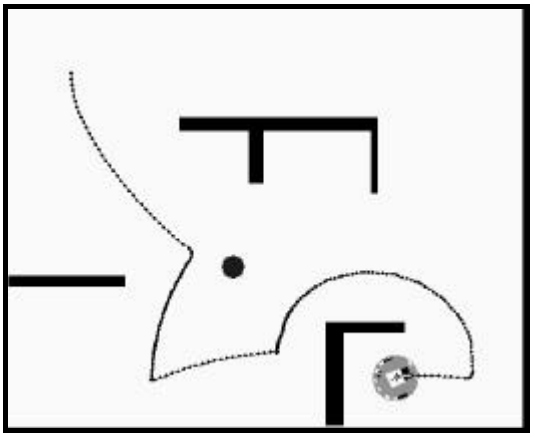

Fig: 17. The best solution 13-genes (Second experiment)

Third experiment: this is a novel model; we used a chromosome with 13-genes and F_N: fitness function as is shown in fig 11, and the following parameter:
Population size 50 individual, crossover rate 80%; mutation rate 5%; 500 generation number; collision threshold l= 150; time either necessary time to get his goal or 1000 steps. Fig 18 and 19 show the best chromosome fitness in each generation and the average of the all the chromosomes in each generation

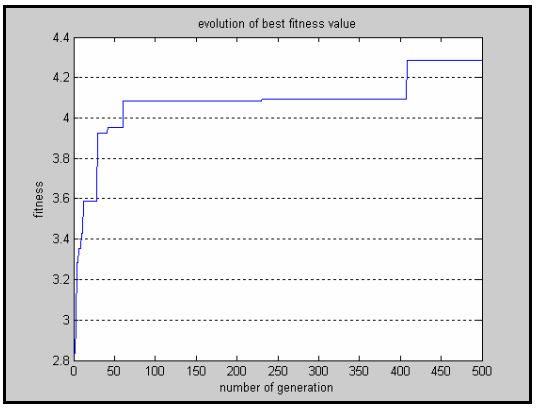

Fig: 18. Evolution of best chromosome (13 genes) in 500 generation. (third experiment)

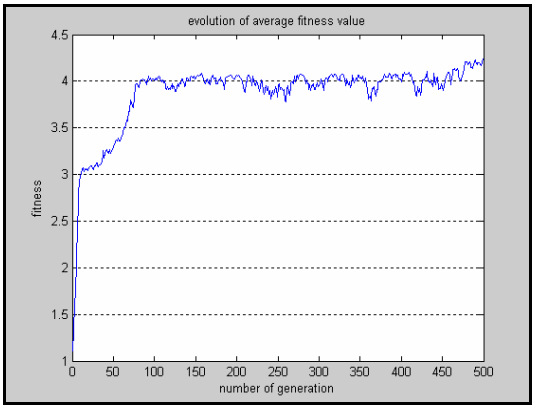

Fig: 19. Evolution of average fitness chromosome (13 genes) in 500 generation. (third experiment)

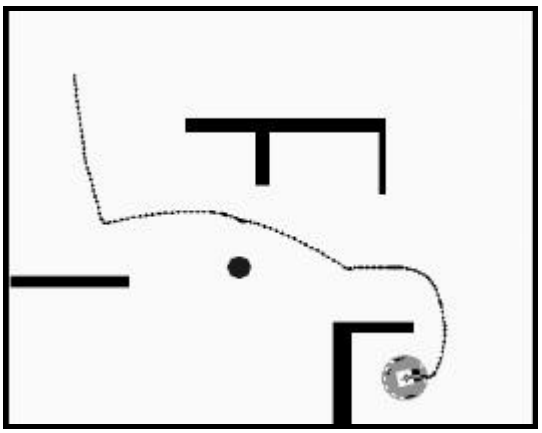

Fig: 20. The best solution 13- genes. (third experiment)

Where the start point is the beginning of the path and the goal point is when he arrives. Whenever the dots are close to each other it means the robot runs slow and when the dots are far to each other that mean the robot goes faster.

Where Fig. 12 and 13 represent respectively the Evolution of the best chromosome and Evolution of average fitness chromosome in the first experiment. We obtain the best solution from 5000 chromosomes that is represented in Fig. 14.

Where Fig. 15 and 16 represent respectively the Evolution of best chromosome and Evolution of average fitness chromosome in the second experiment. We obtain the best solution from 25000 chromosomes. That is represented in Fig. 17.

Where Fig. 18 and 19 represent respectively the Evolution of best chromosome and Evolution of average fitness chromosome in the third experiment. We obtain the best solution from this experiment that is represented in Fig. 20. In fig 14, 17 and 20 we observe that fitness has an important role to improve the path which we can see clear in fig 20 where GA teaches the robot to get his target faster with intelligent behaviour than the two previous experiments thus the robot has short and fast path, where all the path at this Fig. almost dot point that means the robot travels with high speed, and when there is a little bit solid segment that means the robot travels in slow speed.

4.2 simulations

In Fig. 21 we have a problem that the robot encounters in cycling mode and the robot will never get his target because the robot enters into the target zone with target orientation 90 degrees. Thus the problem is when the target is very close the robot he couldn't get his target and continuing in cycle movement around it target as shown in Fig. 21.

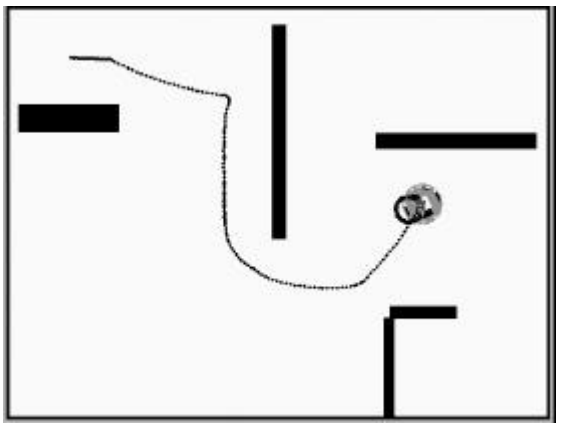

Fig: 21. Robot was encounter cycling case.

The experiments confirmed the power of the genetic algorithm in many new environments. We applied the best solution that we have obtained from the third experiment (13 genes) in different areas as are shown in fig 21, 22 and 23. We can observe that the robot solves it problem with out advance knowledge about his environment, and in Fig. 22, 23 GA demonstrates his navigation ability.

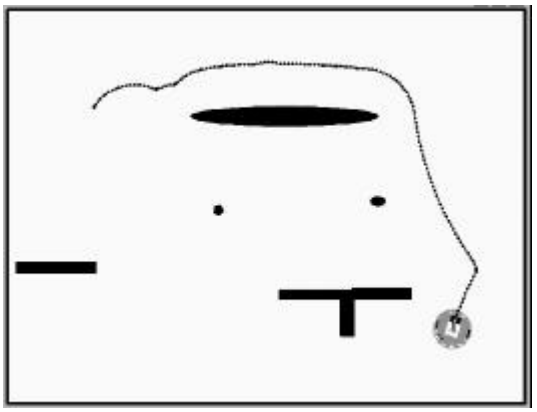

Fig: 22. Robot is in new environment.

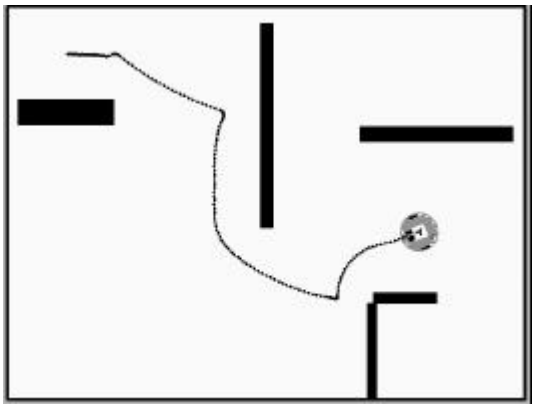

Fig: 23. Robot gets his target (no obstacle near his target).

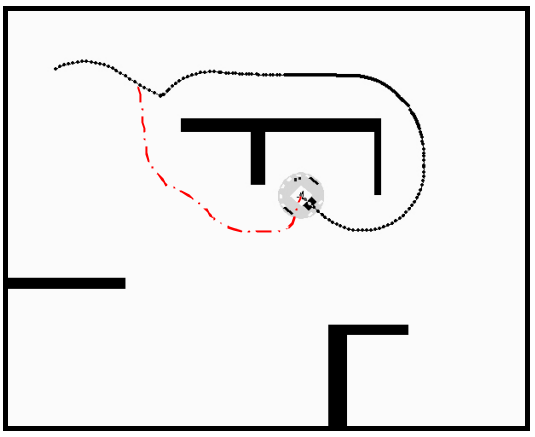

Fig: 24. Robot gets his target with the solution 11-genes.

In Fig. 24, 25 we applied the best solution that we have obtained from first experiment to control the robot in new area with different start point condition and other target point. We see clear the robot solved his problem and got it target.

We observe the robot has walked slowly when the target point is behind or between some obstacles. This behaviour is the mixture from two behaviours the obstacles avoiding and the target steering. One of the model's disadvantages is that the robot could avoid the obstacle in an intelligent way as the supposed path dash dot line. That is because the fitness has neither consideration for the type of distant (path) nor the required time that the robot wanted to get his target.

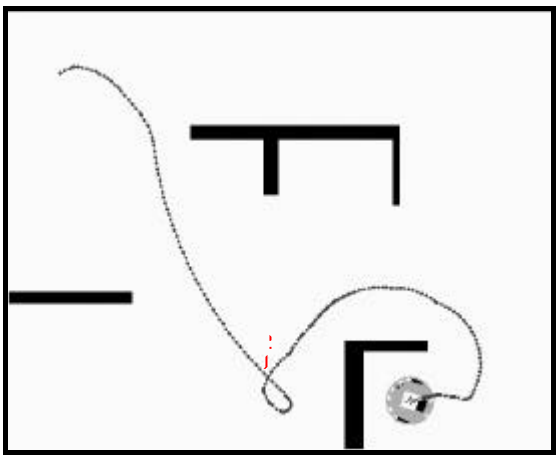

Fig: 25. Robot gets his target with the solution 11-genes.

5. CONCLUSION

The experiments improved the genetic algorithm power of navigation shown in new areas. The robot was put in new areas, which have different conditions the start point, situated different obstacles and the target point, thus the robot solved the problem with out any advance knowledge about it environment, and here GA demonstrates his navigation ability.

In many cases the robot fell in cycling mood around its target. This problem occurs when the target is near same obstacles, and maybe it falls in death zone, also it depends on the obstacle shape and it target position. The difficult problem in these models is when the robot encounters an obstacle; the robot avoided it without any consideration to it target. This problem is caused because the robot had no advance knowledge about his area.
The experiments result present that the fitness have an important role to improve the path. These fitness teach the robot to get it target at a short and fast path. When all the paths almost dot point that means the robot travels with high speed, and when there is a little bit solid segment that means the robot travels in slow speed.

In the first and second model result we observe that the paths are roundabout, and they have many solid segments. In the second model the problem is when the robot encounters an obstacle, he avoided it hardly because his ability to direct to it target faster.
The important development in these experiments is that we introduce the distant and the time in the fitness function that is trained the robot to get it target at short time and short path. We improved the robot's performance and it ability and it became more intelligent and had a stable and good behaviour.

REFERENCES

[1] Ion Dumitrache – Catalin Buiu "introduction to genetic algorithms" 1995 Bucharest
[2] Roland siegwart illah R.Nourbakhsh "Introduction to autonomous mobile robots" Mit press 2004
[3] Stefano Nolfi and Dario Floreano Evolutionary Robotics "the Biology, Intelligence, and Technology of Self-Organizing Machines" 2000
[4] Thomas braunl "Mobil robot design and applications with embedded system" Springer 2003 [5] Simon X. Yang, Yanrong Hu, "robot path planning in unstructured environments using a knowledge-based genetic algorithm" Copyright c_2005 IFAC.
[6] Carlos E. Thomaz, Marco Aurélio C. Pacheco, Marley Maria B.R. Vellasco, "Mobile Robot Path Planning Using Genetic Algorithms"
Departamento de Engenharia Elétrica Pontifícia Universidade Católica - PUC/Rio, Brazil
Departamento de Engenharia de Sistemas e Computação Universidade do Estado do Rio de Janeiro - UERJ, Brazil
http://www.ica.ele.puc - rio.br/publicacoes/download/cnf_0114.
[7] http://www.ie.ncsu.edu/mirage/ GAToolBox/gaot/
[8] http:// www.k-team. com/kteam/index.php?site=1&rub=2&page=179&version=EN
Khepera Documentation
Khepera user manual
Khepera II Programming Manual
[9] http://www.ica.ele.puc- rio.br/publicacoes/download/cnf_0106.pdf
[10] http://www.tstorm.se/projects/kiks/
[11] http://www.k- team.com/download/kameleon/matlab/readme.txt