A Genetic_FPGA Algorithm Path planning Of an Autonomous Mobile Robot

OUARDA HACHOUR
Department Of Electrical and Electronics Engineering
Faculty Of Engineering Sciences
Signal and System Laboratory
Boumerdes University
Boulevard de l'independence Boumerdes 35000
ALGERIA

Abstract: -This paper proposes Genetic Algorithms (GAs) for path Autonomous Mobile Robot (AMR). This approach has an advantage of adaptivity such that the GA works perfectly even if an environment is unknown. First, we present a software implementation GA path planning in a terrain. The results gotten of the GA on randomly generated terrains are very satisfactory and promising . Second, we discuss extensions of the GA for solving both paths planning and trajectory planning using a Single Static Random Access Memory (SRAM) for Field Programmable Gate Array (FPGA). This new design methodology based upon a VHDL description of the path planning has the two (02) advantages : to present a real autonomous task for mobile robots, and being generic and flexible and can be changed at the user demand. The results gotten are promising.

Key-Words: - Genetic Algorithm (GA), Autonomous Mobile Robot (AMR), FPGA implementation, VHDL, Synthesis, and Galileo.

1 Introduction

Motion planning is one of the important tasks in intelligent control of an autonomous mobile robot . It is often decomposed into path planning and trajectory planning. Path planning is to generate a collision free path in an environment with obstacles and optimize it with respect to some criterion. Trajectory planning is to schedule the movement of a mobile robot along the planned path . A wide variety of approaches have been considered, but these can broadly be categorized into on-line and off-line techniques. However, few algorithms have been developed for on-line motion planning in a time-varying or unknown terrain.

A robotic systems capable of some degree of self-sufficiency is the overall objective of an AMR and are required in many fields[2,3]. The focus is on the ability to move and on being self-sufficient to evolve in an unknown environment for example. Thus, the recent developments in autonomy requirements, intelligent components, multi-robot system, and massively parallel computer have made the AMR very used, notably in the planetary explorations, mine industry, and highways [1,9].

This paper deals with the intelligent path planning of AMR in an unknown environment. The aim of this paper is to develop an AMR for the AMR stationary obstacle avoidance to provide them more autonomy and intelligence .This technology GA based on intelligent computing as becoming useful as alternate approach may be able to replace the classical approaches such as: recognition, learning, decision-making , and action (the principle obstacle avoidance problems) can be realized in efficient manner.

Recently, applications of genetic algorithms to path planning or trajectory planning have been recognized. GA is a search strategy using a mechanism analogous to evolution of life in nature. The GAs, which are evolutionary have recently emerged from study of the evolution mechanisms and are searching strategies suitable for finding the globally optimal solution in a large parameter space. They are based on learning mechanisms. It has widely been recognized that GA works even for complex problems such that traditional algorithms cannot find a satisfactory solution within a reasonable amount of time.

To benefit from using such technologies, another technological advance in hardware has revealed the soft-computing in order to give a high speed processing that could be provided through massively parallel implementation management and so more autonomy requirements such as power, thermal and communication management are obtained in real time. GA approach can be implemented either in analogue or digital way. Analog circuits are sensitive to noise and temperature changes and inter-chip variations make manufacturing functionally identical circuits very difficult. Digital integrated circuits, on the other hand, is easily manufactured and are functionally identical. Nowadays,

FPGA are gaining momentum in digital design. They are used for a wide range of application including rapid prototype, glue logic for microprocessor, and hard–wired simulation. Moreover, the FPGA circuit offers attractive features for hardware designer in comparison with other VLSI techniques. Then, this new design methodology of FPGA offers more performance than the software implementation (an easiness of implementation, shortest possible time, and rapid execution by AMR).In this paper, we discuss the possibility to deal with hurdle by proposing a new approach permitting the mapping of an entire navigation approach (planning, intelligent control for obstacle avoidance) into a single Xilinx's.

2 Motion planning

The navigation planning is one of the most vital aspect of an autonomous robot . In most practical situations, the mobile robot can not take the most direct path from start to the goal point. So, path finding techniques must be used in these situations, and the simplest kinds of planning mission involve going from the start point to the goal point while minimizing some cost such as time spent, chance of detection, etc.

Path planning in spatial representation often requires the integration of several approaches. This can provide efficient, accurate, and consist navigation of a mobile robot. It is sufficient for the robot to use a topological map that represents only the areas of navigation (free areas , occupied areas of obstacles). It is essential the robot has the ability to build and uses models of its environment, that enable it to understand the environment's structure. This is necessary to understand orders, plan and execute paths [4].

Many researches which have been done within this field, some of them used a "visibility graph" to set up a configuration space that can be mapped into a graph of vertices between which travel is possible in a straight line. The disadvantage of this method is time consuming. At the opposite, some researches have been based on dividing the world map into a grid and assign a cost to each square. Path cost is the sum of the cost of the grid squares through which the path passes. A grid model has been adopted by many authors, where the robot environment is divided into many squares and indicated to the presence of an object or not in each square. A cellular model, in other hand, has been developed by many researchers where the world of navigation is decomposed into cellular areas, some of which include obstacles. More, the skeleton models for map representation in buildings have been used to understand the environment's structure, avoid obstacles and to find a suitable path of navigation. These researches have been developed in order to find an efficient automated path strategy for mobile robots to work within the described environment where the robot moves.

In this paper, a simple and efficient navigation approach for autonomous mobile robot is proposed in which the robot navigates, avoids obstacles and attends its target. Note that, the algorithm described here is just to find a feasible and flexible path from initial area source to destination target area, flexible because the user can change the position of obstacles it has no effect since the environment is unknown. This robust method can deal a wide number of environments and gives to our robot the autonomous decision of how to avoid obstacles and how to attend the target. More, the path planning procedure covers the environments structure and the propagate distances through free space from the source position. For any starting point within the environment representing the initial position of the mobile robot, the shortest path to the goal is traced. The algorithm described here therefore is to develop a method for path planning by using simple and computationally efficient-way to solve path planning problem in an unknown environment without consuming time, lose energy, un-safety of the robot architect.

3 The Proposed GA For Motion Planning Of AMR

3.1 Path planning

Assume that path planning is considered in a square terrain and a path between two locations is approximated with a sequence of adjacent cells in the grid corresponding to the terrain. The length A(α,β) from cell ″α″ to its adjacent cell ″β″ is defined by the Euclid distance from the center cell ″α″ of one cell to the center cell ″β″ of another cell. Each cell in this grid is assigned of three states : *free, occupied,* or *unknown* otherwise. A cell is *free* if it is known to contain no obstacles, *occupied* if it is known to contain one or more obstacles. All other cells are marked *unknown*. In the grid, any cell that can be seen by these three states and ensure the visibility constraint in space navigation.

We denote that the configuration grid is a representation of the configuration space. In the configuration grid starting from any location to attend another one, cells are thus belonging to reachable or unreachable path. Note that the set of reachable cells is a subset of the set of free configuration cells, the set of unreachable cell is a subset of the set of occupied configuration cells. By selecting a goal that lies within reachable space, we ensure that it will not be in collision and it exists some "feasible path" such that the goal is reached in the environment. Having determined the

reachability space, the algorithm works and operates on the reachability grid . This one specifies at the end the target area. The detection of the three states is done by the different color of pixels of those belonging to the area obstacle. Generally, the detected different colors of pixels have the same luminous intensity for every free path (a less difference). The other color neighbors are belonging to obstacle area. This detection is based on the game of every detected color of pixel. We separate between the set of luminous intensity of free path of pixels with those belonging to the set of luminous intensity of obstacle and unknown area. This separation is very useful to get a meaning of segmentation.

A grid of (i X j) dimension of free path is denoted by "***X***," an occupy grid of (l X k) is denoted by "***Y*** " . An obstacle is collection of hazardous cells in the "***Y***" ***grid*** **.**A path from start cell "C" to destination cell "D" that the detected color of "***X***" does not interest any detected color "***Y***". the path is said to be monotone of free cells "***X***" with respect to i-coordinates if no lines parallel to k-axis cross the j-axis (see figure 1).

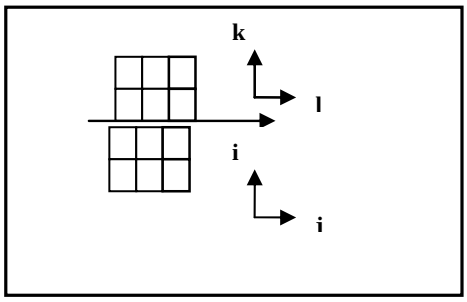

Fig. 1 : an example of no intersecting in unknown environment

The proposed algorithm here relies on number of cells and iterates, as follows :

1) i by j grid, start cell a in the grid.
2)Detect free destination in the grid (free cells).
3)Detect the collection of cells in the grid corresponding to obstacle area (hazardous occupied cells area) and unknown cells.
4)A path from "C" to "D" such that the total of neighboring cells are detected free.
5)If the collection of free cells is continuous, detect all neighboring on the same destination until the target is reached.
6)If the collection of free cells is discontinuous, change the direction and continue on another free continuous collection of cells.

To maintain the idea ; we have created several environments which contain many obstacles. The search area (environment) is divided into square grids. Each item in the array represents one of the squares on the grid, and its status is recorded as walkable or unwalkable area (obstacle).

The robot can identify three colors inside our environment: dark, yellow and green. The dark color is interpreted as an obstacle area; whereas the yellow color represents the free trajectory to attend the given target, and the green color refers to the area target (this is the first part of the main project to be after developed more, i.e., we start by this principle to give after more intelligence to our AMR). The robot starts from any position then it must move by sensing and avoiding the obstacles. The trajectory is designed in form of a grid-map, when it moves it must verify the adjacent case by avoiding the obstacle that can meet to reach the target. We use an algorithm containing the information about the target position, and the robot will move accordingly.

3.2 The proposed GA approach

A genetic algorithm for an optimization problem consists of two major components. First, GA maintains a population of individual corresponds to a candidate solution and the population is a collection of such potential solutions. In GA , an individual is commonly represented by a binary string the mapping between solutions and binary strings is called a "coding". GA has been theoretically and empirically proven to provide robust search capabilities in complex spaces offering a valid approach to problems requiring efficient and effective searching[7].

Before the GA search starts, candidates of solution are represented as binary bit strings and are prepared. This is called a population . A candidate is called a chromosome (in our case: the path is a "chromosome" and positions are the "genes"). Also, an evolution function, called fitness function, needs to be defined for a problem to be solved in order to evaluate chromosome. As fitness function, we should define distance for each chromosome to give an evaluation function. This evaluation is the goal of the GA search and goes as follows: two (02) chromosomes are chosen randomly from a population are mated and they go through operations like the crossover to yield better chromosomes for next generations. This is repeated until about twelve populations with new chromosomes. To determine execution of the GA, we must specify a stopping criterion, in our case, it could be determine, by a probabilistic function: as we have four chromosomes and we choose randomly two chromosomes, we have used the combination given by the theorem in the probability: 2^n -4 (here n is the number of the all paths), we subtract with number four because it is impossible to combine and to compare one path with itself. The crossover is the comparison operator (see Fig.2). Therefore, after several generations of GA search (The problem of mutation, see Fig.3), relatively low fitness of chromosomes remain in a population and some of them

are chosen as the solution of the problem (the most preferable path.

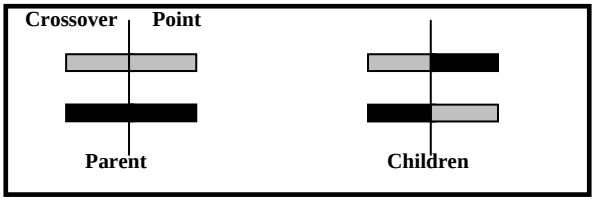

Fig.2 : Example of Crossover on singlepoint

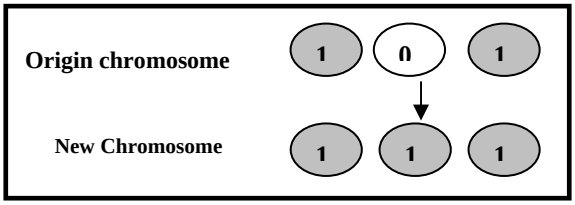

Fig. 3: Example of Mutation in the second bit.

3.3 Path planning based on GA

First, we need to choose a coding which maps a path from start cell "C" to destination cell "D" into a binary string . We can represent an arbitrary path by using a binary string of "variable length" . To simplify the problem, we assume that a path from C to D is either i-monotone or j-monotone (but not necessarily both). Obviously, not all paths are monotone. The path can be represented by a column-wise (or row-wise) sequence of (i-1,j-1) pairs of direction and distance such that each pair corresponds to each column (or each row, respectively) . Thus, the path can encode into a binary whose length is proportional to j and fixed.The first bit B1 indicates that path is x-monotone if B1 =1, and is y-monotone if B1=0. A block of (n+1) bits represents distance and direction on each column or row, where (n+1) are pairs of distance and direction. In case of B1 =1 : the first 2 bits of a block denote the direction ;e.g., 00(vertical), 01 (diagonal), 10 (horizontal). In the case of B1=0; we denote 00 (vertical), 10 (diagonal) and 01 (horizontal) (see figure 4). The aim of this technique is clearly obvious when we see the figure 5 (the sub-procedure for obstacle avoidance). The other bits of the block denote the distance is denoted by 1 if free, o if it is belonging to obstacle area.

The population size is computed by $2^{(n+1)}$ bits. The likelihood of optimality which is the estimated probability of finding an optimal solution within g generations computed by : ($2^{(n)}/\ 2^{(g+1)}$). The mutation rate is a=0,1 crossover rate b=0,9 and win rate w=(1-a)---(1-b).this implies that GA can find an optimal solution with high probability if it is executed respectively and to a realized and b realized.

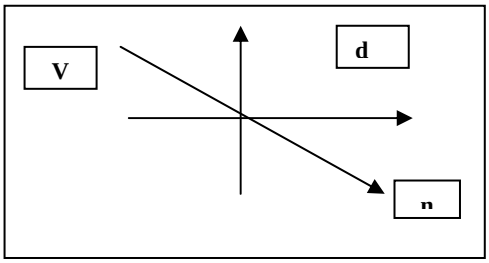

Fig.4: an example of pairs of distance and direction

In order to evaluate, the average performance of our GA over various environments, we observed simulation of the GA for 16 environments. We can change the position of obstacles so we get other different environments. These environments were randomly generated .To find a new optimal path after insertion of deletion of an obstacle , we measure the number of generations of candidates. The coding of GA is to affect label 0 for free cell and 1 for hazardous cell. This way of work is very useful later if the substring is inherited to new generations by genetic operators. We generate (04) four paths and we detect the best one to be optimized between them. The fourth are shown in the Fig.5 and Fig.6. These fourth paths are called sub optimal paths. Every suboptimal is presented by its fitness function (number of detected pixels of free path) and is quite different from each other. Hence, a mobile robot detects unknown hazardous obstacle on the path. The generation of several paths gives at the end the best optimal path with low fitness function (e.g. the shortest path).

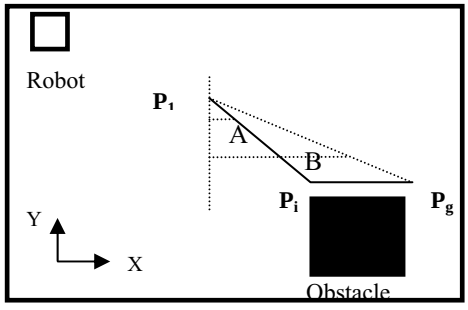

Fig. 5: Vehicle obstacle avoidancemode

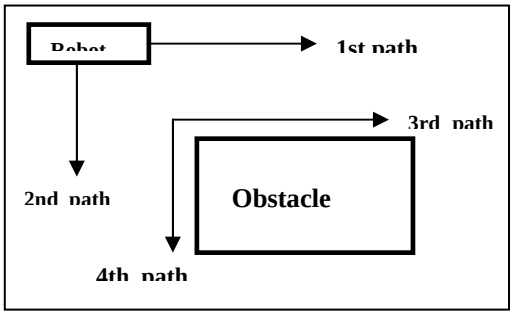

Fig. 6: an example of generation of the fourth Paths

4 A VLSI Path Planning of AMR

The GA principle and work are tested in more unknown environments. The creation starts with empty environment (no intelligent navigation) and finishes until we get the high order of dispositions (e.g. the maximum number of obstacles). The operation examines the crossover and mutation operators and various parameters values for each operator. Simulation results in 16 or more unknown environments shown that after 2^4 generations the optimal path is gotten (4 is number of paths, 2 : is two paths). The parent (02 paths) are combined together that crossover operator is applied regarding to their fitness functions) (shortest path) is realized, then, the short one is stored with low fitness function (low number of pixels). We choose between two parents (two paths) and we combine between them (regarding their fitness function) until we get the best one suitable path to be taken into account to navigate intelligibly in unknown environment.

4 Digital Implementation Of GA

Configurable hardware is an approach for realizing optimal performance by tailoring its architecture to the characteristics of a given problem. FPGAs Implementation of *GA* is very attractive because of the high flexibility that can be achieved through the reprogramability nature of these circuits[9]. The complexity of VLSI circuits is being more and more complexes. Nowadays, the key of the art design is focused around high level synthesis which is a top down design methodology, that transform an abstract level such as the VHDL language (acronym for Very High Speed Integrated Circuits Hardware Description Language) into a physical implementation level[5,6,8,9]. In addition, the synthesis tools allow designers to realize the mainly reasons: the need to get a correctly working systems at the first time, technology independent design, design reusability, the ability to experiment with several alternatives of the design, and economic factors such as time to market. In this section, we present a new design methodology of GA based upon a VHDL description and using a synthesis tool Galileo. The result is a netlist ready for place and root using the XACT. The intended objective is to, realize architecture that taxes into account the parallelism, performance, flexibility and their relationship to silicon area. In this section, we discuss the possibility to permitting the mapping of an entire *GA* into a single Xilinx's FPGA.

4.1 Design methodology

The proposed design method for the *GA* implementation is illustrated in fig. 6 as a process to follow.This status is followed by the VHDL description of the navigation approach. Then the VHDL code is passed through the synthesis tool Galileo. The result is a netlist ready for place and root using the XACT tool. At this level, verification is required before final FPGA implementation. The simplified model of GA is presented in Fig.7. Thus, we can represent it in its hardware equivalent model, and the top view of architecture is represented in Fig.8.

4.2 VHDL Description of GA

Synthesis of this design is achieved by using the VHDL language with Register Transfer Logic (RTL) style description. The choice of VHDL comes from its emergence as an industry standard. The RTL style is used because it is well adapted for synthesis. The description of the *GA approach* begins by creating the components; the flexibility of the design is introduced by *generic* statement.

4.3 Synthesis and implementation

The described methodology has been used for FPGA using the synthesis tool Galileo. At this level, and depending on the target technology, which is in our case, the FPGA Xilinx XC4000 family, the synthesis tool proceeds to estimate area in term of CLBs (Configurable Logic Blocs). The table I presents the best synthesis results for *GA Approach* (given by the process). The output (XNF file) generated by the synthesis tool is passed trough the XACT place and tool. The resulting FPGA implementation of navigation approach of the whole architecture has been into a single FPGA mapped. Using the XACT, the netlist are placed and rooted and the area of configurable logic blocs is almost occupied by the architecture (the small juxtaposed squares).

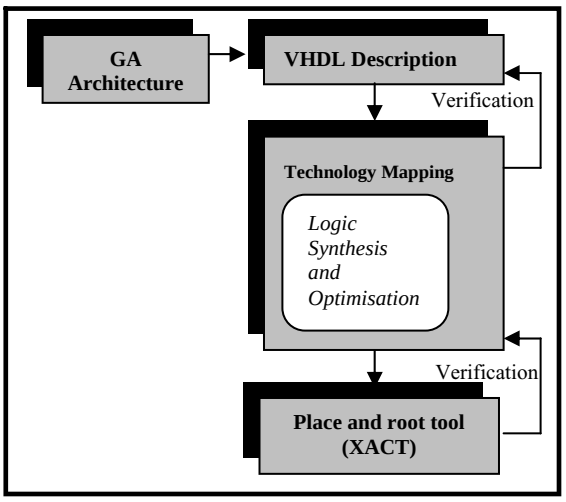

Fig.7: Design methodology of the *GA approach*

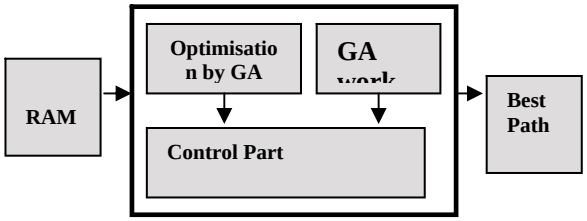

Fig.8 : Architecture of GA

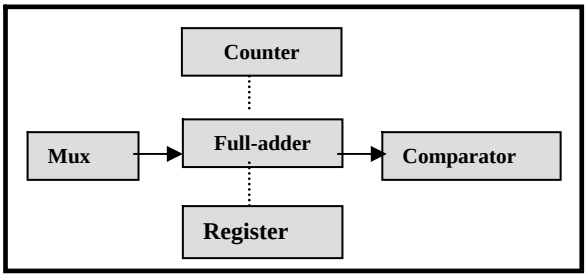

Fig.9: The Top view of GA structure

Design	***AREA (in term of CLBs)***
GA	***192***

Table I : Synthesis results

4 Conclusion

In this paper, we have presented a hardware implementation of navigation approach of an autonomous mobile robot in an unknown environment. The proposed approach can deal a wide number of environments. This system constitutes the knowledge bases of *GA approach* allowing to recognize situation of the target localization and obstacle avoidance, respectively. Also, a new design methodology of *GA* is introduced based upon a VHDL description and using a synthesis tool Galileo. The top down of this design based on logic synthesis has allowed a single chip FPGA implementation. The proposed VHDL description has the advantage of being generic and can be changed at the user demand. Depending on the final performance requirements, *GA* can be implemented using software tools supported by the standard microprocessor or by the hardware tools using FPGA technology, but the earlier offers more performance, flexibility and high speed processing into hardware, using the VLSI Technology FPGA (the concept of specification task and the reduce time of treatment). The primary results of the digital implementation show that the "intelligence-autonomy-economy" is achieved. However in the future, it is necessary to use a "micro-robot" in hostile environment and space exploration or other applications by using advanced micro-product control systems.

References

[1]D.Estrin, D.Culler, K.Pister, PERVASIVE Computing IEEE, 2002,pp. 59-69.

[2]T.Willeke, C.Kunz, I.Nourbakhsh, The Personal Rover Project : The comprehensive Design Of a dometic personal robot, Robotics and Autonomous Systems (4), Elsevier Science, 2003, pp.245-258.

[3]L . Moreno, E.A Puente, and M.A.Salichs, : World modelling and sensor data fusion in a non static environment : application to mobile robots, in *Proceeding International IFAC Conference Intelligent Components and Instruments for control Applications*, Malaga, Spain, 1992, pp.433-436.

[4]S.Florczyk, *Robot VisionVideo-based Indoor Exploration with Autonomous and Mobile Robots*, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim, 2005.

[5] R. Airiau, J.M Berger, V. Olive, *Circuit synthesis with VHD,* Kluwer Academic Publishers, 1994.

[6]S.D.Brown, R.J., J.Francis Rose, and Z.G.Vranesic : *Field-Programmable Gate Array*, *Kluwer Academic Publishers*, 1997.

[7]A.Gonzalez and R.Perez. : SLAVE : A genetic learning System Based on an Iterative Approach, *IEEE, Transaction on Fuzzy systems*, Vol 7, N.2, April 1999, pp.176-191.

[8]J.Legenhausen, R.Wade, C.Wilner, and B. Wilson,: *VHDL for programmable logic*, Addison- Wesley, 1996.

[9]O.Hachour and N.Mastorakis, IAV : A VHDL methodology for FPGA implementation, *WSEAS transaction on circuits and systems,* Issue5, Volume3,ISSN 1109-2734, pp.1091-1096.