A Motion Planning System for Mobile Robots

Adem TUNCER, Mehmet YILDIRIM, Kadir ERKAN
Networked Control Systems Laboratory, Kocaeli University, Umuttepe 41380, Turkey
adem.tuncer@kocaeli.edu.tr, myildirim@kocaeli.edu.tr, erkan@kocaeli.edu.tr

***Abstract*—In this paper, a motion planning system for a mobile robot is proposed. Path planning tries to find a feasible path for mobile robots to move from a starting node to a target node in an environment with obstacles. A genetic algorithm is used to generate an optimal path by taking the advantage of its strong optimization ability. Mobile robot, obstacle and target localizations are realized by means of camera and image processing. A graphical user interface (GUI) is designed for the motion planning system that allows the user to interact with the robot system and to observe the robot environment. All the software components of the system are written in MATLAB that provides to use non-predefined accessories rather than the robot firmware has, to avoid confusing in C++ libraries of robot's proprietary software, to control the robot in detail and not to re-compile the programs frequently in real-time dynamic operations.**

***Index Terms*—genetic algorithm, mobile robot, motion planning.**

I. Introduction

Recently, interest of researchers on autonomous vehicles increases with technological developments. There are many studies in the literature about autonomous vehicles. One of the main subjects on autonomous vehicle is path planning. Path planning tries to find a feasible path for mobile robots to move from a starting node to a target node in an environment with obstacles [1].

The path planning environment can be either static or dynamic. In the static environment, the whole solution must be found before starting execution. However, for the dynamic or partially observable environments re-planning are required frequently in each step and it needs more planning update time.

There are so many methods that have been developed to overcome the path planning problem for mobile robots. Each method differs in their effectiveness depending on the type of application environment and each one of them has its own strengths and weaknesses. Compared to traditional search and optimization methods, such as calculus-based and enumerative strategies, the evolutionary algorithms are robust, global and generally more straightforward to apply in situations where there is little or no prior knowledge about the problem to solve [2].

In the last decade, genetic algorithms have been widely used to generate the optimal path by taking the advantage of its strong optimization ability [3]. Genetic algorithms have been recognized as one of the most robust search techniques for complex and ill-behaved objective functions. The basic characteristic that makes the GA attractive in developing near-optimal solutions is that they are inherently parallel search techniques [4, 5]. They can search all working environment simultaneously in a parallel manner and so they can reach a better solution more quickly.

The Pioneer P3-DX of Mobile Robots is a popular research robot that is programmable and suitable for classroom and laboratory use. So many researchers have used this robot in their study [6,7,8,9]. ARIA is an object-oriented robot control application-programming interface for intelligent robots of Mobile Robots. Programmers working with ARIA should be familiar with using typical C++ concepts, including using classes and objects with simple inheritance, pointers, memory management, the STL containers, and the compiling and linking process [10]. However, MATLAB, which is the one of popular academic software, is a high-level language and interactive environment to perform computationally intensive tasks faster than with traditional programming languages such as C, C++, and FORTRAN. With the MATLAB, a programmer can program and develop algorithms faster than with traditional languages because there is no need to perform low-level administrative tasks, such as declaring variables, specifying data types, and allocating memory. Moreover, MATLAB executes commands or groups of commands one at a time, without compiling and linking [11].

In this study, a motion planning system for a mobile robot is introduced. A camera and digital image processing is used for localization of the mobile robot itself, obstacles and the target. The Pioneer P3-DX of Mobile Robots is used for indoor applications. A genetic algorithm is used for collision-free path planning. Grid-based environment model, which is frequently used in indoor applications, is used as the motion area of mobile robot. A graphical user interface (GUI) is designed for the motion planning system that allows the user to interact with the robot system and to observe the robot environment. Including the image processing, path planning, and communication with the robot and the GUI, all the software components of the system is written in MATLAB instead of using C++ libraries of ARIA. Programming in MATLAB provides to use non-predefined accessories rather than the robot firmware has, to avoid confusing in C++ libraries, to control the robot in detail and not to re-compile the programs frequently in real-time dynamic operations.

II. Motion Planning System

ARIA is an object-oriented robot control application that has a programming library (SDK) for C++ programmers who want to access their P3-DX platform and accessories. However, the robot firmware does not perform any high-level robotic tasks. Rather, it is the job of an intelligent client running on a connected PC to perform this application-level robotic control strategies and tasks, such as

This work was supported by Kocaeli University's Scientific Research Projects and No. 2010/97.

Digital Object Identifier 10.4316/AECE.2012.01010

obstacle detection and avoidance, localization, mapping, intelligent navigation and camera control [12].

Indoor motion planning of mobile robot deals with the issues of localization, path planning and the motion control. Fig. 1 shows the motion planning system used in this study. Localization is the determination of the positions of mobile robot, obstacles and the target. Generally positions are given by a user or identified by a camera. In the case of real-time and dynamic working, such as moving obstacles or targets are used in the environment, localization by means of a camera should be preferred.

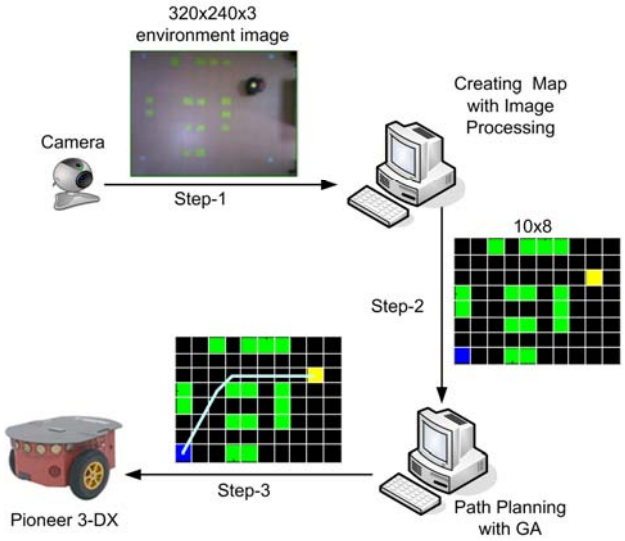

Figure 1. Proposed motion planning system for mobile robot

Because real-time and dynamic applications are made in this study, a camera and image processing techniques are used for localization. The camera is mounted on the ceiling and it sends the real-time images of the environment to a computer. Each object is labeled with a different color on it; green for the obstacles, blue for the target and yellow for the mobile robot. In order to determine the heading angle of the mobile robot, it is also labeled with a red color. Direction of the line that connects the centers of yellow and red colored circles on the robot equals to the heading angle of it.

Image processing program running on the computer, which is written in MATLAB, takes the instantaneous images from the camera, as shown on the upper left image of Fig. 2. It identifies the colors on the objects and the color of the base. This process is sensitive to the homogeneity of light and the reflections in the environment. The program determines the coordinates of each object which has a color different from the base color, as shown in the upper right image of the figure. Coordinate determination is a complicated problem, because the real environment and its image do not overlap due to the direction of the camera. Therefore, coordinate transformation should be done by using corner coordinates of the real environment and the image. At the last, image processing program matches the coordinates of colors with a grid based map, as shown on the lower right image. In this way, localization is realized and the map is produced.

The second step of the motion planning system shown in Fig. 1 is to send the map to the path planning process. In this study, genetic algorithm is used to determine the path which mobile robot goes through it. Path planning with the genetic algorithm is explained in detail in the following section.

In the last step, in order to control the robot platform, a client program, which is written in MATLAB, sends command packets through the robot connection. This can be done using direct commands. Direct commands consist of two-byte packet header, one-byte byte count, one-byte command number, one-byte argument type, n-byte argument and two-byte checksum, as defined by the robot's operating system ARCOS [13]. The direct command method allows sending any unusual or special command directly to the robot platform, without any intervening processing.

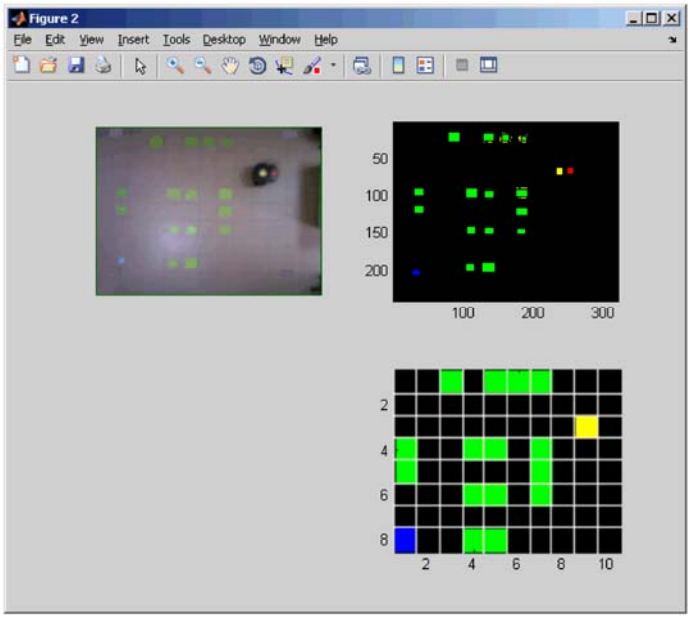

Figure 2. Map production by means of image processing

III. PATH PLANNING WITH GENETIC ALGORITHMS

GA is a parallel and global search technique that emulates natural genetic operators. Because it simultaneously evaluates many points in the parameter space, it is more likely to converge toward the global optimum. It is not necessary that the search space to be differentiable or continuous. GA applies operators inspired by the mechanics of natural selection in a population of gene string which is encoding the parameter space. At each generation, it explores different areas of the search space, and then directs the search to regions where there is a high probability of finding a better solution.

A. Representation of Environment and Chromosome

Many path planning methods use a grid-based model to represent the environment space. It has been determined that calculation of distance and representation of obstacle is easier with grid-based representation. The grid-based environment space is represented in two ways, by the way of coordinate plane [4,14,15] or by the way of orderly numbered grids [1,16,17]. Coordinates can be represented with both the binary or decimal numbers. Fig. 3 shows the binary, decimal and orderly numbered grid representation of the path planning environment. The shadowed grid on the environment shows the infeasible obstacle area and blank grid shows the feasible area where mobile robots can move freely.

A chromosome represents a candidate solution [18] for the path planning problem. A chromosome or a path consists of a starting node, a target node and the hopping nodes which mobile robot across to them. These nodes or steps in the path are called as genes of the chromosome. Different coding methods are used to create chromosomes (Fig. 4), depending on the representation method of the environment. Binary coded string method [4,19,20] is used in general, however decimal coded string method is also used [1,16,17] and it is thought as to be more flexible. Decimal coding needs less computational overhead in time and space.

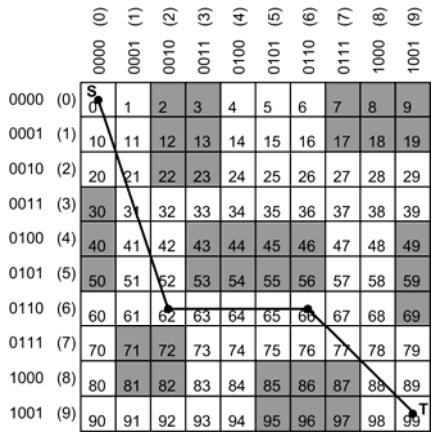

Figure 3. Binary, decimal and orderly numbered representations of the environment

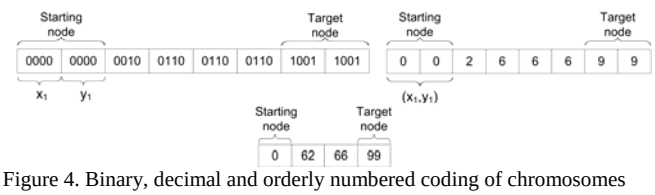

Figure 4. Binary, decimal and orderly numbered coding of chromosomes

B. Initialization of Population

The initial population is generally generated randomly. Some of the generated chromosomes may include infeasible paths which intersect an obstacle. An optimal or near optimal solution can be found by genetic operators, even though the initial population includes infeasible paths. However, this process reduces the search capability of the algorithm and increases the time to find an optimal solution. Furthermore, crossover of two infeasible chromosomes may generate new infeasible paths. In order to solve this problem, each chromosome must be checked whether it intersects an obstacle, when generating the initial population. If it is, intersected gene of the chromosome is changed randomly with feasible one. It is determined that starting the genetic algorithm with the feasible initial population is fairly beneficial, as also stated in the [16,17].

C. Fitness Function and Selection

The purpose of the path planning problem is to find an optimal path between a starting and a target node. Optimal path may be the shortest [21], the least time and energy requiring path to trip on it. Generally, in the path planning problems, the objective function is considered as a shortest path. After moving from previous node to the next node, *x-y* coordinates and heading angle of mobile robot changes as shown in Fig. 5. In this study, the objective function value of a chromosome used in genetic algorithm is given in Equation (1) and (2).

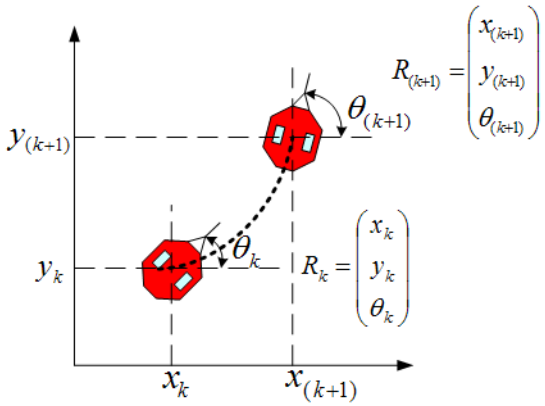

Figure 5. Change in mobile robot position

$$f = \begin{cases} \sum_{k=1}^{n-1} d(p_k, p_{k+1}) & \text{, for feasible paths} \\ \sum_{k=1}^{n-1} d(p_k, p_{k+1}) + penalty & \text{, for infeasible paths} \end{cases} \quad (1)$$

$$d(p_k, p_{k+1}) = \sqrt{(x_{(k+1)} - x_k)^2 + (y_{(k+1)} - y_k)^2} \quad (2)$$

Where; f is the fitness function, p_k is the k^{th} gene (node) of the chromosome, n is the length of the chromosome, d is the distance between two nodes, x_k and y_k are robot's current horizontal and vertical positions, x_{k+1} and y_{k+1} are robot's next horizontal and vertical positions. The direction of the robot is determined by Equation (3).

$$\alpha = \tan^{-1} \frac{(y_{(k+1)} - y_k)}{(x_{(k+1)} - x_k)} \quad (3)$$

Objective function value is defined as the sum of distances between each node in a path. If there is an obstacle in the direction of the robot, a penalty is added to objective function value. The penalty value should be greater than the maximum path length on the environment. In order to find an optimal path, the algorithm searches for a chromosome whose penalty is eliminated.

D. Crossover and Mutation Operators

Generally, crossover combines the features of two parent chromosomes to form two offspring. Single-point crossover operator is used in this study (see Fig. 6). The genes of two chromosomes after the crossover point are swapped. All candidate chromosomes in the population are subjected to the random mutation after the crossover operation. This is a random bit-wise binary complement operation or a random small change in a gene, depends on the coding of chromosomes. Mutation is applied uniformly to all genes of all individuals in the population with a probability of mutation rate. The mutation operation expands the search space to regions that may not be close to the current population, thus ensuring a global search [17]. Mutation operation increases the diversity of the population and avoids the premature convergence.

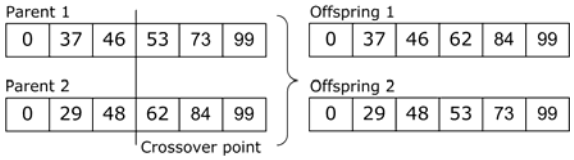

Figure 6. Single-point crossover

IV. Graphical User Interface For Motion Planning System

Graphical user interface (GUI) allows users to interact with electronic devices with images rather than text commands. A GUI represents the information and actions available to a user through graphical icons and visual indicators as opposed to text-based command line interfaces. In this study, a GUI is designed for coordinating and observing the proposed motion planning system.

The GUI screen, which is shown in Fig. 7, includes a menu bar on the top of the screen. When *Environment Settings* is selected in the menu, environment settings section appears at the left and an environment section appears at the right of GUI window. The environment section shows a grid based map on which there are the starting node (location of the robot), the obstacle nodes and the target node. In the settings section, a user can change the dimensions of the environment by means of *x* and *y* parameters. The locations of the starting, obstacle and target nodes can also be set by selecting the node type first and then mouse clicking on the map. Moreover, real time environment, the locations of the robot, the obstacles and the target can be taken from the camera by selecting on the settings section. The environment is cleared by clicking on the *Clear* button.

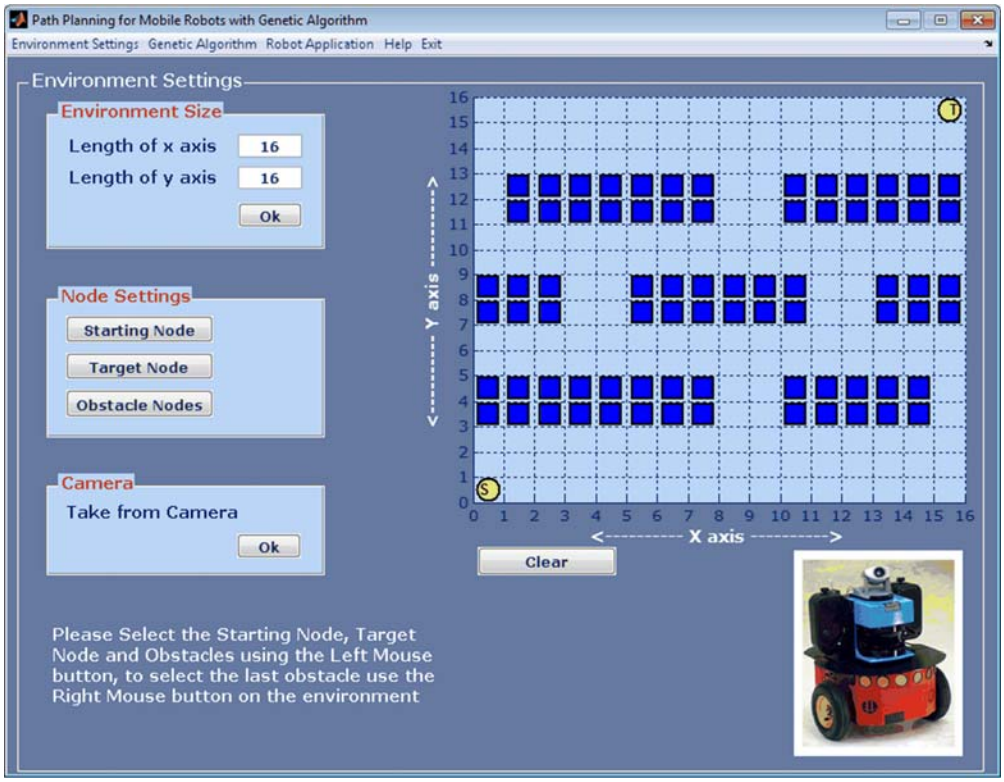

Figure 7. Environment settings on the GUI

When *Genetic Algorithm* is selected in the menu, genetic algorithm settings section appears at the left and the environment previously described appears at the right of GUI window (see Fig. 8). Genetic algorithm settings include the population size, the crossover and mutation rate, the number of maximum generation loop and the number of steps. Type of the chromosome coding is also selected in the genetic algorithm settings section. After clicking on the *Find Path* button, genetic algorithm starts to generations and

shows the determined path on the map, in each generation.

Upon completion of the generations, objective function value, nodes and the generation number of the best path are displayed on the *Simulation Results* section. *Show Evolution* demonstrates the objective function value versus generation number graph.

When *Robot Application* is selected in the menu, robot settings section appears at the left and the environment with the found path on it appears at the right of GUI window (see Fig. 9). Robot settings include the moving velocity, the rotating velocity and the heading angle of the mobile robot. The path to be sent to the robot is selected by clicking the radio buttons. User can prefer the path determined by the genetic algorithm or the path drawn by clicking on the grids on the map. Furthermore, without sending a path, the robot can be controlled manually by entering a value in the center box and clicking a direction button. *Left* and *Right* buttons rotate the robot by the degree of value entered in the center box. Forward and Backward buttons are used for moving the robot. In any case, sending a path to the robot or controlling the robot manually, the action taken by the robot is exhibited in the *Display* section, sequentially.

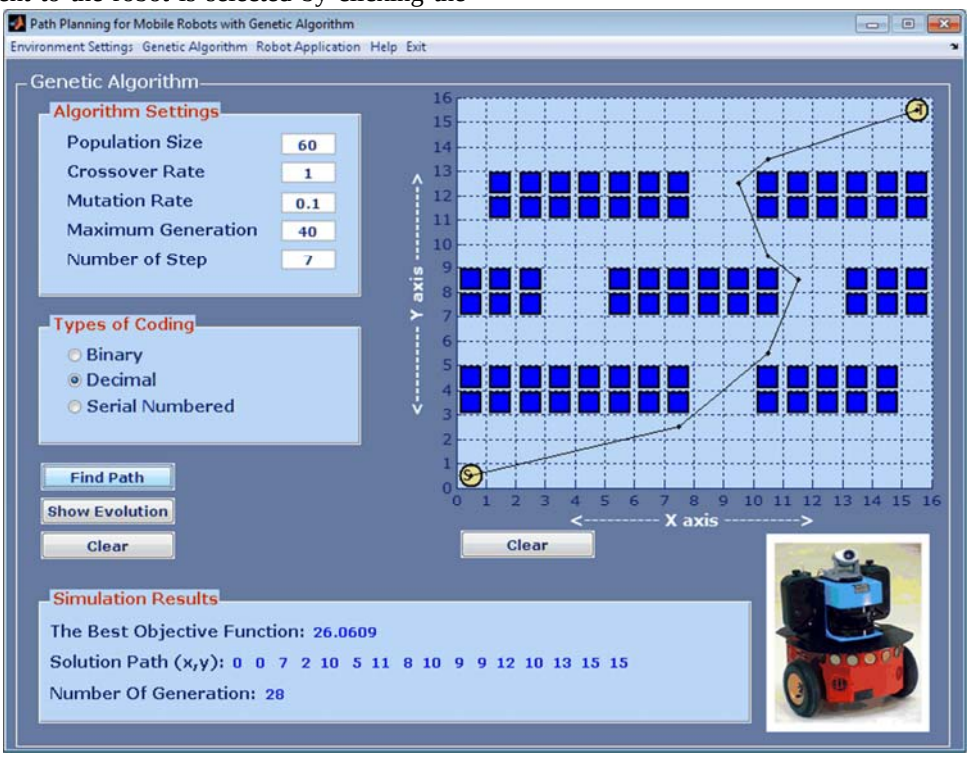

Figure 8. Genetic algorithm settings on the GUI

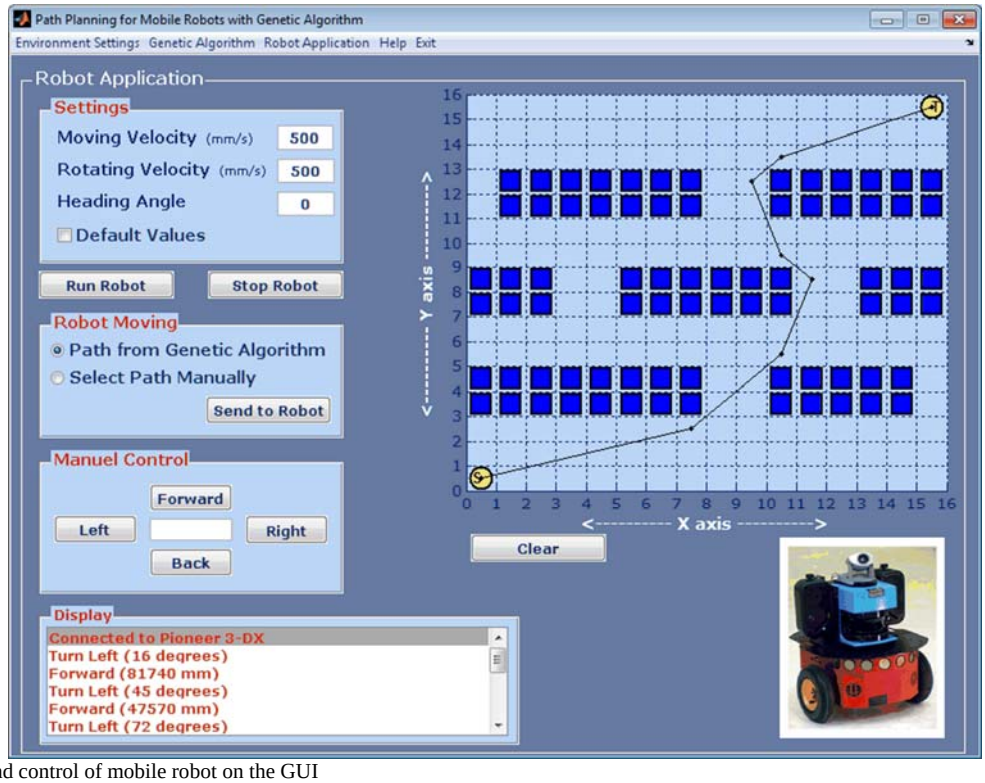

Figure 9. Settings and control of mobile robot on the GUI

V. CONCLUSION

In this study, a motion planning system for a mobile robot is proposed. Indoor motion planning of mobile robot deals with the issues of localization, path planning and the motion control. Because real-time and dynamic applications are made in this study, a camera and image processing techniques are used for localization. A genetic algorithm is used to find a feasible path for the mobile robot to move from a starting node to a target node in an environment with obstacles. In order to control the robot platform, a client program sends command packets through the robot connection by using direct commands. Furthermore, a graphical user interface is designed for the motion planning system that allows the user to interact with the robot system and to observe the robot environment.

All the software components of the system are written in MATLAB that provides to use non-predefined accessories rather than the robot firmware has, to avoid confusing in C++ libraries of robot's proprietary software, to control the robot in detail and not to re-compile the programs frequently in real-time dynamic operations.

REFERENCES

[1] Y. Hu and S. X. Yang, "A knowledge based genetic algorithm for path planning of a mobile robot," *Proceedings of the 2004 IEEE, Int. Conf. on Robotics & Automation*, Vol. 5, pp. 4350-4355, 2004. [Online].Available: http://dx.doi.org/10.1109/ROBOT.2004.1302402

[2] J. Tu and S. X. Yang, "Genetic algorithm based path planning for a mobile robot," *Robotics and Automation. Proceedings. ICRA '03. IEEE Int Conf. on*, Vol. 1, pp. 1221-1226, 2003. [Online]. Available: http://dx.doi.org/10.1109/ROBOT.2003.1241759

[3] I. Al-Taharwa, A. Sheta and M. Al-Weshah, "A mobile robot path planning using genetic algorithm in static environment," *J of Computer Science*, Vol. 4, pp. 341-344, 2008. [Online]. Available: http://dx.doi.org/10.3844/jcssp.2008.341.344

[4] A. Elshamli, H. A. Abdullah and S. Areibi, "Genetic algorithm for dynamic path planning," *Electrical and Computer Engineering, Canadian Conference on*, Vol. 2, pp. 677-680, 2004. [Online]. Available: http://dx.doi.org/10.1109/CCECE.2004.1345203

[5] S. M. H. Nabavi, A. Kazemi and M. A. S. Masoum, "Social welfare improvement by TCSC using real code based genetic algorithm in double-sided auction market," *Advances in Electrical and Computer Engineering*, Vol. 11, pp. 99-106, 2011. [Online]. Available: http://dx.doi.org/10.4316/aece.2011.02016

[6] H. J. Chang, C. S. G. Lee, Yung-Hsiang Lu, Y. C. Hu, "P-SLAM: simultaneous localization and mapping with environmental-structure prediction," *IEEE Transactions on Robotics*, Vol. 23, No. 2, pp.281-293,2007. [Online]. Available:http://dx.doi.org/10.1109/TRO.2007.892230

[7] Y. Zhuang, M. Gu, W. Wang and H. Yu, "Multi-robot cooperative localization based on autonomous motion state estimation and laser data interaction," *Science China Information Sciences*, Vol. 53, No. 11, pp. 2240-2250, 2010. [Online]. Available: http://dx.doi.org/10.1007/s11432-010-4096-4

[8] H. Teimoori and A. V. Savkin, "Equiangular navigation and guidance of a wheeled mobile robot based on range-only measurements," *Robotics and Autonomous Systems*, Vol. 58, pp. 203-215, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2009.09.004

[9] A. Yufka, O. Parlaktuna, "Performance comparison of bug algorithms for mobile robots," *5th International Advanced Technologies Symposium (IATS'09)*, Karabuk, Turkey, May 13-15, 2009.

[10] http://robots.mobilerobots.com/wiki/ARIA#Documentation

[11] http://www.mathworks.com/products/matlab/description2.html

[12] http://www.ai.rug.nl/vakinformatie/pas/content/Aria-manual/main.html# AriaPackage

[13] Pioneer 3 Operations Manual, Version 6, MobileRobots Inc., September 2010.

[14] T. W. Manikas, K. Ashenayi and R. L. Wainwright, "Genetic algorithms for autonomous robot navigation," *Instrumentation & Measurement Magazine, IEEE*, Vol. 10, pp. 26-31, 2007. [Online]. Available: http://dx.doi.org/ 10.1109/MIM.2007.4428579

[15] M. Naderan-Tahan and M. T. Manzuri-Shalmani, "Efficient and safe path planning for a mobile robot using genetic algorithm," *Evolutionary Computation, CEC '09. IEEE Congress on*, pp. 2091-2097, 2009. [Online]. Available: http://dx.doi.org/ 10.1109/CEC.2009.4983199

[16] Z. Yao and L. Ma, "A static environment-based path planning method by using genetic algorithm," *Computing, Control and Industrial Engineering (CCIE), 2010 International Conference on*, Vol. 2, pp. 405-407, 2010. [Online]. Available: http://dx.doi.org/ 10.1109/CCIE.2010.220

[17] Q. Li, W. Zhang, Y. Yin, Z. Wang and G. Liu, "An improved genetic algorithm of optimum path planning for mobile robots," *Intelligent Systems Design and Applications, ISDA '06. Sixth International Conference on*, Vol. 2, pp. 637-642, 2006. [Online]. Available: http://dx.doi.org/10.1109/ISDA.2006.253911

[18] E. Gelenbe, P. Liu and J. Laine, "Genetic algorithms for route discovery," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol. 36, pp. 1247-1254, 2006. [Online]. Available: http://dx.doi.org/10.1109/TSMCB.2006.873213

[19] K. Sugihara and J. Smith, "Genetic algorithms for adaptive motion planning of an autonomous mobil robot," *Computational Intelligence in Robotics and Automation, CIRA'97, Proceedings, IEEE International Symposium on*, pp. 138-143, 1997. [Online]. Available: http://dx.doi.org/10.1109/CIRA.1997.613850

[20] G. Nagib and W. Gharieb, "Path planning for a mobile robot using genetic algorithms," *Electrical, Electronic and Computer Engineering, ICEEC '04, International Conference on*, pp. 185-189, 2004. [Online]. Available: http://dx.doi.org/ 10.1109/ICEEC.2004.1374415

[21] E. Masehian and D. Sedighizadeh, "Multi-objective PSO- and NPSO-based algorithms for robot path planning," *Advances in Electrical and Computer Engineering*, Vol. 10, pp. 69-76, 2010. [Online]. Available: http://dx.doi.org/10.4316/aece.2010.04011