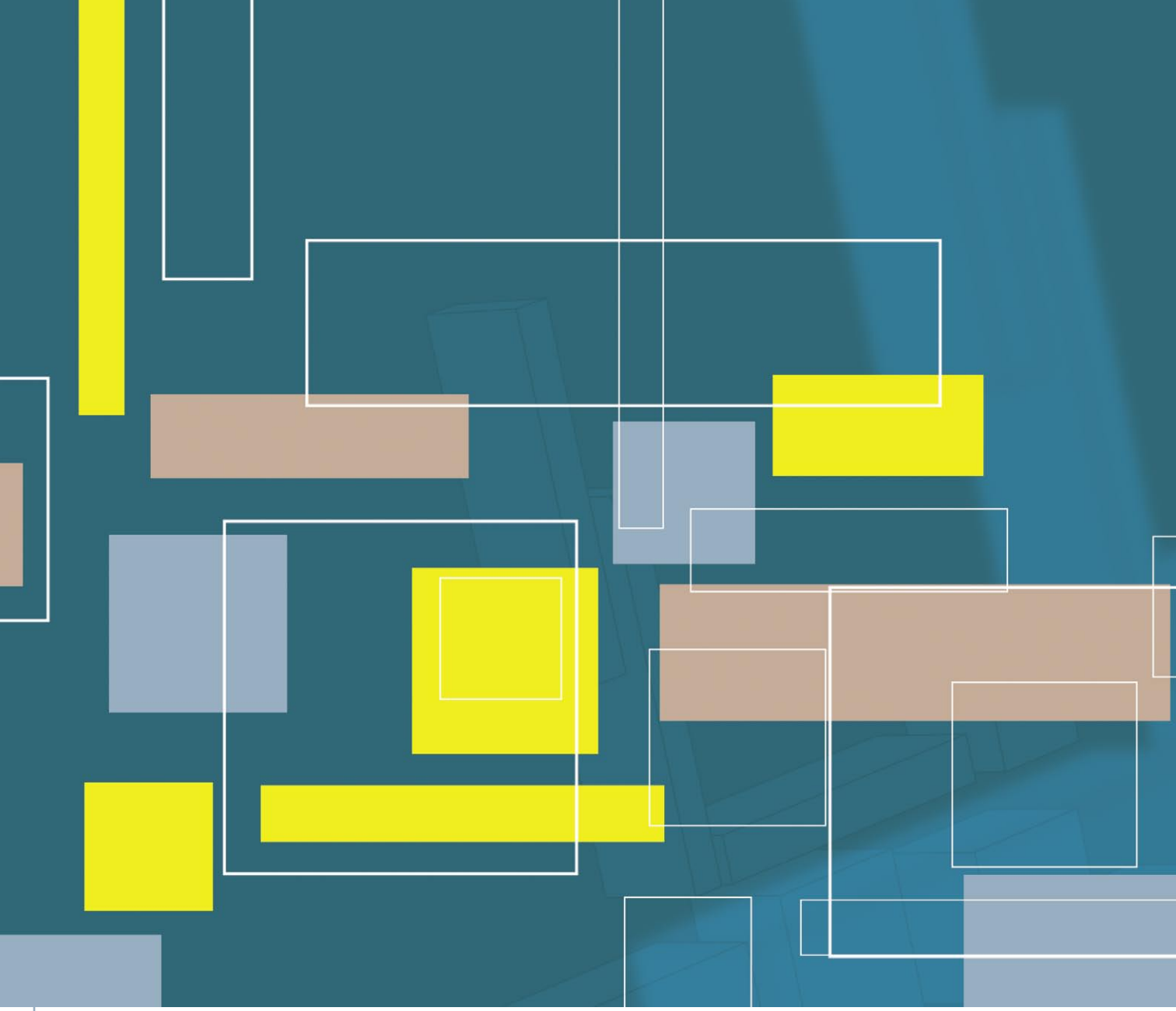


Genetic Algorithms for Autonomous Robot Navigation

Theodore W. Manikas, Kaveh Ashenayi, and Roger L. Wainwright

Engineers and scientists use instrumentation and measurement equipment to obtain information for specific environments, such as temperature and pressure. This task can be performed manually using portable gauges. However, there are many instances in which this approach may be impractical; when gathering data from remote sites or from potentially hostile environments. In these applications, autonomous navigation methods allow a mobile robot to explore an environment independent of human presence or intervention. The mobile robot contains the measurement

1094-6969/07/$25.00©2007IEEE

Note that the switching points are part of the evolutionary process and vary from chromosome to chromosome.

of genetic algorithms is an example of machine intelligence applications to modern robot navigation. Genetic algorithms are heuristic optimization methods, which have mechanisms analogous to biological evolution. This article provides initial insight of autonomous navigation for mobile robots, a description of the sensors used to detect obstacles and a description of the genetic algorithms used for path planning.

Autonomous Robot Navigation

Autonomous navigation implies that a robot must decide how to travel through a given environment [2]. While basic information may be available to the robot about the navigation area boundaries, unknown obstacles may exist within the navigation area. This is called an uncertain environment: the robot must be able to detect and maneuver around these obstacles to reach its target point.

The navigation environment in which the robot and obstacles both exist is called the "world space." A path between the starting and target points that avoids collisions with obstacles is said to be "feasible." Robot navigation methods need to solve the path-planning problem, which is to generate a "feasible" path and optimize this path with respect to specific criteria.

Many path-planning methods use a grid-based model to represent the world space (Figure 1). In this model, the world space is partitioned into grids, where the size of each grid depends on the particular specifications of the autonomous

device and records the data then either transmits it or brings it back to the operator.

Sensors are required for the robot to detect obstacles in the navigation environment, and machine intelligence is required for the robot to plan a path around these obstacles [1]. The use

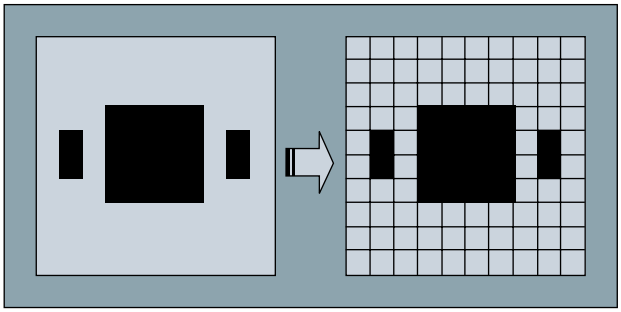


Fig. 1. World space example with grid system representation.

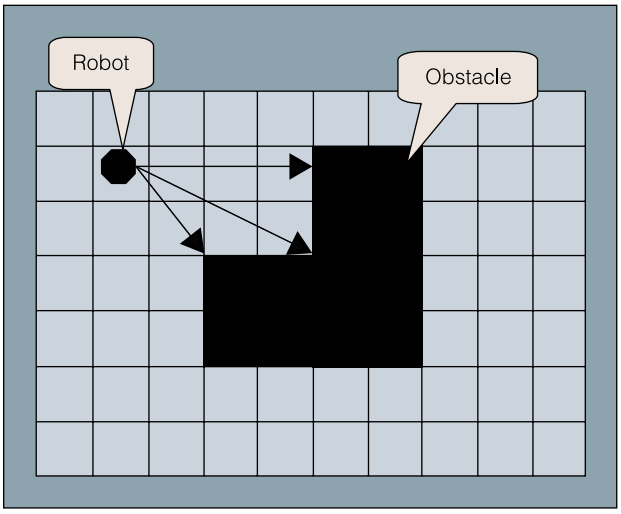


Fig. 2. Example of a robot sensing obstacles in the navigation environment.

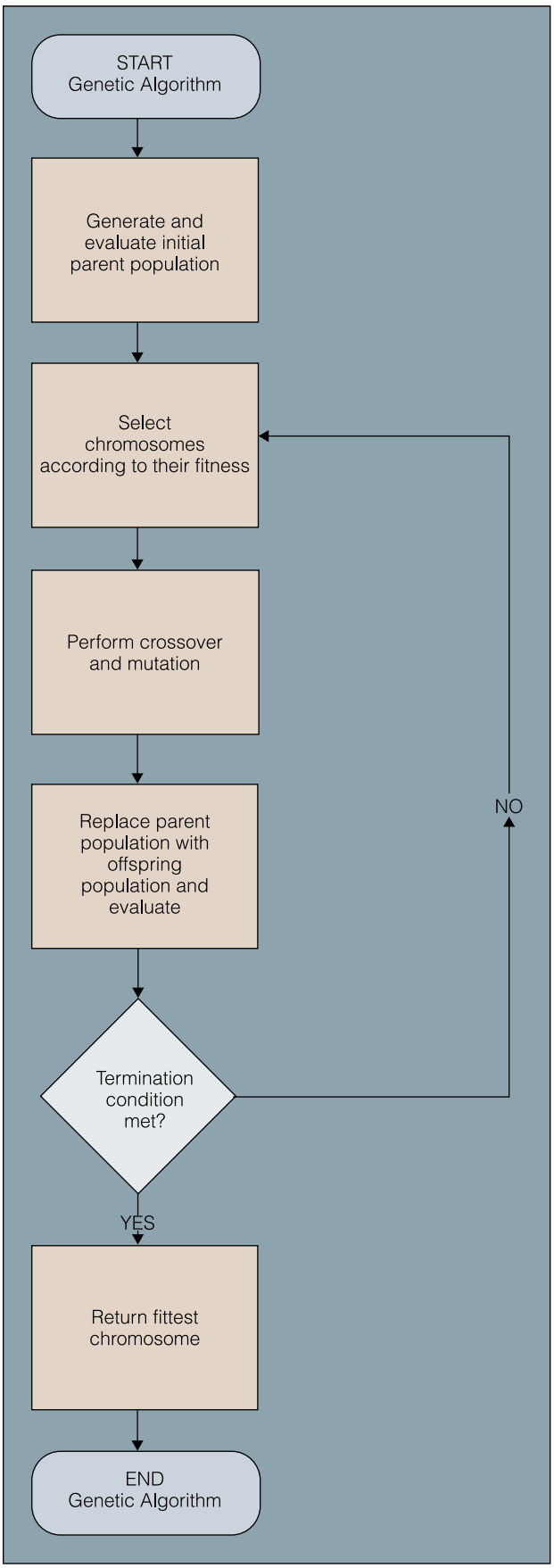


Fig. 3. Flow chart of a basic genetic algorithm.

vehicle and the navigation area. An obstacle may occupy one or more grids, depending on the size of the obstacle relative to the size of the vehicle. Squares, fully or partly blocked by obstacles, are identified as occupied cells when detected in the search space. The robot can move on all free cells, where the center of the robot moves along an imaginary line from the center of one cell to the center of another cell, either in a vertical or horizontal direction.

Sensing the Navigation Environment

Autonomous navigation requires the robot to interact with its environment and to adapt to changing conditions. This means that sensors must be mounted on the robot to detect and locate obstacles in the navigation space (Figure 2). Types of sensors that are commonly used in robot navigation include contact, orientation, proximity (capacitive and magnetic), imaging, ultrasonic and infrared sensors, as well as laser range finders and cameras.

Since each sensor type has its strengths and weaknesses (suitable for a different application), an autonomous robot system will typically combine various sensor types for effective obstacle mapping. For example, proximity detectors are used to identify items in the proximity (short range) of the robot. Orientation sensors such as gyroscopes and global positioning systems (GPS) are used to provide the robot with data on its orientation and direction.

Imaging sensors such as IR cameras and Omni cameras are used to provide visual data in visible as well as the IR frequency ranges (to detect temperature differentials). This information, in combination with other sensors, can be used to help the robot to avoid obstacles such as potholes and bodies of water in an unknown terrain. The imaging sensors can be used for medium- and long-range detection.

Ultrasonic sensors are widely used for obstacle detection as a result of their simplicity and relatively low cost [3]. An ultrasonic sensor system emits a sonar signal and receives its echo from an object. The object distance is determined by measuring the time difference between the signal emission and the echo reception.

While ultrasonic sensors are useful for measuring the distance of an obstacle from the robot, they are less successful at detecting object profiles (such as edges) because of sonar wave reflections. To overcome this limitation, IR sensors may be used in addition to ultrasonic sensors [4]. An infrared sensor system emits a pulse of infrared light that is reflected back from an object. The angle of this reflection that is received at the sensor yields information about the object distance and profile. One disadvantage of IR sen-

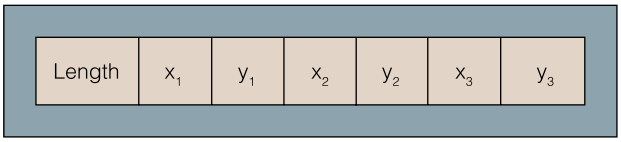


Fig. 4. Burchardt's path chromosome structure [8].

sors is that they can become severely corrupted by other light sources. Therefore, a robot navigation system should typically contain several types of sensors, as each sensor has its advantages and disadvantages.

The navigation environment in which the robot and obstacles both exist is called the "world space."

Planning Navigation Paths Using Sensor Information

The path-planning problem is often divided into global and local planning approaches. Global planning optimizes the overall traveled distance, while local planning determines how to navigate around obstacles. The global approach constructs a world model based on sensory information and uses this model to plan a global path. Since the construction and maintenance of a global map may become computationally complex for a robot, the local approach is often used to plan paths around specific obstacles. Local path planning uses the obstacle location and profile information obtained from the sensors to determine a feasible navigation path for the robot. The path-planning problem has been shown to be NP (Nondeterministic Polynomial-time)-hard [5], which means that many of the traditional methods can become computationally intractable. Genetic algorithms have been shown to be effective in solving NP-hard problems; thus, they have often been used for local path planning in contemporary robot navigation algorithms [7].

Genetic algorithms are heuristic optimization methods, the mechanisms of which are analogous to biological evolution [6]. Figure 3 shows a flow chart that describes the basic operations of a genetic algorithm. The algorithm starts with a population of individuals (chromosomes). Each individual represents a possible solution for the given problem. For robot navigation, an individual may represent a path between the starting and target points. Each individual is assigned a fitness value, based on how well the individual meets the problem objectives. Using these fitness values, individuals are selected to be parents. These parents form new individuals, or offspring, via crossover and mutation. Parent selection, crossover, and mutation operations continue for numerous iterations (generations) until the algorithm converges to an optimal or near-optimal solution.

The dominant task in developing a genetic algorithm to solve a particular problem is the development of the chromosome structure and the operators that process this structure, such as the fitness function and crossover and mutation operators. For robot navigation, the goal is to determine a feasible path as quickly as possible. Therefore, the chromosome must be able to represent a valid path in the navigation space. In addition, the structures of the chromosome and fitness function must allow for efficient computation. These factors govern the development of genetic algorithm path planners.

Path Planning Using Genetic Algorithms

Various genetic algorithm methods have been developed to solve the path-planning problem for autonomous robot navigation. Using the grid-based model, the world space can be viewed as a set of (x,y) coordinate points, or as a set

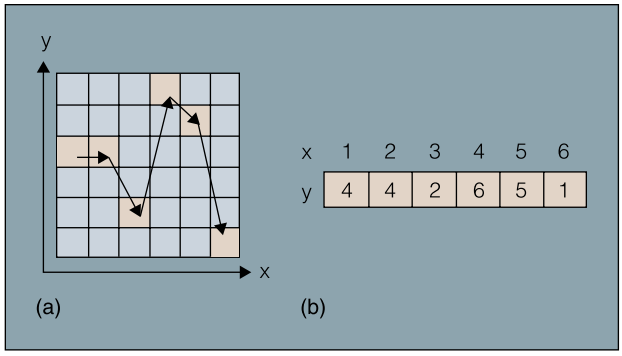


Fig. 5. (a) Sample path; (b) fixed-length chromosome representation.

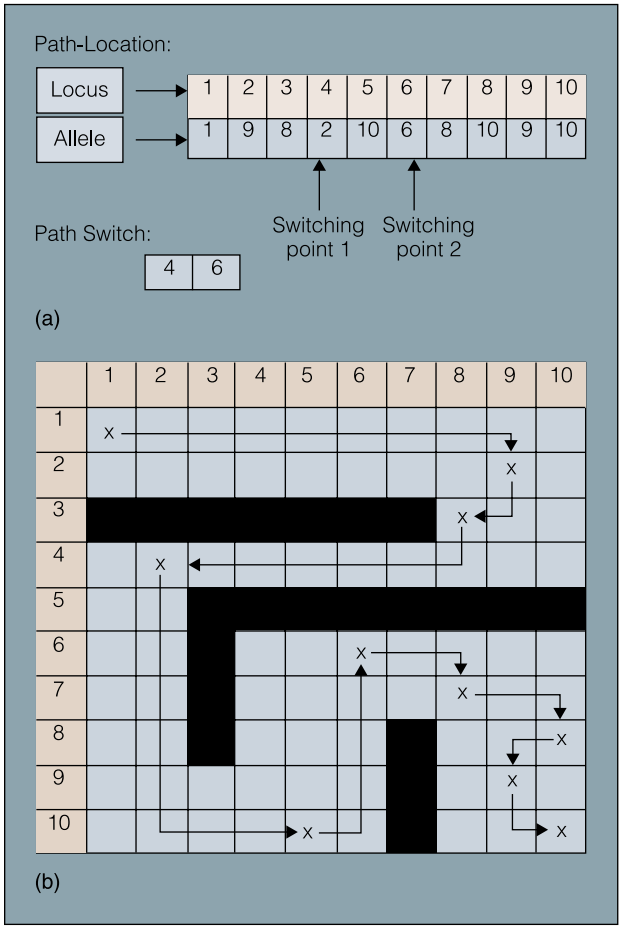


Fig. 6. (a) Chromosome with switching points; (b) resultant path.

of rows and columns. Often the grid is viewed as a square, in which the number of rows equals the number of columns [7]. The chromosome structure used by many genetic algorithm path planners is a value-encoded scheme: x and y coordinate information for the path points is contained in the chromosome structure. Figure 4 shows the chromosome structure used by Burchardt and Salomon [8]. The first gene of the chromosome contains the value of the chromosome length, which indicates the number of path points. Each subsequent pair of genes contains the (x,y) coordinates for each path point. The path fitness is based on both path length and feasibility, with a significant penalty for paths that collide with obstacles. This fitness function helps the genetic algorithm identify a collision-free path through the world space. Alvarez, Caiti, and Onken [7] use a similar representation, with (x,y,z) coordinates, since their genetic algorithm was developed for an autonomous undersea vehicle. Path fitness is determined by feasibility and the estimated energy required by the vehicle to travel this path.

Orientation sensors such as gyroscopes and global positioning systems (GPS) are used to provide the robot with data on its orientation and direction.

Encoding path points into a chromosome is an intuitive and straightforward approach. However, differences in path lengths may result in differences in the number of chromosome genes. For a genetic algorithm to function efficiently, it is desirable for all chromosomes to have the same number of genes, especially for crossover and mutation operations. Figure 5 shows an example of a fixed-length chromosome, based on the model of Geisler and Manikas [9]. Given a navigation environment that is modeled by N rows, a path in that environment is represented by a chromosome with N genes. Each gene position (locus) corresponds to an x-coordinate, while each gene value (allele) corresponds to a y-coordinate within that column. Figure 5(a) shows a path in a 6 × 6 navigation space, modeled by the chromosome shown in Figure 5(b). This chromosome represents a path that starts at point (1,4) and travels along intermediate points (2,4), (3,2), (4,6), and (5,5) to reach its target at point (6,1).

While this chromosome structure is easier to handle for a genetic algorithm, a main limitation of this structure is that it requires all paths to be x-monotone: $x_{i+1} > x_i$. Depending on the obstacle configuration in the world space, this restriction may not allow the genetic algorithm to find a feasible path. Sedighi et al. [10] address this limitation by allowing paths to switch between x-monotone and y-monotone ($y_{i+1} > y_i$) orientations. This is accomplished by adding "switching points" to the chromosome structure to identify where the path switches orientation.

Figure 6 shows an example for this model. The chromosome structure (Figure 6[a]) contains two parts: path-location and path-switch. The path-location part works similar to the way it does in the Geisler and Manikas model, except that it represents both x- and y-monotone subpaths within the main path. For x-monotone orientation, each locus represents a column index, while each allele represents a row within that column. For y-monotone orientation, each locus represents a row index, while each allele represents a column within that row. There are two switching points that identify the loci on the path-location part where the path orientation switches. Note that the switching points are part of the evolutionary process and vary from chromosome to chromosome. Up to two switching points are allowed. However, a given chromosome may evolve only one switch point, or perhaps none.

Figure 6(b) shows the navigation path that corresponds to the chromosome shown in Figure 6(a). Note that the world space origin is in the upper left corner, as per the specification of Sedighi et al. [10]. The path always starts at (1,1) with y-monotone orientation. The direction bit for locus 1 is 1, so the navigation direction to the next point is horizontal, then vertical. The next point is identified by locus 2 in the path-location chromosome, with an allele of 9. Since the orientation is y-monotone, this corresponds to row 2, column 9, or point (2,9) in the world space. The path continues through points (3,8) to (4,2).

The first switching point is at locus 4 of the path-location chromosome. Thus, at point (4,2) the orientation changes from y-monotone to x-monotone. The next point is identified by locus 5 in the path-location chromosome, with an allele of 10. Since the orientation is now x-monotone, this corresponds to column 5, row 10, or point (10,5) in the world space. The path continues to point (6,6), where the next switch point occurs.

The second switching point is a locus 6 of the path-location chromosome, so the path now changes back to a y-monotone orientation. This means that locus 7, allele 8 corresponds to row 7, column 8, or point (7,8) in the world space. The path continues through points (8,10) and (9,9) to the target point (10,10). As with the other genetic algorithm path planners, path feasibility is important when evaluating an individual. Additional goals are to minimize the total path length and the number of turns required by the robot.

Conclusion

This paper has provided an overview of applications of genetic algorithms to autonomous robot navigation. Many of these methods have been tested using simulated environments. The genetic algorithm optimization method has been previously shown to be effective in solving NP-hard problems such as path planning. However, these algorithms may take some time to converge to an optimal solution, which will affect the speed of robot navigation. After the genetic algorithm path planners have been verified using simulations,

these algorithms will need to be tested on actual robots and modified as necessary to ensure acceptable real-time navigation performance.

References

[1] K. Warwick and S.J. Nasuto, "Historical and current machine intelligence," IEEE Instrum. Meas. Mag., vol. 9, (no. 6), pp. 20–26, Dec 2006.

[2] J.-C. Latombe, *Robot Motion Planning*, Boston, MA: Kluwer Academic Publishers, 1991.

[3] S. Choi, T. Jin, and J. Lee, "Obstacle avoidance algorithm for visual navigation using ultrasonic sensors and a CCD camera," Artif. Life Robotics, vol. 7, (no. 3), pp. 132–135, Sep 2003.

[4] M. Mazo, "An integral system for assisted mobility [automated wheelchair]," *IEEE Robotics Automation Mag.*, vol. 8, (no. 1), pp. 46–56, Mar 2001.

[5] Y.K. Hwang and N. Ahuja, "Gross motion planning—A survey," *ACM Comput. Surv.*, vol. 24, (no. 3), pp. 219–291, Sep 1992.

[6] M. Mitchell, *An Introduction to Genetic Algorithms*, Cambridge, MA, MIT Press, 1996.

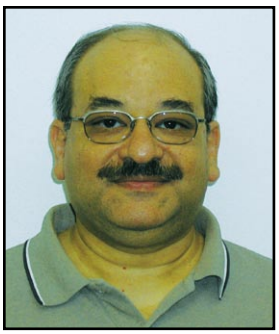
[7] A. Alvarez, A. Caiti, and R. Onken, "Evolutionary path planning for autonomous underwater vehicles in a variable ocean," *IEEE J. Oceanic Eng.*, vol. 29, (no. 2), pp. 418–429, Apr 2004.

[8] H. Burchardt and R. Salomon, "Implementation of path planning using genetic algorithms on mobile robots," in *Proc. 2006 IEEE Congress Evolutionary Computation*, Piscataway, NJ, IEEE, 2006, pp. 1831–1836.

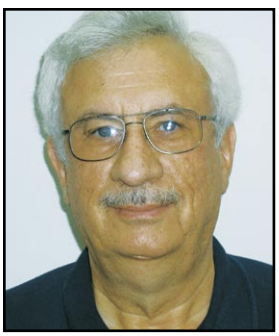
[9] T. Geisler and T.W. Manikas, "Autonomous robot navigation system using a novel value encoded genetic algorithm," in *45th IEEE Int. Midwest Symp. Circuits and Systems*, Piscataway, NJ, IEEE, 2002, pp. 45–48.

[10]K.H. Sedighi, K. Ashenayi, T.W. Manikas, R.L. Wainwright, and H.-M. Tai, "Autonomous local path planning for a mobile robot using a genetic algorithm," in *Proc. 2004 IEEE Congress Evolutionary Computation*, Piscataway, NJ, IEEE, 2004, pp. 1338–1345.

Theodore W. Manikas (theodore-manikas@utulsa.edu) has been an assistant professor of electrical engineering at The University of Tulsa since 2000. Prior to that, he was a systems analyst at the NSABP Biostatistical Center, University of Pittsburgh. His research interests include genetic algorithms, robotics, and VLSI design. He is a registered Professional Engineer in Oklahoma.



Kaveh Ashenayi (kash@utulsa.edu) has been with The University of Tulsa since 1986. He is currently a professor with the Electrical Engineering Department. His research interests include intelligent systems, robotics, control, and power systems. He is a registered Professional Engineer in Oklahoma.



Roger L. Wainwright (rogerw@utulsa.edu) has been with The University of Tulsa since 1974. He is currently a professor of computer science and the chair of the Mathematical and Computer Sciences Department. His research interests include analysis of algorithms, combinatorial optimization, genetic algorithms, and robotic navigation.