ISSN:1306-3111
e-Journal of New World Sciences Academy
2011, Volume: 6, Number: 4, Article Number: 1A0269

Metin Toz
Pakize Erdoğmuş
İbrahim Şahin
Duzce University
metintoz@duzce.edu.tr
pakizeerdogmus@duzce.edu.tr
ibrahimsahin@duzce.edu.tr
Duzce -Turkey

ENGINEERING SCIENCES
Received: September 2011
Accepted: October 2011
Series : 1A
ISSN : 1308-7231

A NEW EDUCATIONAL TOOLBOX FOR SOLVING ROBOTIC OPTIMIZATION PROBLEMS USING GA AND PSO

ABSTRACT

Two fundamental problems in robotics are the path planning for mobile robots to achieve a given task and the inverse kinematics for serial robots. In this study, these two problems were solved with two different heuristic optimization methods which are Genetic algorithm (GA) and Particle Swarm Optimization (PSO). GA and PSO results were compared in terms of the computational complexity of the algorithms and the feasibility of the solutions. According to the results, PSO outperforms GA with respect to both comparison criteria. Also, a new toolbox for using these optimization algorithms in robotics was developed for educational purposes. The toolbox was designed to simulate the solution of path planning problem in different environments and the inverse kinematics of a 6 Degrees of Freedom (DOFs) Puma robot with offset wrist.

Keywords: Mobile Robots, Obstacle Avoidance, GA, Serial Robots, Inverse Kinematics, Path Planning, PSO,

ROBOTİK PROBLEMLERİNİN GA VE PSO İLE ÇÖZÜMÜ İÇİN EĞİTİM AMAÇLI YENİ BİR YAZILIM

ÖZET

Mobil robotlarda verilen bir işi yerine getirmek için gerekli yol planlamasının yapılması ve seri robotlarda ters kinematik hesaplamalar robotik alanın iki temel problemidir. Bu çalışmada bu iki problem iki farklı sezgisel optimizasyon yöntemi olan Genetik Algoritma (GA) ve Parçacık Sürü Optimizasyonu (PSO) kullanılarak çözülmüştür. GA ve PSO'dan elde edilen sonuçlar hesaplama yükü ve çözümlerinin uygunluğu açısından karşılaştırılmıştır. Sonuçlar, her iki karşılaştırma kriterine göre de PSO'nun GA'ya göre daha iyi sonuç verdiğini göstermiştir. Bu çalışmada ayrıca GA ve PSO optimizasyon tekniklerinin robotikte kullanımı için eğitim amaçlı bir yazılım geliştirilmiştir. Yazılım, değişik ortamlarda mobil robot yol planlama problemi ve 6 serbestlik dereceli eklem kaçıklı bileğe sahip Puma türü bir robotun ters kinematik probleminin çözümlerinin benzetimini gerçekleştirecek şekilde tasarlanmıştır.

Anahtar Kelimeler: Mobil Robotlar, Engelden Sakınma, GA Seri Robotlar, Ters Kinematik, Yol Bulma, PSO

1. INTRODUCTION (GİRİŞ)

In general, robotic education requires solving several non-linear computationally expensive problems. Since these problems hard to solve using paper and pencil, several software tools have been developed for all aspects of the education to help both instructors and the students. Two of these problems which students faced with in their robotic education are path planning for mobile robots and inverse kinematics for serial robots. The path planning for mobile robots is finding an optimum path between a starting point and a goal point while the path satisfies certain optimization criteria such as being the shortest path and avoiding the collisions [1]. Robot kinematics refers to robot motions without considering the forces. The forward kinematics is the process of finding the robot's end effector's position and orientation using the given robot parameters and the joint's variables. The inverse kinematics is about finding the robot's joints variables when the robot's parameters and the desired position of the end effector are given [2]. There are many studies in the literature presents development of computer simulations or specifically designed software tools for educational purposes in robotic. Some examples can be given as following; Kucuk and Bingul was presented a robot toolbox named as ''ROBOLAB'' for simulation of the kinematical analysis of fundamental robot manipulators [3]. Toz and Kucuk was presented the next version of the ROBOLAB which is for dynamics simulation of fundamental robot manipulators using Langrange-Euler and Newton-Euler formulations [4]. In [5] Çakır and Butun was presented an educational tool for robotic with flexible structure using quaternion algebra. Nayar was introduced ROBOTECT which was designed for modeling, visualization and performance analysis of serial-link manipulator arms [6].

In this study, path planning problem for mobile robots, and inverse kinematic problem for a 6 DOFs puma robot with an offset wrist were solved with two different heuristic algorithms, GA and PSO, and a new toolbox was developed based on Matlab GUI (Graphical User Interface) to assist robotic instructors and students. Also, a comparison between GA and PSO was presented. Since, the efficiency of the stochastic methods is mainly related to the computational complexity of the algorithms and the feasibility of the solutions, in the present work, algorithm execution time and fitness values calculated through the proposed fitness functions were used as comparison criteria for GA and PSO. The obtained results showed that both algorithms produced acceptable solutions on both problems.

This paper consists of seven sections. Significance of the research is explained in the second section. In the third section, two algorithms, GA and PSO, are briefly described. The definitions and the solutions of the path planning problem for mobile robots in 2D environments and the inverse kinematic problem of a 6 DOFs Puma robot with offset wrist were presented in the fourth section. In fifth section, the comparison results of GA and PSO are given. The designed toolbox is introduced in the sixth section. Finally, the study is concluded with a conclusions section.

2. RESEARCH SIGNIFICANCE (ÇALIŞMANIN ÖNEMİ)

Two fundamental problems in robotics are path planning for mobile robots and inverse kinematics for serial robots. Both problems are considered as computational complex problems. Solving these problems becomes harder when the environment contains moving obstacles in path planning problem and when the number of joints of the robot increases in the inverse kinematic problem. Teaching the solution methods of these problems in robotic education is a huge challenge for

the instructors. In this study, a software toolbox in Order to help the educators was developed. The toolbox is able to solve the problems using GA and PSO which are the two very well known optimization algorithms and let the user visually observe the solutions. At the same time, this study presents important comparison results for the performances of GA and PSO on these problems.

3. DESCRIPTION OF THE GA AND PSO (GA VE PSO'NUN TANIMI)

In this section, we described the fundamentals of GA and PSO heuristics briefly and presented their pseudo codes used through this research work.

3.1. Genetic Algorithms (Genetik Algoritma)

Genetic algorithms (GAs) were invented by John Holland in 1970s [7] and have been utilized to solve many kind of optimization problems including combinatorial problems having a huge number of possible solutions such as reactive compensation placement, expansion planning, maintenance scheduling and so forth [8]. GA is based on theory of evolution. It uses evolutionary operators such as crossover and mutation to find an optimal solution to a given problem. The algorithm starts with a population of possible solutions and goes on using the fitness function and evolutionary operators to produce new offspring. This cycle repeats until a stopping criterion is reached. Pseudo code of the GA used in this study is shown in Figure 1.

```
init_Popu← Produce initial solution ()
(while stopping criteria not true do)
    for i=1 to generation_number
        calculate fitness function values of init_popu
        choose parents with roulette wheel
        (Parent 1 and Parent 2 are chosen)
        Elitism(Select the best ones in the population)
        Crossover(Create two children for each parent)
        Mutation(Gens are changed with mutation_rate)
        The new ofspring is created
    Next
```

Figure 1. Pseudo code for GA
(Şekil 1. GA için sözde kod)

3.2. Particle Swarm Optimization (Parçacık Sürü Optimizasyonu)

Particle Swarm Optimization (PSO) developed by Kennedy and Eberhart [9-10] based on the analogy of bird flocking or fish schooling [8]. Each particle in a swarm is a candidate solution of the problem and flies in an n dimensional search space. The algorithm starts with a population of particles. In every iteration, each particle keeps track of its coordinates and updates its velocity and position information according to the best solutions both it has achieved (pbest) and all particles in the swarm have achieved (gbest) so far. The formulations for velocity and position values are presented in Equation (1) and Equation (2) [11]. Pseudo code of the PSO used in this study is shown in Figure 2.

$$v_{id}(t+1) = wv_{id}(t) + c_1 rand[p_{id}(t) - x_{id}(t)] + c_2 rand[g_{id}(t) - x_i(t)] \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t) \qquad 1 \le i \le n \quad (2)$$

Where, v_i :The velocity of each particle, x_i :The position of each particle, d :Number of dimensions, w :Inertia weight factor, c_1 :The

cognitive learning rate, p_i :pbest value of ith particle, g_i :gbest value of the swarm, c_2 :The social learning rate, $rand$:Random value between 0 and 1.

```
Generate initial P particle swarm's random positions and velocities
do
  for i=1:P
    Evaluate fitness function for each particle
    If fitness(Pi) <fitness(Gbest) then Gbest=Pi
    If fitness(Pi) <fitness(Pbest(i)) then Pbest(i)=Pi
    Update velocity of the particle i
    Update position of the particle i
  end
Until stopping criteria is not true
```

Figure 2. Pseudo code for PSO
(Şekil 2. PSO için sözde kod)

4. PROBLEMS (PROBLEMLER)

4.1. The Path Panning Problem (Yol Planlama Problemi)

Path planning is one of the most studied topics for mobile robots. In the literature several methods have been proposed. Generally, these methods are classified in two categories, global and local path planning, according to the characteristics of the environment. The global method is based on the model of a static environment, the features of which are completely known. Although a mobile robot can find a globally optimum path using this method, the local path planning is required if the environment is dynamic and features of which are partially or entirely unknown [12]. In this study, global path planning was performed. The robot's environment was defined as a 10x10 unit size square in 2D coordinate system, and the obstacles were described as polygons in the same square. The coordinates of the robot's starting point, goal points, and vertexes of obstacles were assumed to be known. According to these known coordinates, a path is planned, so that, the starting and the goal points were connected through via points. The number of via points in the path can vary. A sample planned path is presented in Figure 3.

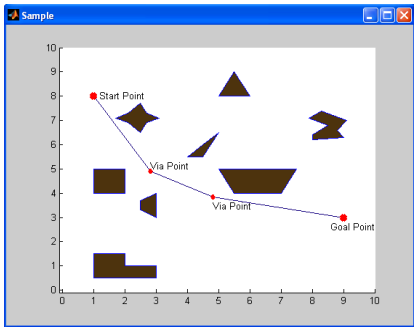

Figure 3. A sample planned path
(Şekil 3. Örnek bir yol planlaması)

4.2. Fitness Function for the Path Planning Problem (Yol Planlama Problem İçin Uygunluk Fonksiyonu)

Defining the fitness function is an important part of the solution process of the problems solved using heuristic optimization techniques. Concerning the path planning problem studied in this work, the fitness function was constructed with two factors in mind, the path length and the penalty for collision avoidance. The sum of these

two terms creates the final fitness value for the planned path. The path length is basically computed by means of Euclidean distance given in Equation (3). The total path length is calculated by summing the distances between two consecutive points including starting and goal points.

$$d = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{3}$$

Where (x_i, y_i) is the first end-point and (x_{i+1}, y_{i+1}) is the second end-point of a part of the path. In contrast to path length, collision avoidance is more complicated. Guaranteeing that the path does not go through any obstacles requires using a powerful technique. So, we used a technique based on a vector cross product property for avoiding collisions.

Obstacle avoidance procedure is started by determining the types of the obstacles, concave or convex. In this procedure, all vectors are defined in 3D coordinate system and the third element values (z coordinate values) are set to zero. In order to determine the obstacle types, cross product of two consecutive edge vectors is performed for every obstacle in the environment. If all signs of the cross products are the same for a given obstacle that means that the obstacle is convex. On the other hand, if the obstacle has mixture of cross product signs, in this case, the obstacle is concave [13]. After that, an obstacle area is determined for every obstacle. For a convex obstacle, the area is the same area enclosed by the edges of the obstacle, but for a concave obstacle, before determining the area, the vertexes of the obstacle which cause concaveness are removed and the area is determined using the remaining vertexes of the obstacle. As shown in Figure 4, the obstacle on the left is a convex obstacle. So the vertex list of the obstacle is not changed. On the other hand, the obstacle on the right is a concave obstacle. V6 is the vertex that causes concaveness, so this vertex is removed from the vertex list of the obstacle before the obstacle area is determined. The final area of the second obstacle includes both S1 and S2 areas. After the obstacle areas are determined, firstly, a *p* vector between two consecutive via points is defined as shown in Figure 5. Then, a rectangular path area is determined which is constructed by maximum and minimum values of the coordinates of the *p* vector. For every obstacle which is located totally or partially in the path area, additional vectors are defined from the first via point (P1) of the vector *p* to each vertex of the obstacle area. Two sample vectors, *a* and *b*, are depicted in Figure 5.

As shown in Figure 5, since the obs1 is entirely located outside of the path area there is no need to determine whether the path segment crosses this obstacle or not. As a result, the obstacle is ignored for this segment of the path. But, obs2 and obs3 must be taken into account because these obstacles are entirely or partially located in the path area. If we consider obs2, which is crossed by the *p* vector, the signs of the cross products of *p* and *a*, and *p* and *b* are opposite of each other. This case is represented in Equation (4).

$$sign(\vec{p} \times \vec{a}) = -sign(\vec{p} \times \vec{b}) \tag{4}$$

Three vertexes of the obs2 area are on the right hand side of the *p* vector while one vertex is on the left hand side. This means that the signs of the cross products are different and obs2 is absolutely on the path segment. Considering the obs3 area, all vertexes of the area are located on the left hand side of the *p* vector. Hence, the signs of the all cross products will be the same. That means that the obs3 is not on the path segment.

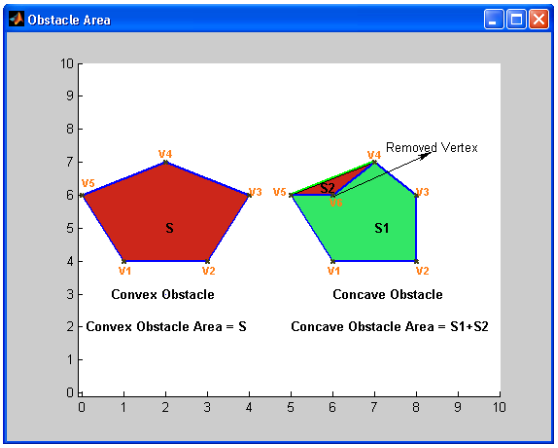

Figure 4. Determination of obstacle areas
(Şekil 4. Engel alanlarının belirlenmesi)

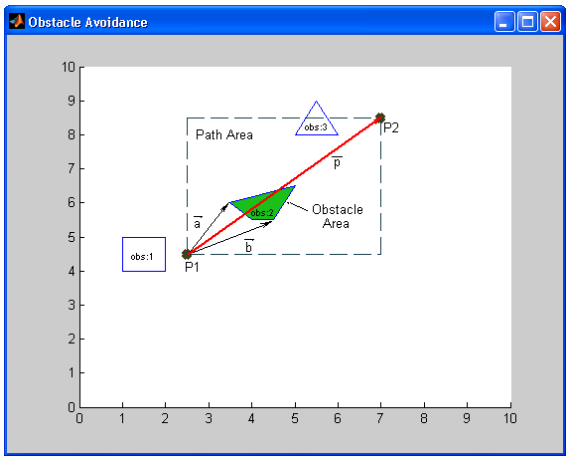

Figure 5. A sample path area and vectors related this area
(Şekil 5. Örnek bir yol alanı ve bu alanla ilgili vektörler)

For each obstacle, which is crossed by the path segment, a penalty value is calculated. While calculating it, first, a rectangular penalty area is formed using minimum and maximum coordinate values of the obstacle area. Second, the perpendicular distance between the center of the penalty area and the path segment is calculated using *a* and *p* vectors as shown in Figure 6. Using these distance values calculated for each path segment, the final fitness value for the whole path is calculated through Equation (5). Considering the fitness function described above, the problem solved in this study is about finding a minimum acceptable value.

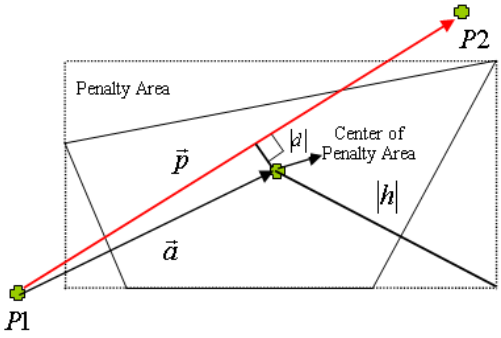

Figure 6. Distances between the center of penalty area and the *p* vector
(Şekil 6. Ceza sahası merkezi ile *p* vektörü arasındaki dik uzaklık)

$$f = \sum_{i=1}^{k} L_i + \sum_{j=1}^{n} c_p \left(1 + h_j - d_j\right)^2 \qquad i, j = 1, 2, 3...n \qquad (5)$$

Where;

$|d| = \dfrac{|\vec{a} \times \vec{p}|}{|\vec{p}|}$: Distance between the center of the penalty area and the path segment.

$|h|$: Distance between the center and a corner of the penalty area.

L : The length of ith segment of the path,

k : The number of segments of the path,

$$L = \begin{cases} L & , \quad L < 3d_u \\ p_p L & , \quad L \geq 3d_u \end{cases}$$

p_p : The penalty constant for length of the path,

d_u : The Euclidean distance between the starting and the goal points.

n : Number of collisions.

c_p : The penalty constant for collision with an obstacle.

4.3. The Inverse Kinematic Problem for PUMA Robot (PUMA Robotu için Ters Kinematik Problem)

Robot kinematics is related to the robot motions, without consideration of the forces. The forward kinematics is determination of the robot's end effector's position and orientation while the robot parameters and the joint's variables are given. The inverse kinematics is about finding the robot's joints variables when the robot's parameters and the desired position of the end effector are given [2].The solution of the robot's forward kinematics is always possible and unique. On the other hand, generally, the inverse kinematics problem has several solutions [2]. The most known and used method for solving robot kinematics problems is Denavit-Hartenberg method. In this method, several parameters, namely DH parameters, are defined according to the robot's parameters and coordinate systems placed on the robot joints as shown in Figure 7. These parameters are used to define transformations in between the coordinate systems using 4x4 transformation matrices, Equation (6). The multiplication of the all matrices of the robot gives the final position and orientation of the robot's end effector [2].

$$T = \begin{bmatrix} & R & & P \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Where, T and R are 4x4 transformation and 3x3 rotation matrices between two consecutive links respectively while P is 3x1 position vector. The T matrix can be defined using DH parameters as in Equation (7).

$$T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos a_{i-1} & \cos\theta_i \cos a_{i-1} & -\sin a_{i-1} & -\sin a_{i-1} d_i \\ \sin\theta_i \sin a_{i-1} & \cos\theta_i \sin a_{i-1} & \cos a_{i-1} & \cos a_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

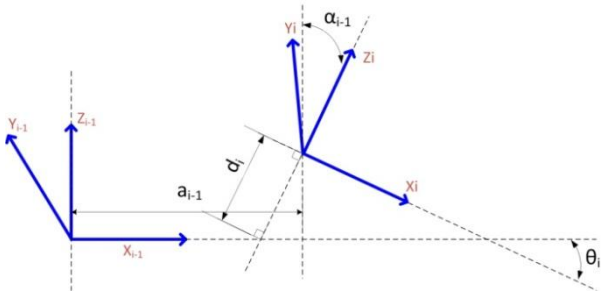

Figure 7. DH parameters between two coordinate systems
(Şekil 7. İki koordinat sistemi arasındaki DH parametreleri)

The selected robot is a nearly PUMA type robot. Differently from the original PUMA robot, the robot used in this study is equipped with an offset wrist. The inverse kinematics of the robot equipped with such a wrist is quite complicated and can be computationally cumbersome [14]. The robot's link parameters and coordinate systems placements can be seen in Figure 8. Furthermore, according to Figure 8, the DH parameters of the robot can be defined as given in Table 1.

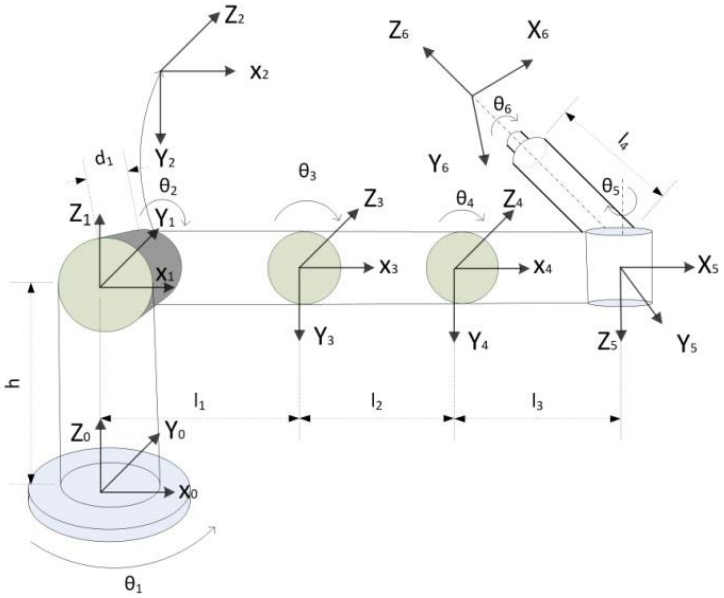

Figure 8. The nearly Puma robot's link parameters and coordinate systems placement
(Şekil 8. PUMA Robotun bağ parametreleri ve koordinat sistemleri yerleşimi)

Table 1. The DH parameters of the robot
(Tablo 1. Robotun DH parametreleri)

İ	a_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	h	θ1
2	-π/2	0	d1	θ2
3	0	l1	0	θ3
4	0	l2	0	θ4
5	-π/2	l3	0	θ5
6	π/2	0	l4	θ6

The forward kinematics of the robot is the forward multiplication of transformation matrices of the robot given by Equation (8).

$$ {}^{0}_{6}T = {}^{0}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T\ {}^{3}_{4}T\ {}^{4}_{5}T\ {}^{5}_{6}T \tag{8} $$

From the inverse kinematic problem of the robot, it can be seen that each individual of the population should have six joint constraints. They are θ_1, θ_2, θ_3, θ_4, θ_5, and θ_6. So, an individual can be a 1x6 vector. Each component of the individual corresponds with one of the joint's constraints. The range of each constraint can be defined separately. However, for the sake of simplicity, the range of the joints is defined between [-π, π]. A sample individual can be seen in Figure 9.

π	-π/4	π/2	0	π/5	0

Figure 9. A sample individual
(Şekil 9. Örnek bir aday çözüm)

4.4. Fitness Function for the Inverse Kinematic Poblem (Ters Kinematik Problem için Uygunluk Fonksiyonu)

The fitness function is defined using the desired position values of the end effector of the robot and the obtained position values from the individuals using forward kinematic equations of the robot. The Euler distance between the desired position of the end-effector and the results obtained from an individual can be defined as follows.

Let the desired position of the end effector and the obtained position knowledge using an individual be Pd and Pi where, $P_d = [x_d \quad y_d \quad z_d]$, $P_i = [x_i \quad y_i \quad z_i]$. The Euler distance between these two points in 3D space can be calculated using Equation (9). The individual that offer the smallest distance is the most convenient candidate of the solution. The fitness function for the problem can be formulated as in Equation (10).

$$ d_i = \sqrt{(x_d - x_i)^2 + (y_d - y_i)^2 + (z_d - z_i)^2} \tag{9} $$

$$ O_i = p d_i^{\,2} \tag{10} $$

Where O_i is fitness function value, p is penalty constant, and d_i is the Euler distance calculated using Equation (9) for i^{th} individual.

5. COMPARISON OF GA AND PSO ON PATH PLANNING and INVERSE KINEMATIC PROBLEMS (YOL PLANLAMASI VE TERS KİNEMATİK PROBLEMLERİNDE GA VE PSO'NUN KULLANIMININ KARŞILAŞTIRILMASI)

In this section, the path planning problem for mobile robots and the inverse kinematic problem for the puma robot are solved with GA and PSO utilizing the fitness functions described above and a comparison between these algorithms according to final object function values and the algorithm's execution times are presented. Both algorithms were run on the same computer which was equipped with a 2.80 GHz Pentium 4 microprocessor and 2 GB RAM memory.

The path planning problem was solved in a very complicated environment which contains 15 polygonal obstacles. Each algorithm was run 100 times. The parameters of GA and PSO defined for this problem are presented in Table2. As listed in the table, values of common parameters which are number of individuals in the populations, number of iterations, cp, pp, number of obstacles, number of via points, start point, goal point, and gene code type were determined to be the same for both heuristic methods. The evolution of the best fitness values and execution times are presented in Figure 10 and 11 for both algorithms respectively. The solutions of each algorithm for all runs were plotted on two separate figures, GA's solutions in Figure 12 and PSO's in Figure 13.

Table 2. Parameters of GA and PSO for path planning problem
(Tablo 2. Yol planlama problem için GA ve PSO parametreleri)

Parameters	Number of individuals	Number of Iterations	cp	pp	Number of Obstacles	Number of Via Points	Start Point	Goal Point	Gene Code Type	Mutation Rate	Crossover Rate	Selection Type	Mutation Type	c1	c2	wmin	wmax	vmax
GA	50	100	120	2.2	15	5	(0.1,0.1)	(8.5,8)	Real Values	0.4	0.9	R. Wheel	One Point	**	**	**	**	**
PSO	50	100	120	2.2	15	5	(0.1,0.1)	(8.5,8)	Real Values	**	**	**	**	2	2	0.4	1.2	1

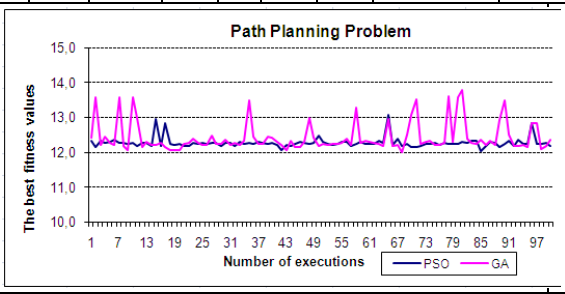

Figure 10. Evolution of the best fitness values obtained from GA and PSO for Path Planning Problem.
(Şekil 10. Yol planlama probleminin çözümünde GA ve PSO ile elde edilen en iyi uygunluk değerleri)

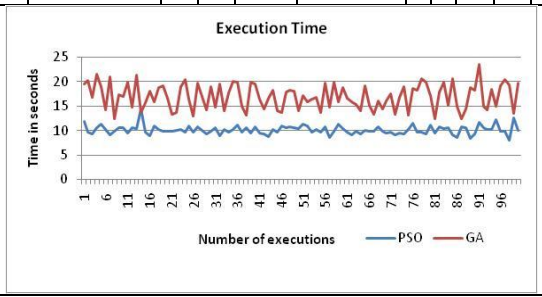

Figure 11. Execution times obtained from GA and PSO for Path Planning Problem.
(Şekil 11. Yol planlama probleminin GA ve PSO ile çözümü için harcanan zaman)

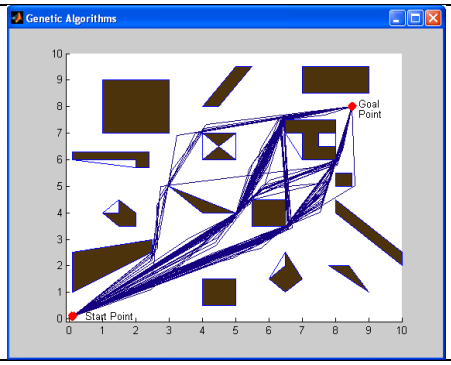

Figure 12. All solutions of GA for Path Planning Problem
(Şekil 12. Yol planlaması problem için GA tarafından üretilen tüm çözümler)

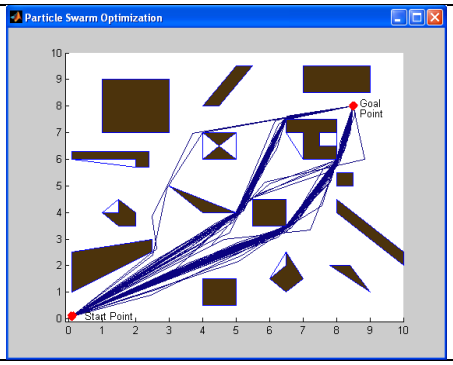

Figure 13. All solutions of PSO for Path Planning Problem
(Şekil 13. Yol planlaması problem için PSO tarafından üretilen tüm çözümler)

The second problem is the inverse kinematic problem and it was also solved with the two meta-heuristics. The robot parameters were the world coordinates=[0 0 0], zero thetas=[0 0 0 0 0 0], goal point=[10 5 60], h=30, d1=5 and l1=l2=l3=l4=10. The parameters used in both algorithms are given in Table 3.

Table 3. The parameters for GA and PSO for the inverse kinematic problem
(Tablo 3. Ters kinematik problem için GA ve PSO parametreleri)

Parameters	Number of individuals	Number of Iterations	Penalty	World Coordinates	Goal Coordinates	Gene Code Type	Mutation Rate	Crossover Rate	Selection Type	Mutation Type	c1	c2	wmin	wmax
GA	100	1000	10	[0 0 0]	[10 5 60]	Real Values	0.2	0.9	R. Wheel	One Point	**	**	**	**
PSO	100	1000	10	[0 0 0]	[10 5 60]	Real Values	**	**	**	**	2	2	0.4	0.9

The evolution of the best fitness values and execution times for the inverse kinematic problem were presented in Figure 14 and 15 for both algorithms, respectively.

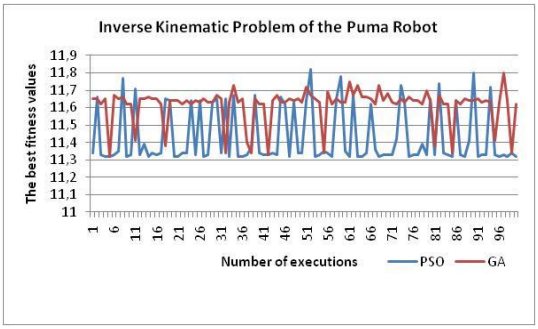

Figure 14. Evolution of the best fitness values obtained from GA and PSO for the inverse kinematic problem
Şekil 14. Ters kinematik problemin çözümünde GA ve PSO ile elde edilen en iyi uygunluk değerleri)

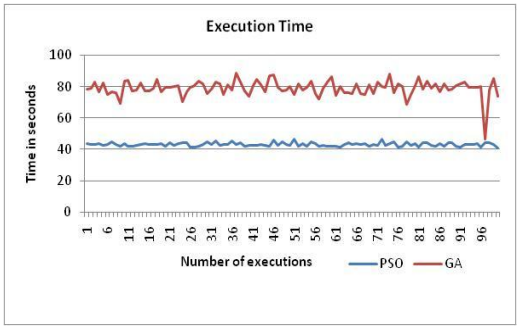

Figure 15. Execution time obtained from GA and PSO for the inverse kinematic problem.
(Şekil 15. Ters kinematik problemin GA ve PSO ile çözümü için harcanan zaman)

The analysis of overall results indicated that both meta-heuristic algorithms found the nearly optimum solutions to both problems. Moreover, both algorithms were generated similar results in terms of the best fitness values. However, in terms of execution time, PSO outperformed GA. Also, in the path planning problem, PSO outperformed GA for both comparison criteria.

6. THE TOOLBOX (YAZILIM)

The toolbox was developed for solving and simulating the path planning problem for mobile robots and the inverse kinematic problem for a puma robot with offset wrist using the two optimization algorithms, PSO and GA. The user interface of the toolbox is shown in Figure 16.

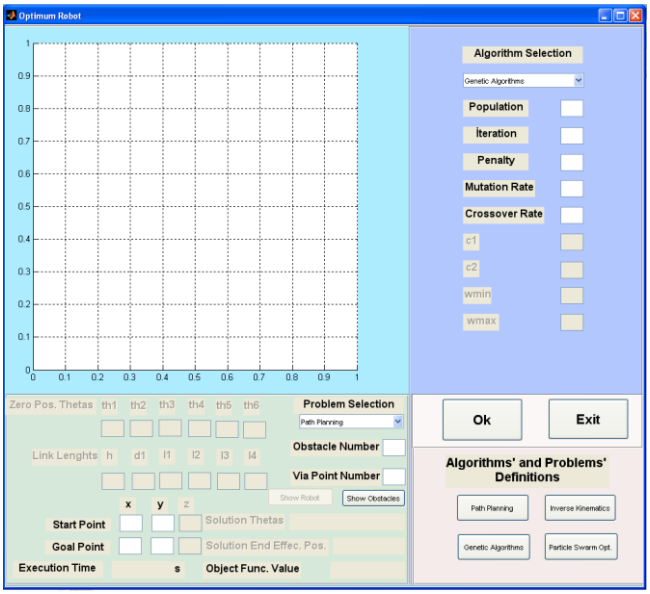

Figure 16. The toolbox user interface
(Şekil 16. Yazılımın kullanıcı arayüzü)

The toolbox is composed of four parts. The first one is the algorithm Selection section. This section is simply let the user to select one of the optimization algorithms, GA and PSO. After selecting the algorithm type, common parameter and algorithm specific parameters become active so that user can enter desired parameters correctly.

Second section is designed to select and form the sample problem. The tool offers two options for selecting problem types which are the path planning and the inverse kinematic problems. When path planning is selected, the user should define the number of obstacles in the environment, the start and goal points on the path and number of via points on the path. The obstacles are selected randomly from a pre-defined obstacle list and if desired can be previewed before the problem is solved using "Show Obstacles" button.

When the inverse kinematic problem of the puma robot is selected the user should define the link length parameters which construct the puma robot, world coordinates which are the coordinates of the first link of the robot, zero thetas the angles for the zero position of the robot, and the goal point coordinates which are the desired coordinates of the end effector of the robot. If desired, the user can preview the robot with currently defined parameters in 3D environment before solving the problem.

The third section of the user interface was designed for visualizing the results. The both algorithm's results are drawn in graph area of this section. Path planning problem results are drawn using 2D graphical objects while the inverse kinematic problem results are drawn using 3D objects. Since the toolbox was designed for educational purposes, the fourth section formed for reaching the brief information about the problems and the algorithms. There is a push button for each topic in this section and if the user clicks one of these buttons, a text file containing brief information about the topic is displayed. Two different simulation examples are presented to show the toolbox functionality.

The first example is about the path planning problem. In this example, the environment was containing 10 differently shaped obstacles and the number of via points was selected as 2 while the start and the goal point coordinates were defined as [0 0] and [8 8] in 2D environment, respectively. The problem was solved using GA and the parameter of the algorithm were defined, population=100, iteration number=100, penalty=1000, crossover rate=0.9 and mutation rate=0.1. After defining the parameters, when the "Ok" button was clicked, the problem was solved with GA and the results are displayed in the graph area of the user interface. Also the final object function value which is 12.0545 and the algorithm execution time which is 33.6395 sec. are displayed in the related areas of the second section of the interface as shown in Figure 17.

The second example is about the inverse kinematic problem for the puma robot. This problem was solved using PSO. Through this problem, first, the robot's parameters were defined, zero_thetas=[0 0 0 0 0 0], h=10, d1=4, l1=l2=l3=l4=5 and world coordinates=[0 0 0]. The desired end-effector coordinates were defined as [5 5 10]. Second, the algorithm parameters were defined, population=100, iteration number=100, penalty=1000, c1=2, c2=2, wmin=0,4 and wmax=0,9. Similar to the first example, when the "Ok" button was clicked, the problem was solved and final status of the robot was drawn on the graph area using the joint angels determined by the algorithm which were displayed in the related are of the second section of the user interface. The resulting parameters for this case are the final object function value=0.386079, execution time =0.946098 sec, solution thetas=[0.19 -0.95 1.90 1.28 -.157 0] in radians and solution end effector coordinates= [5.0 4.99 10.02] as shown in Figure 18. It can be obviously seen that the algorithm solved this very nonlinear problem with just [0 0.001 0.02] error.

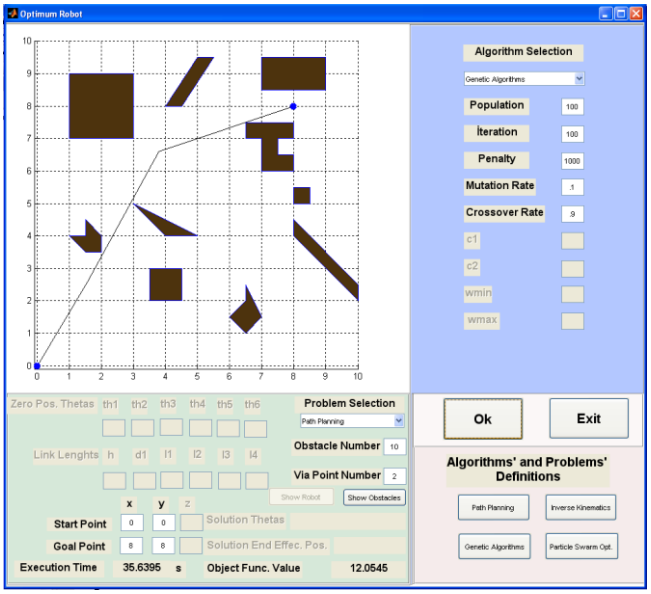

Figure 17. Solving the path planning problem using GA
(Şekil 17. Yol planlaması probleminin GA ile çözümü)

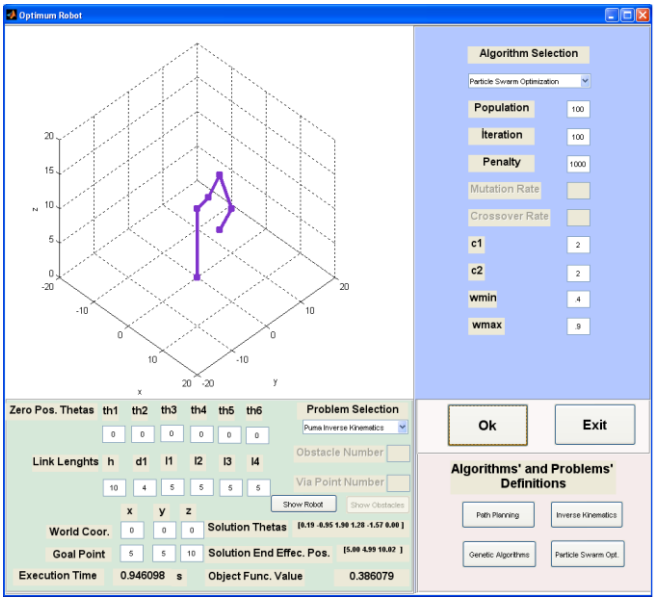

Figure 18. Solving the inverse kinematic problem of the Puma robot using PSO
(Şekil 18. Ters kinematik problemin PSO ile çözümü)

7. CONCLUSIONS (SONUÇLAR)

In this work, the path planning problem for mobile robots and the inverse kinematic problem for a 6 DOFs puma robot with offset wrist were solved with GA and PSO, which are two well known evolutionary optimization methods. A new and helpful toolbox is developed for educational purposes of using optimization algorithms in robotic. PSO and GA are compared for the two problems according to the best fitness values and algorithm's execution times. Both algorithms produced acceptable solutions on both problems. However, the results showed that PSO outperformed GA in terms of execution time. Moreover,in path planning problem, PSO produced better results than GA in terms of best fitness values.

REFERENCES (KAYNAKLAR)

1. Qin, Y., Sun, D., Li, N., and Cen, Y., (2004). Path Planning for Mobile Robot using the Particle Swarm Optimization with Mutation Operator, International Conference on Machine Learning and Cybernetics, Vol. 4, pp.2473–2478.
2. Kucuk, S. and Bingul, Z., (2006). Robot Kinematics: Forward and Inverse Kinematics, In: Industrial Robotics: Theory, Modelling and Control.
3. Kucuk, S. and Bingul, Z., (2010), An off-line robot simulation toolbox, Computer Applications in Engineering Education, Vol. 18, pp. 41–52.
4. Toz, M. and Kucuk, S., (2010), Dynamics simulation toolbox for industrial robot manipulators. Computer Applications in Engineering Education, Vol.18, pp. 319–330.
5. Cakir, M. and Butun, E., (2007), An educational tool for 6-DOF industrial robots with quaternion algebra. Computer Applications in Engineering Education, Vol.15, pp. 143–154.
6. Nayar, H.D., (2002), Robotect: Serial-link manipulator design software for modeling, visualization and performance analysis, 7th International Conference on Control, Automation, Robotics and Vision,Vol.3, pp 1359_1364.
7. Mitchell, M., (1999), An Introduction to Genetic Algorithms, MIT Press.
8. Lee, K.Y. and El-Sharkawi, M.A., (2008), Modern Heuristic Optimization Techniques Theory and Applications to Power Systems, IEEE Press.
9. Eberhart, R. and Kennedy, J., (1995), A New Optimizer Using Particle Swarm Theory, In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39-43.
10. Kennedy J and Eberhart R (1995), Particle Swarm Optimization, In Proceedings of IEEE International Conference on Neural Networks, Vol. 4, pp.1942-1948.
11. Qu, C. and Lin, W., (2006), Comparison between PSO and GA for Parameters Optimization of PID Controller, International Conference on Mechatronics and Automation, pp.2471–2475.
12. Sedighi, K.H., Ashenayi, K., Manikas, T.W., Wainwright, R.L., and Tai, H.M., (2004), Autonomous Local Path Planning for a Mobile Robot Using a Genetic Algorithm, The Congress on Evolutionary Computation, Vol. 2, pp. 1338–1345.
13. http://paulbourke.net/geometry/clockwise/ (29.09.2011)
14. Kucuk, S. and Bingul, Z., (2005), The Inverse Kinematics Solutions of Fundamental Robot Manipulators with Offset Wrist, Proceedings of the 2005 IEEE International Conference on Mechatronics. Pp.197-202