Robotica (2014) volume 32, pp. 77–95. © Cambridge University Press 2013
doi:10.1017/S0263574713000635

Immune–wavelet optimization for path planning of large-scale robots

Saeid Asadi, Vahid Azimirad*, Ali Eslami and Saeid Karimian Eghbal

Biomechatronics Laboratory, School of Engineering Emerging Technologies, University of Tabriz, Tabriz, Iran

(Accepted May 31, 2013. First published online: July 19, 2013)

SUMMARY
In this paper an optimal path planning method based on a new evolutionary algorithm is presented for higher order robotic systems. It is a combination of immune system and wavelet mutation. By increasing the system's dimensions, the complexity of algorithm grows linearly. The obtained results have been compared with other optimal path producing algorithms, and its excellence in terms of optimality has been proved. Strengths of this method are simplicity in large-scale path planning, being free of most of the common deadlocks in usual method, and ability to obtain more optimized results than other similar methods. The effectiveness of this approach on simulation case studies for a three-link planar robot and 5 degrees of freedom mobile manipulators as well as an experiment for a mobile robot called K-joniour is shown.

KEYWORDS: Artificial immune system; Wavelet; Path planning; Large-scale robots.

1. Introduction
Nowadays the robotic applications are going toward using the robots with high degrees of freedom and more complicated structure. Producing the optimal path for sophisticated systems, which involve a huge range of industrial systems, has always been a challenging problem. These kinds of robots have been considered due to high efficiency. But by increasing the system's dimension, path planning problem becomes much more complicated. Although great attention is given to robotic path planning, a few researchers have considered complex robotic systems.[1]

An important issue in path planning is optimality. Generally, path optimization is treated with dynamic equations. There are two types of optimal path planning: analytical and evolutionary methods. Working with analytical methods, such as optimal control, is easy as the dynamic equations of robots are analytic, but it must be solved numerically, which may lead to numerical explosion in large-scale mobile robots.[2] Hence, evolutionary algorithms, such as ant colony,[3] particle swarm optimization,[4] and genetic algorithm[5] for path planning of mobile robots are developed. They developed a method based on genetic algorithm for planning paths that are subjected to sub-path constraints. Ching-Chih *et al.*[6] presented a parallel elite genetic algorithm for path planning of mobile robots in cluttered environments. More recently, path planning of mobile robots using artificial immune system (AIS) is studied;[7,8] but only the kinematic model of mobile robot is considered. None of the previous works consider dynamic equations, large-scale systems, mobile manipulators, and wavelet mutation. Although the path planning of simple point-assumed mobile robots is done with AIS, the path planning of dynamic complicated high-order systems with AIS algorithm is yet to be studied.

In this paper a novel method is presented for path planning of large-scale robotic systems based on AIS–wavelet algorithm, which is inspired by biological immune system. The most impressive algorithm for AIS is influenced by the concept of clonal selection.[9] On the other hand, the mutation is

* Corresponding author. E-mail: azimirad@tabrizu.ac.ir

the most important step in evolutionary algorithms. Some methods have been suggested for making appropriate features in mutation to increase algorithm's efficiency.[10] They proposed a particle swarm optimization with wavelet mutation. Furthermore, wavelet mutation was used in differential evolution to control scaling factor.[11] The excellence of the AIS algorithm over other methods has been proved,[12] but it is still not applied on path planning of robots with considering dynamic equations. The proposed method is completely based on an evolutionary algorithm and does not have other methods' problem, such as numerical explosion,[13] while analytical dynamic equations are used. In the proposed method, growing the dimension of the system leads to increase in input and a midpoint in path curvature, named (t_b), for each of the relocating curves, which keeps information of kinematics. Hence, by increasing the degree of freedom, the problem develops linearly and numerical explosion does not occur for the path planning of large-scale robots.

In this method, first the final desired point is obtained, second the speed and acceleration at the starting and ending points can be easily controlled, and finally, a quite smooth path is the result. Hence, the profile is broken into two sections: mutation and revision. In the mutation of input control, optimality is achieved, and in the revision, the path is directed to reach the final point. The value of break point is chosen by the AIS algorithm. Some strengths of this method are as follows: decreasing the cost function more than other similar methods, simple than other similar methods for complicated robots, and being free of sensitivity to initial guesses. While the solution of Boundary Value Problem (BVP) is sensitive to initial guess, solution of Initial Value Problem (IVP) is not so. Furthermore, in BVP problems the wrong initial guess leads to singularity errors. Using IVP in the proposed method, the problem is solved without sensitivity to initial guess and singularity. Besides, the IVP is solved for many times in the AIS iterations. Hence, singularity error could not be generated in this method.

The rest of paper is organized as follows: In Section 2 the AIS algorithm is explained based on wavelet mutation. In Section 3 the path planning problem is clarified, which is continued by a new evolutionary method as a solution. Section 4 contains the results of simulations and the comparison with other methods as well as the results of path planning algorithm. Results of experimental test are discussed in Section 5. Finally, Section 6 concludes the paper.

2. Artificial Immune System (AIS) Algorithm

Artificial Immune System is based on the function of body's last line of defense. There are three defensive layers for human body against foreign substances (called pathogens), in which the third (last) layer plays the major role. This layer has some cells, named B cells (lymphocytes). Antigens are pieces of proteins suspended in the blood separated from infectious foreign elements. Using antigens, immune system is able to distinguish foreign cells from inner cells. When an antigen (a solution to the problem) is recognized by B cells, they begin to proliferate (clons). Many of these proliferated cells become a special kind of cells called plasma (resolvent), and rest of these become memory cells. The plasma cells stick to antigens and destroy the cells that are related to antigens, and the memory cells keep the history of this illness. In AIS, the whole process is repeated till the termination condition is satisfied. Unlike the genetic algorithm, in the AIS algorithm the new generation usually has more adjustment with the environment. Besides, the Continuous Clonal Selection Algorithm (CCSA)[12] has better performance in comparison to other similar evolutionary techniques.

2.1. Modeling aspect

In this paper the CCSA based on wavelet mutation is used. It is inspired by clonal selection algorithm, wavelet mutation, and negative selection approach. The algorithm has the below-mentioned components.

2.1.1. Initializing. The following parameters are initialized:

- n: the number of main population.
- β: the coefficient of reproduction.
- N_{gen}: the maximum permitted amount of repetition of algorithm.
- F_{aff}: the maximum permitted affinity among members of population.

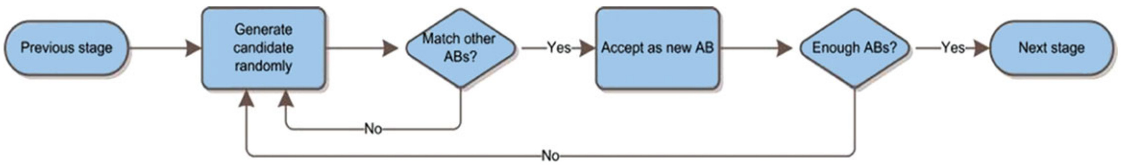

Fig. 1. (Colour online) AB's generation process based on negative selection approach.

2.1.2. Generating the initial population. Having the proper initial population is essential for increasing the speed of algorithm. Hence, in order to distribute the population in search space, the concept of negative selection is used, which consists of the following three basic steps (Fig. 1):

(a) The random generating of AB candidates in $[-1, 1]$.
(b) Measuring the distance between candidate AB and other accepted candidates.
(c) The acceptance of mentioned AB, provided its distance from others is less than F_{aff}.

2.1.3. Proliferation. Each member of the population is reproduced for $n_c = \beta \times n$ times for clone production.

2.1.4. The mutation of produced clones. All members of the clone except the main member are mutated. By selecting the proper method of mutation, the speed of convergence is increased. In this algorithm, another mutation is done through the wavelet theory. The wavelet mutation operation produces an adaptive amount of mutation for each repetition. Suppose parameter P must be mutated. Hence,

$$\hat{p} = p + rand(1) \times \mu. \tag{1}$$

The $rand(1)$ generates a random number from $[0, 1]$, and μ is the adaptive mutation rate obtained from the wavelet theory,

$$\mu = \psi_{a,0}(\alpha), \tag{2}$$

$$\mu = \frac{1}{\sqrt{a}}\psi\left(\frac{\alpha}{a}\right), \tag{3}$$

where ψ is the mother wavelet, which could be selected from various types of mother wavelets. For wavelet mutation application, dilation parameter is used and translation parameter is assumed zero. Here Morlet wavelet is used.[14] The Morlet wavelet is a sine wave multiplied by a Gaussian envelope. Furthermore, it is a complex wavelet that can be decomposed in two parts: real and imaginary. In this paper, the real part is used as defined in Eq. (4):

$$\psi(x) = e^{-x^2/2}\cos(5x). \tag{4}$$

Therefore,

$$\mu = \frac{1}{\sqrt{a}}e^{-(\frac{\alpha}{a})^2/2}\cos\left(5 \times \frac{\alpha}{a}\right), \tag{5}$$

where α is generated randomly from interval $[-2.5a, 2.5a]$ because over 99% of the total energy of the Morlet mother wavelet function is in the interval $[-2.5, 2.5]$ as $a = 1$. Consequently, μ is randomly produced by Eq. (5) in the interval $[-1, 1]$. Moreover, a is dilation (scale) parameter of wavelet, which is defined in monotonic increasing function as in Eq. (6):

$$a = e^{-\ln(g) \times \left(1 - \frac{\mathrm{gen}}{N\mathrm{gen}}\right)^{\xi} + \ln(g)}, \tag{6}$$

where g is the upper limit of parameter a, and ξ is the shape parameter of monotonic increasing function. It means that in early iterations of algorithm, a produces a coarse-tuning mutation rate and

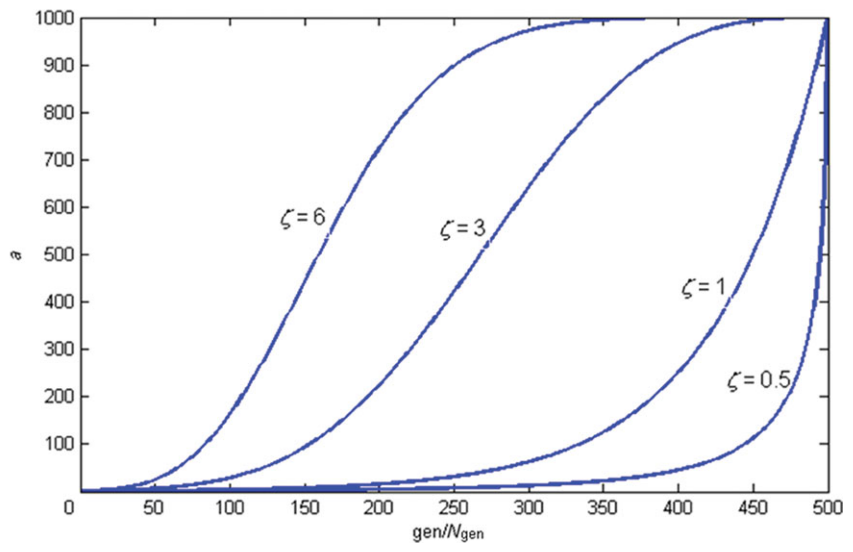

Fig. 2. (Colour online) Effects of various values of ξ with respect to gen/N_{gen} ($g = 1000$).

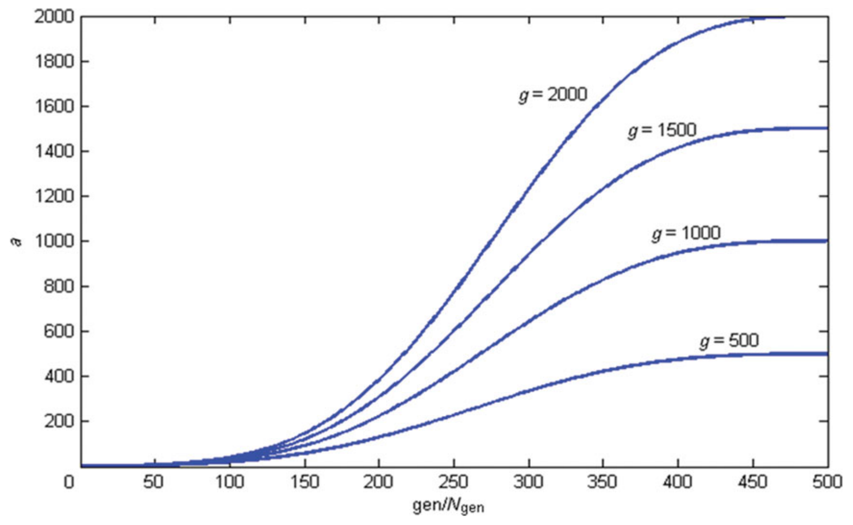

Fig. 3. (Colour online) Effects of various values of g with respect to gen/N_{gen} ($\xi = 3$).

by increasing iterations, a tends to upper limit and the mutation rate dwindles. However, performance of the wavelet mutation operation depends on selection of g and ξ and total number of iterations, N_{gen}. Figs. 2 and 3 respectively show the effects of shape parameter ξ and upper limit g with respect to gen/N_{gen}. ξ and g are optimized by CCSA-WM too. Note that for achieving property 1, clone size (CS) must be large enough as in Eq. (7):

$$\frac{1}{\text{CS}} \sum_{\text{CS}} \mu = 0 \text{ for CS} \to \infty. \tag{7}$$

2.1.5. Evaluation of each clone. The members of each clone are given out to the cost function (or fitness function), then in each clone the member that makes the least cost is replaced by the main member of the clone.

2.1.6. Improving main population. The main members of clones are replaced with the main population. Then the worst members (the members that generate greater cost in comparison to other members) are replaced by new members in the main population using the method explained

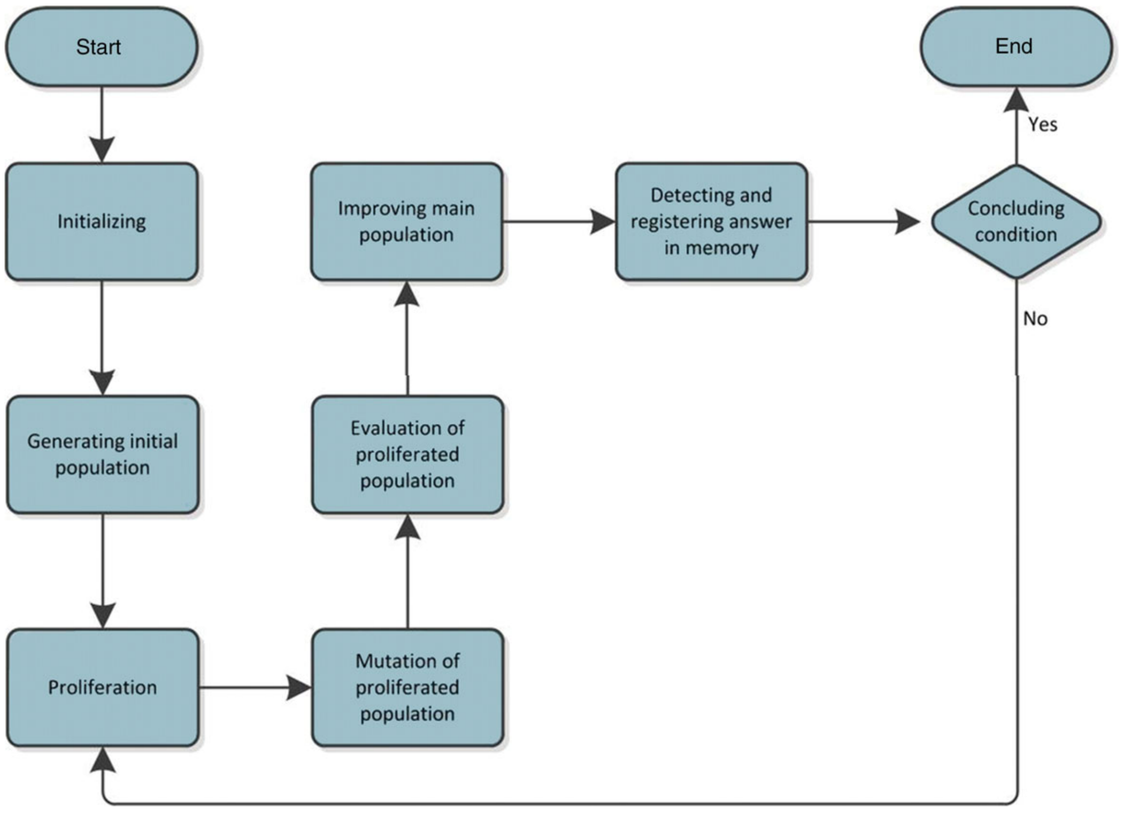
Start
Initializing
Generating initial population
Proliferation
Mutation of proliferated population
Evaluation of proliferated population
Improving main population
Detecting and registering answer in memory
Concluding condition
Yes
No
End

Fig. 4. (Colour online) The flowchart of CCSA.

in stage 2 (negative selection method). The best members (the members that generate less cost in comparison with other members) are nominated as the solution.

2.1.7. Detecting and registering answer(s) in memory. In two cases the nominated answer in stage 6 is registered in the memory as the solution(s), i.e., if the memory is empty or the nominated answer has less cost in comparison with other answer(s) then that answer(s) is/are already registered in the memory.

2.1.8. Evaluation of termination condition. If the gen (the counter of repetition) is smaller than N_{gen}, go to step 3, else go to end. Consequently, the flow chart of algorithm is illustrated in Fig. 4.

3. Path Planning of Manipulator

3.1. Defining anti-bodies to the algorithm

The parameters should be defined as anti-bodies. Here is a brief definition as a summary. For each degree of freedom, following parameters must be defined as anti-bodies: the curves that keep the information about the angular speed and acceleration (as discussed in Section 3.2) and the inputs (Us) (as discussed in Section 3.3). At the final phase, the problem is solved through the optimizing algorithm of immune system.

3.2. The initial estimation of controlled parameters

It is supposed that the desired paths are curves. It is indicated in the explanation part of the immune system algorithm that the initial estimation of anti-bodies is necessary for starting. A polynomial of degree 3 or 5 is used as an initial estimation.[15] The primary position is specified as a collection of joint angles. The goal is to detect a function for each joint in a way that its amount at t_0 be equal to joint's initial location and at t_f it be equal to the considered target location for the joint. Many paths can be expected for such a function (Fig. 5).

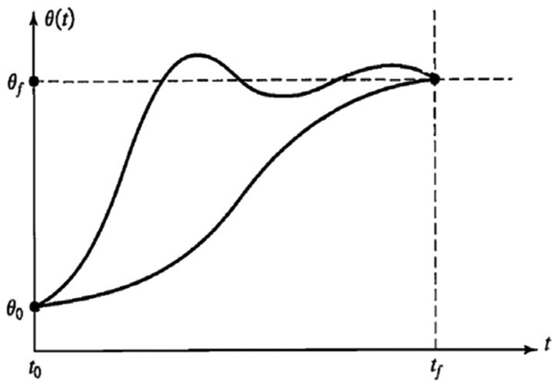

Fig. 5. Some instances of possible paths.

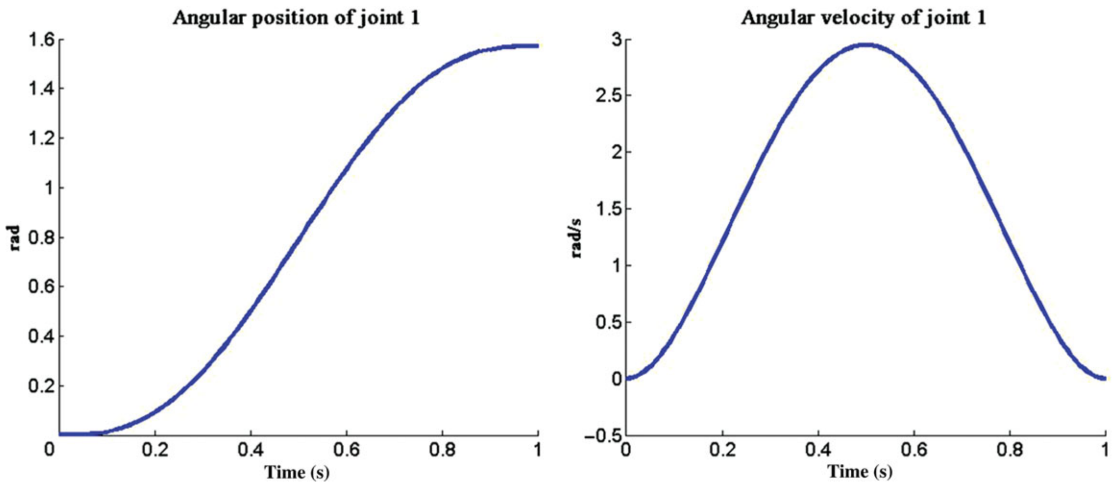

Fig. 6. (Colour online) Displacement and speed of the first joint.

If the accelerations and velocities at the starting and ending points of the path are given, a polynomial is used to produce the initial guess as the optimization algorithm initialization. Figure 6 shows an example; the initial and final points are 0 and $3\pi/4$, and the initial and final speed and acceleration are considered as zero.

The same action for every independent degree of freedom is repeated. For generating velocities, derivatives of position curves are taken.

3.3. *Input control variable production*

After generating the path for all independent path variables, the system's dynamic equations are used to generate the needed input torque (U) for each joint (Appendix A and B). Figure 7 is obtained for the first joint.

3.4. *The optimization phase*

During the algorithm, the input (U) must be mutated. The $e \times m$ matrix of U is mutated using the AIS algorithm (e is the number of inputs and m is the number of internal points from start to end). In order for U to remain smooth after mutation, some spots are used. The spots are some random points. The spots will be dealt as weight spots, and to create the controlling U's curve, a resultant curve of these spots is used. For instance, Fig. 8 shows 10 random points in the interval [0, 1] and a polynomial regression fitted to the data set. In the case of muting U, the points are selected in order to be close to U's curve.

After changing U, all its dependent and related variables will change. Now with the new U, we face a problem with specific split boundary value problem in two distinct points. In this phase the following two solutions can be used:

(a) Reach an acceptable solution by solving the problem with split BVP at two distinct points.

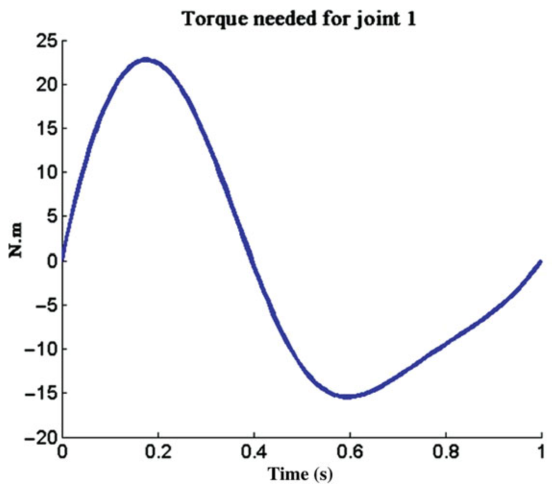

Fig. 7. (Colour online) The joint's torque.

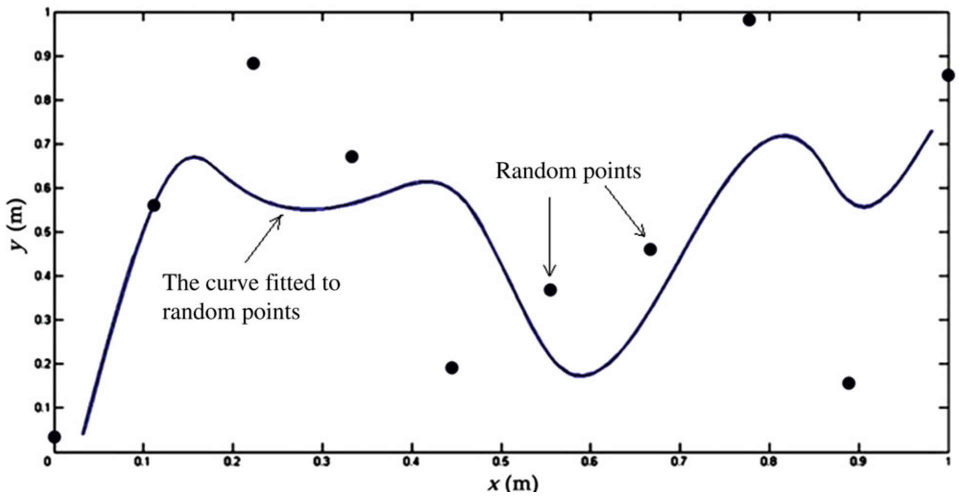

Fig. 8. (Colour online) A polynomial regression fit to the data set (10 random points in the interval [0, 1]).

The basic problem of this approach is the limitation of solving a BVP problem. There is no analytical method for nonlinear BVP problem and these problems must be solved numerically.

There are two other limitations in BVP solving method: (a) The high sensitivity of solving algorithms to the initial estimation; and (b) the deadlocks that may occur during the solving process like singularity. Because of these limitations it is not possible to use this method in repetitive algorithms since the hypotheses of the algorithm are changing constantly and it will be stopped at the middle of the work.

(b) The problem is treated as an IVP with initial conditions at one point.

Since the desired ending points will not be achieved, a method for an appropriate mutation, "multi-phase mutation," and avoiding the local trap is used. To avoid local optimization following instruction is proposed.

Instruction: Consider the assumptive t_b point in interval (t_a, t_c) in Fig. 9. The goal is reaching the θ_{fd} (the dots). After mutation of U, the curve leans toward θ_f (the solid line). In this phase by using the previously noted 5-degree polynomial, a curve is drawn from t_b (the new start point) to θ_{fd} (the end point) when the speed and acceleration are definite in t_b and t_c (the dash line in Fig. 9). Creating a 5-degree polynomial is discussed in Appendix C. The last phase is named as "revising the finals." The summary of the method is as follows: First, the final desired point is obtained, second, the speed and acceleration at the starting and ending points can be easily controlled, and finally a quite smooth path is the result (Fig. 9).

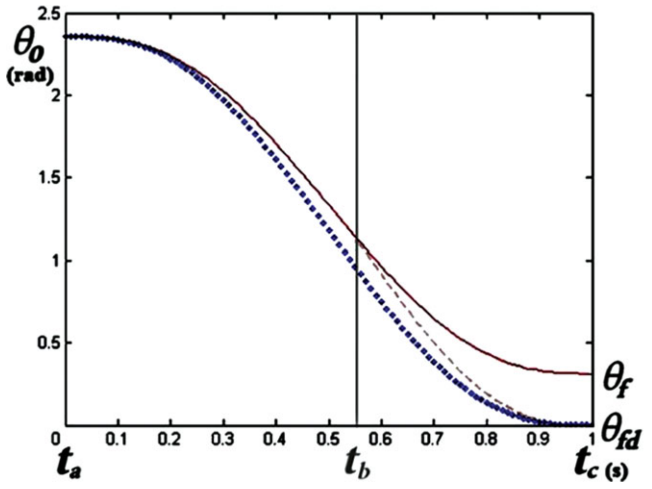

Fig. 9. (Colour online) How to correct the extremity of the path after muting U.

The value of break point (t_b) is chosen by the AIS algorithm. Hence, the profile is broken into two sections: mutation and revision. In the mutation of U, optimality is achieved, and in the revision the path is directed to reach the final point. With this approach it is possible to have both the optimality and the reaching to the final points simultaneously. The other advantage of this method is that all of the resultant paths in each phase are valid, since revising the final points happens in all the steps. Accordingly, it will be possible to find a more optimized path in each phase. After revising the path, we come back to the first step with a much better estimation than the initial one and again repeat the process up to the point the algorithm stops.

3.5. Generalizing the algorithm to non-holonomic and complicated robots

Increasing the dimension of the system leads to increase the following parameters: (a) Inputs (U), and (b) the t_b point for each one of the relocating curves, which keeps the information about the angular speed and acceleration. Summation of these parameters for a single degree of freedom is not a large number. Superiority of the evolutionary algorithms must be judged in a comparative way. In the most evolutionary algorithms, increasing the dimension of the system is equal to adding so many points in the system's motion curves per each degree of freedom. Nevertheless, in the proposed method in the algorithm, the increased element is only one point in the each motion curve (t_b). This means that by increasing degrees of freedom the problem develops linearly. In the other explanation it means absence of numerical explosion. For non-holonomic systems, optimization on the controlled parameters is performed and the dependent parameters of the system are simply implemented using the system's dynamic equations. In the simulation results, a non-holonomic system with 5 degrees of freedom (DOF) is implemented.

4. Simulation Results and Comparison

4.1. Optimal search performance benchmarking

In order to study the performance of CCSA-WM, it is compared with some other evolutionary algorithms. Table I demonstrates benchmark functions that are used to prove the performance of CCSA-WM. It is compared with HAIS and standard GA[16] for Sphere and Rosenbroke functions.

These algorithms were performed with 500 generations over 30 trials. Tables II and III show simulation parameters and results of CCSA-WM, HAIS and GA, respectively.

Table III shows that CCSA-WM has better results than HAIS and GA in Sphere and Rosenbroke benchmark functions (in terms of mean and variance results). Therefore, CCSA-WM is more precise and accurate than HAIS and GA. Moreover, CCSA-WM uses smaller population size, so it leads to higher speed than HAIS and GA to obtain solution. The shape parameters ξ and g are optimized

Table I. The benchmark functions.

Test functions	Objective functions	No. of variables	Bound	Global optimal	Type
Sphere function	Min: $f(x) = \sum_{i=1}^{n} x_i^2$	30	$\lvert x_i \rvert \leq 10$	$x_i = 0$	Single objective
Rosenbroke function	Min: $f(x) = \sum_{i=1}^{n-1} [100 \times (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	5	$\lvert x_i \rvert \leq 2.048$	$x_i = 1$	Single objective
ZDT1	$f_1(x) = x_1;\ f_2(x) = g(x)[1 - \sqrt{x_1 / g(x)}]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	30	$0 \leq x_i \leq 1$	Convex	Multi-objective
Constraint_Ex	$f_1(x) = x_1$ $f_2(x) = (1 + x_2)/x_1$ subject to : $\begin{cases} g_1(x) = x_2 + 9x_1 \geq 6 \\ g_2(x) = -x_2 + 9x_1 \geq 1 \end{cases}$	2	$0.1 \leq x_1 \leq 1$ $0 \leq x_2 \leq 5$	For :$0.39 \leq x_1^* < 0.67$ $x_2^* = 6 - 9x_1^*$ For :$0.67 \leq x_1^* \leq 1$ $x_2^* = 0$	Multi-objective with constraints

Table II. The simulation parameters of CCSA-WM, HAIS, and GA for Sphere and Rosenbroke functions.

Parameters	CCSA-WM	HAIS	GA
Population size	5	25	50
Archive size	3	25	-
Crossover rate	-	0.5	0.5
Mutation rate	Adaptive	0.1	0.1

Table III. The mean and variance of results from CCSA-WM, HAIS, and GA for Sphere and Rosenbroke functions.

	Sphere function		Rosenbroke function	
Algorithms	Mean	Variance	Mean	Variance
CCSA-WM	5.52E-18	5.27e-37	8.81E-20	5.42e-39
HAIS	0.000161	3.54E-08	2.193986	1.758235
GA	306.7045	3438.832	38.84761	1800.696

Table IV. Optimized wavelet mutation parameters.

Benchmark functions	ξ	G
Sphere function	1.4296	$9.7417E+19$
Rosenbroke function	1.18	$6.49332E+19$
ZDT1	0.6583	$4.9070E+19$
Constraint_Ex	8.8320	$4.9500E+19$

Table V. Simulation parameters of CCSA-WM, HAIS, EMOCA, NSGA-II, SPEA-II, and PAE.

Methods	Population size	Archive size	Crossover rate	Mutation rate
CCSA-WM	5	3	-	Adaptive
HAIS	100	100	0.7	0.3
EMOCA	100	100	0.9	$1/n$[a]
NSGA-II	100	100	0.9	$1/n$
SPEA-II	100	100	0.9	$1/n$
PAES	100	100	0.9	$1/n$

[a]n: Number of decision variables.

by CCSA-WM, therefore Table IV shows optimized wavelet mutation parameters for benchmark functions.

As another comparison, ZDT1 and Constraint_Ex are used as multi-objective with constraint benchmark functions. In these cases the CCSA-WM is compared with HAIS, EMOCA, NSGA-II, SPEA-II, and PAE.[16] The simulation parameters for these methods are shown in Table V.

The multi-objective problems, such as ZDT1 and Constraint_Ex, generate a set of optimal front, and in this case to compare the performance of methods, generational distance[17] and spread metric[18] are used. The generation distance (GD) measures distance between the known Pareto front and the true Pareto optimal as in Eq. (8):

$$\mathrm{GD} = \sqrt{\sum_{i=1}^{m} d_i^2}, \tag{8}$$

where m is the number of solutions, and the parameter d_i denotes Euclidean distance between the solution of algorithm and the nearest solution of true Pareto optimal in objective space. Moreover,

Table VI. Mean and variance of generational distance over 30 trials for ZDT1 benchmark function.

Method	Mean	Variance
CCSA-WM	8.025E-4	2.10E-7
HAIS	0.024	4.9E-5
EMOCA	0.029	0.0082
NSGA-II	0.034	0.0055
SPEA-II	0.0432	0
PAES	0.032	1E-5

Table VII. Mean and variance of generational distance over 10 trials for Constraint_Ex benchmark function.

	Generational distance		Spread metric	
Method	Mean	Variance	Mean	Variance
CCSA-WM	1.019E-6	2.28E-12	0.3992	0.0093
HAIS	0.61929	0.01225	1.20124	0.0018

Table VIII. Physical statistics of a three-link holonomic manipulator.

Parameter	Value	Unit
Length of links	$l_1 = 0.4$, $l_2 = 0.5$, $l_3 = 0.5$	m
Mass of links	$m_1 = 12$, $m_2 = 10$, $m_3 = 5$	kg
Moment of inertia of links	$I_1 = 0.16$, $I_2 = 0.2083$, $I_3 = 0.1042$	kg m^2

the diversity of solutions is measured by spread metric,

$$\Delta = \frac{\sum_{l=1}^{L} d_l^e + \sum_{i=1}^{m} \left| d_i - \bar{d} \right|}{\sum_{l=1}^{L} d_l^e + m \times \bar{d}}. \tag{9}$$

The parameter $d_l^e$is the Euclidean distance between extreme solution of the Pareto optimal front and the nearest set of solutions corresponding to the nth objective. A smaller value of Δ demonstrates a more uniform spaced set of solution. Table VI shows the results of ZDT1 benchmark problem over 30 trials.

The results of Constraint_Ex, which are obtained over 10 trials, are presented in Table VII.

As shown in Tables VI and VII, the CCSA-WM has more accuracy and precision in comparison to other algorithms. Furthermore, the CCSA-WM utilizes the lowest population size, therefore the speed of obtaining solutions increases. Consequently, the CCSA-WM obtains accurate and precise solutions in single and multi-objective (with constraint) problems with minimum elapsed time by using the lowest population size.

4.2. Path planning results for the three-link planar holonomic manipulator

Suppose the three-link planar robotic arm with parameters according to Table VIII starts to move from configuration $[0, 0, -\pi/2]$ to $[\pi/2, \pi/4, -\pi]$ and the speed and angular acceleration at the beginning and end of the path are considered as zero. The obtained results for states and inputs are shown in Fig. 10.

4.3. Path planning of non-holonomic mobile manipulator with 5-DOF

This method can be applied on systems with high degrees of freedom. As an example, the proposed algorithm is applied for a non-holonomic two-link mobile manipulator with 5-DOF. The robot's parameters are given in Table IX.

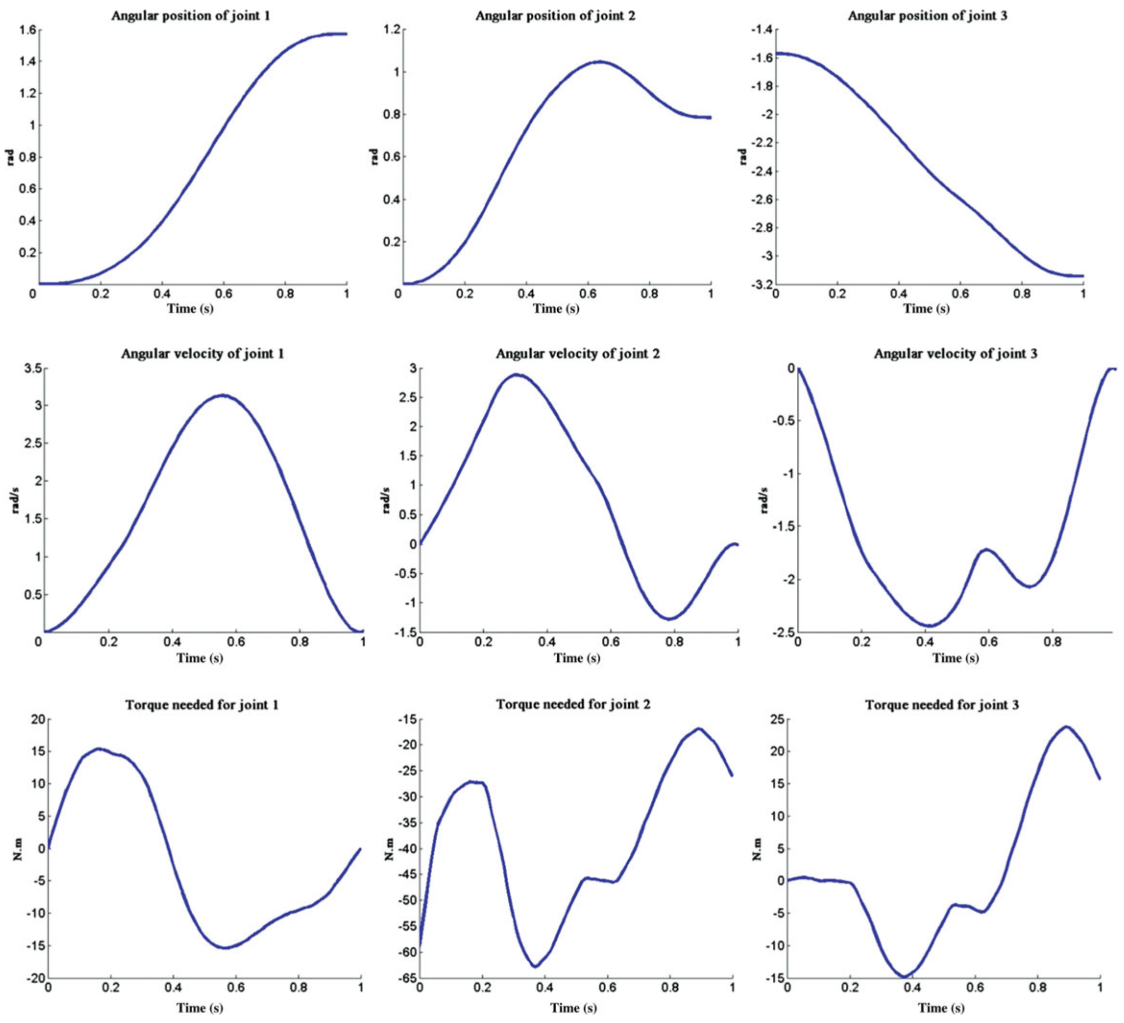

Fig. 10. (Colour online) The displacements, speeds, and torques of a three-link holonomic manipulator.

All results are compared with the method developed in Das *et al.*,[8] which is based on optimal control. The starting configuration is$[X_0,\ Y_{0,0},\ \theta_{r0},\ \theta_{l0},\ \theta_{10},\ \theta_{20},\ \dot{\theta}_{r0},\ \dot{\theta}_{l0},\ \dot{\theta}_{10},\ \dot{\theta}_{20}] = \left[0,\ 0,\ 0,\ 0,\ 0,\ \frac{\pi}{2}-0.1,\ -\pi+0.1,\ 0,\ 0,\ 0,\ 0\right]$, and the end configuration is

$$[X_f, Y_{f,f}, \theta_{rf}, \theta_{lf}, \theta_{1f}, \theta_{2f}, \dot{\theta}_{rf}, \dot{\theta}_{lf}, \dot{\theta}_{1f}, \dot{\theta}_{2f}] = [0.8,\ -0.35,\ -\frac{\pi}{4},\ \times, \times, 0,\ 0,\ 0,\ 0,\ 0,\ 0].$$

The obtained results for the angular positions of wheels and joints, angular velocity of wheels and joints as well as the torque needed for each joint and wheel are shown in Figs. 11–13.

Equation (10) shows the norm of all input torques as a cost function in $m = 3$ s with $k = 101$ points,

$$\sum_{\substack{i\ =\ 1\\ j\ =\ 0\\ k}}^{\substack{i\ =\ e\\ j\ =\ m}} U^2_{(i,j)} \qquad \begin{cases} i = 1, 2, \ldots, e \\ j = 0,\ \text{step},\ 2\ \times\ \text{step}, \ldots, m \end{cases}. \tag{10}$$

In this paper minimizing the needed torque integrals is proposed. Hence, Eq. (10) is used as the cost function. Size of the steps is considered to be 0.03 s and the number of input torques is four

Table IX. Physical parameters of a non-holonomic two-link mobile manipulator.

Parameter	Value	Unit
Length of links	$l_1 = l_2 = 0.5$	m
Mass of links 1, 2	$m_1 = 5, m_2 = 3$	kg
Moment of inertia of links 1, 2	$I_1 = 0.416, I_2 = 0.0625$	kg m^2
Mass of base	94	kg
Mass of wheels	5	kg
Moment of inertia of base	6.609	kg m^2
Moment of inertia of wheels	0.01	kg m^2
*b**	0.171	m
*r**	0.075	m
*d**	0	m

**b* is the distance of wheels, *r* is the radius of wheels, and *d* is the distance between wheels and the center of mass.[13]

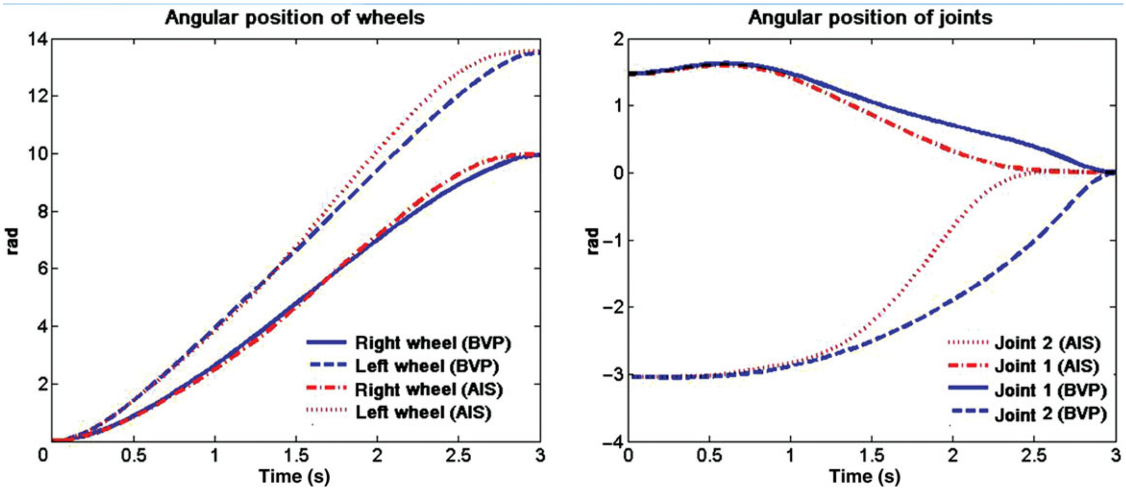

Fig. 11. (Colour online) Angular position of wheels and joints.

($e = 4$). Here a limitation is applied to the system to settle down the system at the end of the process. This limitation results in increasing of cost. The amount of velocities and accelerations had to be zero at the end of the scope whereas the BVP method had no limitation. Nonetheless, as shown in the Fig. 13, the cost related to the method developed in Das *et al.*[8] is equal to 0.2098, while the cost related to the presented method equals to 0.1522, which is 27.45% better than results of Das *et al.*[8]

5. Experimental Test

To verify the applicability of simulation results, the proposed method is implemented on a non-holonomic wheeled mobile robot, named K-joniour. The robot is controlled using a Peripheral Interface Controller (PIC) microcontroller. Fig. 14 shows K-joniour mobile robot.

The kinematics and the kinetic model are derived, and optimization and programming are done and implemented on the robot. The characteristics of the parts of K-junior are shown in Table X.

To test the robot, test bed and plot are designed to run the robot 10 times from the start to end points. Suppose that initially the mobile base with coordinates is P0 ($x = 0$ m, $y = 0$ m, $\varphi = 0$ rad), and it must reach the point pf P0 ($x = 0.7$ m, $y = 0.4$ m, $\varphi = 0$ rad) at $tf = 5$ s. For experimental setup, offline data from simulations are used as input for the microcontroller program. The experiments are repeated 10 times, and the mean values as well as desired path are plotted (shown in Fig. 15). In each run, one set of wheels' velocity is read and sent to the robot's motion controller board. The path of mobile robot resulted from simulations in comparison with experimental test is shown in Fig. 15.

As shown in Fig. 15, similar path resulted from simulations is tracked. In the experimental implementation, the maximum value of deviation from simulation results is 0.07 m.

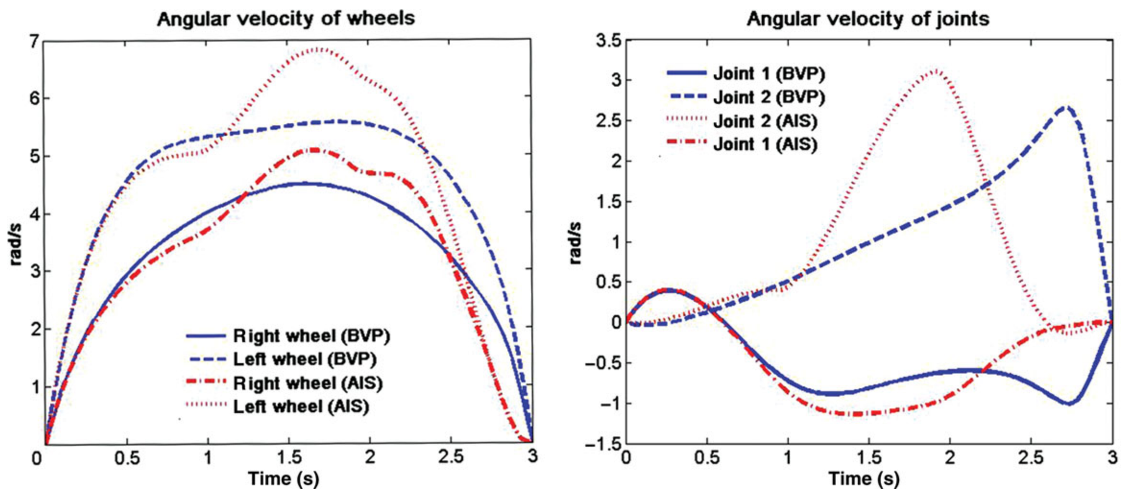

Fig. 12. (Colour online) Angular velocity of wheels and joints.

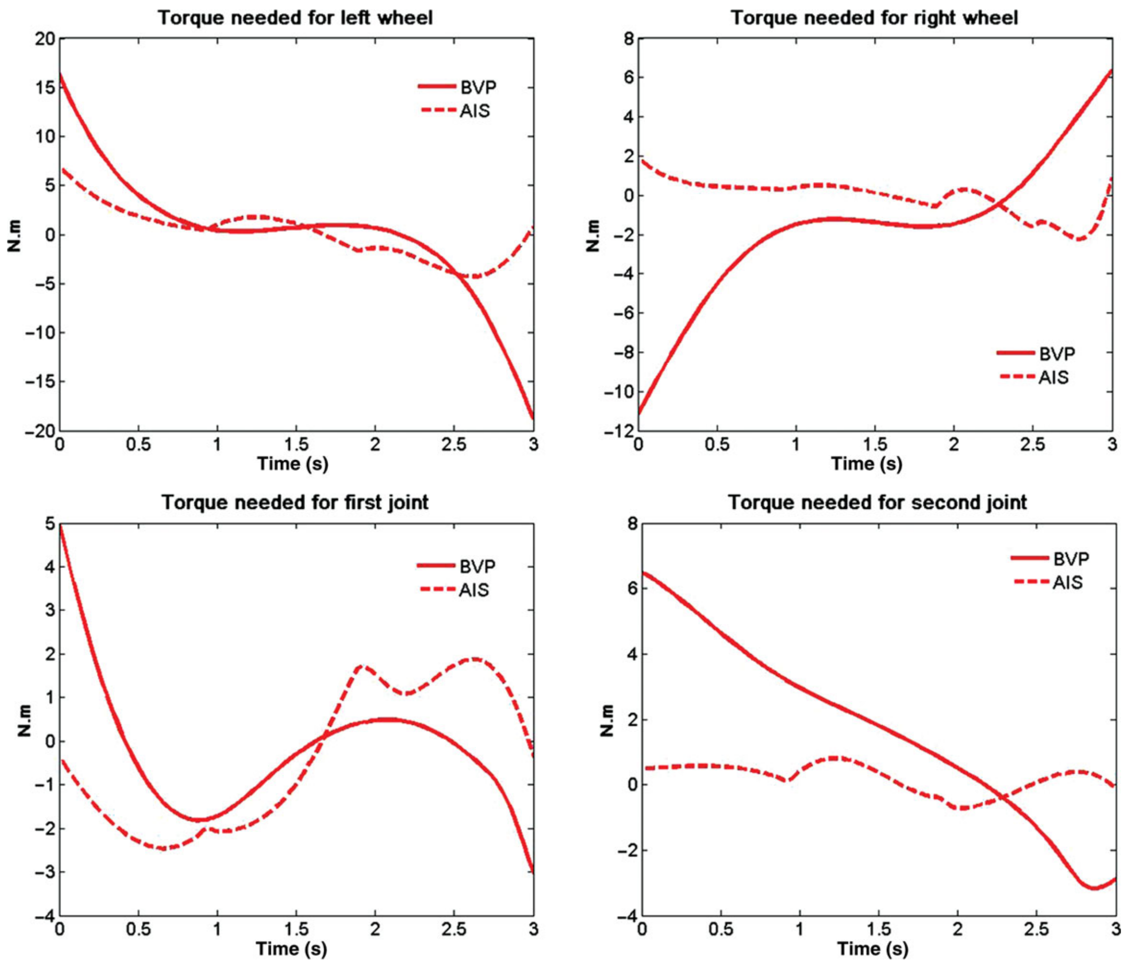

Fig. 13. (Colour online) Torque needed for joints and wheels.

6. Conclusions

A new optimal path planning procedure based on a new evolutionary algorithm is presented for higher order robotic systems. It does not suffer from other shortages of evolutionary algorithms such as numerical explosion, because by increasing the system's dimension, the complexity of algorithm grows linearly. Hence, the new AIS algorithm based on wavelet mutation is explained. The main idea

Table X. Parameters and inertia properties of K-joniour robot.

Parameter	Value	Unit
Mass of base	0.60	kg
b^*	0.07	m
r^*	0.01	m
d^*	0.02	m

$^*b$ is the distance of wheels, r is the radius of wheels, and d is the distance between wheels and the center of mass.[13]

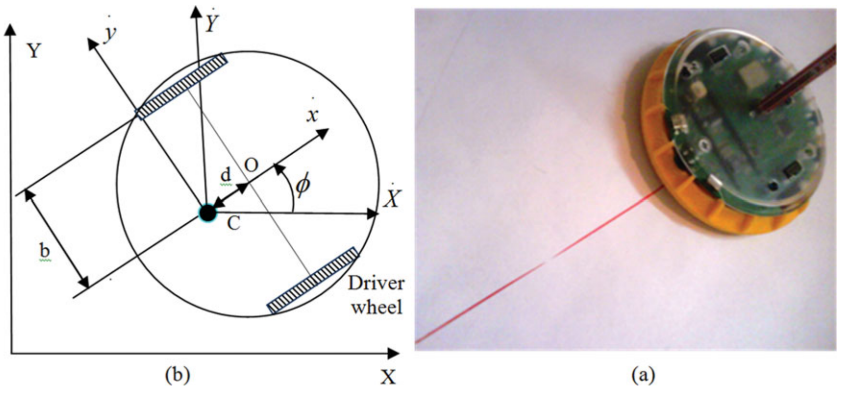

Fig. 14. (Colour online) Experimental system: (a) K-joniour robot, (b) schematic of the robot and path generating process.

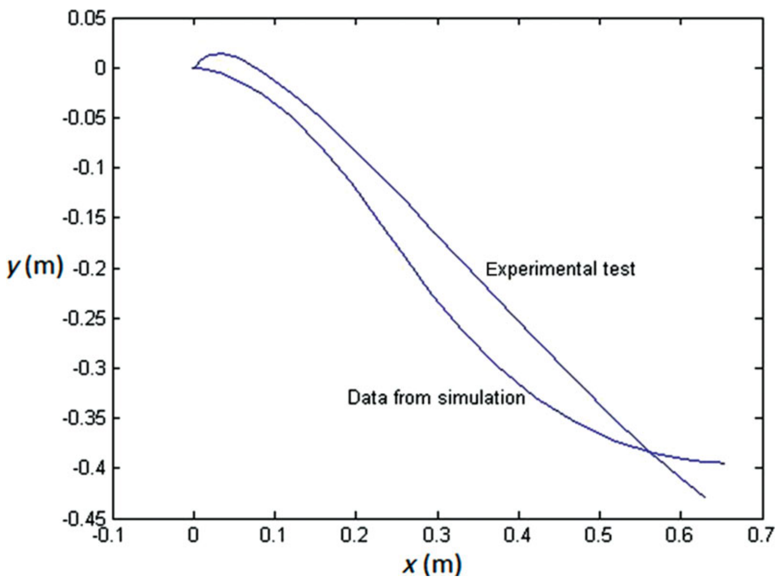

Fig. 15. (Colour online) Comparison of path of the center of wheels in experimental test and simulation.

is to break the input profile into two subsections and specifying the location of break point as an anti-body by using CCSA-WM. According to the results of the simulations, it has very significant excellences over other similar algorithms. In this algorithm, by using wavelet mutation whole of the space search is covered properly in CCSA. Therefore, repeatability in convergence to the solution is achieved. Moreover, the simple view of procedure in CCSA leads to the solution by the lowest population size, so the speed of convergence is increased. Then the path-planning problem was clarified by the new method. The results of simulations were compared with other methods, which showed effective decrease in the cost function. In addition, the proposed algorithm is compared with an optimal control-based algorithm. The results showed about 27% decrease in the cost function. The novelties and benefits of the proposed approach and its superiority regarding other approaches are as follow:

1. The proposed approach is very simple in comparison to other similar algorithms in this field. Especially in the case of the high-order systems, the path planning methods are mostly complicated. Moreover, the algorithm is completely based on an evolutionary algorithm. Therefore, the algorithm can be performed on simple hardware with very simple structure.
2. Complicated mathematics is not used in the proposed approach. Nevertheless, according to the cost function, the results of the optimizations are better than other optimization methods in this field.
3. The BVP-based solving methods have some serious problems. (a) The high sensitivity of solving algorithms to the initial estimation, (b) the dead locks that may occur during the solving process like singularity. In the proposed approach, these problems are eliminated. Solution of BVP is sensitive to initial guess,[13] but solution of IVP is insensitive to initial guess. In this paper a method based on IVP is used to solve BVP. On the other hand, in BVPs, the wrong initial guess leads to singularity errors. Using IVP in the proposed method, no doubt the problem is solved with the boundary points. They are amended in another phase of the algorithm. Hence, there is no worry about the singularity errors. Besides, the IVP is solved many times in AIS iterations. A singularity error within iterations would be a deadlock to the whole process and the outcomes of the method.
4. Despite other evolutionary algorithms, by increasing the degree of freedom, the problem develops linearly. Therefore, numerical explosion does not occur for path planning of large-scale robots.

There has been no practical analytical method for nonlinear boundary value problems, so these problems should be solved numerically. Moreover, the evolutionary methods do not guarantee the optimality and convergence to global extreme and superiority of these methods must be judged in a comparative way. However, according to the resulted variance in the CCSA-WM method and the simulation results with standard benchmarks, the CCSA-WM is a more reliable algorithm in comparison with other similar methods. In addition, the path planning method is applied to a 5-DOF system mobile manipulator and compared with an optimal control-based algorithm. The simulation showed that the cost function has 27.45% better results than optimal control. Finally, its applicability is proved by experimental implementation on a mobile robot.

References

1. P. Raja and S. Pugazhenthi, “Optimal path planning of mobile robots: A review,” *Int. J. Phys. Sci.* **7**(9), 1314–1320 (2012).
2. L. Moreno, J. M. Armingol, S. Garrido, A. De La Escalera and M. A. Salichs, “A genetic algorithm for mobile robot localization using ultrasonic sensors,” *J. Intell. Robot. Syst.* **34**, 135–154 (2002).
3. M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepulveda and P. Melin, “Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation,” *Appl. Soft Comput.* **9**, 1102–1110 (2009).
4. W. Xianxiang, G. Baolong and W. Juan, “Mobile robot path planning algorithm based on particle swarm optimization of cubic splines,” *Robot* **31**(6), 556–560 (2009).
5. J. S. Gyorfi, D. R. Gamota, S. M. Mok, J. B. Szczech, M. Toloo and J. Zhang, “Evolutionary path planning with subpath constraints,” IEEE Trans. Electron. Packag. Manuf. **33**, 143–151 (2010).
6. T. Ching-Chih, H. Hsu-Chih and C. Cheng-Kai, “Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation,” *IEEE Trans. Ind. Electron.* **58**, 4813–4821 (2011).
7. Y. Bin Hou, W. Wang and X. Yue Lu, “Mobile robot path planning and research in the improved artificial immune algorithm,” *Adv. Mater. Res.* **466**, 864–869 (2012).
8. P. K. Das, S. K. Pradhan, S. N. Patro and B. K. Balabantaray, “Artificial immune system-based path planning of mobile robot,” *Stud. Comput. Intell.* **395**, 195–207 (2012).
9. L. N. de Castro and F. J. Von Zuben, “Immune-inspired somatic contiguous hypermutation for function optimization,” *IEEE Trans. Evolut. Comput.* **6**, 239–251(2002).
10. S. Ling and F. Leung, “An improved genetic algorithm with average-bound crossover and wavelet mutation operations,” *Soft Comput.* **11**(1), 7–31 (2007).
11. J. Lai, F. Leung and S. Ling, “A New Differential Evolution with Wavelet Theory-Based Mutation Operation,” **In**: *Proceedings of the IEEE Congress on Evolutionary Computation* (Trondheim, 2009) pp. 1116–1122.
12. S. Asadi, V. Azimirad and A. Eslami, “A Novel Real-time Global Optimal Path Planning and Trajectory Method Based on Adaptive Dijkstra-Immune Approach for Mobile Robot,” **In**: *Proceedings of the IEEE International Conference on Advanced Intelligent Mechatronics (AIM)* (Budapest, 2011) pp. 1093–1098.

13. M. H. Korayem, V. Azimirad, H. Vatanjou and A. H. Korayem, "Maximum load determination of nonholonomic mobile manipulator using hierarchical optimal control," *Robotica* **30**, 53–65 (2011).
14. I. Daubechies, *Ten Lectures on Wavelets* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992).
15. J. Crage, John, *Introduction to Robotics-Mechancs and Control* (Pearson, Upper Saddle River, NJ, 2005).
16. E. Y. C. Wong, H. S. C. Yeung and H. Y. K. Lau, "Immunity-based hybrid evolutionary algorithm for multi-objective optimization in global container repositioning," *Eng. Appl. Artif. Intell.* (Elsevier), **22**, 842–854 (2009).
17. D. A. V. Veldhuizen and G. B. Lamont, "On Measuring Multiobjective Evolutionary Algorithm Performance," **In**: *Proceedings of Evolutionary Computation Congress* (La Jolla, CA, 2000) pp. 204–211.
18. K. Deb, A. Pratap, S. Agarwal and T. A. M. T. Meyarivan, "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II," *IEEE Trans. Evolut. Comput.* **6**(2), 182–197 (2002).

Appendix A: Dynamic modeling of the n-link planar serial manipulator

The n-link manipulator could be considered as a large-scale system, which is shown in Fig. 16 and presented in Eq. (11) as Lagrangian form.

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = \tau, \tag{11}$$

where $q, \dot{q}, \ddot{q} \in R^n$ are angular position, velocity, and accelerator of joints respectively. The $M(q) \in R^{n\times n}$ is symmetric and positive definite inertia matrix and $C(q, \dot{q}) \in R^{n\times n}$is a matrix of coriolis and centrifugal terms. The state space form of two-link manipulator is written as Eq. (12):

$$X_1 = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix},$$

$$\begin{cases} \dot{X}_1 = X_2 \\ \dot{X}_2 = M(q)^{-1} \times [\tau - G(q) - C(q, \dot{q})X_2] \end{cases}. \tag{12}$$

Appendix B: Dynamic modeling of the non-holonomic mobile manipulator

Mobile manipulator is a combination of a mobile platform and a fixed manipulator mounted on it (Fig. 17).

In order to optimize trajectory of mobile manipulator, the dynamic equations of motions are needed. The overall motion equations are Eqs. (13) and (14). Details could be found in Korayem *et al.*,[13]

$$M_{v1}(q_v)\ddot{q}_v+V_{v1}(q_v, \dot{q}_v)+V_{v2}(q_r, q_v, \dot{q}_r, \dot{q}_v)=E_v\tau_v-A^T\lambda-M_{v2}(q_r, q_v).\ddot{q}_v-R_v(q_r, q_v)\ddot{q}_r, \tag{13}$$

where M_{v1} and V_{v1} are the mass inertia and velocity dependant of the platform respectively, M_{v2} and V_{v2} represent the inertial term and centrifugal and coriolis terms due to the presence of manipulator, τ_v is the input torque to the vehicle, E_v is a constant matrix, λ denotes the Lagrange multipliers

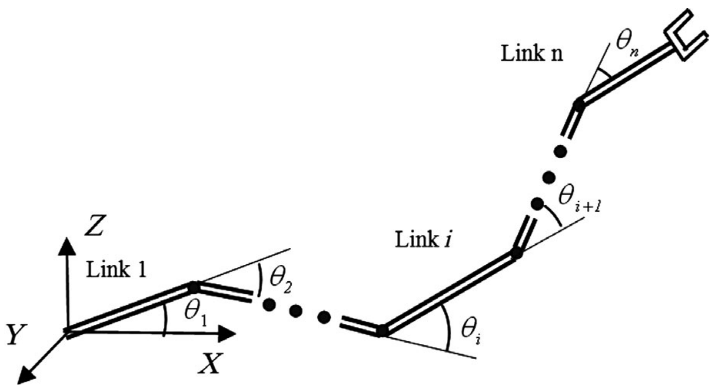

Fig. 16. Serial link manipulator.

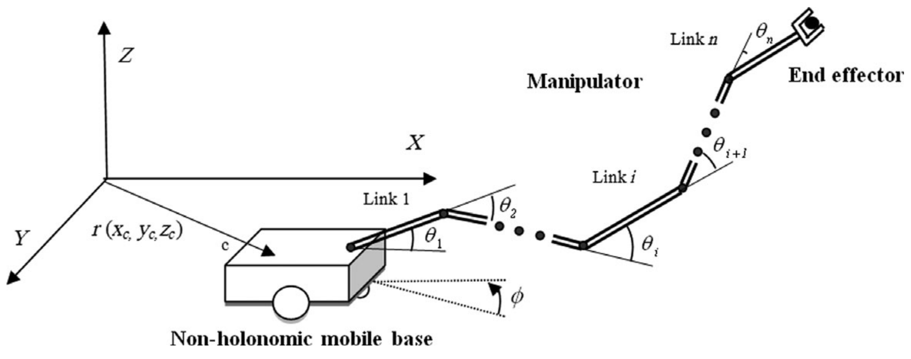

Fig. 17. Wheeled mobile platform with a serial manipulator.

corresponding to the kinematic constraints, and R_v represents the inertia matrix, which reflects dynamical effect of arm motion on the vehicle. Also, dynamic equation for the mounted arm is

$$M_r(q_r)\ddot{q}_r + V_{r1}(q_r, \dot{q}_r) + V_{r2}(q_r, \dot{q}_r, \dot{q}_v) = \tau_r - R_r(q_r, q_v)\ddot{q}_v, \tag{14}$$

where q_r denotes the n-dimensional coordinates of the manipulator, M_r is the inertial matrix, V_{r1} represents the nonlinear terms, V_{r2} denotes the centrifugal and coriolis terms caused by the angular motion of the platform, τ_r is the torque of the manipulator, and R_r is the inertial matrix that represents effects of the vehicle motion on the manipulator. The dynamic equations of non-holonomic mobile manipulator are represented in the state space form as follows:[13]

$$\dot{X} = \begin{bmatrix} Sv \\ (S^T MS)^{-1}(-S^T M\dot{S}v - S^T V) \end{bmatrix} + \begin{bmatrix} 0 \\ (S^T MS)^{-1} \end{bmatrix} \tau, \tag{15}$$

where $X = \begin{bmatrix} q^T & v^T \end{bmatrix}^T$ is the state vector of the system and is equal to

$$q = \begin{bmatrix} X & Y & \varphi & \theta_r & \theta_l & \theta_1 & \cdots & \theta_n \end{bmatrix}, \quad v = \begin{bmatrix} \dot{\theta}_r & \dot{\theta}_l & \dot{\theta}_1 & \cdots & \dot{\theta}_n \end{bmatrix}, \tag{16}$$

where M is the mass matrix, and V is related to nonlinear terms.

Appendix C: Creating a 5-degree polynomial

It is supposed that the acceleration and velocity at the start and end of the path are given as in Eq. (17):

$$\begin{aligned} &\theta(0) = \theta_0, \quad \dot{\theta}(0) = \dot{\theta}_0, \ddot{\theta}(0) = \ddot{\theta}_0, \\ &\theta(t_f) = \theta_f, \ \dot{\theta}(t_f) = \dot{\theta}_f, \ddot{\theta}(t_f) = \ddot{\theta}_f. \end{aligned} \tag{17}$$

Therefore, the 5-degree polynomial is as in Eq. (18), and the speed and acceleration can be calculated by Eqs. (19) and (20):

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5, \tag{18}$$

$$\dot{\theta}(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4, \tag{19}$$

$$\ddot{\theta}(t) = 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3. \tag{20}$$

Combination of Eqs. (17)–(20) leads to the system of equations as Eq. (21):

$$\begin{array}{l} \theta\left(0\right)=a_0,\ \dot{\theta}\left(0\right)=a_1, \ddot{\theta}\left(0\right)=2a_2 \\ \theta\left(t_f\right)=a_0+a_1t_f+a_2t_f^2+a_3t_f^3+a_{42}t_f^4+a_5t_f^5 \\ \dot{\theta}\left(t_f\right)=a_1+2a_2t_f+3a_3t_f^2+4a_4t_f^4+5a_5t_f^5 \\ \ddot{\theta}\left(t_f\right)=2a_2+6a_3t_f+12a_4t_f^2+20a_5t_f^3 \end{array}. \tag{21}$$

After solving the above system of equations, the results could be presented as Eq. (22):

$$\begin{array}{l} a_0=\theta_0, a_1=\dot{\theta}_0, a_2=\ddot{\theta}_0/2 \\ a_3=(20\theta_f-20\theta_0-\left(8\dot{\theta}_f+12\dot{\theta}_0\right)t_f-(3\ddot{\theta}_0-\ddot{\theta}_f)t_f^2)/2t_f^3 \\ a_4=(30\theta_0-30\theta_f+\left(14\dot{\theta}_f+16\dot{\theta}_0\right)t_f+(3\ddot{\theta}_0-2\ddot{\theta}_f)t_f^2)/2t_f^4 \\ a_5=(12\theta_f-12\theta_0-\left(6\dot{\theta}_f+6\dot{\theta}_0\right)t_f-(\ddot{\theta}_0-\ddot{\theta}_f)t_f^2)/2t_f^5 \end{array}. \tag{22}$$