Applied Mathematics and Computation 217 (2011) 8997–9008

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

On solving the forward kinematics of 3RPR planar parallel manipulator using hybrid metaheuristics

Rohitash Chandra [a,*], Luc Rolland [b]

[a] Department of Computing Science and Information Systems, College of Business, Hospitality and Tourism Studies, Fiji National University, Suva, Fiji
[b] Faculty of Engineering, Toros University, Mersin, Turkey

ARTICLE INFO

Keywords:
Forward kinematics
Planar parallel manipulators
Real-coded genetic algorithm
Simulated annealing
Teamwork collaborative hybrid metaheuristics
Relay collaborative hybrid metaheuristics

ABSTRACT

Hybrid metaheuristics have been applied with success in solving many real-world problems. This work introduces hybrid metaheuristics to the field of kinematics problem, in particular, for solving the forward kinematics of the 3RPR parallel manipulator. It implements a combination of genetic algorithms and simulated annealing into two popular hybrid metaheuristic techniques. They are combined as teamwork and relay collaborative hybrid metaheuristics and compared to the performance of genetic algorithms and simulated annealing alone. The results show that the meta-heuristic approaches give robust and high quality solutions. Genetic algorithms and teamwork collaborative metaheuristics showed better performance than simulated annealing and relay collaborative metaheuristics. The given metaheuristic methods obtain all the unique solutions and comparisons with algebraic methods show promising results.

1. Introduction

The use of metaheuristic search algorithms have gained attention in the recent decades due to the failure of conventional methods to solve NP-hard combinatorial and difficult real-world optimization problems. Metaheuristic search techniques such as genetic algorithms and simulated annealing have been applied to a number of real-world application problems. Evolutionary computation is solely inspired from biological evolution and uses a population of solutions which are evolved and improved. Evolutionary algorithms are often called *metaheuristic* algorithms which are computational methods that optimize a problem by iteratively improving the solution [1].

Evolutionary algorithms have shown their strength in obtaining solutions of high accuracy in optimization problems [2,3], therefore, it is reasonable to apply them in solving problems that involve non-linear equation systems which are still an open problem. In robotics, genetic algorithms have been applied for solving the forward kinematics of parallel manipulators (FKP). Initial work was done by Boudreau and Turkkan who used a real-coded genetic algorithm (RCGA) for solving the FKP of 3-RPR manipulators [4] where it was reported that genetic algorithms are more time consuming than Newton–Raphson's method [5]. Newton's method suffers from Jacobian inversion problems, numeric instabilities related to the application of floating numbers, and convergence problems when not provided with a good initial position in the search space; therefore, evolutionary algorithms are preferred. The FKP of planar and spatial manipulators has also been approached with genetic algorithms, the 3RPR by Boudreau and Turkkan [6] and the ideal Gough platform (SSM) by Omran et. al. [7]. It has been reported that the differential evolution algorithm converges to a solution relatively faster than binary-coded genetic algorithm for the SSM hexapod manipulator [8].

* Corresponding author.
E-mail address: rohitash_c@yahoo.com (R. Chandra).

0096-3003/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2011.03.106

Robot kinematics problems are mostly associated with solving a system of non-linear equations and are rarely treated as a direct optimization problem. Therefore, the problem of solving a system of non-linear equation has to be converted into an optimization problem where the objective function describes the entire robot system kinematics [6,8].

The simulated annealing algorithm is a stochastic search method which performs a random global search during the initial stage and local greedy search during the later stage. Recently, the use of simulated annealing has decreased as population based genetic algorithms are more popular in real-world application problems. This work shows the contribution of simulated annealing in solving the FKP of 3RPR manipulators and compares its results with genetic algorithms. We also solve the problem by combining genetic algorithm and simulated annealing into two different hybrid metaheuristic approaches which are called *teamwork* and *relay* collaborative hybrid metaheuristics. The results obtained from the metaheuristic methods are verified against that of the algebraic methodology used by Rolland [9]. Preliminary results have been presented in [10] and a substantial extension of the results are given in this work.

The rest of the paper is organized as follows: In Section 2 an overview of the hybrid metaheuristic techniques with emphasis on genetic algorithms and simulated annealing is presented. Section 3 gives the details of the hybrid metaheuristic techniques used in this study. Section 4 presents the forward kinematics problem of parallel manipulators and its conversion into an optimization problem. Section 5 presents the results, their analysis and discussion while Section 6 concludes the paper with directions for future research.

2. Background: hybrid metaheuristics

2.1. Evolutionary algorithms

Evolutionary algorithms are nature inspired computational paradigms which use a population of candidate solutions that are improved over time with evolutionary operators. The major evolutionary algorithms are genetic algorithms [2,3], evolution strategies [11], evolutionary programming and genetic programming [12]. The similarity between most evolutionary algorithms is that they use a population of solutions and the differences lie in the representation of the problem and the way new solutions are created. Evolutionary algorithms are considered to be global search methods and can be used for problems that are noisy and change over time. Evolutionary algorithms are problem independent. They can be applied to different problems irrespective of their nature and hence can be used for problems that are non-differentiable. The term *generation* is mainly used for an iteration in which a population of new individuals are created in the evolutionary process.

Over the years, the original genetic algorithm (GA) introduced by Holland [2] has been modified significantly in order to suit the real-world optimization challenges faced by engineers and scientists. Traditionally, candidate solutions used binary encoding. One important alteration is the introduction of real coded genetic algorithms (RCGA) which implement real numbers in their candidate solutions rather than binary encoding. Real-valued encoding use representations that are easier to understand and useful in engineering applications.

Genetic algorithms use the population of candidate solutions that are called *individuals or phenotypes*. The new solutions are called *offspring* which are built from combining candidate solutions that are known as *parents*. Each individual is evaluated according to its *fitness*. The RCGA employs *genetic operators* such as *selection, crossover* and *mutation* to create offspring.

The basic idea used in RCGA optimization is given in Algorithm 1. Initially, a number of candidate solutions constitute a population. After each generation, the algorithm evaluates each individuals according to its fitness and employs genetic operators to produce offspring from selected parents. The fitness function measures solution quality which is problem dependent. The offspring are added into the population while sometimes, least fit individuals are discarded. The process is repeated until the algorithm obtains a sufficiently good solution.

Algorithm 1. Real-Coded Genetic Algorithm

Initialize Population (P)
Evaluate fitness
while Not Termination **do**
  for each *Individual* in P **do**
1. Evaluate fitness
2. Select *Parents*
3. Apply *Crossover* and produce *Offspring*
4. *Mutate* the Offspring
  end for
  Update P
end while

The choice of the appropriate genetic operator is important as it directly influences the convergence of the genetic algorithm. However, different forms of the main genetic operators are needed according to the type of the genetic algorithm and the nature of the optimization problem. An overview of the main components of a genetic algorithm are discussed below.

1. *Initialization:* At the initialization stage, candidate solutions or individuals are randomly generated. The number of individuals in the population are determined according to the problem, and in many cases, empirically evaluated in trial experiments. In some cases, the candidate solutions are seeded in the area of search space where the desired solution is likely to be found.
2. *Selection:* The selection operator plays an important role as it ensures that qualities from the fittest individuals always survive in future generations. Some common selection strategies are rank selection [13], fitness proportionate or roulette wheel selection [14], tournament selection [14] and the elitist strategy [15]. The roulette wheel selection is used in this work. It gives priority to those individuals with greater fitness; however, less fit individuals are chosen occasionally as they may contain useful genetic material for the future.
3. *Reproduction using crossover:* The main reproduction operators are *crossover* and *mutation*. The crossover operator exchanges genetic material from selected parents and forms either a single or multiple offspring. Some common crossover operators for real-coded genetic algorithms are flat crossover, [16], simple crossover [17,18], arithmetic crossover [18], linear breeder crossover [19], and Wright's heuristic crossover which has also shown superior performance compared to binary GA for a set of optimization problems [3]. The Wright's heuristic crossover operator is used in this work.
4. *Reproduction using mutation:* The mutation operator provides random diversity in the population. This is important when the algorithm gets trapped in a local minimum. Some common mutation operators include random uniform mutation and Michalewicz non-uniform mutation [18,20]. In uniform mutation, a random number in the range of $[a,b]$ is added to a selected gene where a and b are the highest and lowest values in the individual, respectively. In non-uniform mutation, the strength of mutation is decreased as the number of generation increases. The uniform mutation operator is used in this work.
5. *Termination:* In the genetic algorithm, the evolutionary process continues until the termination condition is satisfied. The search is terminated when one of the following is true. (1) Fixed number of generations or function evaluations is reached, (2) a solution with a desired fitness is found, or (3) the highest ranking solutions come to a point where there is no change of fitness indicating no improvement in candidate solutions.

2.2. Simulated annealing

The simulated annealing algorithm (SA) has been inspired by the physical annealing process of molten metals. It has been developed independently by Kirkpatrick [21] and Cerny [22] and is a generalization of the Metropolis Monte Carlo method [23].

The simulated annealing algorithm acts like a pure randomization algorithm in the initial phase and turns into a pure greedy algorithm in the final phase as shown in Algorithm 2. At each temperature level, the search is carried out for a certain number of iterations, this is known as the *Markov chain length* (MCL). The algorithm progresses with a random initial solution and continues by generating a new solution S^{new} in the neighborhood of S. Afterwards, the energy function values, $E(S)$ and $E(S^{new})$ are determined for calculating the change in the energy function, $\Delta E = E(S^{new}) - E(S)$. The improved solution is always accepted. The solution without improvement is accepted according to the probability, $P(-\Delta E, T) = min(1, exp^{(-\Delta E/T)})$ where T is the parameter called temperature. The solution is accepted if P is greater than *random*() which returns a random value in the range $[0,1]$. This implies that the algorithm moves uphill at certain times during its downhill search. An uphill move may help the algorithm escape from a local minimum. As the temperature decreases, the probability of accepting weaker solutions decreases, i.e. those with higher energy value.

Algorithm 2. Simulated Annealing

```
Generate initial solution S
Get initial temperature T_0 and set T = T_0
while Not Termination do
  while i < MCL do
    1. Generate neighbour S^new of S
    2. Calculate ΔE = E(S^new) − E(S)
    if ΔE ⩽ 0 then
      Set S = S^new and E(S) = E(S^new)
    end if
    if P(−ΔE, T) > random () then
      Accept solution
    end if
    3. Increment MCL
  end while
  Decrement T according to the cooling rate
end while
```

The initial temperature, the Markov chain length and the cooling temperature rate are important variables in the design of the simulated annealing algorithm. The neighbour of the new solution is generated usually by adding a small real number with some distribution to each particle in the new solution. The algorithm termination criteria can either be when ΔE is less than a threshold ω or when a minimum temperature is reached or when the algorithm reaches the maximum number of iterations. The Markov chain length varies according to the difficulty of the problem.

2.3. Hybrid metaheuristics

Metaheuristics (MH) refer to the family of search algorithms which have improved basic heuristic methods by extending exploration capabilities [24]. The term *metaheuristic* was first introduced by Glover in 1986 [25] which is derived from the composition of two ancient Greek words. The word heuristic is derived from the verb *heuriskein* which means "to find " while *meta* means "beyond, in an upper level". Metaheuristic methods have been useful for approximately solving difficult optimization problems. They do not guarantee a global optimal solution, however, near global optimal solutions can be found faster in comparison to conventional approaches for combinatorial optimization problems [26].

Hybrid metaheuristics (Hybrid MHs) refer to the group of hybrid algorithms where two or more metaheuristic search techniques are combined to solve difficult problems [27]. They provide *diversification* and *intensification* during the search process for obtaining a stronger solution. Diversification refers to the ability to visit many different regions in the search space while intensification refers to the ability to obtain high quality solutions in a particular search space [27]. Diversification is often referred to as *global search* or *exploration*. Intensification is referred to as *local search, local refinement* or *exploitation*. The main problem of hybrid metaheuristics is to efficiently balance the effects of diversification and intensification in the search process. The goal is to reach a global optimum by obtaining quality solutions in lesser optimization time.

2.4. Collaborative hybrid MHs

Collaborative hybrid HMs are based on the exchange of information between different metaheuristics which run sequentially or in parallel [28]. Collaborative hybrid MHs are further divided into subcategories that include teamwork and relay collaborative MHs.

Teamwork collaborative MHs use several metaheuristics which include evolutionary algorithms that work in parallel [29,30]. Gradual distributed real-coded genetic algorithms use several different sub-populations [30]. These sub-populations are employed for diversification and intensification and evolve in parallel with the migration of information. The development of the hybrid algorithm has been implemented in parallel where simulated annealing is employed with lower temperature after the use of genetic recombination operators such as crossover and mutation over the whole population [31]. The *COSEARCH* method, proposed by Talbi and Bachelet [32], implements the combination of an EA, Tabu search and a local search procedure which communicate via an adaptive memory that contains the search history.

Relay collaborative MHs use a pipelined process to execute metaheuristic search methods where the output of each method is used as the input of the other. They follow the heuristic where exploration is done in the initial stages and exploitation in later stages. Chelouah et al. presented a hybrid MH algorithm divided into two main stages where the first stage launch a specified genetic algorithm for diversification, then a second stage takes the best solution and proceeds with a local search procedure for intensification [33]. Moreover, the initial global search with a high temperature in the simulated annealing is replaced with a faster heuristic method such as the genetic algorithm. Then the solution is presented with a lower initial temperature to the simulating annealing algorithm which further improves the quality of the solution [34]. This work uses genetic algorithms as a global search technique in the first stage and later provides a solution to the simulated annealing algorithm with a lower temperature for further refinement of the solution.

2.5. Integrative hybrid MHs

Integrative hybrid MHs use embedded methods such as local improvement of candidate solutions by an inner optimization algorithm and exact techniques for searching very large neighbourhoods [28]. Memetic algorithms are well known methods for this category where a master MH is used for diversification and a subordinate MH is used for intensification. Memetic algorithms address the shortcomings of EAs in balancing diversification and intensification. Memetic algorithms [35] typically combine population-based evolutionary algorithms with local refinement in order to provide an improved global solution.

3. The proposed collaborative MHs for the FKP

This paper implements the *teamwork* and *relay* collaborative metaheuristics which employ genetic algorithms and simulated annealing.

3.1. Teamwork collaborative MH

The teamwork hybrid MH combines both algorithms referred to as Teamwork Genetic Algorithm and Simulated Annealing (Teamwork GA–SA). Note that simulated annealing is considered as global search methods which use high temperature

for diversification and intensification follows with lower temperature. Given a lower temperature in the initial stage, simulated annealing can also be used as a local search procedure. In hybrid teamwork GA–SA, the genetic algorithm becomes the master global search which employs the simulated annealing as a local search technique during the evolutionary search process.

Teamwork GA–SA employs the SA algorithm in addition to mutation operation after the crossover. At each generation of the GA, SA is launched with a lower temperature for a specified number of iterations. The lower temperature deploys a greedy local search in the SA. The local search intensity (LSI) defines the duration of the search employed by the SA. The LSI is determined by the number of iterations (N) in the SA. The local search rate (LSR) determines the rate or frequency of the local search. The LSR is determined by the probability with which the local search is applied to all the offspring at every generation. The LSI and LSR are the two most important parameters which must be evaluated in order to achieve quality solutions including lower optimization time. The details of the hybrid teamwork GA–SA is given in Algorithm 3.

Algorithm 3. Hybrid Teamwork GA–SA

```
Initialize Population (P)
while Not Termination do
  while No. Offspring < Size (P) do
    1. Evaluate fitness
    2. Selection of Parent 1 and Parent 2
    3. Employ Crossover and produce a single Offspring
    if LSR then
      4. Present the Offspring to SA with LSI of N Iterations
    end if
    5. Apply Mutation on the refined Offspring
  end while
end while
```

The inner-loop of Algorithm 3 begins by initializing the population of individuals with real random numbers in some specified range or distribution. The individuals are evaluated by calculating their fitness with the given fitness function. Once the individuals are evaluated, the algorithm proceeds as a standard GA which employs genetic operators such as selection and crossover to create offspring for the population. In our implementation, the roulette wheel selection is used and the selection operator chooses two parents. The crossover operator combines both parents and produces a single offspring. According to the LSR, the offspring is presented to the SA algorithm with a specified initial temperature. The LSI defines the duration in the SA deployment in terms of N. The refined offspring is exposed to mutation and added to the population. The inner-loop is repeated until the number of offspring equals to the size of the population and this makes up a generation. The procedure is repeated until the outer-loop terminates according to the termination criteria.

The Wright's heuristic crossover operator has shown superior performance in comparison with other crossover operators for a set of optimization problems [3,36], therefore, it is used in all the GA instances. The Wright's heuristic operator produces a single offspring given two parents. For a pair of parents $x^1 = (x_1^1, x_2^1, x_3^1, \ldots, x_n^1)$ and $x^2 = (x_1^2, x_2^2, x_3^2, \ldots, x_n^2)$, an offspring is produced as shown in Eq. (1).

$$y_i = r(x_i^1 - x_i^2) + x_i^1 \tag{1}$$

where r is a real random number belonging to $[0,1]$ and x^1 is the parent with the best fitness. The *uniform mutation* operator is also used in the hybrid teamwork GA–SA [18] which adds small real-random numbers to selected variables. For instance, given an offspring, $x = (x_1, x_2, x_3, \ldots, x_n)$, the mutated offspring becomes $y = (x_1, x_2, y_3, \ldots, y_n)$, where, $y_i = x_i + r$ given that r is a small real random number in the interval $[-b, b]$, where b is the number chosen according to the problem.

3.2. Relay collaborative MH

This work also implemented the relay hybrid MH algorithm which is referred to as relay genetic algorithm and simulated annealing (Relay GA–SA). GA and SA are combined in a two-stage process. Initially, the GA is launched and when a specified fitness is reached, the best individual from the population is given as an initial solution in the search space for the SA algorithm which further refines the solution. The SA uses a lower initial temperature for intensification with a greedy search.

The neighborhood of the new solution is important for the simulated annealing algorithm and must be tailored to problems which need quality solutions with high accuracy. The neighborhood in each solution depends on real random numbers in the interval $[-a, a]$. In this work, a is adapted and reduced after every t iterations. In this way, the new solutions in the beginning of the search are exposed to significant changes in the variables as the neighborhood interval for the new solutions is large. In the final stages, the neighborhood interval is reduced which enforces minor changes and significant refinement.

3.3. Performance evaluation

The main performance measures will be the following parameters:

1. The duration of the optimization in terms of number of function evaluations;
2. The success rate;
3. The solution quality in terms of error defined by the fitness function;
4. The solution accuracy against the exact results.

The success rate determines how well the particular algorithm can guarantee a solution within a specified time. A run is considered successful if a desired solution defined by an error is found before the maximum time is reached in n number of independent experimental runs. The desired solution is specified by a predefined minimum error according to the fitness function which express the kinematics of the 3RPR parallel manipulator.

4. Application: forward kinematics of the 3RPR parallel manipulator

The idea to design planar parallel mechanisms can be traced back as early as in the beginning of the 40's with Pollard and his *five-bar mechanism*.[1] More generally, parallel mechanisms have then attracted much attention since they have been successfully applied as flight simulators [37]. Even though they were extensively studied, the impact of planar parallel manipulators in the industry remains limited.

Typically, planar parallel manipulators are characterized by one base, one mobile platform and three kinematics chains which lie in one plane, namely the 3RPR, [38]. Moreover, the manipulator end-effector displacements are restricted to that same plane as shown in Fig. 1. The design hypotheses states that the bodies are infinitely rigid and the joints do not yield friction nor play. The redundant manipulators shall not be studied in this article. Moreover, the number of actuated and measured joint variables equals the number of end-effector *degrees-of-freedom* (DOF).

4.1. The kinematics of parallel manipulators

Any manipulator is characterized by its mechanical configuration parameters and the posture variables. The configuration parameters are thus $\mathbf{OA}_{|R_f}$, the base attachment point coordinates in R_f (the base reference frame, located at **O**), and $\mathbf{CB}_{|R_m}$, the mobile platform attachment point coordinates in R_m (the mobile platform reference frame, located at **C**). The kinematics model variables are the joint coordinates and end-effector generalized coordinates. The joint variables are described as L_i, the prismatic joint or linear actuator positions. The generalized coordinates are expressed as $\vec{X}$, the end-effector position and orientation.

The kinematics model is an implicit relation between the configuration parameters and the posture variables, $F(\vec{X}, \overline{L}, \mathbf{OA}_{|R_f}, \mathbf{CB}_{|R_m}) = 0$ where $\overline{L} = \{L_1, L_2, L_3\}$.

This article shall only concentrate on the forward kinematics problem (**FKP**) as shown in Fig. 2. Usually the *inverse kinematics problem* is required to model the **FKP** and is defined as: *given the generalized coordinates of the manipulator end-effector, find the joint positions*. Accordingly, the *forward kinematics problem* is defined as: *given the joint positions, find the generalized coordinates of the manipulator end-effector*. In the majority of parallel manipulator cases, the **FKP** is a difficult problem [39].

4.2. Vectorial formulation of the basic kinematics model

Containing as many equations as variables, vectorial formulation constructs an equation system [40], Fig. 3, as a closed vector cycle between the following points: A_i and B_i, kinematics chain attachment points, O the fixed base reference frame and C the mobile platform reference frame. For each kinematics chain, an implicit function $\vec{A_iB_i} = U_1(X)$ can be written between joint positions A_i and B_i. Each vector $\vec{A_iB_i}$ is expressed knowing the joint coordinates $\vec{L_i}$ and X giving function $U_2(X, \vec{L})$. The following equality has to be solved: $U_1(X) = U_2(X, \vec{L})$. The distance between A_i and B_i is set to L_i. Therefore, the end-effector position X or C can be derived by one platform displacement $\vec{OC}$ and then one platform general rotation expressed by the rotation matrix $\mathcal{R}$. Vectorial formulation in Eq. (4) evolves as a displacement based equation system using the relation given in Eq. (2).

$$\vec{A_iB_i} = \vec{OC} + \mathcal{R}\vec{CB_i} - \vec{OA_i} \tag{2}$$

As given in Eq. (2), for this planar manipulator, the rotation matrix $\mathcal{R}$ is expressed in terms of one rotation parameter as in Eq. (3).

$$\mathcal{R} = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix} \tag{3}$$

$$L_i^2 = \|A_iB_i\|^2 \tag{4}$$

[1] W-L-G. Pollard, Spray painting machine, August 26, 1940, USA Patent No. 2,213,108, Evanston, ILL, USA.

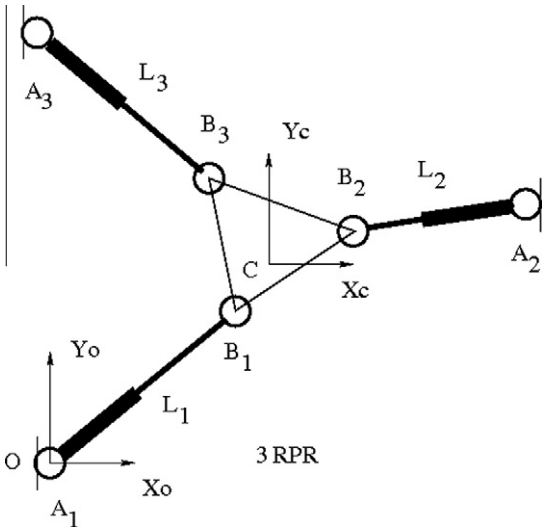

Fig. 1. The general planar manipulator and the typical 3RPR tripod [9].

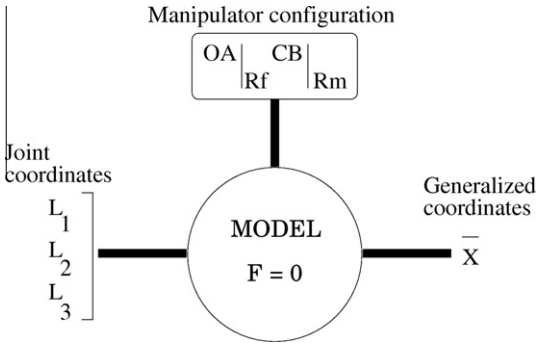

Fig. 2. Kinematics model [9].

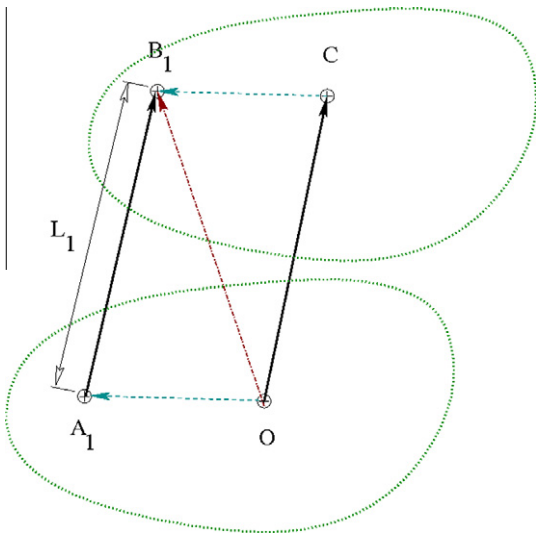

Fig. 3. The vectorial formulation for one kinematics chain [9].

For each distinct platform point $\vec{OB}_{i|_{R_f}}$ with $i = 1,\ldots,3$, each kinematics chain can be expressed using the distance norm constraint [41]:

4.3. The inverse kinematics problem

A review of planar parallel manipulators shows us that the majority of the proposals fall into the four following classes: 3RPR, 3RRP, 3PRR and 3RRR, [9]. We will only study the 3RPR. For each kinematics chain, the 3RPR manipulator is constituted by a prismatic actuator located between two ball joints fixed on the base and the platform. Taking Eq. (2) and substituting it in Eq. (4), one obtains the expression in Eq. (5).

$$L_i^2 = \| \vec{OC} + \mathcal{R}\vec{CB}_i - \vec{OA}_i \|^2, \quad i = 1 \ldots 3 \tag{5}$$

4.4. The forward kinematics and conversion to optimization problem

The **IKP** expression gives an algebraic system comprising the first three equations in terms of the classical variables: x_c, y_c, θ, Eq. (5). Being only applicable to optimization problems, the simulated annealing and genetic algorithm handles a energy or fitness function to be maximized or minimized, respectively. Therefore, we need to effectively convert the problem which solves a system of equations into an optimization problem. Hence, only the inverse kinematic model is required from which we can easily derive an objective function, also called the fitness function. The fitness function represents the total error on each leg lengths. Let Lg_i be the leg length of kinematics chain i which is given as input of the problem. Therefore, the fitness function is given in Eq. (6).

$$\sum_{i=1}^{3} (L_i - Lg_i)^2 \tag{6}$$

If we set $H_i = L_i^2$ coming from Eq. (5), the fitness function is updated as given in Eq. (7).

$$\sum_{i=1}^{3} (sqrt(H_i) - Lg_i)^2 \tag{7}$$

5. Experiments, results and analysis

In this section, simulated annealing, genetic algorithms, teamwork and relay hybrid metaheuristic methods are used for solving the FKP of the 3RPR manipulator. The fitness function is derived from the inverse kinematics of the tripod 3RPR parallel manipulator as shown in Eq. (7). The following steps are taken in this section:

1. The genetic algorithm is used for solving the FKP problem. We evaluate the best population size.
2. Simulated annealing is used for solving the FKP problem. We evaluated the best Markov chain length.
3. The hybrid relay GA–SA is used. We evaluate when it is best to change from GA to SA in terms of the error.
4. The hybrid teamwork GA–SA is used. We evaluate the *local search rate* and *intensity*.
5. All the metaheuristic methods are compared in terms of optimization time taken and the solution accuracy.
6. The solutions obtained by the metaheuristic methods are further compared with the exact Algebraic method.

One example of the resolution on a typical 3RPR manipulator configuration is examined. The manipulator base coordinates of the joint center position $OA_{|_{R_f}}$ in the base reference frame R_f and the mobile platform coordinates of the joint center position $CB_{|_{R_m}}$ in the platform reference frame R_m, and the minimum bar lengths are shown in Table 1.

The joint variables are the kinematics chain lengths. The following leg lengths are used, $L := [100, 120, 150]$. All experiments initialize the population with real numbers in the range of $[-50, 50]$. The three selected kinematics variables represent the end-effector position and orientation, being x, y and θ. The search is terminated when the fitness in terms of the error gets lower than 1e−20. The experiments are done on an IBM compatible personal computer with *i3* processors with the Linux operating system. The computation time is given in seconds.

5.1. Genetic algorithm

We first use the genetic algorithm with Wright's heuristic crossover and uniform mutation. The crossover rate of 0.9 and uniform mutation with the rate of 0.1 are used. The high crossover rate ensures that maximum global search is achieved. These values showed good performance in trial experimental runs. Table 2 displays the optimization time and successful runs in 100 experiments (Success/100) in terms of the population size. The subscript in the attribute Time (S) given in seconds reports the 95% confidence interval. The population size of 40 led to the minimum computation time and hence is used in the rest of the experiments. Note that all experimental runs could reach the minimum error before the maximum time.

Table 1
The planar 3RPR configuration table.

Base joints (mm)	$A_1(x)$	$A_1(y)$	$A_2(x)$	$A_2(y)$	$A_3(x)$	$A_3(y)$
	0	0	200	0	0	200
Platform joints (mm)	$B_1(x)$	$B_1(y)$	$B_2(x)$	$B_2(y)$	$B_3(x)$	$B_3(y)$
	0	0	50	0	40	40

5.2. Simulated annealing

The goal of the simulated annealing algorithm is to reach the minimum error of 1e-20 within the maximum time in terms of number of iterations which was set to 300000. The neighborhood of the new solution is restricted to the interval $[-a,a]$, where $a = m * (1/k)$, k is incremented after every 1000 iterations and m is numerical constant. We used the value of 5 for m. The results are shown in Table 3 which gives the best value for the Markov chain length (MCL). The results show that the simulated annealing algorithm fails to converge to the minimum error within the maximum time in 30 independent experimental runs. The Markov chain length of 10 has taken the least optimization time and will be used in the rest of the experiments. In Table 3, the subscripts reports the 95% confidence interval for the errors and the computation time.

5.3. Hybrid relay GA–SA

The hybrid relay GA-SA implemented GA in the initial stage and SA in the later stage. It is important to evaluate when to change from GA to SA. We observe the error and make the change when the FKP problem reaches a given error as indicated by the *error change* in Table 4. The results show that the hybrid relay GA–SA fails to converge to the minimum error within the maximum time. The error change of 10 has taken the least optimization time. The subscript in Time (S) given in seconds reports the 95% confidence interval and the number of successful runs in 30 experiments is given by (Success/30). In Table 4, the subscripts reports the 95% confidence interval for the errors and the computation time.

5.4. Hybrid teamwork GA–SA

In the hybrid teamwork GA-SA, the new solutions in the simulated annealing algorithm are generated by adding a small real number in the fixed range of $[-0.01, 0.01]$ to all variables in the solution. This value is used to ensure that solutions are refined with minor changes. The Markov chain length of 10 is used in the SA. Given a local search rate of 0.5 and initial temperature of 10 in the SA, Table 5 shows response times and success rates in terms of the LSI; the subscripts reports the 95% confidence interval for computation time. These parameters were determined in trial experiments. Table 5 shows the performance of two different methods with different local search intensity (LSI). In Method 1, the LSI is applied from the beginning of the evolutionary search while in Method 2, the local search is employed during the later stages when the fitness goes below 1. Note that all experimental runs could reach the minimum error given by the fitness before the maximum time.

Table 6 displays the local search rate (LSR). The results show that the local search rate of 0.2 produces the best results implying that local search should be applied to just a few offspring in the population.

Table 2
Evaluation of the population size for the genetic algorithm.

Pop. size	Time (S)	Success/100
20	$5.50_{0.25}$	100
40	$3.41_{0.13}$	100
60	$3.74_{0.12}$	100
80	$4.14_{0.12}$	100
100	$4.66_{0.14}$	100

Table 3
Evaluation of the Markov chain length (MCL) in simulated annealing.

MCL	Error	Time (S)	Success/30
10	$3.99e{-}10_{1.12e-10}$	$7.28_{0.46}$	0
20	$7.50e{-}11_{1.72e-11}$	$14.44_{0.92}$	0
30	$1.85e{-}11_{3.53e-12}$	$21.71_{1.39}$	0
40	$8.21e{-}12_{2.41e-12}$	$28.95_{1.86}$	0
50	$4.31e{-}12_{1.14e-12}$	$36.12_{2.32}$	0

Table 4
Evaluation of the error change in relay GA–SA.

Error change	Error	Time (S)	Success/30
10	$4.17e{-}10_{1.03e-10}$	$7.49_{0.48}$	0
1	$3.38e{-}10_{7.61e-11}$	$7.58_{0.49}$	0
0.1	$4.00e{-}10_{9.51e-11}$	$7.68_{0.49}$	0
0.01	$4.41e{-}10_{9.65e-11}$	$7.87_{0.51}$	0
0.001	$4.65e{-}10_{1.24e-10}$	$7.89_{0.51}$	0
0.0001	$3.97e{-}10_{1.05e-10}$	$8.04_{0.52}$	0

5.5. Comparison of the metaheuristic methods

We evaluate the performance of the four metaheuristic methods in terms of least optimization time and least error in solution accuracy. Table 7 shows the mean and 95 percent confidence interval for 100 experimental runs for each metaheuristic method. The success rate shows how well the given method can guarantee a solution when given any random seed within the search space.

The given FKP problem has 2 unique solutions which are given in Table 8 for coordinates x, y, and orientation θ with the respective error. In order to verify the solutions obtained by the respective methods, we revert to an Algebraic method which can compute the certified exact results [9]. The exact results using the Algebraic method are also shown in Table 8. Note that solutions 1 and 2 seem identical, however, the error is different. The solutions seem identical as only 4 decimal places have been reported. The solution are different for more than 4 decimal places which gives the difference in the error of the solutions.

5.6. Discussion

The results in Table 7 show that genetic algorithms and hybrid teamwork GA–SA have been able to give solutions to the desired fitness or error with the least optimization time. The simulated annealing and hybrid relay GA–SA showed similar results. They have not been able to converge to the desired fitness within the maximum time. This implies that the crossover operator has global and local search capabilities which obtains quality solutions. This is evident according to the results in

Table 5
Evaluation of the local search intensity (LSI) for hybrid teamwork GA–SA.

LSI	Method 1		Method 2	
	Time (S)	Success/30	Time (S)	Success/30
50	$17.38_{1.63}$	30	$3.90_{0.38}$	30
100	$26.44_{2.55}$	30	$4.38_{0.41}$	30
150	$34.94_{3.34}$	30	$4.99_{0.39}$	30
200	$42.08_{4.46}$	30	$5.38_{0.43}$	30

Table 6
Evaluation of the local search rate (LSR) for hybrid teamwork GA–SA.

LS rate	Time (S)	Success/30
0.2	$3.69_{0.29}$	30
0.4	$3.93_{0.31}$	30
0.6	$4.15_{0.33}$	30
0.8	$4.11_{0.32}$	30
0.9	$4.21_{0.36}$	30

Table 7
Hybrid meta-heuristic methods for the FKP of 3RPR manipulator.

Method	Error	Time (S)	Success/100
GA	$6.83e{-}22_{4.72e-23}$	$3.60_{0.15}$	100
SA	$4.28e{-}10_{6.11e-11}$	$7.54_{0.14}$	0
Relay GA–SA	$3.92e{-}10_{4.88e-11}$	$7.65_{0.15}$	0
Teamwork GA–SA	$6.32e{-}22_{5.15e-23}$	$3.74_{0.13}$	100

Table 8
Optimal x, y and θ values from the best experimental run for each method.

Method	Solution	x	y	θ(deg)	Error
GA	1	97.9961	19.9189	85.0176	8.51e−22
	2	52.8610	84.8865	−33.4615	2.41e−22
SA	1	97.9961	19.9189	85.0176	3.70e−10
	2	52.861	84.8865	−33.4615	2.31e−10
Relay GA–SA	1	97.9961	19.9189	85.0176	2.70e−10
	2	52.861	84.8865	−33.4615	3.371e−10
Teamwork GA–SA	1	97.9961	19.9189	85.0176	8.51e−22
	2	52.8610	84.8865	−33.4615	2.41e−22
Algebraic method	1	97.9911	19.9193	85.0154	–
	2	52.8654	84.9536	−33.4487	–

generic algorithms and hybrid teamwork GA–SA where crossover operator with a high crossover rate has been employed throughout the entire search process. In simulated annealing, the global search is ensured using high temperature, however, the greedy local search with low temperature is limited as the SA does not have a good heuristic to produce new and refined high quality solutions. This is the reason the hybrid relay GA–SA did not converge to the desired fitness and the SA algorithm on its own could not ensure refined local search. In the hybrid teamwork GA–SA, the SA is employed in parallel with the crossover operation, which exchanges genetic information as a team and enables convergence. On its own, SA performed poorly when compared to genetic algorithms. SA has not been able to achieve the desired solution and at the same time takes more time to converge.

The solutions given by the metaheuristic methods are similar to that of the exact Algebraic method implying that the results are reachable and close to exactness. We also note that all the metaheuristic approaches were successful in finding the two distinct solutions in multiple runs. This was done by running multiple experiments with different initial positions in the search space. In this way, the metaheuristic algorithm converges towards the solution nearest to the initial search position. They are applicable in solving the forward kinematics of parallel manipulators as they are problem independent and do not require the problem to be differentiated or suffer from Jacobian inversion problems. Note that the two solutions reported in this work were not found in the same population. They were obtained from different populations given by multiple independent experimental runs.

6. Conclusions

The results demonstrate that the genetic algorithm and hybrid teamwork GA–SA has provided the best solutions in terms of the solution accuracy and optimization time when compared to simulated annealing and hybrid relay GA–SA. The crossover operator has played an important role in global and local search with simulated annealing in hybrid teamwork GA–SA. The rate and intensity of local search is important and must be tailored to the problem. The hybrid methods, in general, were not successful in lowering the optimization time of genetic algorithms. This is because they use two search algorithms which add to the optimization time.

This work has also shown that meta-heuristic algorithms, either single solution or population based, are able to provide distinct solutions given multiple trial runs with different initial search positions. This is of advantage to multi-modal optimization problems that handle non-linear equation systems where more than one distinct solution is present.

Note that any local search method can be used instead of simulated annealing in the hybrid teamwork or relay metaheuristic methods. Genetic algorithms can be replaced by any other evolutionary algorithms for real-parameter optimization. The main limitation of this study is the optimization time which restricts the implementation of the FKP of the 3RPR in real-time. We achieved an average optimization time of 3–4 s, however; for real-time implementation, the optimization time of a few milliseconds is desired. We have showed that metaheuristic methods are able to give robust and high quality solutions without facing convergence problems. This opens up the road for further research where the optimization time can be reduced for real-time implementations.

The use of metaheuristic methods is promising for optimization problems in areas of robotics in general. In future work, it may be useful to use hybrid meta-heuristic methods in solving other kinematics problems.

References

[1] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: overview and conceptual comparison, ACM Comput. Surv. 35 (2003) 268–308.
[2] J.H. Holland, Adaptation in Natural and Artificial Systems, MIT Press, Cambridge, MA, USA, 1992.
[3] A.H. Wright, Genetic algorithms for real parameter optimization, in: Foundations of Genetic Algorithms, Morgan Kaufman, 1991. pp. 205–218.
[4] R. Boudreau, N. Turkkan, Solving the forward kinematics of parallel manipulators with a genetic algorithm, J. Robotics Syst. 13 (2) (1995) 111–125.
[5] T.J. Ypma, Historical development of the Newton–Raphson method, SIAM Rev. 37 (4) (1995) 531–551. <http://dx.doi.org/10.1137/1037125>.
[6] D.S. Boudreau R., N. Turkkan, Etude comparative de trois nouvelles approches pour la solution du problème géométrique direct des manipulateurs parallèles, Mech. Machine Theor. 33 (5) (1998) 463–477.
[7] A. Omran, G. El-Bayiumi, M. Bayoumi, A. Kassem, Genetic algorithm based optimal control for a 6-dof non redundant stewart manipulator, Int. J. Mech. Syst. Sci. Eng. 2 (2) (2008) 73–79.

[8] H.M. Wang X., Y. Cheng, On the use of differential evolution for forward kinematics of parallel manipulators, Appl. Math. Comput. 205 (2) (2008) 760–769.
[9] L. Rolland, Synthesis on the forward kinematics problem algebraic modeling for the planar parallel manipulator, Adv. Robotics 20 (9) (2006) 1035–1065.
[10] R. Chandra, M. Zhang, L. Rolland, Solving the forward kinematics of the 3RPR planar parallel manipulator using a hybrid meta-heuristic paradigm, Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA (2009) 177–182.
[11] I. Rachenberg, (in german) evolutionsstrategie optimierung technischer systeme nach prinzipien der biologischen evolution, Ph.D. thesis, Reprinted by Fromman-Holzboog, 1973.
[12] L.J. Fogel, A.J. Owens, M.J. Walsh, Artificial Intelligence through Simulated Evolution, John Wiley, New York, USA, 1966.
[13] D. Whitley, The genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best, in: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, 1989, pp. 116–121.
[14] J.E. Baker, Reducing bias and inefficiency in the selection algorithm, in: Proceedings of the Second International Conference on GA, Lawrence Erlbaum Associates, Inc. Mahwah, NJ, USA, 1987, pp. 14–21.
[15] K.A. DeJong, An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. thesis, 1975.
[16] N.J. Radcliffe, Equivalence class analysis of genetic algorithms, Complex Syst. 5 (1991) 183–205.
[17] D.E. Goldberg, Real-coded genetic algorithms, virtual alphabets, and blocking, Complex Syst. 5 (1991) 139–167.
[18] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, thrid ed., Springer-Verlag, London, UK, 1996.
[19] D. Schlierkamp-Voosen, Strategy adaptation by competition, Proceedings of the Second European Congress on Intelligent Techniques and Soft Computing (1994) 1270–1274.
[20] Z. Michalewicz, M. Schoenauer, Evolutionary algorithms for constrained parameter optimization problems, Evol. Comput. 4 (1) (1996) 1–32.
[21] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680, doi:10.1126/science.220.4598.671.
[22] V. Černý, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm, J. Optim. Theor. Appl. 45 (1) (1985) 41–51, doi:10.1007/BF00940812.
[23] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (6) (1953) 1087–1092, doi:10.1063/1.1699114.
[24] F.W. Glover, G.A. Kochenberger, Handbook of Metaheuristics, Springer, 2003.
[25] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (1986) 533–549.
[26] C. Blum, A. Roli, Metaheuristics in combinatorial optimization: overview and conceptual comparison, ACM Comput. Surv. 35 (2003) 268–308.
[27] M. Lozano, C. Garcia-Martinez, Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: overview and progress report, Comput. Oper. Res. 37 (3) (2010) 481–497.
[28] G. Raidl, A unified view on hybrid metaheuristics, in: F. Almeida, M. Blesa Aguilera, C. Blum, J. Moreno Vega, M. Prez Prez, A. Roli, M. Sampels (Eds.), Hybrid Metaheuristics of Lecture Notes in Computer Science, vol. 4030, Springer, Berlin Heidelberg, 2006, pp. 1–12.
[29] M. Tomassini, Parallelism and evolutionary algorithms, IEEE Trans. Evol. Comput. 6 (2002) 443–462.
[30] F. Herrera, M. Lozano, Gradual distributed real-coded genetic algorithms, IEEE Trans. Evol. Comput. 4 (2000) 43–63.
[31] S.W. Mahfoud, D.E. Goldberg, Parallel recombinative simulated annealing: a genetic algorithm, Parallel Comput. 21 (1) (1995) 1–28. <http://dx.doi.org/10.1016/0167-8191(94)00071-H>.
[32] E.-G. Talbi, V. Bachelet, Cosearch: a parallel cooperative metaheuristic, J. Math. Model. Algorithms 5 (1) (2006) 5–22.
[33] R. Chelouah, P. Siarry, Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions, European J. Oper. Res. 148 (2) (2003) 335–348. Sport and Computers.
[34] E. Rodriguez-Tello, J.-K. Hao, J. Torres-Jimenez, An effective two-stage simulated annealing algorithm for the minimum linear arrangement problem, Comput. Oper. Res 35 (10) (2008) 3331–3346. <http://dx.doi.org/10.1016/j.cor.2007.03.001>.
[35] P. Moscato, On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms, Tech. Rep., 1989.
[36] P.C. Pendharkar, J.A. Rodger, An empirical study of impact of crossover operators on the performance of non-binary genetic algorithm based neural approaches for classification, Comput. Oper. Res. 31 (4) (2004) 481–498. <http://dx.doi.org/10.1016/S0305-0548(02)00229-0>.
[37] D.A. Stewart, Platform with 6 degrees of freedom, Proceedings of IMechE 180 (9) (1965) 371–386.
[38] C. Gosselin, J.P. Merlet, The direct kinematics of planar parallel manipulators: special architectures and umber of solutions, Mech. Machine Theor. 29 (1994) 1083–1097.
[39] M. Raghavan, B. Roth, Solving polynomial systems for the kinematic analysis and synthesis of mechanisms and robot manipulators, Trans. ASME 117 (1995) 71–79.
[40] D.E. Parrish, R., B.R., An actuator extension transormation for a motion simulator and an inverse transformation applying Newton–Raphson's method, Tech. Rep., D-7067 NASA, 1972.
[41] J.P. Merlet, Les Robots Parall'les, Hermès (1997).