Evolutionary Design of Inverse Dynamics Controllers for Multiple Arm Co-operating Manipulator Systems

Anjan Kumar Swain[1], Urmila Bjanja[2], and Alan S. Morris[3]

[1] Institute of Management Technology, Nagpur, India
akswain@imt.ac.in
http://www.imtnagpur.ac.in
[2] Indira Gandhi Institute of Technology, Sarang, Dhenkanal, India
bhanjaurmila@hotmail.com
[3] Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, United Kingdom
a.morris@sheffield.ac.uk

Abstract. The In this paper a novel hybrid evolutionary algorithm (HEA) method is used to find optimal feedback gains for inverse dynamics controllers. It is shown that HEA efficiently finds the optimal feedback gains to improve the performances of inverse dynamics controllers for end-effector tracking control in highly complex, nonlinear, multi-arm manipulator systems. The feedback gain tuning is studied for constrained, co-operating manipulator systems for both fixed-base and free-floating, multi-arm, space manipulator systems under zero gravity.

1 Introduction

The inverse dynamics control algorithms used to control robot manipulators are primarily concerned only with decoupling and compensating the nonlinearities. In order for inverse dynamics controllers to be able to improve the overall performance of the system, they must be given tunable controller parameters. The controller parameters are usually decided by trial-and-error through extensive simulations and experiments. But, in the case of multi-arm manipulator systems, selecting good controller parameters by a trial-and-error process becoming excessively difficult. Thus, the PD-based inverse dynamics controllers for robot control normally use only two parameters, one each for the proportional and derivative part. In contrast, controlling complex, multi-arm manipulator systems with only two gains may not be sufficient to provide the desired precision. On the other hand, using a number of gains that is equal to the number of controlled variables increases the complexity of finding optimal gains to an even greater extent. In addition, with the increase in the number of parameters, the search space increases rapidly, thereby increasing the problem complexity. Further, it has been shown that the computational complexity of multi-arm manipulator systems is very high, and when combined with evolutionary tuning, the overall computational requirements increase by a large extent [1]. Hence, even though using only two gains

Bhanu Prasad (Editor): IICAI-05, pp. 54-71, 2005.

makes control difficult, the use of more than two gains makes the computational burden unmanageable.

Evolutionary computation (EC) methods have previously been applied to tune the feedback gains of only relatively simple robotic manipulator systems. Kim and Shim [2] used an evolutionary programming method to tune gain levels of a mobile robot posture controller with an aim to achieve shortest paths using minimum time and energy. Ge et al. [3] reported the gain tuning of single link flexible manipulators using a real number GA. They experimented on a task of controlling the tip of the manipulator to reach a predefined position in minimum time with minimal overshoot and oscillation. Katic [4] reported a connectionist controller design for a robotic system applied to a compliant task by combining a real-valued GA with neural classification and learning control techniques. The GA is used to tune the feedback controller gains, neural topology, and weights. Porter and Allaoui [5] used an adaptive $(\mu + \lambda)$ evolutionary method without recombination to design proportional, integral, and derivative (PID) controllers for a three-joint revolute robotic manipulator. The developed controller was used for the tracking control of the end-effector along a straight-line trajectory.

Inverse dynamics controllers use the entire dynamic model of complex manipulator systems to control their motion. However, the model of a multi-arm manipulator system requires the computation of many vector and matrix operations that automatically demands higher computational power [1]. The complex mathematical representation issues of the dynamics of multi-arm manipulator systems are discussed in section II. Further, during the evolutionary tuning of the feedback gains, it is necessary to compute all the model calculations repeatedly. Thus, the computational requirements increase enormously. This might be the reason why EC methods for the tuning of the gain parameters have been restricted to comparatively simple single-arm manipulator systems comprising of very few links. Further, the problems of standard EC methods for the tuning of the feedback gains of multi-arm manipulator systems are due to their high computational requirements, low convergence rate and premature convergence. Thus, efficient EC-algorithms are needed that can efficiently address the gain tuning of multi-arm manipulator systems. Hence, to meet this need a highly efficient hybrid evolutionary algorithm (HEA) has been developed to search for the optimal feedback gains of the inverse dynamics controllers. The faster convergence and high accuracy features of HEA compared with other EC methods can address the computational complexity arising due to the complex mathematical model of the multi-arm manipulator systems. These issues are further discussed in section III. Subsequently, the design issues pertaining to the evolution-based inverse dynamics controllers are discussed in section IV. The experimental set-up and other system parameters needed for the numerical simulation of the evolutionary gain tuning procedure and subsequent control of multi-arm manipulator systems are presented in section V. Then, in sections VI, the results of the numerical simulations are presented for unconstrained manipulators and constrained, co-operating, manipulator systems, respectively. Finally, in section VII, conclusions of this research work are presented.

2 Multi-arm Manipulators

2.1 Dynamic Model

A generalized representation of the dynamic model of a multi-arm, co-operating manipulator system can be expressed by the following set of equations [1]:

$$\begin{aligned} \hat{\mathbf{T}} - \mathbf{J}^{\mathrm{T}} \mathbf{f}_{\mathrm{e}} &= \mathbf{M}\, \ddot{\mathbf{q}} + \mathbf{C}, \\ \dot{\mathbf{V}}^{\mathrm{e}} &= \mathbf{W}\, \dot{\mathbf{V}}^{\mathrm{o}} + \dot{\mathbf{W}}\, \mathbf{V}^{\mathrm{o}} \\ &= \mathbf{J}_{\mathrm{b}}\, \dot{\mathbf{V}}_{\mathrm{b}} + \mathbf{J}_{\mathrm{q}}\, \ddot{\mathbf{q}} + \dot{\mathbf{J}}_{\mathrm{b}}\, \mathbf{V}_{\mathrm{b}} + \dot{\mathbf{J}}_{\mathrm{q}}\, \dot{\mathbf{q}}, \\ \mathbf{M}_{\mathrm{o}}\, \dot{\mathbf{V}}^{\mathrm{o}} + \mathbf{b}_{\mathrm{o}} &= \mathbf{W}^{\mathrm{T}}\, \mathbf{f}_{\mathrm{e}}, \end{aligned} \tag{1}$$

where $\mathbf{M} = \Phi^{\mathrm{T}}\, \mathbf{X}^{\mathrm{T}} (\mathbf{M}_{\mathrm{q}}^{-1} + \mathbf{X}_{\mathrm{b}}\, \mathbf{M}_{\mathrm{b}}^{-1}\, \mathbf{X}_{\mathrm{b}}^{\mathrm{T}})^{-1} \mathbf{X}\, \Phi$ is the generalized inertia tensor, $\mathbf{J}^{\mathrm{T}} = \Phi^{\mathrm{T}}\, \mathbf{X}^{\mathrm{T}}\, (\mathbf{M}_{\mathrm{q}}^{-1} + \mathbf{X}_{\mathrm{b}}\, \mathbf{M}_{\mathrm{b}}^{-1}\, \mathbf{X}_{\mathrm{b}}^{\mathrm{T}})^{-1} \mathbf{M}_{\mathrm{q}}^{-1}\, \mathbf{D}$ is the transpose of the generalized Jacobian matrix; $\mathbf{C} = \Phi^{\mathrm{T}}\, \mathbf{X}^{\mathrm{T}}\, (\mathbf{M}_{\mathrm{q}}^{-1} + \mathbf{X}_{\mathrm{b}}\, \mathbf{M}_{\mathrm{b}}^{-1}\, \mathbf{X}_{\mathrm{b}}^{\mathrm{T}})^{-1} \{ \dot{\mathbf{X}}_{\mathrm{b}}\, \mathbf{V}_{\mathrm{b}} + \mathbf{M}_{\mathrm{q}}^{-1}\, \mathbf{b} - \mathbf{X}_{\mathrm{b}}\, \mathbf{M}_{\mathrm{b}}^{-1}\, \mathbf{b}_{\mathrm{b}} + \dot{\mathbf{X}}\, \Phi\, \dot{\mathbf{q}} \}$ is the coriolis and centrifugal force vector; $\mathbf{T}$ is the torque vector, $\mathbf{V}_{\mathrm{b}}$ and $\mathbf{V}^{\mathrm{e}}$ are the base and end-effector velocity vectors, respectively; $\dot{\mathbf{V}}_{\mathrm{b}}$, $\dot{\mathbf{V}}^{\mathrm{e}}$ and $\dot{\mathbf{V}}^{\mathrm{o}}$ are the base, link and object accelerations, respectively; $\mathbf{J}_{\mathrm{b}}$ and $\mathbf{J}_{\mathrm{q}}$ are the base and link Jacobian matrices representing the end-effector velocity in the end-effector frame; $\mathbf{W}^{\mathrm{T}}$ is the transformation matrix that transforms end-effector quantities onto the object center of mass; $\mathbf{f}_{\mathrm{e}}$ is the end-effector force, $\mathbf{M}_{\mathrm{o}}$ is the object inertia matrix; and $\mathbf{b}_{\mathrm{o}}$ is the object bias force vector. All the quantities for an m-manipulator system with n-links each are based on the following definition [1]:

$\mathbf{X}$ = diag($\mathbf{X}_1 \ldots \mathbf{X}_m$) is the composite link transformation matrix, Φ = diag($\Phi_1 \ldots \Phi_{\mathrm{m}}$) is the composite modal matrix, $\dot{\mathbf{q}} = [\dot{\mathbf{q}}_1^{\mathrm{T}} \ldots \dot{\mathbf{q}}_{\mathrm{m}}^{\mathrm{T}}]^{\mathrm{T}}$ is the composite joint velocity vector, $\mathbf{b} = [\mathbf{b}_1^{\mathrm{T}} \ldots \mathbf{b}_{\mathrm{m}}^{\mathrm{T}}]^{\mathrm{T}}$ is the composite link bias force vector, $\mathbf{B}$ = diag($\mathbf{B}_1 \ldots \mathbf{B}_{\mathrm{m}}$) and $\mathbf{M}_{\mathrm{q}}$ = diag ($\mathbf{M}_1 \ldots \mathbf{M}_{\mathrm{m}}$), and where

$$\mathbf{X}_j = \begin{bmatrix} {}^1_1\mathbf{X}_j & \mathbf{0} & \mathbf{0} & . & . & . & \mathbf{0} \\ {}^2_1\mathbf{X}_j & {}^2_2\mathbf{X}_j & \mathbf{0} & . & . & . & \mathbf{0} \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ {}^n_1\mathbf{X}_j & {}^n_2\mathbf{X}_j & . & . & . & . & {}^n_n\mathbf{X}_j \end{bmatrix}, \tag{2}$$

$\Phi_j = \text{diag}({}^1\Phi_j \ldots {}^n\Phi_j)$, $\mathbf{b}_j = [{}^1\mathbf{b}_j^T \ldots {}^n\mathbf{b}_j^T]^T$, $\mathbf{B}_j = [0 \ldots 0 \; {}^{n+1}_n\mathbf{X}_j]$ and $\mathbf{M}_j = \text{diag}({}^1\mathbf{M}_j \ldots {}^n\mathbf{M}_j)$.

The spatial velocity, acceleration and force vectors of the ith link of the jth robot resolved in the ith link frame are defined by 6×1 vectors ${}^i\mathbf{V}_j$, ${}^i\dot{\mathbf{V}}_j$ and ${}^i\mathbf{f}_j$, respectively, and ${}^i\Phi_j$ represents the matrix of free mode vectors of the ith joint of the jth robot. The 6×6 spatial transformation matrix ${}^i_{i-1}\mathbf{X}_j$ transforms a spatial vector from the (i-1)th co-ordinate frame to the ith co-ordinate frame of the jth robot and is defined as

$${}^i_{i-1}\mathbf{X}_j = \begin{bmatrix} {}^i_{i-1}\mathbf{R}_j & \mathbf{0} \\ {}^i_{i-1}\mathbf{R}_j \, {}^i_{i-1}\tilde{\mathbf{p}}_j^T & {}^i_{i-1}\mathbf{R}_j \end{bmatrix}, \tag{3}$$

where ${}^i_{i-1}\mathbf{R}_j$ is a 3×3 rotation matrix from the (i-1)th link frame to the ith link frame for the jth robot; ${}^i_{i-1}\mathbf{p}_j$ is a 3×1 vector from the origin of the (i-1)th link frame to the origin of the ith link frame for the jth robot; $\tilde{\mathbf{p}}$ for a vector $\mathbf{p} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^T$ is a 3×3 anti-symmetric matrix defined as

$$\tilde{\mathbf{p}} = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}. \tag{4}$$

The 6×6 spatial inertia matrix of the ith link of the jth robot is denoted by ${}^i\mathbf{M}_j$ and is defined as

$${}^i\mathbf{M}_j = \begin{bmatrix} {}^i\mathbf{I}_{jj} & {}^i m_j \, {}^c_i\tilde{\mathbf{l}}_j \\ -{}^i m_j \, {}^c_i\tilde{\mathbf{l}}_j & {}^i m_j \mathbf{E}_3 \end{bmatrix}, \tag{5}$$

where ${}^{i}m_{j}$ is the mass of the ith link of the jth robot; ${}^{i}\mathbf{I}_{j}$ is the inertia tensor of the ith link of the jth robot at the ith frame origin; ${}^{c}_{i}\mathbf{l}_{j}$ is the distance from the ith frame origin to the center of mass of the ith link of the jth robot; and $\mathbf{E}_{3}$ is a 3×3 identity matrix.

2.2 Inverse Dynamics Controllers

The block diagram representation of the inverse dynamics control method for co-operating manipulators is shown in Fig.1. The controller accepts the values of all the desired and actual object and/or base position and orientation related information along with the end-effector forces to calculate the resolved acceleration vector **a**. Then, the block representing the inverse dynamics control law provides the necessary torques to be applied to the active joints and/or base. The inverse dynamics controller compensates the motion-inducing component $\mathbf{f}_{e}^{m}$ of the end-effector force $\mathbf{f}_{e}$ ($\mathbf{f}_{e}$ = $\mathbf{f}_{e}^{m}$ + $\mathbf{f}_{e}^{i}$) and allows the uncompensated internal forces $\mathbf{f}_{e}^{i}$ to maintain grip on the object.

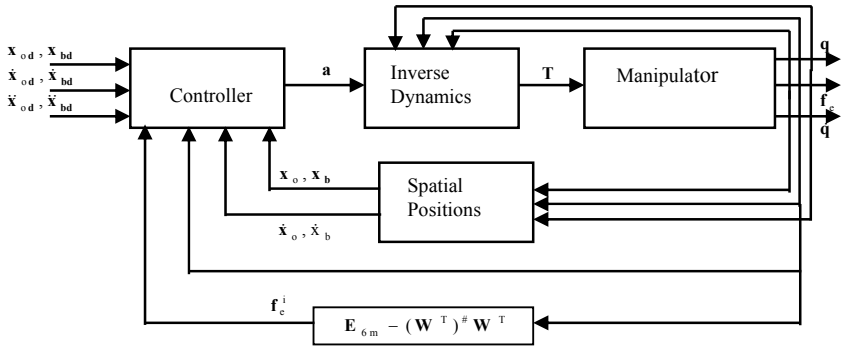


Fig. 1. Block diagram of inverse dynamics control scheme for co-operating manipulator systems.

3 Hybrid Evolutionary Algorithm

The hybrid evolutionary algorithm (HEA) [7] is based on the concept of pseudoglobal optimum. The pseudoglobal optimum is defined as the optimum individual in a particular generation of any population-based algorithm [6]. Usually, the global optimum of the problem at hand is not known in advance, hence in each generation the fittest

individual is identified as the global optimum for that generation, which is known as the pseudoglobal optimum. Thus, the pseudoglobal optimum is dynamic in nature. Now, in each generation, every individual's goal is to achieve the pseudoglobal position. This is achieved by varying the step size in proportion to the absolute genotypic distance of each object variable from the respective object variable of the fittest individual. This allows independent variations for each object variable of an individual. This feature can be represented by a factor σ_{ij} :

$$\sigma_{ij} \propto |p_{ij} - p_{kj}|, \tag{6}$$

where p_{kj} is the kth row and jth column element of an $\mu \times n_o$ population matrix P, with μ and n_o as the number of individuals in the population pool and the number of object variables in an individual, respectively; k is the index for the best quality individual; and |.| denotes an absolute value.
Hence, Eq.(6) can be represented as

$$\sigma_{ij} = k\,|p_{ij} - p_{kj}|\ , \tag{7}$$

where $k = \beta\sqrt{\frac{\omega}{\pi}}$ is a proportionality constant, with $\beta = 0.1$ and ω as the width of the user defined search domain.

Then, to direct the solution in the direction of the pseudoglobal optimum so as to speed up the process of convergence, a directionality feature associated with each pair of individual parent genes has been introduced, determined with respect to the genes of the pseudoglobal optimum individual. The direction (or the sign) of the generated absolute Gaussian random variable is decided according to the position of the particular genes of the parent individual with respect to the genes of the pseudoglobal optimum. This directionality feature can be expressed as

$$dir(p_{ij}) = sgn(p_{ij} - p_{kj})\ , \tag{8}$$

where "sgn" calculates the sign of the argument within the bracket.

In Eq.(7), the genotypic distance for the best quality individual in the population pool is zero, which in turn forces σ_{ij} to zero. This does not update the object variables of the fittest individual. This is undesirable in the context that the fittest should have more opportunity to exploit its own neighborhood to generate better offspring (local search). To circumvent this, a constant offset z has been introduced. Hence, for the fittest individual $\sigma_{ij} = z$, this small standard deviation increases the probability of producing offspring in a very close vicinity of the parent. Whereas, for other individuals, the exploration of new and unknown areas of the search domain increases as the genotypic distance increases that progressively decreases the exploitation capability. Now, σ_{ij} can be written as

$$\sigma_{ij} = k\,|p_{ij} - p_{kj}| + z. \quad (9)$$

The standard Cauchy random variables $C_{ij}(0,1)$, $\forall i \in \{1, \ldots, \mu\}$ and $\forall j \in \{1, \ldots, n_o\}$ are used to guide the search process to generate offspring. Hence, the offspring can be represented as

$$p_{ij} = p_{ij} - \sigma_{ij}\,\mathrm{dir}(p_{ij})\,C_{ij}(0,1). \quad (10)$$

Because the crossover operator generates offspring within a defined initial rectangular hyperbody, this effectively generates offspring constrained to lie within the parent object boundaries. However, in the proposed DSR operator without the directional feature, the offspring can be anywhere in the search space. The incorporation of the directional feature provides a restriction on the generated offspring such that the offspring will no longer be generated in the direction quite opposite to the fittest individual. Thus, it adds the ability to explore the search space more than that of the conventional recombination operators used in ES. Between a parent and its offspring, the fittest survives, which makes the operator elitist.

The mutation operation used here is similar to that used in a basic EP [8]. Here, a problem dependent deterministic factor γ has been formulated, which is then used along with the randomness of the standard Cauchy distribution C(0,1) to escape from the local optima so that there is increased probability of directing the solution process toward the global optimum. Here, γ is selected such that it is directly proportional to the square root of the fitness score and inversely proportional to the problem dimensions, and is defined for the ith individual as

$$\gamma_i \propto \frac{1}{n}\sqrt{f(p_i)} = \frac{\alpha}{n_0}\sqrt{f(p_i)}, \quad (11)$$

where $f(p_i)$ is the fitness score associated with the *i*th individual and $\alpha = 0.01$ is a proportionality constant. Hence, the search step size Δx_{ij} can be represented as

$$\Delta x_{ij} = C_{ij}(0,1)\,\frac{\alpha}{n_o}\sqrt{f(p_i)}. \quad (12)$$

As Δx_{ij} is related to the individual fitness score, it will suffer from the problems of solution instability for higher dimensional tasks [6]. This problem has been addressed here by limiting the variation to lie within twice the user-supplied search-width when the fitness score exceeds ten times the value of the search width. This also ensures that no portion of the search space is omitted. Hence, the jth object variable of the ith offspring generated from the corresponding object variable of the *i*th individual p_i can be represented as

$$p_{ij} + C_{ij}(0,1)\frac{\alpha}{n_o}\sqrt{f(p_i)} \tag{13}$$

Then a stochastic selection [8] is used to select m parents for the next generation.

4 Evolutionary Controller Design

The inverse dynamics controllers used for the control of fixed-base and free-floating systems are always exponentially stable for any positive, definite, and constant feedback gain matrices K_p and K_d. Now, arbitrarily selected positive values for K_p and K_d matrices do not necessarily yield better performance in terms of rise time, overshoot, and settling time. In addition, the number of tuning parameters should always be as small as possible for engineering implementations. In this case, the minimum number of tuning parameters is two, i.e., one k_p and one k_d for all the control variables of all the manipulators.

Usually, k_p and k_d values are selected by running the simulation for many different k_p and k_d values, and those values of k_p and k_d which yield the best tracking performance are chosen as the final values. Indeed, the k_p and k_d values were selected exactly this way for the conventional inverse dynamics controllers. However, for the evolution-based inverse dynamics controllers, the tuning parameters k_p and k_d are selected by taking advantage of the highly efficient and fast hybrid EA (HEA) described above for the tracking control of complex multi-arm manipulator systems. The higher speed and sharp convergence features of HEA have been utilized fruitfully to yield better controller performance for highly complex multi-arm manipulator systems.

The major problems of multi-arm manipulator systems are their high computational costs and time requirements to perform the complex computations. Hence, these problems become increasingly difficult when EC methods run the entire manipulator simulation repeatedly while tuning the gains. This problem can be addressed from two angles: (i) using fast and efficient EC-based methods; (ii) using only a small part of the manipulator simulation to guide the feedback gain optimization process. The first issue has been addressed with the use of a very fast and efficient EC-based algorithm called hybrid evolutionary algorithm (HEA). The second issue can be addressed by allowing the manipulator simulation program to run only for the first few seconds. Ideally, this initial run time is taken to be equal to the time period with hand-tuned parameters within which the system error characteristics settle close to the minimum value (i.e., zero in the case of inverse dynamics controllers) [3]. However, the non-optimal gain parameters for trajectory tracking applications, where the orientations are also time varying, exhibit a large settling time with prominent steady-state error [1]. This time period can typically be taken as 2 sec [3]. But for a two second period with sampling time of 0.001sec, it is necessary to calculate the manipulator simulation 2000 times up to a period of two seconds. Thus, this issue should be taken into consideration while designing EC-based inverse dynamics control algorithms to control multi-arm manipulator systems.

The most important part of any evolutionary method is the fitness evaluation. In the feedback tuning, the fitness is evaluated through the simulation of the manipulator system. For clarity of understanding, the pseudocode of the fitness evaluation during the feedback gain tuning of the multi-arm manipulator systems is shown in Table 1. The total fitness score of each individual is the sum of the absolute values of the errors in all the controlled variables. Thus, the fitness-score assigned to the ith gain individual is denoted by Fit_i and is defined as

$$Fit_i = \sum_{k=1}^{n_c} | e_k |, \tag{14}$$

where n_c is the number of controlled variables, and e_k is the error associated with the kth control variable.

Table 1. Pseudocode of fitness evaluation

```
Input:
k_p , k_d , integration step size = step_size = 0.005;
Fitness Score:
for i := 1 to μ do
Initialize(); // Initialize Manipulator Parameters
Abs_error = 0.0; // Total Absolute Error
for t := 1 to t_max do  // t_max is the maximum time allowed for each gain tuning
Path(t); // Define Trajectories
ModelParm(); // Define the Manipulator Model Parameters
Controller(); // Inverse Dynamics Controller
Abs_error = Abs_error + sum of absolute errors on
all the controlled variables;
RungeKutta(&t, step_size); // Integrate the States
Fit [i]  = Abs_error; // Fitness Score
```

5 Numerical Simulation

5.1 Set-up

A two-arm manipulator system with three links per manipulator holding an object has been used to verify the effectiveness of the evolution-based inverse dynamics controllers. All the joints are assumed to be one-degree-of freedom rotational joints with rigid links. The two-arm co-operating manipulator system is shown in Fig.2.

Two manipulators are mounted on a free-base with a base frame $\sum_b$ defined at the center of mass (CM) 'B' of the free base. All the links are assumed to be of equal length ${}^{k+1}_{k}l_j$ = 1.0 m. $\forall j \in \{1, 2\}$ and $\forall k \in \{1, \ldots, 3\}$. The mass and inertia of each link are ${}^{k}m_j$ = 1 Kg. and ${}^{k}I_j$ = 0.33 Kg-m^2, $\forall j \in \{1, 2\}$ and $\forall k \in \{1, \ldots, 3\}$, respectively. It is assumed that both the manipulators are mounted on the base such that their distances from the CM of the base 'B' are equal. Here, this distance ${}^{1}_{b}l_j$ = 1.5m, $\forall j \in \{1, 2\}$. The mass of the base m_b = 200 Kg, and its inertia I_b = 40 Kg-m^2. The mass of the object m_o = 2 Kg, and inertia I_o = 0.5 Kg-m^2. Initially, the object center-of-mass was at (0.0, -1.7320508, 0.0) metres. Corresponding to the object position, the joint angles of manipulator 1 and 2 were (-1.0471976, -1.0471976, -1.0471976) radians and (-2.0943951, 1.0471976, 1.0471976) radians, respectively. The initial object orientation about the z-axis of the base frame $\sum_b$ was zero radians. Also, the orientations of the base frame $\sum_b$ with respect to the world reference frame $\sum_w$ of all the axes were zero radians. For a fixed-base manipulator system, $\sum_b$ and $\sum_w$ were chosen to be coincident with each other. The sampling time for all the simulations described in this research were fixed at 0.001 sec. The diagonal elements of the controller gain matrices $\mathbf{K}_p$ and $\mathbf{K}_d$ were set to 16 and 10, respectively, for all the simulation work for the conventional inverse dynamics controllers.

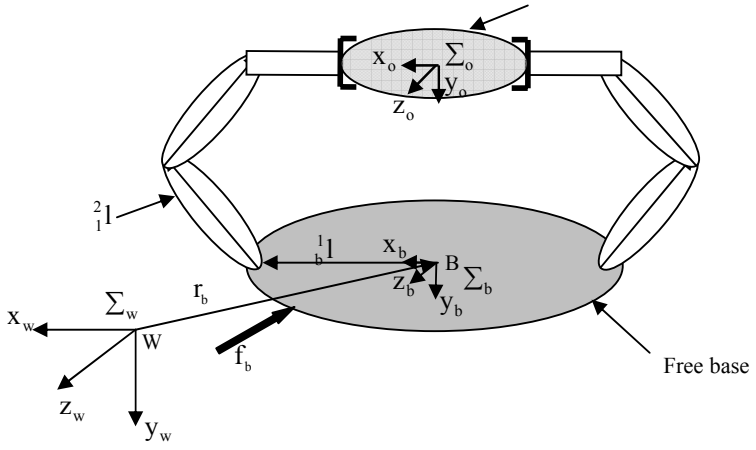


Fig. 2. Two-arm three-link co-operating manipulator system

Every individual in HEA consisted of proportional (k_p) and derivative (k_d) controller gains. Only one k_p and one k_d value were used for all the control variables.

Thus, the number of variables per individual was two. The population pool consisted of twenty individuals. HEA was run for thirty generations for all the problems. The number of individuals and generations were so chosen for two reasons. The first reason is that, in most of the simulation cases, the improvement in error was almost negligible after twenty generations. Secondly, it was observed that it took nearly seventy two hours on a Pentium II, 350 MHz processor for the completion of thirty generations. The constant offset z_c was set to 10^{-4}. The major problem with gain parameter tuning is that there is no fixed feasible region within which the search process can be bound. Hence, an arbitrary feasible region (0, 50) was adopted for the initialization of k_p and k_d values. Further, only positive values were allowed, as positive values can yield uniform stability for the inverse dynamics controller [1]. Thereafter, no such bounds were applied on the search process, i.e., the search process was allowed to search through the entire positive real number space, and in the event of any negative gain values, the gain was given an arbitrarily low value of 0.001. However, very large values of k_p would result in large control torques that might exceed the actuator saturation values. In order to overcome this problem, the torques were constrained to lie between the actuator limits. Thus, a torque was represented as

$$|\mathbf{T}| \leq \mathbf{T}_{max}, \tag{15}$$

where $\mathbf{T}_{max}$ is the maximum torque value that can possibly be applied to safely drive the manipulator joints by the control motors. In this study, $|\mathbf{T}_{max}|$ = 50.0 Nm. For the fitness evaluation, the manipulator simulation was run for a time of t_{max} = 0.5sec with a sampling time of 0.005sec. In the simulation studies of controller parameter optimization, the aim was to minimize the errors in the end-effector/object position and orientation. Thus, the fitness value of each individual was the sum of the absolute errors in all the end-effector/object control variables. Further, in the case of free-flying manipulator systems, where it was desired to keep the base in its original position, the errors in the base control variables were also included in the fitness calculations.

5.2 Test Problem

The test problem chosen to verify the performance of the inverse dynamics control algorithms with motive force compensation is the circle problem, with the desired trajectories of the object specified by

$$\gamma = r_1 \sin(\xi t)$$
$$p_x = d + r\cos(\xi t)$$
$$p_y = -h + r\sin(\xi t)$$

where r is the radius of the circle with center at the co-ordinate point (d, h), r_1 is the amplitude of the orientation variation, $\xi = k\pi$, with k as a constant, γ is the end-

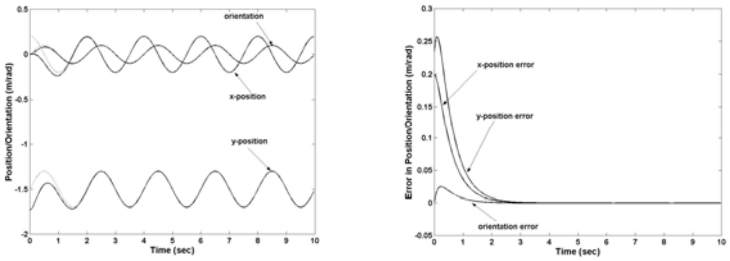
effector orientation about the z-axis, and p_x and p_y are the x- and y-positions of the end-effector. Here, the parameters of the circle were taken as r = 0.2m, r_1 = 0.1m, (d, h) = (0.0, -1.5) and k = 1.

In essence, for a co-operating manipulator system holding a common object whose motion is controlled along the desired path given above, there are two position constraints along the x- and y-axes, and one orientation constraint. Further, there are three internal forces $f^i_{e,x}$, $f^i_{e,y}$, and $f^i_{e,\gamma}$, respectively, due to the three lost degrees of freedom in the presence of constraints. These internal forces cancel each other and are essential to hold the object rigidly.

6 Results and Discussions

6.1 Fixed-base Two-arm Co-operating Manipulator System

The performance of the conventional inverse dynamics controllers for a two-arm co-operating manipulator system rigidly holding an object that tracks a circular contour in space is shown in figs.3 to 7. Fig.3 (a) shows the position and orientation tracking performance of the object. The tracking errors are presented in Fig.3 (b). These results show that the object tracks the desired trajectories perfectly. The circular contour tracking in the x-y co-ordinate plane is shown in Fig.4. The actual trajectory represented by the dashed lines tracks the desired circular contour represented by the solid line in spite of starting at a completely faraway location. Also, the velocity tracking performance is very good, as shown in Fig.5. The input torques are shown in figs.6 (a) and (b). It can be observed from these figures that the second manipulator requires larger torques than the first manipulator. Figs.7 (a) and (b) show the internal forces created during the tracking process. These figures clearly show that the internal forces cancel each other at every instant of time.



(a) Position and Orientation Tracking

(b) Position and Orientation Error

Fig. 3. Response of object position and orientation for a fixed-base configuration. The dashed lines indicate desired and the solid lines indicate actual

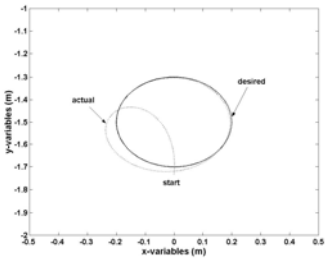


Fig. 4. Motion of the payload in Cartesian space for fixed-base

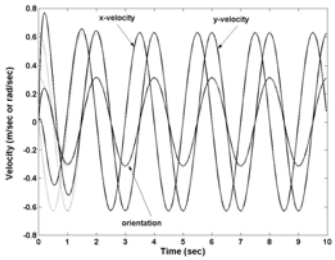
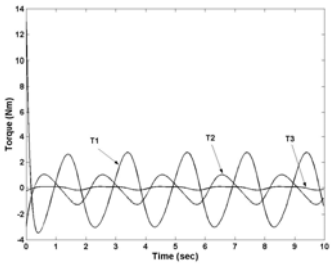
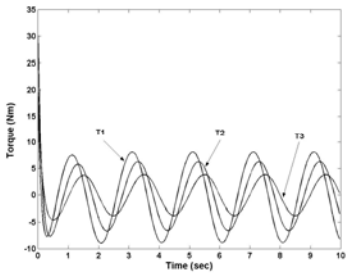


Fig. 5. Object velocity tracking for fixed-base

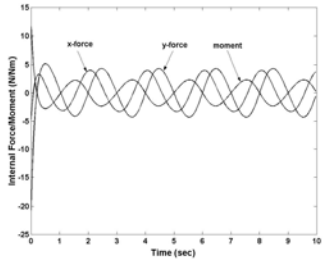


(a) Manipulator 1

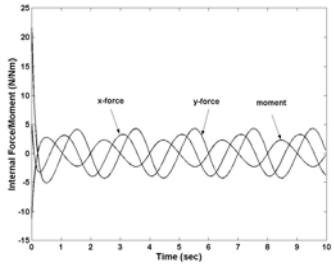


(b) Manipulator 2

Fig. 6. Torque profiles for the fixed-base configuration



(a) At End-effector 1



(b) At End-effector 2

Fig. 7. Internal forces at the end-effector frames for a fixed-base configuration

The evolutionary tuning of the feedback gains k_p and k_d is presented in Figs.8 (a) and (b). The gain values for k_p and k_d after thirty generations were 42950.816466 and 312.897062, respectively. The generation best and average total absolute error characteristics are shown in Fig.9. This shows that the evolutionary learning converges around twenty generations. Fig.8 clearly shows that the k_p and k_d values

vary significantly after the 20th generation in spite of the total absolute error value being constant. This indicates that there are many combinations of k_p and k_d values that yield almost identical performance with apparently different controllers. The performance characteristics with these values of k_p and k_d are shown in Figs.10 to 14. The position and orientation tracking characteristics and their errors are shown in Figs.10 (a) and (b), respectively. These figures indicate that the errors drop to the zero value almost instantly (around 0.02sec), and the end-effectors accurately tracked the desired trajectories. Whereas, in the case of hand-tuned controllers as presented in Figs.3 (a) and (b), the settling time was around two seconds. This excellent tracking performance is further illustrated in Fig.11, which shows the circular contour tracking in the x-y plane. In the case of instantaneous tracking of desired circular trajectories where the initial positions (at 0th instant of time) are away from the required circular contour, it is quite obvious that greater torques will be needed and thus the initial velocities will be higher until the desired contour is reached. This is confirmed by the velocity and torque profiles shown in Figs.12 and 13 respectively. Fig.13 in particular shows perfect velocity tracking performance once the end-effectors have moved onto the desired circular trajectory.

Apart from the initial time period of 0.02sec, the torque requirements are almost identical to those in the hand-tuned case. Further, the large initial values of torques, which directly control the magnitude of end-effector forces acting on the object and which in turn control the magnitude of internal forces on the object, induce relatively large values of internal forces. This is shown in Fig.14. Here, it is important to note that, whenever it is required to have smaller internal forces, the same method can be used directly with lower active torque limits. In this simulation, the torques limits were ± 50 Nm.

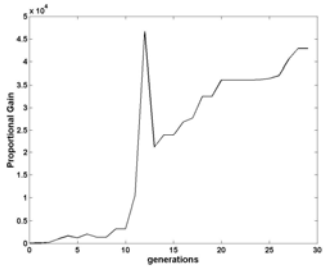


Fig. 8.(a) Best proportional gains for the fixed-base, co-operating system

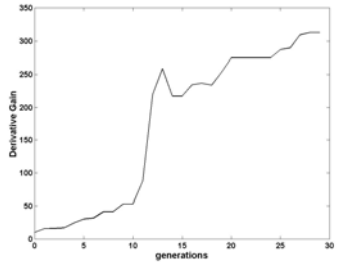


Fig. 8.(b) Best derivative gains for the fixed-base, co-operating system

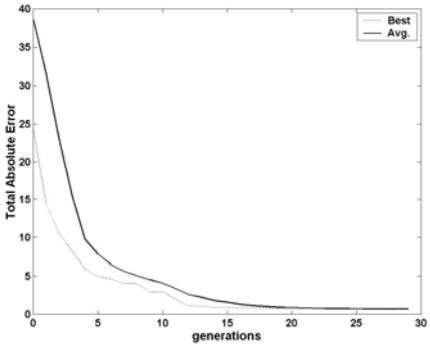
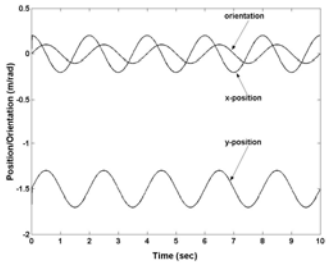
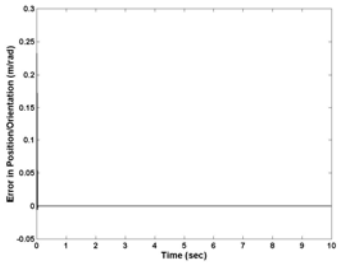


Fig. 9. Best and average of the total absolute errors for the fixed-base, co-operating system



(a) Position and Orientation Tracking



(b) Position and Orientation Error

Fig. 10. Response of object position and orientation for a fixed-base configuration. The dashed lines indicate desired and the solid lines indicate actual

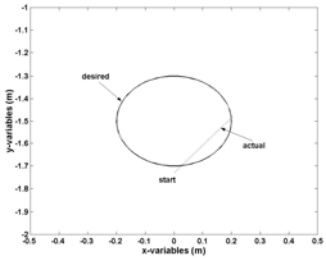


Fig. 11. Motion of the payload in Cartesian space for the fixed-base configuration

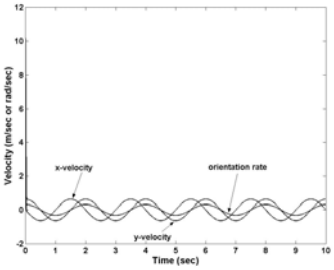
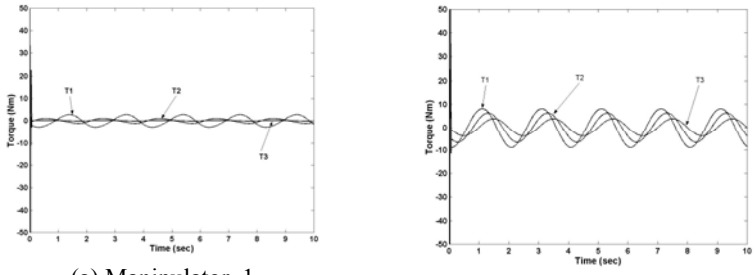
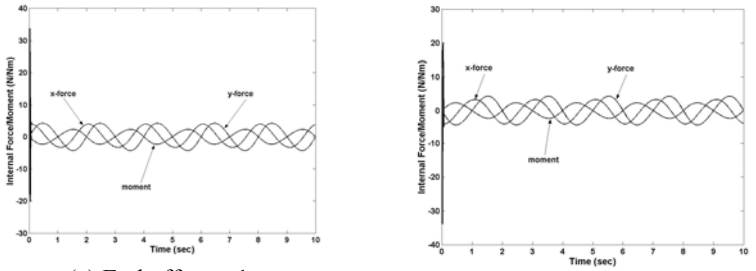


Fig.12. Object velocity tracking for fixed-base configuration



(a) Manipulator 1 (b) Manipulator 2

Fig. 13. Torque profiles for the fixed-base configuration



(a) End-effector 1 (b) End-effector 2

Fig. 14. Internal forces at the end-effector frames for a fixed-base configuration

6.2 Free-floating Two-arm Co-operating Manipulator System

Interestingly enough, all the performance characteristics for the free-floating system exactly matched those for the fixed-base manipulator system. Hence, those figures are not repeated. Essentially, this shows that, in free-floating, co-operating space manipulator systems, the tracking control does not need any extra energy to perform exactly the same job compared to the fixed-base co-operating manipulator systems. The base motion with conventional inverse dynamics controllers while tracking the circular contour is shown in Fig.15. Also, the forces of interaction between the manipulators and the base are shown in Fig.16.

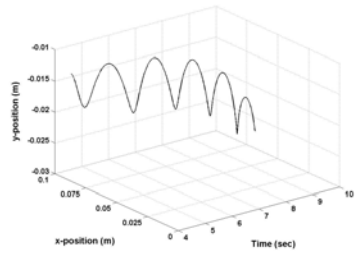


Fig. 15. Base motion in the free-floating configuration

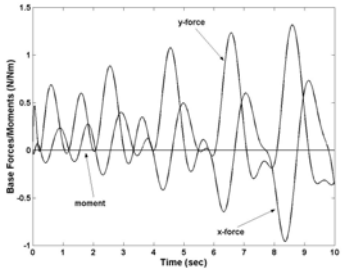


Fig. 16. Base interaction forces in the free-floating configuration

In the case of evolution-based inverse dynamics controllers, all the performance characteristics were also exactly the same as for the fixed-base, co-operating manipulator systems. Hence, only the base interactions, which are not present in its fixed-base counterpart, are shown in Figs.17 and 18. Comparing the base motions of evolutionary-tuned and hand-tuned control systems shows that the displacement in the y-direction in the former case is greater compared to the latter case and vice-versa. Throughout the simulation period, the base interaction forces in the case of the hand-tuned system are slightly less than those of the evolutionary-based system. However, all other performances in the fixed-base case are extraordinarily better, as discussed before.

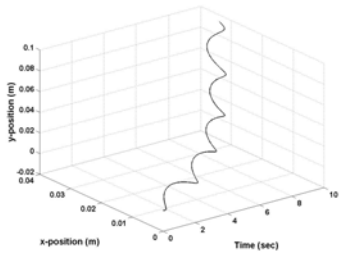


Fig. 17. Base displacements in the free-floating configuration

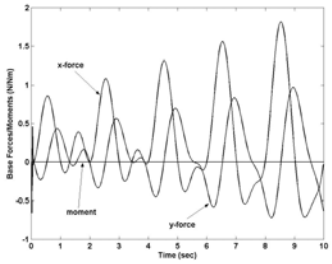


Fig. 18. Base interaction forces in the free- floating configuration

7 Conclusions

In this paper, the most difficult part of any inverse dynamics controller, the feedback gain tuning, has been discussed. A highly efficient hybrid evolutionary algorithm (HEA) was used to tune the feedback gains. The higher speed and greater accuracy features of HEA were utilized to tune the gains within a few generations. It was ob-

served that twenty generations with a population size of only twenty individuals are sufficient to find the optimal gains of complex, multi-arm manipulator systems. In all the simulations, the limits on the motor torques were the only mechanism used to avoid limit any exceptionally rise of proportional gains. Such limiting values of the torques also limit the end-effector and internal forces in a co-operating manipulator system. In all the simulations, the aim was to verify the potentials of the HEA method for gain tuning to improve the end-effector position and orientation tracking performance. For co-operating, fixed-base and free-floating manipulator systems, the tracking performance with evolution-based controllers was very good. Overall, it was shown that the system performances were greatly improved with evolutionary tuning.

References

1. Swain, A.K.: Dynamic modelling and control of robotic manipulators with an investigation of evolutionary computation methods. PhD dissertation, University of Sheffield, United Kingdom, 2001.
2. Kim, J. –H. and Shim, H. -S.: Evolutionary programming-based high precision controller design. Proc. Evolutionary Programming V: Proceedings of Fifth Annual Conference on Evolutionary Programming, Fogel, L.J., Angeline, P.J. and Bäck, T., eds, Cambridge, MA: MIT Press, 1995.
3. Ge. S.S., Lee, T.H. and Zhu, G.: Genetic algorithm tuning of Lyapunov-based controllers: An application to a single-link flexible robot system. IEEE Transactions on Industrial Electronics, vol. 43, no. 5, pp.567-574, 1996.
4. Katic, D.: Genetic algorithm tuning of connectionist controller for compliant robotic tasks. Proceedings of the 14th World Congress of IFAC, pp.155-160, 1999.
5. Porter, B. and Allaoui, C.: Evolutionary design of high-accuracy digital trajectory-tracking controllers for robotic manipulators. Proceedings of the Fourteenth World Congress of IFAC, 1999, pp.25-30.
6. Swain, A.K. and Morris, A.S.: A novel hybrid evolutionary programming method for function optimisation. Proc. Congress on Evolutionary Computation (CEC2000)- IEEE/IEE Intl. Conf. on Evolutionary Computation, San Diego, USA, July, 2000.
7. Swain, A.K. and Morris, A.S.: A hybrid evolutionary algorithm for multimodal function optimisation. Proc. Genetic and Evolutionary Computation Conference (GEECO-2000) Late-breaking Papers, Las Vegas, Nevada, July 8-12, 2000.
8. Fogel, D.B.: Evolutionary computation: towards a new philosophy of machine intelligence. IEEE press, Piscataway, N.J., USA, 1995.