Robotica (2009) volume 27, pp. 677–687. © 2008 Cambridge University Press
doi:10.1017/S0263574708005055 Printed in the United Kingdom

Using a genetic algorithm to fully optimise a fuzzy logic controller for a two-link-flexible robot arm

V. B. Nguyen and A. S. Morris*

Department of Automatic Control and Systems Engineering, University of Sheffield, UK.

(Received in Final Form: July 23, 2008. First published online: September 8, 2008)

SUMMARY

Flexible manipulators have received wide attention because they surpass their rigid counterparts in many criteria. Unfortunately, traditional controller design methods for flexible arms implemented by human experts are usually tedious and intractable. In order to improve system behaviour, this paper proposes schemes based on genetic algorithms (GAs) which optimise the parameters of a fuzzy logic controller for a robotic manipulator with two-link flexibility and two-joint elasticity. Two alternative GA-optimised schemes are simulated and their control behaviour is compared with that of human-expert-designed controller.

KEYWORDS: Flexible manipulator; Fuzzy logic controller; Genetic algorithm; Optimisation.

1. Introduction

The dynamics and control of flexible robotic manipulators have received great attention for more than 30 years due to their superiority over rigid counterparts in criteria such as payload-to-arm ratio, operation speed and energy consumption. In order to investigate the behaviour of such manipulators, the dynamic equations are usually based on either the Assumed Modes Method (AMM)[1–3] or the Finite Element Method (FEM).[4,5] It has been shown that the AMM is superior for modelling multi-link flexible manipulators both in terms of its lesser computational complexity and also because it gives greater insight into the dynamic structure of the manipulator system.[6] Therefore, this paper uses a two-link flexible manipulator model based on the AMM and Euler–Lagrange equations published earlier.[7]

There is a large amount of published literature that describes the application of various forms of traditional control techniques to flexible robotic manipulators. These achieve varying levels of success when the controlled system is well defined. However, in practice, the various uncertainties that exist in a real system often seriously degrade the performance of such traditional controllers. In consequence, much effort has recently been directed towards improving system performance by using controllers based on intelligent soft-computing tools such as fuzzy logic and neural networks in the controller.

Recently, controllers based on intelligent tools such as fuzzy logic and neural network have also been widely used as effective methods for improving system performance. Moudgal *et al.*[8] implemented a 'fuzzy model reference learning controller' for a flexible robot. They concluded that the automatically tuned controller could adapt to various payloads to improve system behaviour compared with that of the manually constructed fuzzy controller. The fuzzy logic technique has also been expressed as a fuzzy-logic-based $(PI+D)^2$ control scheme for a 'shoulder–elbow-like' single flexible-link robot manipulator.[9] Their results showed that the controller allowed the arm to track a specific set-point with satisfactory accuracy and robustness, as well as totally suppressing arm vibration. Lin[10] created a hierarchical fuzzy logic controller using a singular perturbation approach for a robotic manipulator modelled with link structural flexibility. The author decomposed the large-scale system into a finite number of reduced-order subsystems in order to control the force and the position of the end point. In another adaptive control scheme to track the endpoint of a two-link flexible robot,[11] a fuzzy logic system was implemented to generate an output variable in response to elastic deformations of the links and to adapt the Jacobian transpose control law. In a different approach, Yeşildirek *et al.*[12] designed a neural network controller for feedback linearisation in the slow subsystem, with a PD controller for outer-loop for tracking and a robustifying term to ensure closed-loop stability. Although the neural network acting as a multi-layer non-linear net requires no offline learning phase and specific initial parameters, the tracking error and control signal still guaranteed boundedness. Talebi *et al.*[13] utilised a feedforward neural network associated with dynamical elements to build recurrent networks. This method can be considered firstly as an offline learning engine which computes the deflections due to link flexibility and generates parameters to calculate the forward dynamics. It can also be regarded as an online identification scheme for designing an artificial neural-network-based controller. More recently, a robust control scheme for a two-link flexible manipulator with neural network-based quasi-static distortion compensation has been reported.[14] This study divided the system into two parts; firstly, a decomposition-based robust controller was created for the slow subsystem, and secondly, an H^∞ control was outlined for the fast subsystem. The control techniques described earlier use advanced methods to overcome challenges in robotic manipulator systems with link flexibility. However, parameter design in these controllers is still based on engineering insight, and their potential ability to improve system performance has not yet been fully exploited.

* Corresponding author: E-mail: a.morris@sheffield.ac.uk

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

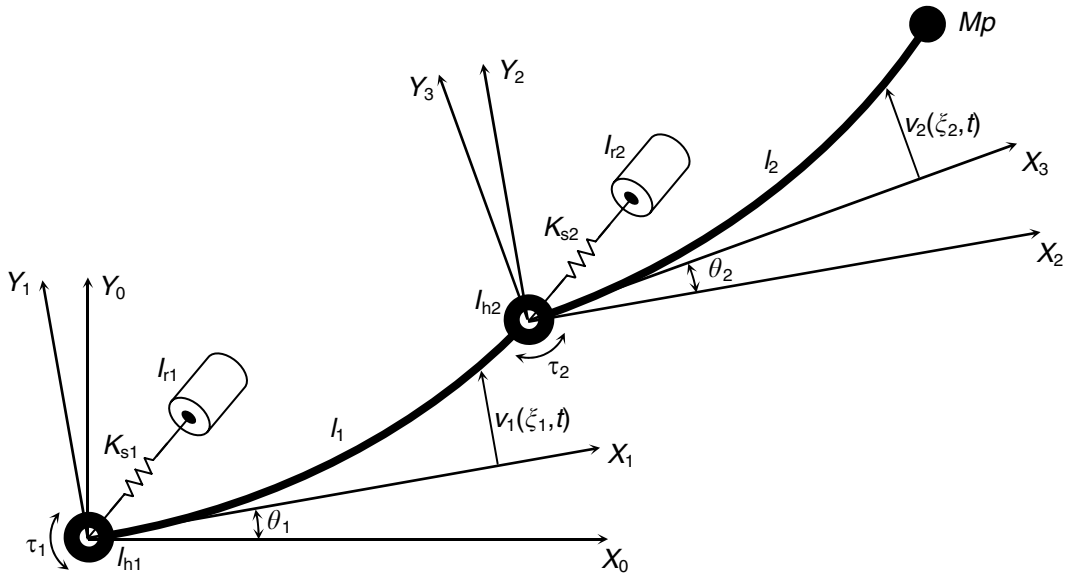


Fig. 1. Structure diagram of two-flexible-link and joint robotic arm.

The optimisation and searching abilities of genetic algorithms (GAs) are also effective tools for improving the behaviour of robot manipulator systems. GAs were pioneered by Holland,[15] inspired by Darwin's theory of biological evolution tools. Besides their success in solving search problems, they have also been applied for automatically designing fuzzy logic controllers. Buckley[16] had the idea of applying a GA to various fuzzy optimisation problems, showing that the algorithm could bring good approximate solutions for applications such as fuzzy optimisation, the fuzzy maximum flow problem, fuzzy regression and tuning a fuzzy logic controller. A simultaneous design of membership functions and rule set for a fuzzy logic controller was implemented to examine the applicability of GAs for cart and truck controllers.[17] However, the GAs were only used to determine the base lengths of triangular fuzzy linguistic variables and the initial rule set. Gürocak[18] introduced a GA-based method for modifying the parameters of a PD-like fuzzy logic controller, where the controller was considered as a multivariate function and optimised systematically for a second-order system and an inverted pendulum problem. For robotic manipulators, various studies have developed GA-based control schemes with different structures. One of these presented a computer-implemented procedure to create a hierarchical hybrid fuzzy-PID controller for a two-link robotic manipulator.[19] The procedure utilised GAs to optimise a fuzzy logic engine that chose parameters for a PID controller, and results showed very good stabilisation and tracking performance. Unfortunately, all of the contributions discussed earlier were only applied to simple systems or rigid robotic manipulators. However, despite the fact that GAs require very large computation time, their success in simple systems suggests that they can play a vital role in optimising other very complex systems such as flexible robotic manipulators.

This paper therefore proposes GA schemes to optimise the structure and parameters of a fuzzy logic controller for an AMM model of a robot manipulator with two flexible links and two elastic joints. The model shown in section 2 is based on the Euler–Lagrange equation and the AMM to find the dynamical equations of the arm. Section 3 presents the concepts of GAs and fuzzy logic controllers. Methods to tune the fuzzy logic membership function parameters and rule set using a GA are then described in section 4. In the first of these (GAMFLC), the widths and peak positions of the membership functions are tuned simultaneously by the GA. In the second scheme (GaMRFLC), which is effectively an extension of the first scheme, the GA alters the rule set of the fuzzy controller as well as the membership functions. Section 5 firstly presents and discusses simulation results for a conventional, expert-designed, PD-like fuzzy logic controller (ExFLC) and then compares these with the GAMMFLC and GAMRFLC GA-optimised schemes.

2. Flexible Manipulator Modelling

Figure 1 shows a structural diagram of the two flexible links/two elastic joints manipulator simulated. It is assumed that the longitudinal deflections of the links are negligibly small and gravity terms are omitted since the arm only moves in the horizontal plane. The flexible joints are considered as linear torsional springs with coefficients K_{si} connecting motor shafts with inertias I_{ri} and link hubs with inertias I_{hi}. The angles between the inertial coordinate frames ($O_0X_0Y_0$, $O_2X_2Y_2$) and the moving frames ($O_1X_1Y_1$, $O_3X_3Y_3$) of link 1 and link 2, respectively, are θ_i. Link parameters consist of lengths l_i, masses m_i, and uniform flexural rigidities EI_i. The end-effector at the free end of link 2 carries a payload M_p.

According to Meirovitch,[20] deflections of the free ends of links are $v_i(\xi_i, t)$ given as

$$v_i(\xi_i, t) = \sum_{j=1}^{n} \phi_{ij}(\xi_i) q_{ij}(t) \tag{1}$$

where $\boldsymbol{\varphi}_i(\xi_i) = [\,\phi_{i1}\ \phi_{i2}\ \ldots\ \phi_{in}\,]^T$ is a mode shape function, $\mathbf{q}_i(t) = [q_{i1}\ q_{i2}\ \ldots\ q_{in}]^T$ is a time varying modal function, i is the link index and j is the number of modes. ξ_i is the distance

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

of the deflection point of ith link.

$$\phi_{ij}(\xi_i) = C_{ij}[\cos(\beta_{ij}\xi_i) - \cosh(\beta_{ij}\xi_i)] + \sin(\beta_{ij}\xi_i) - \sinh(\beta_{ij}\xi_i) \tag{2}$$

where $C_{ij} = (\cos(\beta_{ij}l_i) + \cosh(\beta_{ij}l_i))/(\sin(\beta_{ij}l_i) - \sinh(\beta_{ij}l_i))$, $\beta_{ij}^4 = (\omega_{ij}^2\rho_i)/\mathrm{EI}_i$ and ω_{ij} is natural frequency of ith link and jth shape mode.[21]

The dynamic equations of the system are derived from the total kinetic energy and total potential energy of the arms. The kinetic energy of the whole arm is related to links, hubs, rotors and payload.[7]

$$K = K_\mathrm{l} + K_\mathrm{h} + K_\mathrm{r} + K_\mathrm{p} \tag{3}$$

where K_l, K_h, K_r and K_p are the kinetic energy of the links, hubs, rotors and payload, respectively. The energy caused by motion of the rotors can be calculated as

$$K_\mathrm{r} = \sum_i \frac{1}{2} J_i \dot{\alpha}_i^2 \tag{4}$$

where $J_i = \delta_i^2 I_{ri}$, with δ_i and I_{ri} being the gear ratios of the motors and inertias of the rotors, respectively; α_i is the angle of ith rotor. The kinetic energy due to the motion of the hubs is given by

$$K_\mathrm{h} = \sum_i \frac{1}{2} I_{\mathrm{h}i} \dot{\theta}_i^2 \tag{5}$$

with $I_{\mathrm{h}i}$ being the inertias of the hubs. The energy created when the links move can be written as

$$K_\mathrm{l} = \sum_i \frac{1}{2} \rho_i \int_0^l \dot{\boldsymbol{\sigma}}_i^T \dot{\boldsymbol{\sigma}}_i \, \mathrm{d}\xi \tag{6}$$

According to Fig. 1, $\boldsymbol{\sigma}_i(\theta_i, \xi_i, t)$ are the vectors of ξ_i on coordinate $O_1X_1Y_1$ and $O_3X_3Y_3$. They can be expressed as[22]

$$\boldsymbol{\sigma}_i(\theta_i, \xi_i, t) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \begin{bmatrix} \xi_i \\ v_i(\xi_i, t) \end{bmatrix}, \tag{7}$$

$$\dot{\boldsymbol{\sigma}}_i(\theta_i, \xi_i, t) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \begin{bmatrix} -\dot{v}_i(\xi_i, t) \\ \xi_i\dot{\theta}_i + \dot{v}_i(\xi_i, t) \end{bmatrix}. \tag{8}$$

Substituting for $\dot{\boldsymbol{\sigma}}_i(\theta_i, \xi_i, t)$ from (8) in (6) yields

$$K_\mathrm{l} = \sum_i \frac{1}{2} \rho_i \int_0^l \left[v_i^2(\xi_i, t)\dot{\theta}_i^2 + \xi_i^2\dot{\theta}_i^2 + \dot{v}_i^2(\xi_i, t) + 2\zeta_i\dot{\theta}_i v_i(l_i, t) \right] \mathrm{d}\zeta_i. \tag{9}$$

Similarly, the energy due to the motion of the payload can be calculated from (9) with $\xi_i = l_i$,

$$K_\mathrm{p} = \sum_i \frac{1}{2} M_\mathrm{p} \left\{ v_i^2(l_i, t)\dot{\theta}_i^2 + l_i^2\dot{\theta}_i^2 + \dot{v}_i^2(l_i, t) + 2l_i\dot{\theta}_i\dot{v}_i(l_i, t) \right\}. \tag{10}$$

The total potential energy is expressed by

$$P = P_\mathrm{links} + P_\mathrm{joints} \tag{11}$$

where P_links and P_joints are caused by the bending deflections of the links and the elastic deformations of the joints, respectively, and are given by

$$P_\mathrm{links} = \sum_i \frac{1}{2} \int_0^{l_i} \mathrm{EI}_i \left(\frac{\partial^2 v_i(\xi_i, t)}{\partial \xi_i^2} \right)^2 \mathrm{d}\xi_i, \tag{12}$$

$$P_\mathrm{joints} = \sum_i \frac{1}{2} K_{\mathrm{s}i} (\alpha_i - \theta_i)^2. \tag{13}$$

In addition, when the system moves, a part of the total energy D is consumed by friction from the bearings and the contact of links with the environment

$$D = \sum_i \frac{1}{2} \mathbf{q}_i^T \mathbf{K}_{\mathbf{m}i} \mathbf{q}_i \tag{14}$$

where $\mathbf{K}_{\mathrm{m}i} = \mathrm{diag}(k_{ij})$, and k_{ij}, $j = 1 \ldots n$ are damping coefficients. The dynamic model is obtained by substituting the energy expressions (3)–(14) in Lagrange's equation[20]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\frac{\partial V}{\partial \mathbf{L}}\right] - \frac{\partial V}{\partial \mathbf{L}} + \frac{\partial D}{\partial \mathbf{L}} = \mathbf{F}, \quad V = K - P \tag{15}$$

where $\mathbf{L}$ is the vector of variables in the system,

$$L = [\alpha_1\alpha_2\theta_1\theta_2 q_{11} q_{12} \ldots q_{1n} q_{21} q_{22} \ldots q_{2n}]^T, \tag{16}$$

and $\mathbf{F}$ is the vector of applied control signals,

$$\mathbf{F} = [\, u_1 \; u_2 \; 0 \; 0 \; 0 \; 0 \ldots 0 \; 0 \; 0 \ldots 0 \,]^T. \tag{17}$$

Next, (15) can be expressed as

$$\mathbf{M}\ddot{\mathbf{L}} + \mathbf{C}\dot{\mathbf{L}} + \mathbf{H}\mathbf{L} = -\mathbf{G} + \mathbf{F} \tag{18}$$

or in the following detailed form with $n = 2$:

$$\begin{bmatrix} \mathbf{J} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & M_{11} & M_{12} & \mathbf{M_{13}} & \mathbf{M_{14}} \\ \mathbf{0} & M_{12} & M_{22} & \mathbf{M_{23}} & \mathbf{M_{24}} \\ \mathbf{0} & \mathbf{M_{13}^T} & \mathbf{M_{23}^T} & \mathbf{M_{33}} & \mathbf{M_{34}} \\ \mathbf{0} & \mathbf{M_{14}^T} & \mathbf{M_{24}^T} & \mathbf{M_{34}^T} & \mathbf{M_{44}} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{\alpha}} \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\mathbf{q}}_\mathbf{1} \\ \ddot{\mathbf{q}}_\mathbf{2} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{K_{m1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{K_{m2}} \end{bmatrix}$$

$$\times \begin{bmatrix} \dot{\boldsymbol{\alpha}} \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\mathbf{q}}_\mathbf{1} \\ \dot{\mathbf{q}}_\mathbf{2} \end{bmatrix} + \begin{bmatrix} K_{\mathrm{s}1} & 0 & -K_{\mathrm{s}1} & 0 & \mathbf{0} & \mathbf{0} \\ 0 & K_{\mathrm{s}2} & 0 & -K_{\mathrm{s}2} & \mathbf{0} & \mathbf{0} \\ -K_{\mathrm{s}1} & 0 & K_{\mathrm{s}1} & 0 & \mathbf{0} & \mathbf{0} \\ 0 & -K_{\mathrm{s}2} & 0 & K_{\mathrm{s}2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & A_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & A_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \theta_1 \\ \theta_2 \\ \mathbf{q}_\mathbf{1} \\ \mathbf{q}_\mathbf{2} \end{bmatrix}$$

$$= -\begin{bmatrix} \mathbf{0} \\ g_1(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \mathbf{q}, \dot{\mathbf{q}}) \\ g_2(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \mathbf{q}, \dot{\mathbf{q}}) \\ \mathbf{g_3}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \mathbf{q}, \dot{\mathbf{q}}) \\ \mathbf{g_4}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} + \begin{bmatrix} \mathbf{u} \\ 0 \\ 0 \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{19}$$

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

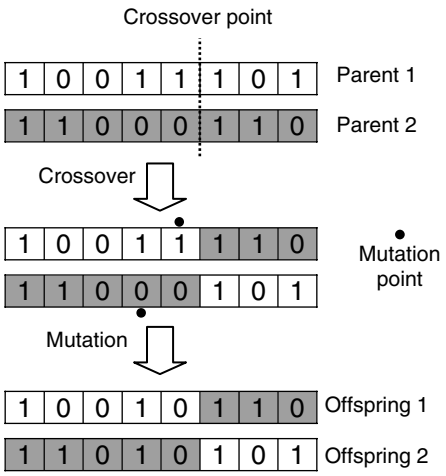


Fig. 2. Genetic algorithm principles.

where $A_1 = \mathbf{EI_1} \int_0^l \boldsymbol{\phi}_{1\xi\xi}^2 \mathrm{d}\xi$ and $A_2 = \mathbf{EI_2} \int_0^l \boldsymbol{\phi}_{2\xi\xi}^2 \mathrm{d}\xi$ Equation (18) can be rewritten in state space form as

$$\dot{\mathbf{X}} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{M}^{-1}\mathbf{H} & -\mathbf{M}^{-1}\mathbf{C} \end{bmatrix} \mathbf{X} + \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1} \end{bmatrix} (-\mathbf{G} + \mathbf{F}),$$
$$\text{where } \mathbf{X} = [\mathbf{L}^T \ \dot{\mathbf{L}}^T] \tag{20}$$

3. GA-Based Optimisation Tool

GAs are effective optimisation tools which use a random mechanism to avoid local minima. They give a new solution from a number of other candidate solutions known as individuals, where each individual has a chromosome and a degree of fitness. Each chromosome is divided into parts as genes and encoded as a string of 0s and 1s; meanwhile fitness is evaluated from genes by a fitness function. A collection of individuals is called a population and normally the population is constant.

The algorithm starts from an initial population in which parents with high fitness are selected to create offspring. Reproductive operations consisting of crossover and mutation are applied to selected individuals (Fig. 2). For crossover, an offspring receives copies of its parents' genes to form a new chromosome. Normally, the crossover point is chosen randomly to simulate the analogous way whereby a parent couple in living organisms contribute their genes to their offspring. For mutation, a bit from a parent is converted to form a new offspring. The function of this operator is to recreate high fitness genes that might have been missed in previous generations.[23]

After the offspring have been created, their fitness is evaluated and compared with that of other individuals in the current population. Individuals with high fitness are added to the new population and low fitness individuals are removed to give space to the offspring.

4. GA Optimised Fuzzy Logic Controllers

Although, fuzzy logic controllers have been applied in a number of complex industrial processes, optimal choice of the linguistic variables, rule set and input/output scale factors is difficult. Traditional design methods are based on empirical knowledge and trials and are thus non-optimal. Such methods are inflexible and often lead to poor controller performance. GAs offer a solution to this difficulty because they can cover a complex space in which an optimal solution is guaranteed. A suitable scheme for implementation of this idea is shown in Fig. 3, where a GA-based optimisation method is used in the automatic design of a fuzzy logic controller. In this, the GA collects information on errors, control torques and system responses and alters the parameters of the membership functions, rule set and scale factors according to the value of the genes in the chromosomes. The grey boxes in Fig. 3 indicate the optimised parts.

4.1. Traditional, expert-designed, PD-like fuzzy logic controller (ExFLC scheme)

In order to demonstrate the benefit of GA optimisation, a traditional, non-optimised, expert-designed, PD-like controller (ExFLC), in which the shapes of membership functions and rules of a PD-like fuzzy logic controller are designed using the control experience of experts, was first constructed and simulated in the usual way.[18,19] In general,

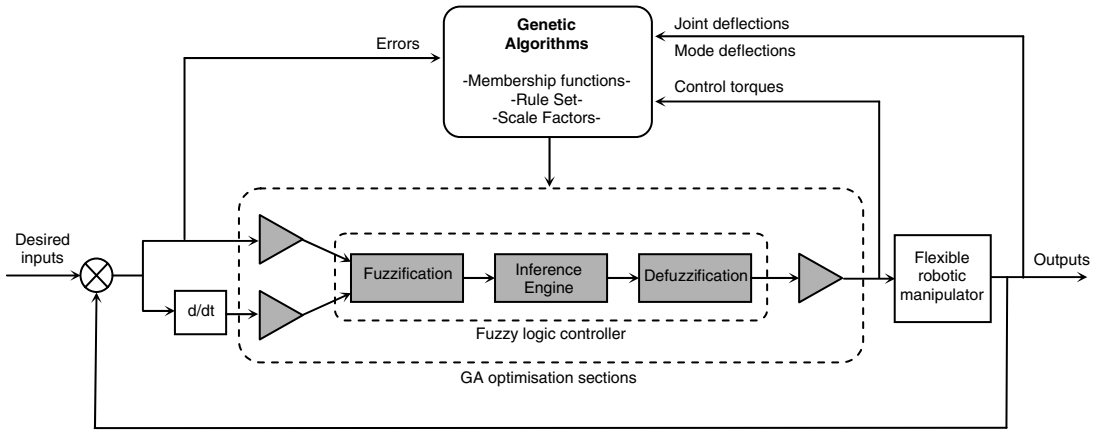


Fig. 3. The structure of a GA-based fuzzy logic controller.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

Table I. The PD-like fuzzy logic controller's rule table.

	Derivative of error						
MFs	NB	NM	NS	AZ	PS	PM	PB
Error							
NB	NB	NB	NB	NM	NM	NS	AZ
NM	NB	NB	NB	NM	NS	AZ	PS
NS	NB	NM	NM	NS	AZ	PS	PM
AZ	NB	NS	NS	AZ	PS	PM	PB
PS	NM	NS	AZ	PS	PM	PB	PB
PM	NS	AZ	PS	PM	PB	PB	PB
PB	AZ	PS	PM	PM	PB	PB	PB

a PD-like fuzzy controller where the membership functions have triangular shapes and rules can be expressed as

IF *error* is *NM* AND *derivative of error* is *PS* THEN *control torque* is *NS*

where error, derivative of error and control torque are input and output variables; NM (negative medium), PS (positive small) and NS (negative small) are linguistic variables; AND and OR are fuzzy logical operators with MIN and MAX definitions, respectively. The outputs of the system are computed by applying MIN operators for implication, MAX for aggregation and the centroid method for defuzzification.

In the controller simulated, the number of input and output membership functions was chosen as 7 and the number of rules used was 49. The distribution of membership functions to rules in order to form the PD-like fuzzy logic controller is given in Table I.[24] There are seven linguistic fuzzy sets for each input and output, namely NB (negative big), NM, NS, AZ (approximate zero), PS, PM (positive medium) and PB (positive big).

4.2. Using GA to optimise membership functions (GAMFLC scheme)

In this first GA-optimised approach, it is assumed that an initial controller has been designed using expert knowledge (ExFLC scheme). The GA optimisation scheme (GAMFLC) adjusts the widths of each membership function from the original left and right widths in the ExFLC scheme (W_{OL} and W_{OR}) to new left and right widths (W_{NL} and W_{NR}) (Fig. 4). Each gene encodes the value of a new width

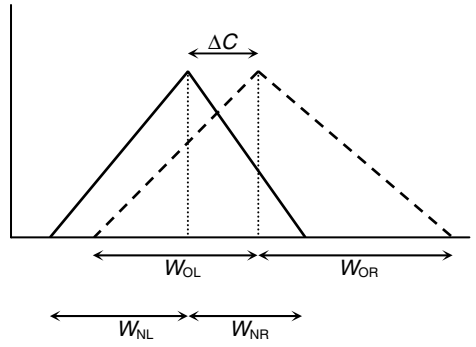


Fig. 4. Membership function parameters.

and the widths of the membership function are changed independently. However, since the system behaviour is also affected by the peak position of each membership function, the optimisation scheme also uses the GA to shift the centre of each membership function by ΔC. Thus, the scheme optimises three parameters of the membership function simultaneously (the left and right widths and peak position), with the expectation of a significant improvement in system performance. It is considered that the number of linguistic variables for two inputs and one output are NI_1, NI_2 and NO. Therefore, the number of genes necessary in each individual in order to encode parameters of the membership functions is $NI_1 \times 3 + NI_2 \times 3 + NO \times 3$, which equals 63 for $NI_1 = NI_2 = NO = 7$.

4.3. Using GA to optimise the rule set as well as the membership functions (GAMRFLC scheme)

For a conventional fuzzy logic controller with two inputs and one output such as the ExFLC scheme described earlier, the number of rules in the fuzzy engine is set to $NI_1 \times NI_2$, where NI_1 and NI_2, are the number of linguistic variables for the two inputs. However, it has been shown that good controller performance can often be achieved with a smaller number of rules in which the consequent of each rule in the reduced rule set is optimised.[24] This suggests that optimisation of the rule set as well as the membership functions may also be beneficial. In order to apply GA-based optimal reduction of the rule set, it is convenient to initially assign a weight of 1 to each rule in the full rule set. The GA then identifies those rules that do not contribute significantly to controller performance and assigns a weight of 0 to them, which effectively removes them from the rule set and results in a reduced rule set. In this way, the space of NO linguistic rule-output variables is progressively altered to find an optimal solution. In this scheme to enhance the rule set, the chromosome of each individual has to encode both the consequent of the rule space and the weights of each rule. Table II illustrates the distribution of genes in the chromosome for the scheme (GAMRFLC) in which the membership function, the rule set and scale factors are modified simultaneously.

The last row in Table II gives three scale factors for the two inputs and one output. In the fuzzifiers and defuzzifiers, the default ranges of linguistic variables are defined from -1 to 1. Thus, the behaviour of the fuzzy system can be varied by adding the scale factors in front of the inputs and after the output.

4.4. Performance measurement using fitness functions

Fitness functions play a crucial rule in GAs because the direction of genetic searching totally depends on these functions. GA-based optimisation usually searches the pertinent parameters of the controller in order to increase system performance. Because the purpose of the optimisation problem is to minimise the fitness of individuals, performance measures of the system are compulsory elements in the fitness functions. In rigid robotic manipulators, a fitness can be considered better if the system has smaller errors, smoother torques and faster response

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

Table II. The function of genes in a chromosome.

Index of genes	Number of genes	Description
$1 \ldots \mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}$	$\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}$	Left widths of membership functions
$\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO} + 1 \ldots 2 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO})$	$\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}$	Right widths of membership functions
$2 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + 1 \ldots 3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO})$	$\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}$	Peak positions of membership functions
$3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + 1 \ldots 3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + \mathrm{NI}_1 \times \mathrm{NI}_2$	$\mathrm{NI}_1 \times \mathrm{NI}_2$	Consequents of rules
$3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + \mathrm{NI}_1 \times \mathrm{NI}_2 + 1 \ldots 3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + 2 \times (\mathrm{NI}_1 \times \mathrm{NI}_2)$	$\mathrm{NI}_1 \times \mathrm{NI}_2$	Weights of rules
$3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + 2 \times (\mathrm{NI}_1 \times \mathrm{NI}_2) + 1 \ldots 3 \times (\mathrm{NI}_1 + \mathrm{NI}_2 + \mathrm{NO}) + 2 \times (\mathrm{NI}_1 \times \mathrm{NI}_2) + 3$	3	Scale factors

time.[19] However, in the case of flexible link and elastic joint manipulators, the fitness function must include joint and link mode deflections as well as the usual rigid case parameters:

$$\text{fitness} = \int_{t_s}^{t_f} (k_{e1}e_1 + k_{e2}e_2 + k_{\tau 1}\tau_1 + k_{\tau 2}\tau_2 + k_{d1}d_1 + k_{d2}d_2 + k_{f1}f_1 + k_{f2}f_2)\,\mathrm{d}t \tag{21}$$

where k_{ei}, $k_{\tau i}$, k_{di}, k_{fi} and e_i, τ_i, d_i, f_i are coefficients and values of errors, control torques, joint deflections, mode deflections of links, respectively. t_s and t_f are the start and finish times of running progress of the system.

5. Simulation and Discussion

The physical parameters of the manipulator arm simulated are given in Table III.[25] A variable step Runge–Kutta integration method was used in the simulations. Only the first two mode shapes were considered because the contribution of higher modes ($n \geq 3$) has been shown to be very small.[7] Since the vibration of the flexible arm is sensitive to the shape of the input signal, smooth and continuous tracking inputs were used in order to reduce the vibration of the end tips of the links. The function of the desired trajectories can be expressed as[26]

$$\theta_t(t) = \theta_b + \theta_s \left(\frac{6t^5}{t_s^5} - \frac{15t^4}{t_s^4} + \frac{10t^3}{t_s^3} \right) \tag{22}$$

where $\theta_t(t)$ is the desired tracking curve; θ_b is the initial value of $\theta_t(t)$; θ_s is the desired set point and t_s is the time of motion from θ_b to θ_s. In this approach, the joints are forced to track from $\theta_b = 0$ to $\theta_{s1} = \pi/8$ radian for joint 1 and $\theta_{s2} = \pi/10$ radian for joint 2 within $t_s = 4$ s.

Table III. The Parameters of the two-flexible-linked two-elastic-joint robotic arm.

Properties	Symbol	Value
Links		
Length (m)	l_1, l_2	0.3
Density ($\mathrm{kg\,m^{-1}}$)	ρ_1, ρ_2	0.06435
Young modulus of elasticity ($\mathrm{N\,m^{-2}}$)	E_1, E_2	69×10^9
Link moment of inertia ($\mathrm{m^4}$)	I_1, I_2	8.93×10^{-13}
Joints		
Rotor inertia ($\mathrm{kg\,m^2}$)	I_{r1}, I_{r2}	0.02
Spring constant ($\mathrm{N\,m\,rad^{-1}}$)	K_{s1}, K_{s2}	2
Payload		
Mass (kg)	M_p	0.01
Damping coefficients	k_{11}, k_{12}	7
	k_{21}, k_{22}	7

The purpose of the simulations carried out was to compare the performance of GA-optimised controllers (GAMFLC, GAMRFLC) with an expert-designed controller (ExFLC). This was achieved by comparing the performance of the three controllers when applied to the same two-flexible-link and two-elastic-joint manipulator model. The GAMFLC scheme involves changing the width and the peak points of the membership functions in both inputs and the output. This GA used 10 individuals in each generation, 1700 generations in the evolution progress and a generation gap of 0.8. The number of variables (genes) was calculated in Section 4.2 as 63, with 20 bits in each variable and thus 1260 (63×20) bits in an individual. For the GAMRFLC scheme, the number of variables increases to 164 (calculated from Table II), and so with 20 bits/variable, the number of bits in an individual was 3280 (164×20).

Evaluation of the fitness function given in (21) poses a particular difficulty in the simulation. Numerical integration tools are normally used to obtain approximate solutions provided that the relationship between the system response and the time domain variable is known. Unfortunately, this relationship is not known for a flexible manipulator because the tracking errors, joint deflections, link-mode deflections and control torques are all unpredictable. A commonly used solution to this problem (e.g. ref. [19], p. 456) is to establish a relationship from measurements made on a real flexible manipulator system using suitable sensors. This allows the real fitness value in (21) to be approximated by the following

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

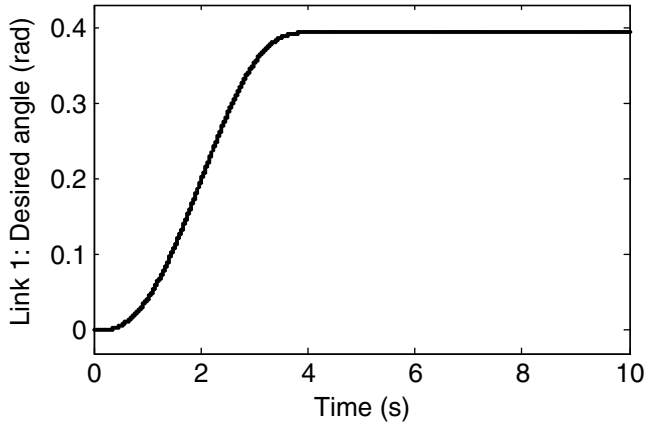


Fig. 5. Link 1: Desired angle.

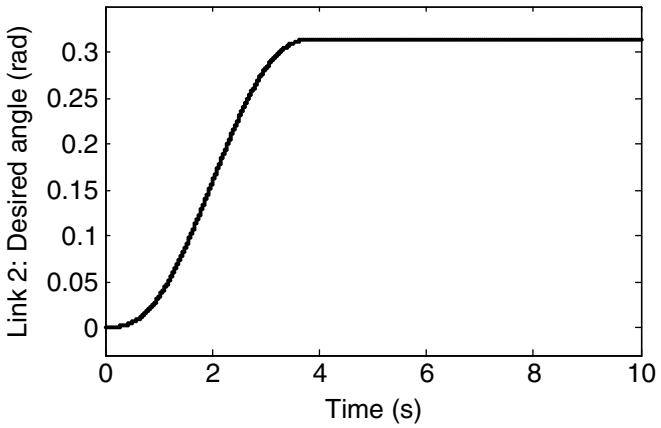


Fig. 6. Link 2: Desired angle.

sum of measures.

$$\text{fitness} = k_{e1}\sum_t |e_1(t)| + k_{e2}\sum_t |e_2(t)| + k_{\tau 1}\sum_t |\tau_1(t)| + k_{\tau 2}\sum_t |\tau_2(t)| + k_{d1}\sum_t |d_1(t)| + k_{d2}\sum_t |d_2(t)| + k_{f1}\sum_t |f_1(t)| + k_{f2}\sum_t |f_2(t)| \quad (23)$$

Coefficients were chosen as $k_{ei} = 1$, $k_{\tau i} = 0.01$, $k_{di} = 1$, $k_{fi} = 100$.

Figures 5 and 6 show the desired trajectory given by (22) with different set points for links 1 and 2, respectively. As can be seen from Figs. 7 and 8, the GAMRFLC scheme gives the smallest tracking errors and oscillations are reduced more quickly compared with the other controllers. The performance of the GAMFLC scheme is less good but still better than the ExFLC scheme.

Figures 9–12 show that the first mode deflections of links 1 and 2 have higher vibration amplitudes than the second mode deflections, irrespective of which controller is used. However, the GAMRFLC scheme achieves the smallest vibration amplitudes for both modes. The performance of GAMFLC in vibration suppression is not as good, but is still better than ExFLC.

Figures 13 and 14 compare the joint deflections obtained with the three controllers. The deflections are smallest for GAMRFLC, slightly larger for GAMFLC and still larger for ExFLC. GAMRFLC also damps out the deflections faster than the other controllers.

Figures 15 and 16 compare the control torque profiles for links 1 and 2, respectively. The peak torque is smallest and time to converge to a constant value is fastest for the GAMRFLC, and thus this scheme achieves the smallest power consumption. GAMFLC is second best in terms of these parameters and ExFLC is worst.

6. Conclusions

The paper has described an investigation into the application of GAs as a tool to optimise fuzzy logic controllers for flexible manipulator systems, with the aim of avoiding the need for engineering insight in the system and thus increasing controller accuracy. Two forms of optimisation scheme have been compared. In the first, both the widths and centre points of membership functions were adjusted simultaneously in order to change the input and output of the fuzzifiers and defuzzifiers in the controller. In the second optimisation scheme, the number of rules, the structure of the rule set and the scale factors were all altered in

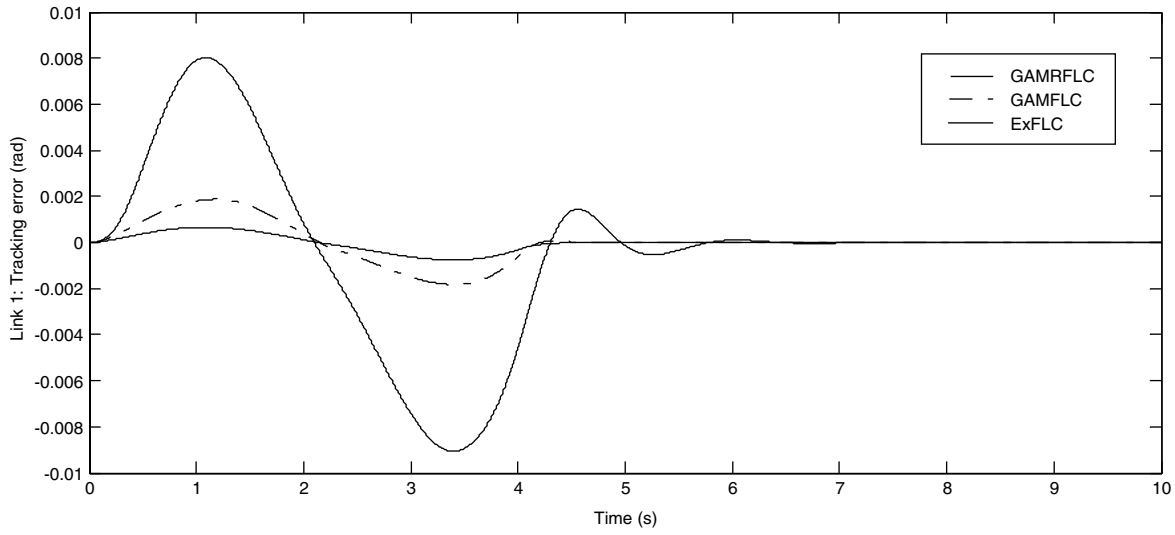


Fig. 7. Link 1: Tracking error.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

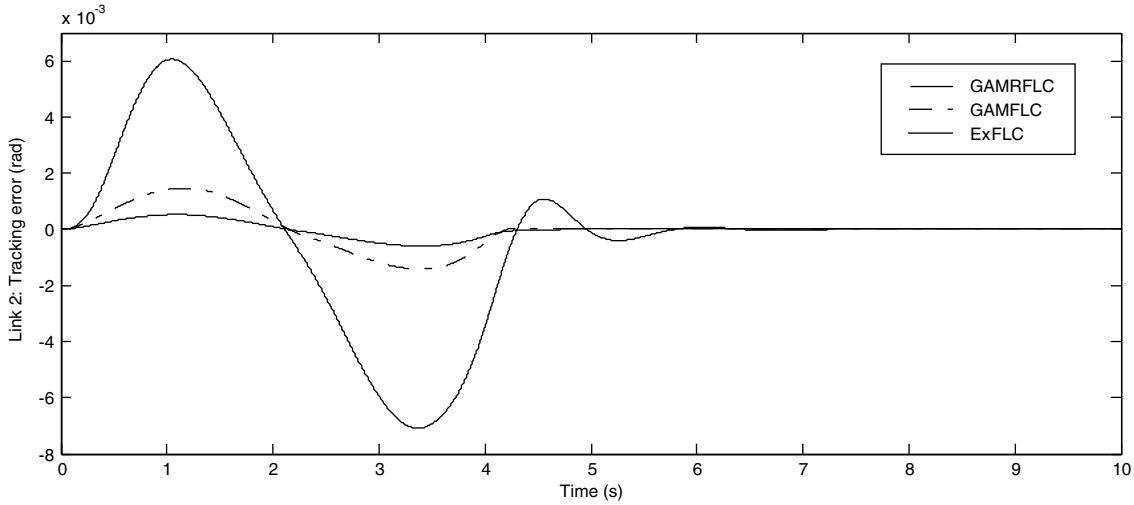


Fig. 8. Link 2: Tracking error.

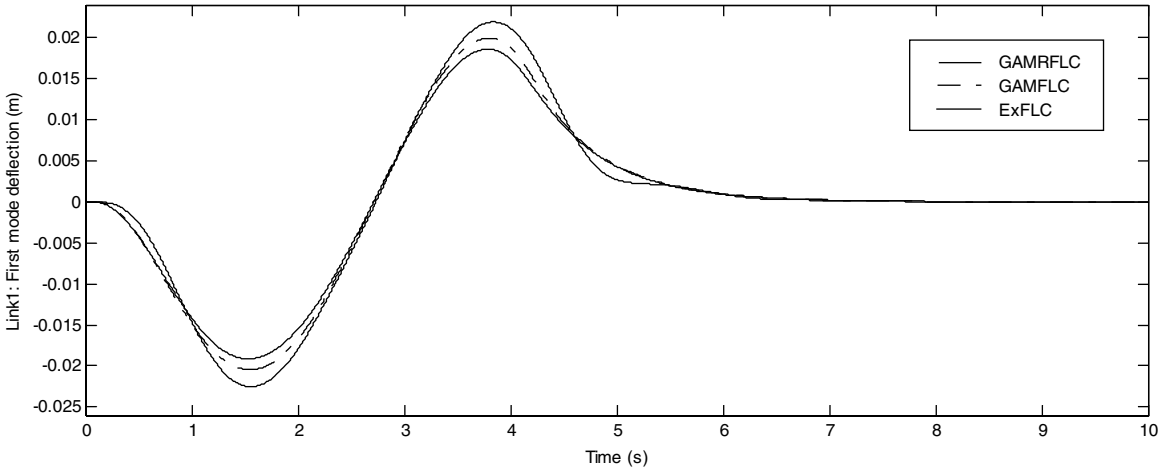


Fig. 9. Link 1: First mode deflection.

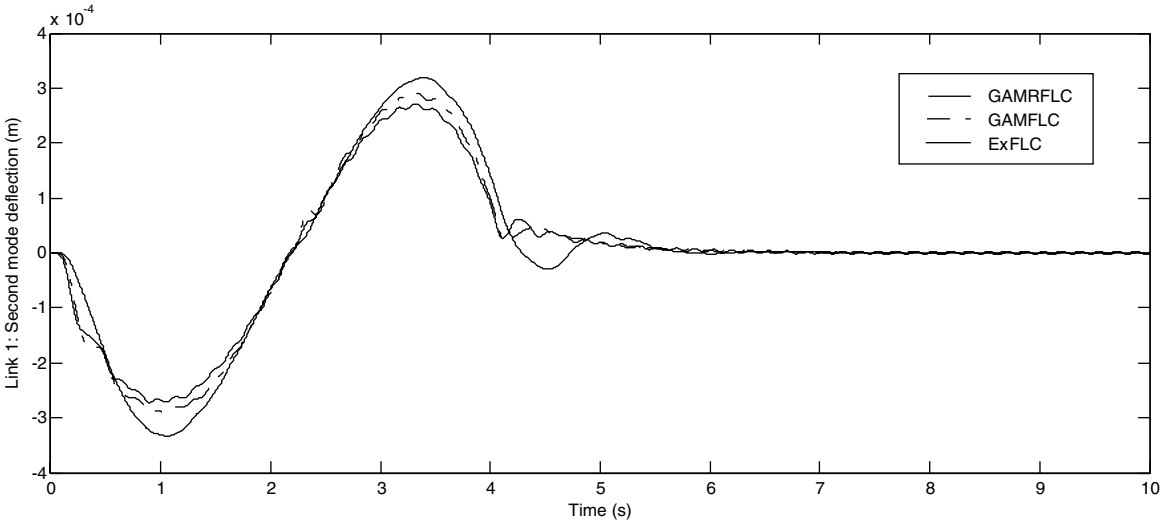


Fig. 10. Link 1: Second mode deflection.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

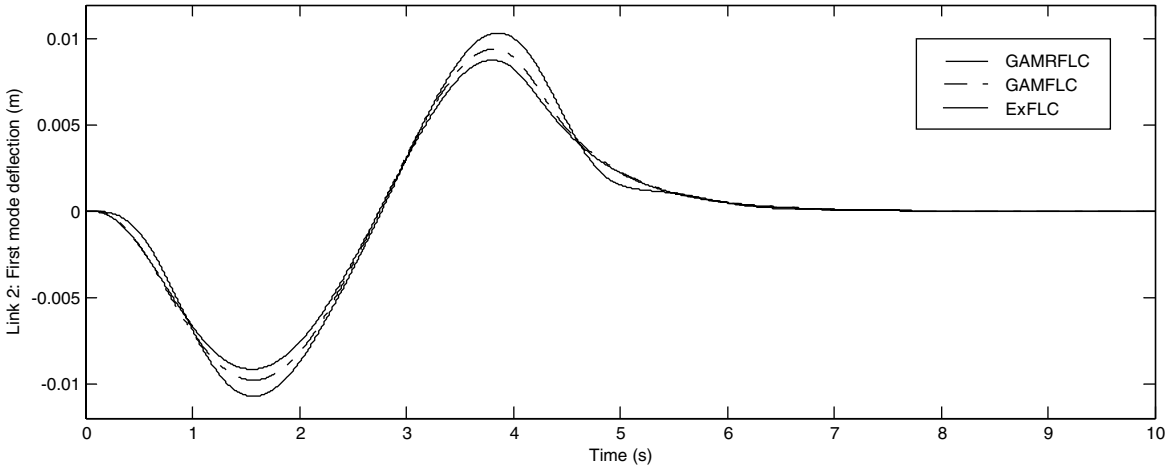


Fig. 11. Link 2: First mode deflection.

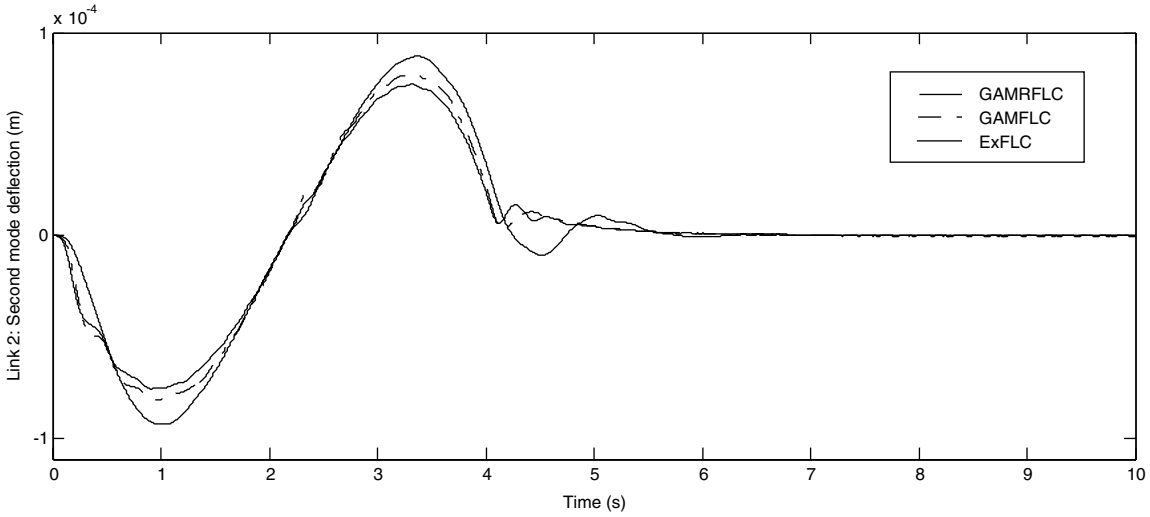


Fig. 12. Link 2: Second mode deflection.

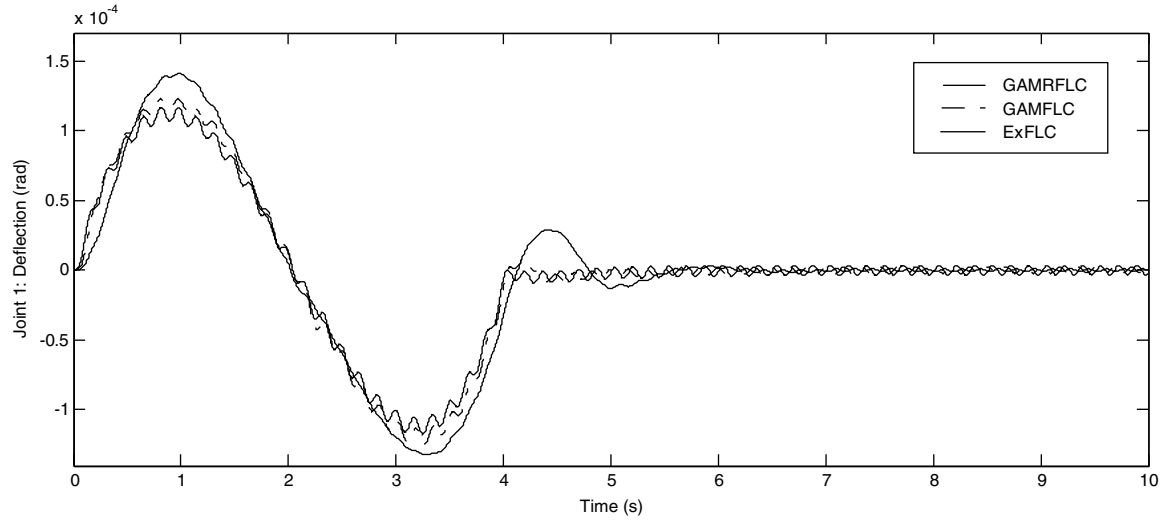


Fig. 13. Joint 1: Deflection.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

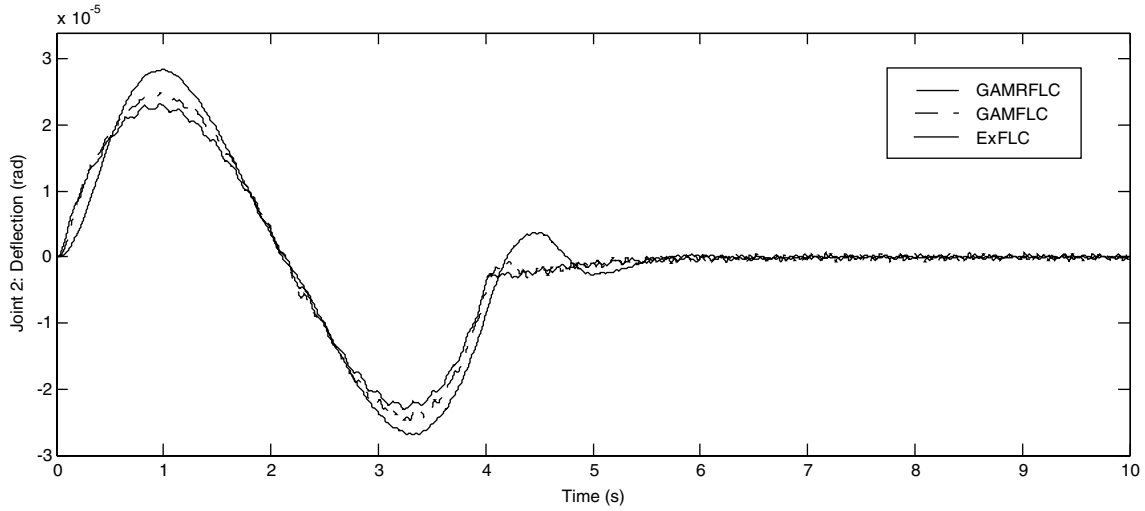


Fig. 14. Joint 2: Deflection.

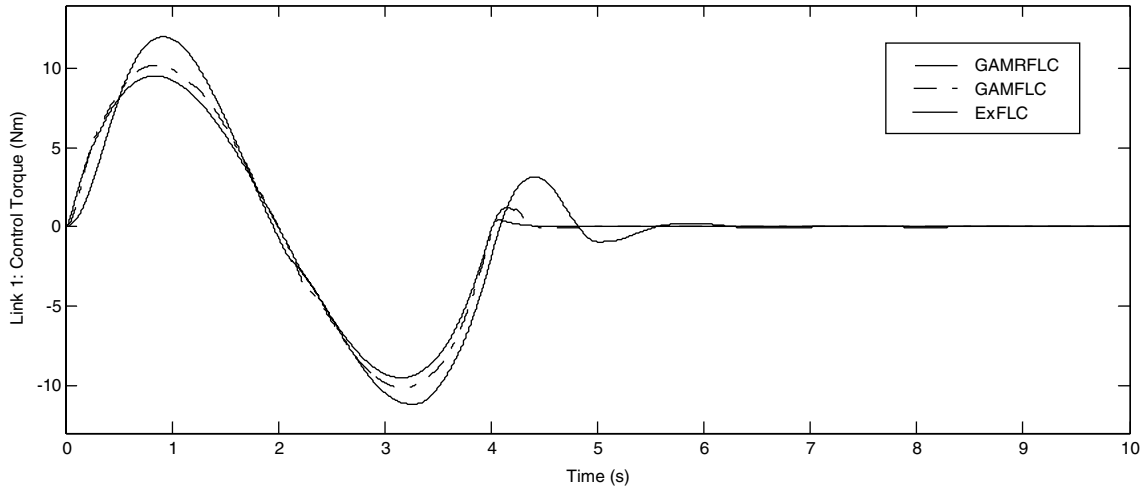


Fig. 15. Link 1: Control torque.

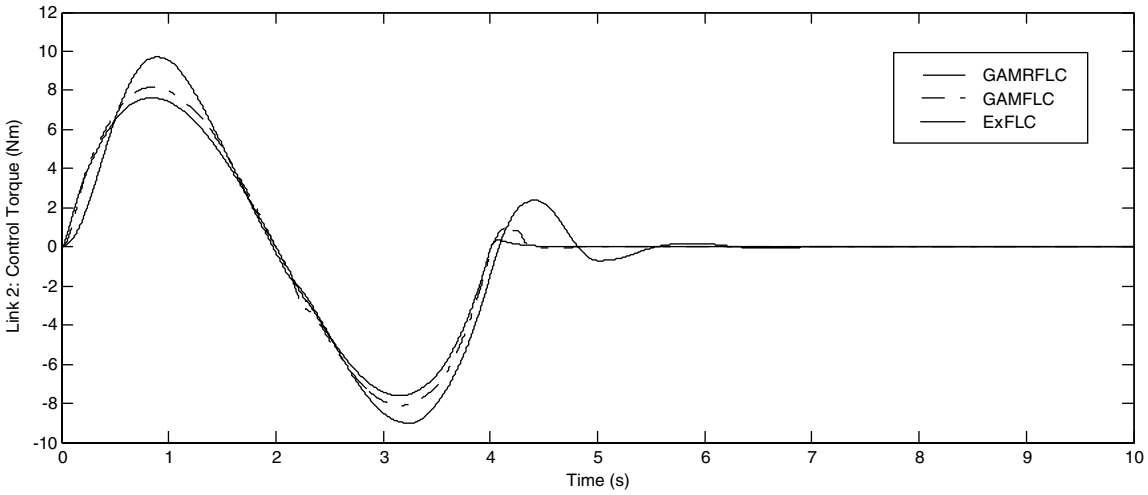


Fig. 16. Link 2: Control torque.

order to optimise the core of the fuzzy inference system associated with modifying the membership functions. The results that are presented and discussed in the last section show the effectiveness and practicality of the optimised fuzzy logic controller schemes compared with human-designed controllers, with particularly good results being given by the second scheme in which the rule set is optimised.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press

References

1. S. S. K. Tadikonda, T. G. Mordfin and T. G. Hu, "Assumed modes method and articulated flexible multibody dynamics," *J. Guidance, Control, Dyn.* **18**(3), 404–410 (1995).
2. F. Karray, S. Tafazolli and W. Guealed, "Robust tracking of a lightweight manipulator system," *Nonlinear Dyn.* **20**(2), 169–179 (1999).
3. J. Martins, M. A. Botto and J. S. D. Costa, "Modeling of flexible beams for robotic manipulators," *Multibody Syst. Dyn.* **7**(1), 79–100 (2002).
4. G. Naganathan and A. H. Soni, "Coupling effects of kinematics and flexibility in manipulators," *Int. J. Rob. Res.* **6**, 75–84 (1987).
5. S. S. Ge, T. H. Lee and G. Zhu, "A nonlinear feedback controller for a single-link flexible manipulator based on a finite element model," *J. Rob. Syst.* **14**(3), 165–178 (1997).
6. R. Theodore and A. Ghosal, "Comparison of the assumed modes and finite element models for flexible multi-link manipulators," *Int. J. Rob. Res.* **14**(2), 91–111 (1995).
7. B. Subudhi and A. S. Morris, "Singular perturbation approach to trajectory tracking of flexible robot with joint elasticity," *Int. J. Syst. Sci.* **34**, 167–179 (2003).
8. V. G. Moudgal, W. A. Kwong, K. M. Passino and S. Yurkovich, "Fuzzy learning control for a flexible-link robot," *IEEE Trans. Fuzzy Syst.* **3**, 199–210 (1995).
9. P. Sooraksa and G. Chen, "Mathematical modeling and fuzzy control of a flexible-link robot arm," *Math. Comput. Model.* **27**, 73–93 (1998).
10. J. Lin, "Hierarchical fuzzy logic controller for a flexible link robot arm performing constrained motion tasks," *IEE Proc. Control Theory Appl.* **150**, 355–364 (2003).
11. A. Green and J. Z. Sasiadek, "Adaptive control of a flexible robot using fuzzy logic," *J. Guidance, Control, Dyn.* **28**, 36–42 (2005).
12. A. Yeşildirek, M. W. Vandegrift and F. L. Lewis, "A neural network controller for flexible-link robots," *J. Intell. Rob. Syst.* **17**, 327–349 (1996).
13. H. A. Talebi, R. V. Patel and H. Asmer, "Neural network based dynamic modeling of flexible-link manipulators with application to the SSRMS," *J. Rob. Syst.* **17**, 385–401 (2000).
14. Y. Li, G. Liu, T. Hong and K. Liu, "Robust control of a two-link flexible manipulator with quasi-static deflection compensation using neural networks," *J. Intell. Rob. Syst.* **44**, 263–276 (2005).
15. J. H. Holland, *Adaptation in Natural and Artificial Systems* (Ann Arbor, MI: University of Michigan Press, 1975).
16. J. J. Buckley, "Fuzzy genetic algorithm and applications," *Fuzzy Sets Syst.* **61**, 129–136 (1994).
17. A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzycontrollers using genetic algorithms," *IEEE Trans. Fuzzy Syst.* **3**, 129–139 (1995).
18. H. B. Gürocak, "A genetic-algorithm-based method for tuning fuzzy logic controllers," *Fuzzy Sets Syst.* **108**, 39–47 (1999).
19. A. Homaifar, M. Bikdash and V. Gopalan, "Design using genetic algorithms of hierarchical hybrid fuzzy-PID controllers of two-link robotic arms," *J. Rob. Syst.* **14**, 449–463 (1997).
20. L. Meirovitch, *Analytical Methods in Vibrations* (New York: Macmillan, 1967).
21. A. S. Morris and A. Madani, "Quadratic optimal control of a two-link robot manipulator," *Robotica* **16**, 97–108 (1998).
22. K. S. Fu, R. C. Gonzalez and C. S. G. Lee, *Robotics: Control, Sensing, Vision and Intelligence* (New York: McGraw-Hill, 1987).
23. M. A. Lee and H. Takagi, "Integrating design stage of fuzzy systems using genetic algorithms," *Second IEEE International Conference on Fuzzy Systems*, San Francisco, CA, USA (1993) pp. 612–617.
24. C. J. Harris, M. Brown and C. G. Moore, *Intelligent Control: Aspects of Fuzzy Logic and Neural Nets* (Singapore: World Scientific, 1993).
25. M. Dadfarnia and N. Jalili, "Lyapunov-based vibration control of translational euler-bernoulli beams using the stabilizing effect of beam damping mechanisms," *J. Vibration Control*, **10**, 933–961 (2004).
26. Y. Aoustin and C. Chevallereau, "The singular perturbation control of a two-flexible-link robot," *Proceedings of IEEE International Conference on Robotics and Automation*, Atlanta, GA, USA (1993) pp. 737–742.

https://doi.org/10.1017/S0263574708005055 Published online by Cambridge University Press