Parametric Modelling of a TRMS Using Dynamic Spread Factor Particle Swarm Optimisation

S. F. Toha[1], I. Abd Latiff, M. Mohamad and M. O. Tokhi
Department of Automatic Control and Systems Engineering
The University of Sheffield
United Kingdom
cop06sft@sheffield.ac.uk[1]

Abstract

System identification in vibrating environments has been a matter of concern for researchers in many disciplines of science and engineering. In this paper, a sound approach for a Twin Rotor Multi-input Multi-Output System (TRMS) parametric modeling is proposed based on dynamic spread factor particle swarm optimization. Particle swarm optimization (PSO) is demonstrated as an efficient global search method for nonlinear complex systems without any a priory knowledge of the system structure. The proposed method formulates a modified inertia weight algorithm by using a dynamic spread factor (SF). The inertia weight plays an important role in terms of balancing both the global and local search. Thus, the usage of dynamic SF is proved experimentally to satisfy main issues of using basic PSO that are trapped in local optima and preservation of diversity. Results in both time and frequency domains portray a very good parametric model that mimic well the behavior of a TRMS. Validation tests clearly show the effectiveness of the algorithm considered in this work.

1. Introduction

A variety of evolutionary algorithms (EAs), operating according to Darwinian concepts have been proposed to solve problems of common engineering applications. Applications often involve automatic learning of nonlinear mapping that govern the behaviour of control systems, as well as parallel search strategies for solving optimization problems. EAs are stochastic search methods that mimic the natural biological evolution and/or the social behavior of species. Such algorithms have been developed to arrive at near optimum solutions to large scale-optimization problems, for which traditional mathematical techniques may fail [1]. Particle swarm optimization (PSO) is one of the relatively new evolutionary computation techniques [2], which has attracted a lot of attention from researches worldwide [3-5]. PSO is a population based algorithm which ensures the convergence of model parameters to the global optimum. It is initialized with a population of random solutions called particles. PSO, unlike other evolutionary computation techniques, offers faster convergence during training and involves low computational complexity.

PSO is therefore embraced in this work in search of a parametric model that can replicates well the behavior of a twin rotor MIMO system. System identification is the route to build a mathematical archetype for the anonymous system by monitoring its empirical input output data. This is accomplished by duly altering the parameters within the model prediction such that for a particular input, the predicted output meets with corresponding actual system output. In a model-based control framework, a pre-requisite to developing an effective control mechanism for a system is to model and predict the behaviour of the system based on given input-output data [6]. Once the model of the system is identified, the output can be envisaged for a given input to the system which is the goal for the system identification problem. A high-fidelity system model is an important first step in control system design and analysis. A number of techniques have been devised by researchers to determine models that best describe input-output behaviour of a system. In many cases when it is difficult to obtain a model structure for a system with traditional system identification techniques, intelligent techniques are desired that can describe the system in the best possible way [7].

In recent years, PSO has been used to solve nonlinear identification and optimization problems in the field of aircraft technology [8-9]. Unlike conventional fixed-wing aircraft, helicopter portrays

978-0-7695-3593-7/09 $25.00 © 2009 IEEE
DOI 10.1109/UKSIM.2009.109

distinct advantages on surveillance and inspection tasks as they can take off and land vertically in limited spaces and easily hover in places above a target. However, helicopters are much more complex in terms of system dynamics and control because the inputs are not directly applied torques of forces, but rather aerodynamic torques or forces created by the main and tail rotors albeit all the aforementioned advantages. A scaled and simplified version of practical helicopter namely twin rotor multi-input multi-output system (TRMS) is used in this work. Although the dynamicsof the TRMS are simpler than those of a real helicopter, they retain the most important helicopter features such as couplings and strong nonlinearities. It can be perceived as an unconventional and complex "air vehicle". Therefore, it is crucial to deduce a good parametric model of the TRMS as attempted in this work.

2. Twin Rotor MIMO System

The TRMS is a laboratory set-up developed by Feedback Instruments Limited [10] for control experiments. Its behaviour in certain aspects resembles that of a helicopter. For example, it possesses a strong cross-coupling between the collective (main rotor) and the tail rotor, like a helicopter. The TRMS used in this work is shown in Figure 1. It is driven by two DC motors. Its two propellers are perpendicular to each other and joined by a beam pivoted on its base that can rotate freely in the horizontal and vertical planes. The beam can thus be moved by changing the input voltage in order to control the rotational speed of the propellers. The system is equipped with a pendulum counterweight hanging from the beam, which is used for balancing the angular momentum in steady-state or with load.

The system is balanced in such a way that when the motors are switched off, the main rotor end of the beam is lowered. The controls of the system are the supply voltages of the motors. It is important to note that the geometrical shapes of the propellers are not symmetric. Accordingly, the system behaviour in one direction is different from that in the other direction. Rotation of a propeller produces an angular momentum which, according to the law of conservation of angular momentum, is compensated by the remaining body of the TRMS beam. This results in interaction between the moment of inertia of the motors with propellers. This interaction directly influences the velocities of the beam in both planes. The measured signals are: position of the beam, which constitute two position angles, and the angular velocities of the rotors. Angular velocities of the beam are software reconstructed by differentiating and filtering the measured position angles of the beam.

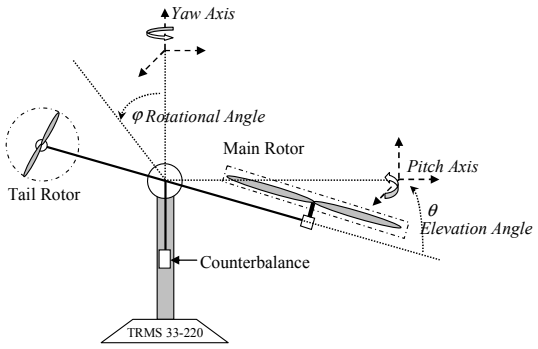

Figure 1: Twin rotor MIMO system

3. PSO Algorithm

Particle swarm optimisation [2] is a population-based evolutionary optimization method, inspired by the collective behaviours of birds and flocks. Through cooperation and competition among the population, population-based optimization approaches often can find very good solutions effectively and efficiently. The PSO algorithm is similar to evolutionary computation in producing a random population initially and generating the next population based on current cost, but it does not need reproduction or mutation to produce the next generation. Thus, PSO is faster in finding solutions compared to other evolutionary computation technique.

3.1. Basic PSO algorithm

In the PSO algorithm, each individual is named as a particle which in fact represents a potential solution to a problem. Each particle is moving, and hence has a velocity. Also, each particle remembers the position it was in and where it had its best result so far. Moreover, the particles in the swarm co-operate and exchange information about what they have discovered within the search region they have visited. A fitness function is the metric to determining the solution's optimality. The fitness of the particles is evaluated in each iteration. The position of the particle with the best fitness, called the global best, is preserved in memory. Each particle also preserves in memory the best position and its best fitness, called the particle best. These best fitness positions influence how the solutions change or particles move in the search space.

When a dimension *d* of the particle's position is updated to a new value in the iteration, the velocity, *v* of the particle decides the new position, *x* at each dimension, *d*. The dimension's position update equation is given by:

$$x_{id_{new}} = x_{id} + v_{id_{new}} \quad (1)$$

When the velocity is updated in the iteration, the velocity, v of the dimension, d of the particles decides where the particle position will move next. It is influenced by the particle best 'pbest' and the global best 'gbest'. The c_1 and c_2 are constant known as acceleration coefficients, d $rand_1$ and $rand_2$ are two separated generated uniformly distributed random numbers in the range of [0,1].

$$v_{id\,new} = v_{id} + c_1 rand_1(p_{id} - x_{id}) + c_2 rand_2(p_{gd} - x_{id}) \quad (2)$$

Equation (2) is used to calculate the particle's new velocity according to its previous velocity and the distances of its current position from its own best experience (position) and the group's best experience. Then the particle flies toward a new position according to equation (1) [11]. The swarm learns the characteristics of the fitness landscape and captures the essence in the particle and global best. This is accomplished through maintaining a local best solution for each particle and the global best solution is maintained after the particles share their own local best before convergence to a solution.

3.2. Dynamic spreading factor algorithm

An inertia weight, w was first brought into PSO equation in 1998 [11-13]. This plays a vital role in balancing both global and local searches. It can be a positive constant or even a positive linear or nonlinear function of time in the velocity equation as:

$$v_{id} = (w * v_{id}) + c_1 rand_1(p_{id} - x_{id}) + c_2 rand_2(p_{gd} - x_{id}) \quad (3)$$

The first part of the equation represents the previous velocity, which provides the necessary momentum for particles to roam across the search space. The second part, known as *cognitive* component, represents the personal thinking of each particle. The cognitive component encourages the particles to move towards their own best position so far. The third part is known as *social* component, which represents the collaborative effect of the particles, in finding the global optimal solution. The social component always pulls the particles towards the global best particle found so far [16].

A new proposed methods will be discussed later to compare with PSO with time-varying inertia weight factor, *w* [11, 12] combined with time-varying acceleration coefficients c_1 and c_2 [14]. The proposed method was implemented in search for parametric model of the TRMS as well as in order to control the global search and the global best solution.

Dynamic spread factor PSO (SFPSO) is proposed in this paper, which introduced a dynamic spread factor in the PSO algorithm [15]. The proposed algorithm is found highly effective in improving major issues in basic PSO that are premature convergence and preservation of diversity. As originally developed, *w* is decreased linearly from 0.9 to 0.4 during a run. Suitable selection of the inertia weight provides a balance between global and local exploration and exploitation and results in less iteration on average to find a sufficiently optimal solution [13]. It is first established that varying the parameters produces superior results compared to the fixed one. Then, the spreading factor is employed to continuously modify the value of inertia weight. In order for particles to keep exploring the search space, it is imperative that they must know their whereabouts and relative distance from each other. The spreading factor measures the distribution of particles in the search space as well as the precision and accuracy of the particles with respect to global optimum.

Precision refers to the maximum distance between particles in the best and worst positions with respect to function fitness. It also describes the spread or distribution of particles in the search space. Accuracy on the other hand refers to the distance of average particle position from the global best particle. This value also refers to the deviation of particles from the global best position. These two important data describe the instantaneous spreading of the particles and are used to calculate the spreading factor (SF) as shown below:

$$SF = 0.5(spread + deviation) \quad (4)$$

The value of SF varies from the maximum range of the search space down to the desired convergence precision. This factor then modifies the inertia weight according to the equation (5)

$$w = \exp(-iter / (SF \times \max\_iteration)) \quad (5)$$

The effect of the spreading factor on the inertia weight can be deduced from Equation (5). As long as the spreading factor remains high, the inertia weight

will be able to maintain its value within the range of 0.95 to 1. The idea is for the particles to keep searching and exploring within the boundary of the search space until convergence to the global optimum is almost in sight. It is likely when all the particles move within the vicinity of global optimum, both the dynamic SF and hence the inertia weight will drop in value drastically. This will not only force all the particles to converge, but also allow the algorithm to achieve extremely high precision.

The dynamic SF alone however, is not sufficient to guarantee convergence. When global optimum has been found towards approaching the end of iterations, it is essential to other particles to forget their own personal best. This is crucial in situations where local optima exist. For this case, particles will be pulled towards both global and local optima simultaneously which is undesirable. Thus the parameter c_1 is reduced to zero from its initial value of 2 as:

$$c_1 = 2 \times (1 - iter / \max\_iteration) \quad (6)$$

where at the same time the value of c_2 is maintained at 2 to preserve the van den Bergh's condition for convergence [16].

4. Parametric Modelling of TRMS

The TRMS set-up is very sensitive to the atmospheric disturbances; hence it was ensured that the identification experiments are conducted in calm air. The body resonance modes of the TRMS lie in a low frequency range of 0 to 3 Hz, while the main rotor dynamics are at significantly higher frequencies [17]. The excitation signal represents voltage input to the main rotor and the output signal represents the elevation angle (pitch angle) in radians. During experimentation, the yaw plane is physically locked, allowing only vertical plane motion. To investigate variations in the resonance modes, the system was excited with a pseudorandom binary sequence (PRBS) of different bandwidths (2 to 20 Hz). A PRBS of 5Hz bandwidth and 100s duration was finally chosen for this analysis. The PRBS magnitude was selected so that it does not drive the TRMS out of its linear operating range. Good excitation was achieved from 0 to 5 Hz, which includes all the important rigid body and flexible modes of the system.

Auto-regressive moving average (ARMA) structures are chosen for the parametric model and expressed as [18]

$$\hat{y}(k) = \sum_{i=1}^{N} a_i \times y(k-i) + \sum_{j=0}^{M} b_j \times u(k-j) + \eta(k) \quad (7)$$

where a_i, b_j are denominator and numerator polynomial coefficients, N and M are number of coefficients in the denominator and numerator polynomials, $y, u, \hat{y}$ and η are measured output, input, predicted output and noise, respectively. The order of the transfer function depends on N. Taking the values of N and M as 4 and 3 and neglecting the noise term η. Equation (6) can be simplified as

$$\begin{aligned} \hat{y}(k) = a_1 y(k-1) - \cdots - a_4 y(k-4) \\ + b_o u(k-1) + \cdots + b_3 u(k-4) \end{aligned} \quad (8)$$

In matrix form, the above equation can be written as

$$\begin{aligned} \hat{y}(k) = -[a_1, a_2, a_3, a_4][y(k-1), y(k-2), y(k-3), \\ y(k-4)]^T + [b_0, b_1, b_2, b_3][u(k-1), u(k-2), \\ u(k-3), u(k-4)]^T \end{aligned} \quad (9)$$

The first four variables are assigned to $b_0, \ldots, b_3$ and the next four to $a_1, \ldots, a_4$ as indicated in equation (7). The difference between the predicted and actual output is recorded as error, $e(k) = y(k) - \hat{y}(k)$, which in turn is used to form the objective function of the optimization process. Mean squared error (MSE) is used in this work and given as

$$f(x) = \frac{1}{s} \sum_{k=1}^{s} (y(k) - \hat{y}(k))^2 \quad (10)$$

where s=1000. The basic flowchart of dynamic SF algorithm can be described as in Figure 2.

5. Results and Discussion

Results of modelling the TRMS in vertical plane motion with parametric modelling using dynamic SFPSO are presented in this section. Model validation is carried out in time and frequency domains through comparative assessment of the system and model responses and correlation tests.

The dynamic SFPSO was compared with global versions of PSO algorithms [11, 12, and 14] and were tested on modelling the TRMS. It was noted that the dynamic SFPSO gave better performance in terms of achieving an MSE of 0.000018325 than the local PSO. Figure 3 shows that local PSO stuck in the local minima while dynamic SFPSO gave better MSE value according to the objective function assigned.

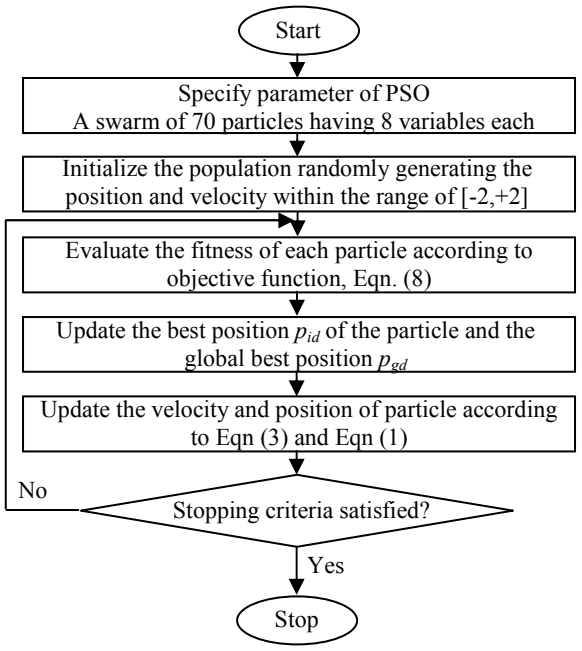

Figure 2: The flow chart of Dynamic SFPSO algorithm

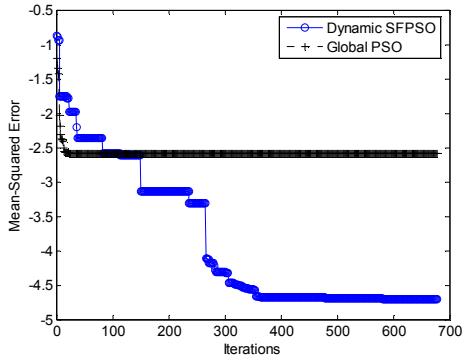

Figure 3: Convergence of SFPSO and local PSO

The eight assigned variables were obtained at the end of maximum iteration of the PSO process. Using these values in equation (6), the transfer function was formed. The discrete transfer function for hovering motion of the TRMS at the sampling time of 0.2s which correspond to 5Hz, thus formed is given as

$$H(z)=\frac{-0.7428z^3-0.5711z^2+0..3548z+0.787}{z^4-0.00492z^3-0.001755z^2-0.001462z+0.001}$$

(11)

This discrete transfer function can be converted to equivalent continuous form s-domain using MATLAB function as

$$H(s)=\frac{-5772s^3+5.369\times10^5 s^2+4.806\times10^7 s-3.315\times10^8}{s^4+391.1s^3+8.193\times10^4 s^2+8.464\times10^6 s+3.76\times10^8}$$

(12)

Figure 4 shows that the predicted output follows the actual output very well in time domain. The pole-zero diagram (Figure 5) shows that all the poles lie inside the unit circle whereas some zeros are outside. This indicates that the model is stable and non-minimum phase. The frequency domain plot (Figure 6) of the predicted and actual outputs indicates that the model has successfully captured the system dynamics surrounding the main resonance mode which is at 0.3497 Hz. The model reached an MSE level of 0.000018325 (Figure 7). Correlation validation of vertical plane motion model is shown in Figure 8. It is noted that all the five correlation functions are within the 95% confidence bands indicating that the model behaviour is unbiased and close to that of the real system.

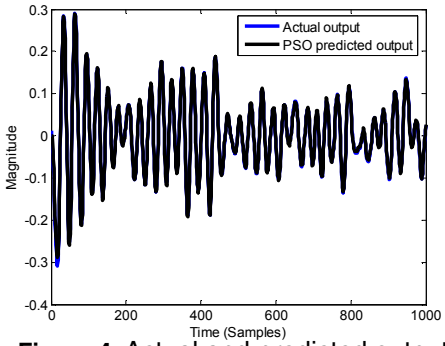

Figure 4: Actual and predicted output

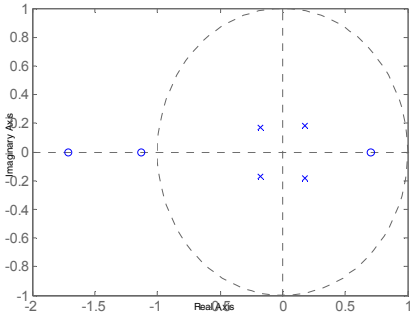

Figure 5: Pole-zero diagram

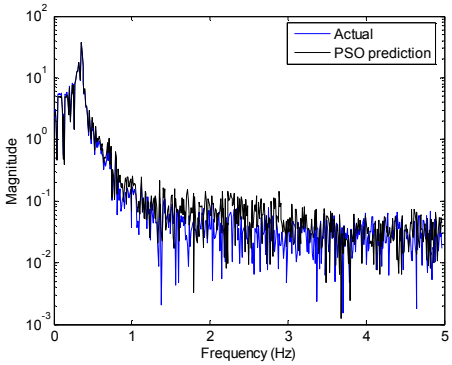

Figure 6: Power spectral density of actual and dynamic SFPSO predicted output

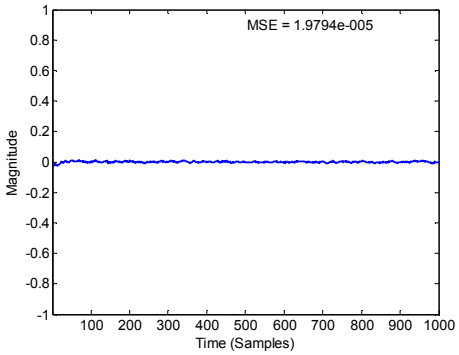

Figure 7: Error between actual and dynamic SFPSO predicted output

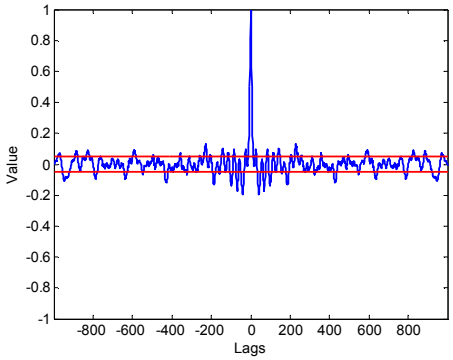

Figure 8: Correlation validation test

6. Conclusion

PSO has successfully been used to derive model of vertical plane hovering motion of the twin rotor system. For the vertical channel, the linear model (transfer function) is a good approximation of the system behaviour in the vicinity of the operating point, mainly to capture the dominant modes of the system. From this modeling result, it is evident that the algorithm with same parameters can extract stable and satisfactory model. The results presented in this paper have demonstrated the potential of using PSO in obtaining accurate dynamic characterization of flexible maneuvering systems.

7. References

[1] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," Advanced Engineering Informatics, vol. 19, 2005, pp.43-53.

[2] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," In Proceedings of the IEEE International Conference on Neural Networks. Perth, Australia, 1995, vol. 4, pp. 1942-1945.

[3] M. S. Alam, and M. O. Tokhi, "Modelling of a twin rotor system: a particle swarm optimisation approach," Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, vol. 221, 2007, pp. 353-375.

[4] G. Panda, D. Mohanty, B. Majhi, and G. Sahoo, "Identification of nonlinear systems using particle swarm optimization technique," In Proceedings of IEEE Congress on Evolutionary Computation (CEC 2007), Singapore, 25-28 September 2007, pp. 3253-3257.

[5] H.-W. Ge, F. Qian, Y.-C. Liang, W.-L. Du, and L. Wang, "Identification and control of nonlinear systems by a dissimilation particle swarm optimization-based elman neural network," Journal of Nonlinear Analysis: Real Worls Applications, vol. 9, 2008, pp. 1345-1360.

[6] K.J. Åström, and P. Eykhoff, "System identification, a survey," Automatica, vol. 7, 1971, pp. 123-162.

[7] S. V. T. Elanayar, and C. S. Yung, "Radial basis function neural networks for approximation and estimation of nonlinear stochastic dynamic systems," The IEEE Transaction on Neural Networks, vol. 5, issue 4, 1994, pp 594-603.

[8] J.-H. Lee, B.-M. Min, and E.-T. Kim, "Autopilot design of tilt-rotor UAV using particle swarm optimization method," In Proceedings of International Conference on Control, Autmation and Systems, Seoul, Korea, 2007, pp. 1629-1633.

[9] F. Sahin, M. C. Yavuz, Z. Arnavut, and Ö. Uluyol, "Fault diagnosis for airplane engines using Bayesian networks and distributed particle swarm optimization," Journal of Parallel Computing, vol. 33, 2007, pp. 124-143.

[10] Feedback Instruments Ltd., *Twin Rotor MIMO System Manual 33-007-0*, Sussex, UK, 1996

[11] Y. Shi, and R. C. Eberhart, "Parameter selection in particle swarm optimization," In Evolutionary Programming VII: Proceedings EP98, New York: Springer-Verlag, 1998a, pp. 591-600.

[12] Y. Shi, and R. C. Eberhart, "A modified particle swarm optimizer," Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, NJ: IEEE Press, 1998b, pp. 69-73.

[13] R. C. Eberhart, and Y. Shi, "Computational intelligence concepts to implementations," Morgan Kauffmann Publisher, 2007.

[14] A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time varying acceleration coefficients," IEEE Transactions on Evolutionary Computation, 2004, vol. 8, issue 3, pp. 240-255.

[15] I. Abd Latiff, and M. O. Tokhi, "Fast convergence strategy for particle swarm optimization using spreading factor," Proceedings of the IEEE Congress on Evolutionary Computation, Trondheim, Norway, 18-21 May 2009, Unpublished.

[16] F. van den Bergh and A. P. Engelbrecht, "Effect of swarm size on cooperative particle swarm optimizer," Proceedings of The Genetic Evolutionary Computation Conference (GECCO 2001), san Francisco, CA, July 2001, pp. 892-899.

[17] S. M. Ahmad, A. J. Chipperfield, and M. O. Tokhi, "Dynamic modelling and linear quadratic Gaussian control of a twin rotor MIMO system," Journal of Systems and Control Engineering, vol. 1, issue 217, 2003, pp 203-227.

[18] L. Ljung, "System identification: theory for the user," Prentice Hall, Englewood Cliffs, NJ, Second Edition, 1999.