ON-LINE DESIGN OF ROBUST FUZZY-LOGIC CONTROL SYSTEMS BY MULTI-OBJECTIVE EVOLUTIONARY METHODS.

P. Stewart (corresponding author) * **D.A. Stone** *
P.J. Fleming **

* *Electrical Machines and Drives Group, Department of Electronic and Electrical Engineering, University of Sheffield. Mappin St. Sheffield S1 3JD U.K.*
e-mail:p.stewart@shef.ac.uk, tel: +44 (0)114 2225841.
** *Department of Automatic Control and Systems Engineering, University of Sheffield. Mappin St. Sheffield U.K.*

Abstract: Evolutionary development of a fuzzy-logic controller is described and is evaluated in the context of hardware in the loop. It had been found previously that a robust speed controller could be designed for a dc motor motion control platform via off-line fuzzy logic controller design. However to achieve the desired performance, the controller required manual tuning on-line. This paper investigates the automatic design of a fuzzy logic controller on-line. An optimiser which modifies the fuzzy membership functions, rule base and defuzzification algorithms is considered. A multi-objective evolutionary algorithm is applied to the task of controller development, while an objective function ranks the system response to find the Pareto-optimal set of controllers. Disturbances are introduced during each evaluation at run-time in order to produce robust performance. The performance of the controller is compared experimentally with the fuzzy logic controller which has been designed off-line. The on-line optimised fuzzy controller is shown to be both robust, possessing excellent steady-state and dynamic characteristics, demonstrating the performance possibilities of this type of approach to controller design.

Keywords: Fuzzy systems, Evolutionary/Genetic algorithms, Methodologies, Models and algorithms.

1. INTRODUCTION

This paper is investigates the potential of multiobjective control design with hardware in the loop. Tuning of PI parameters on-line has been achieved [17] with multiobjective genetic algorithms. Here, the potential of parameter and controller structure tuning on-line is considered. A DC motor dynamometer rig and a microcontroller is used as a platform to develop and assess the control algorithms. In particular, an off-line designed type (fuzzy logic) is considered for performance comparison. An automatic method for fuzzy logic control design is considered, utilising a multiobjective evolutionary algorithm for the optimisation process. Random disturbances with bounds which reflect realistic parameter variations are injected during each on-line assessment with the aim of producing a controller which is also robust to disturbances.

Fuzzy logic control, comprising a fuzzification interface, rule base and defuzzification algorithm [1,2], has been applied to a wide variety of motion control applications [3,4]. A vital region of interest concerns the implementation of the fuzzy

Proceedings 17th European Simulation Multiconference
(c) SCS Europe BVBA, 2003   ISBN 3-936150-25-7

controller. Several different approaches have been postulated to extract the knowledge base from experts or training examples to construct the input-output membership functions and the fuzzy rule-base. These methods can be based on neural networks [5,6] or the application of fuzzy clustering techniques to construct a fuzzy controller from training data sets [7]. It has been observed that the major drawback of most fuzzy controllers and expert systems is the need to predefine membership functions and fuzzy rules. In [5], a method is proposed based on fuzzy clustering techniques and decision tables to derive membership functions and fuzzy rules from numerical data. A natural evolution of the technique was to integrate Genetic Algorithms (GAs) into the Fuzzy logic design process [8,9,10]. The robustness of the GA allows it to cover a multidimensional search space while ensuring an optimal or near-optimal solution, thus simultaneous design of membership functions and fuzzy control rules can be achieved [11]. The development of these techniques to design optimal fuzzy logic controllers has arisen to satisfy the need which exists when expert heuristic knowledge doesn't exist to translate into controller design.

The performance of a particular control design is fundamentally tied to the accuracy of the model upon which it is based. This is especially true for iterative control design and optimisation procedures. The substitution of hardware in the loop for the software model opens up new possibilities for design based on real world perfomance indicies. In this paper the implementation of GA fuzzy design will be evaluated via an on-line experimental DC motor connected to a DC shunt load motor set to introduce dynamic disturbances. The performance of the resulting motion controller is compared with that of a manually tuned fuzzy controller. The results presented here demonstrate a convenient and practical method to produce a robust controller design on a prototype plant.

1.1 *Multiobjective optimisation by evolutionary algorithm*

Evolutionary algorithms are global parallel search and optimisation methods based around Darwinian principles, working on a population of potential solutions to a problem (in this case the on-line design of an optimal fuzzy logic controller via hardware in the loop). Every individual in the population represents a particular solution to the problem, often expressed in binary code. The population is evolved over a series of generations to produce better solutions to the problem. At every generational step, each individual of the population is run on the hardware, and its performance evaluated and ranked via a cost function. Individual performance is indicated by a fitness value, an expression of the solution's suitability in the solution of the problem. The relative degree of the fitness value determines the level of propagation of the individual's genes to the next generation. Evolution is subsequently performed by a set of genetic operators which stochastically manipulate the genetic code. Most genetic algorithms include operators which select individuals for mating, and produce a new generation of individuals. *Crossover* and *Mutation* are two well-used operators. The crossover operator exchanges genetic material between parental chromosomes to produce offspring with new genetic code. The mutation operator makes small random changes to a chromosome. Further repetitions of this process are made in the search for the strongest genetic material. The genetic algorithm explores the multidimensional search-space to find good solutions to the problem. It is possible for the GA to find several dissimilar but equally valid solutions to a single problem due to its use of population, and the competing nature of multiple objectives, since real-world problems involve the simultaneous evaluation of multiple performance criteria. Trade-offs occur between competing objectives with the consequence that it is very rare to find a single solution to a particular problem. In reality a family of *non-dominated* solutions will exist. These *Pareto-optimal* [12,13] solutions are those for which no other solution can be found which improves on a particular objective without a detrimental effect on one or more competing objectives. The designer then has the opportunity to select an appropriate compromise solution from the trade-off family based on a subjective engineering knowledge of the required performance. For example, in this application, it would be expected that a tradeoff will exist between energy consumption and tracking performance. In this case, the designer may be willing to sacrifice a little energy efficiency to achieve a certain tracking metric. Individuals which represent candidate solutions to the optimisation problem (in this case fuzzy controller parameters such as membership functions, rule bases etc.) are encoded as either binary or real number strings, producing an initial population of chromosomes by randomly generating these strings. The population of individuals is evaluated using an objective function which characterises the individual's performance in the problem domain. The experimental system is run iteratively with each individual's set of controller parameters. The objective function determines how well each individual performs based on experimental data (in this case the current and velocity tracking performance and power consumption), and is used as the basis for selection via the assignment of a fitness value. Individuals which perform well are assigned a higher probability of being selected for reproduction. Reproduction of individuals (usually in pairs) is achieved through the application of genetic operators, and the new individuals overwrite their parents in the population vector. The resulting new population contains material exchanged between the parents. Due to the stochastic nature of the GA as a search mechanism, a complete sweep of the global search space is achieved with more likelihood of finding the global minimum than conventional search methods. Whereas conventional methods require well-behaved objective functions, GAs tolerate noisy, discontinuous and even time-varying function evaluations. The motivation in this case for combining GAs with fuzzy logic for control is to investigate a number of factors. Firstly, the design potential which can be gained by removing the need for knowledge solicitation to enable the fuzzy logic design. Secondly to reduce the design time. Thirdly to examine a method for introducing robustness into the fuzzy design. Finally to investigate and define an method for multiobjective controller design where an accurate system model is either unavailable, or runs extremely slowly, a limiting factor in the process of iterative evolutionary design.

1.2 *Hardware overview*

The application consists of a brushed DC permanent magnet field motor fed by a four quadrant DC chopper drive operating at 5kHz. Figure 1 shows a schematic of the on-line control system and hardware setup. The objective is to perform robust closed loop speed control on this motor. The drive motor is connected via a flexible coupling to a field wound DC load motor which itself is fed directly by a 200V DC supply. The disturbance torque from this load motor is independently controllable, based on the applied armature voltage. Current control is embedded in the INTEL 80C196KC microcontroller as is the fuzzy logic velocity controller. The microcontroller also hosts the velocity and current feedback signals from the motor set and chopper drive respectively. The multiobjective optimisation programme runs under *Matlab* [18], and resides on a PC. Candidate controllers are downloaded from this host to the microcontroller via the serial link and on-line debug facility allowing direct access to programme memory. Assessment of the candidate controllers is performed on the PC according to a pre-programmed performance cost function. A National Instruments data acquisition board performs signal acquisition to bring feedback signals into the PC, to facilitate performance evaluation via the objective function.

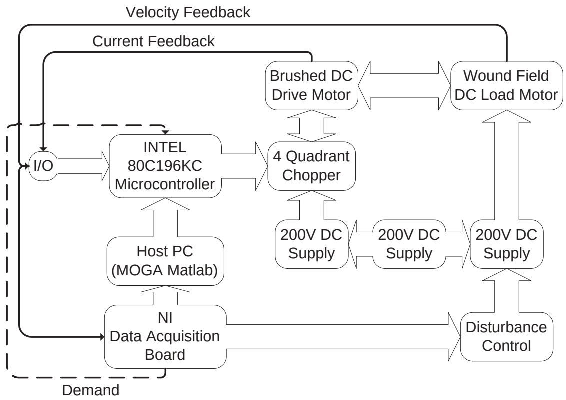

Fig. 1. Online optimisation hardware setup

2. OFF-LINE FUZZY LOGIC CONTROLLER DESIGN

A fuzzy logic velocity control scheme had been developed for this system previously in order to investigate the implementation issues involved with this type of control structure. Although claims are made concerning the reduction of development time [19], in fact the development time to produce the fuzzy controller off-line was significantly greater than the time required to manually produce and tune a robust PID tracking controller, a factor which is exacerbated by the complexity of the design procedure. The designer must choose input and output membership functions, a meaningful rule base, and an effective defuzzification strategy. In essence this requires the implementation of a controller with many degrees of freedom in the design, and consequently a complex implementation to achieve robust design.

An iterative design approach was utilised, to investigate the effects of the various degrees of design freedom in order to design the best controller. The most effective control structure was found to be input membership functions for error $(v(k))$ and change of error $(\Delta v(k))$ at time k, where

$$\Delta v(k) = v(k) - v(k-1) \tag{1}$$

The form of the membership function is shown in figure 2, The input functions are linked to the controller output by a rule base of the form;

- IF error is Positive Big THEN output is Positive Big
- IF error is Positive Small THEN output is Positive Small
- IF error is Zero THEN output is Zero
- IF error is Negative Small THEN output is Negative Small
- IF error is Negative Big THEN output is Negative Big

This rule base is repeated for change of error, and was implemented experimentally, the structure being shown in figure 3. The error and change

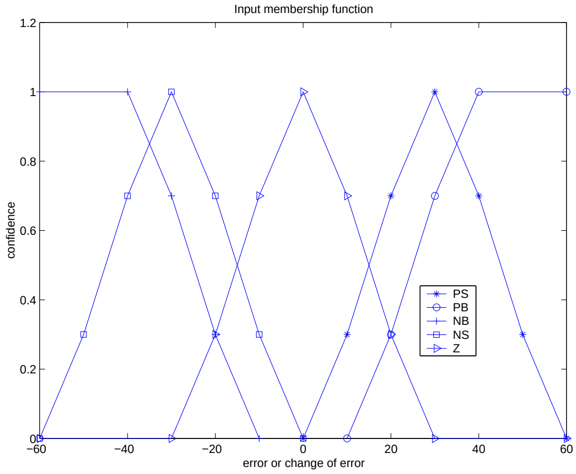

Fig. 2. Input membership functions for v and Δv

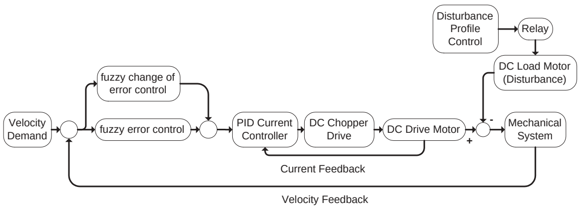

Fig. 3. Fuzzy controller implementation

of error controllers were constructed as follows. The fuzzy inference rule base is implemented using the intersection operator. A matrix of input and output sets included in each rule is constructed. Assuming for example, two classical sets A and B in a universe U, with membership functions μ_A and μ_B, then the minimum operator *intersection* can be defined as [19]

$$\mu_{A\cap B}(x) = min(\mu_A(x), \mu_B(x)) \qquad (2)$$

The overall transfer surface for the controller was achieved by combining the matrix representation of all the individual rules into one overall matrix and applying the maximum operator *union*. This operation exemplifies the Cartesian cross product operator defined on n classical sets $A_1, ..., A_n$ as

$$\begin{aligned} X_{i=1}^{n} &= A_1 \times ... \times A_n \qquad (3) \\ &= ((x_1, ..., x_n) | x_1 \in A_1, ..., x_n \in A_n) \end{aligned}$$

The resulting transfer characteristic for velocity error is shown in figure 4. A corresponding surface consequently exists for change of velocity error. The utilisation of the centre of area defuzzification strategy [19] results in a controller structure shown in (figure 5).The surface provides a nonlinear relationship between velocity error, change of velocity error, and the controller output.

2.1 *Results of off-line fuzzy logic controller design*

The performance of the off-line designed fuzzy logic velocity controller is presented in figure 6

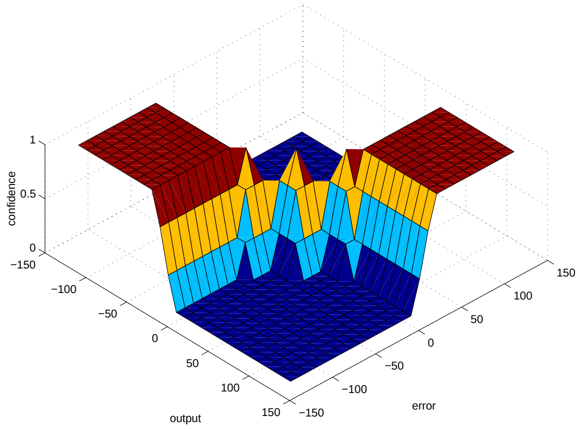

Fig. 4. Fuzzy transfer surface for velocity error

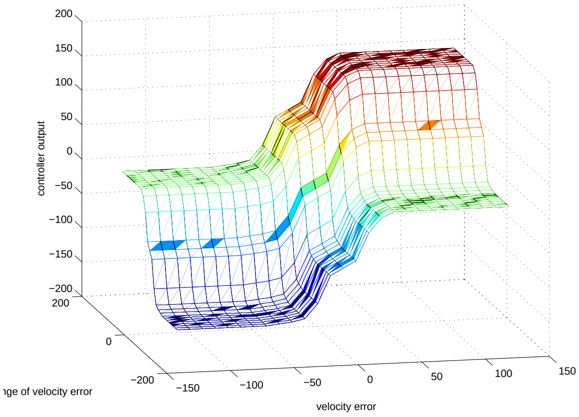

Fig. 5. Fuzzy controller output

for the non-disturbance case, and figure 7 for the case with external disturbance. In this case, a bi-directional velocity demand is supplied to the controller. In both the disturbed and undisturbed state, velocity tracking is comparable both in terms of rise time and steady state accuracy to a standard PID controller. Although it is beyond the central remit of this paper, a substantial amount of time was spent selecting an appropriate defuzzification strategy and the selection of the input-output sets in order to achieve this tracking performance. Consequently, the investigation of an online fuzzy logic design becomes an attractive proposition which is described in the next section. The development for an automatic design scheme with hardware in the loop will be considered and experimentally tested.

3. ON-LINE FUZZY LOGIC CONTROLLER DESIGN

Evolutionary algorithms have been used to optimize various aspects of intelligent control systems. In particular, the algorithm can generate the fuzzy rulebase, and tune the parameters of the associated membership functions. The application of evolutionary algorithms to fuzzy optimisation

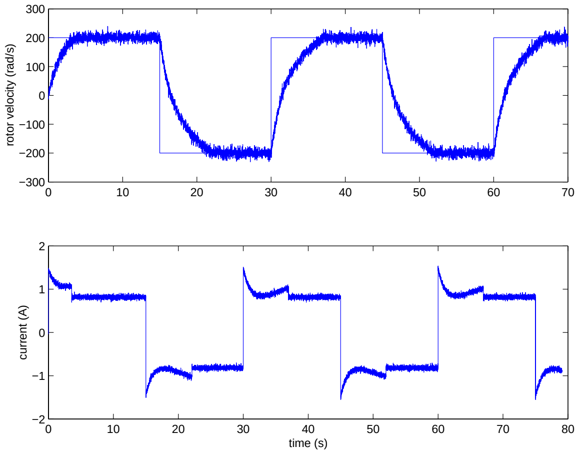

Fig. 6. Off-line designed fuzzy controller performance

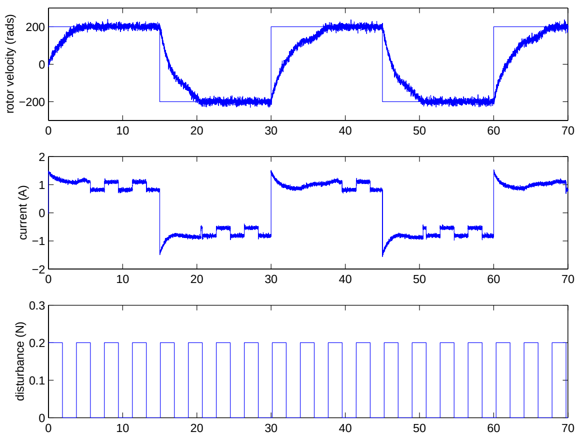

Fig. 7. Off-line designed fuzzy controller performance with external disturbance

is broadly split into two general areas; namely membership function tuning, and rulebase design with tuning. GA has been applied [14] to the off-line tuning of fuzzy membership functions, using a *fuzzy clustering* technique a fuzzy model was developed to describe the friction in a DC-motor system. In this case, the GA was seeded initially by the results obtained by fuzzy clustering. The results were greatly improved over those obtained by the non-tuned version. An *asynchronous* evolutionary algorithm has been used to generate membership functions to facilitate the rapid prototyping of fuzzy controllers [15]. This approach utilized parallel processing, being implemented on a 512 processor CM-5 Connection Machine. The application in question was a simulated space-based oxygen production system. Evolutionary methods have also been used where the derivation of an obvious set of fuzzy rules is not immediately apparent. In this case, the designer may either pre-specify a number of rules, or allow the number of rules to become an extra degree of freedom in the design. In all cases, the computational intensiveness of the designed optimisation technique must be borne in mind, particularly in the case of on-line optimisation.

Due to the considerable computational and experimental considerations implicit in this method, certain constraints are included in the bounds of the decision variable vector in order to bring the automatic design time down to a reasonable level. A flowchart of the experimental setup is shown in

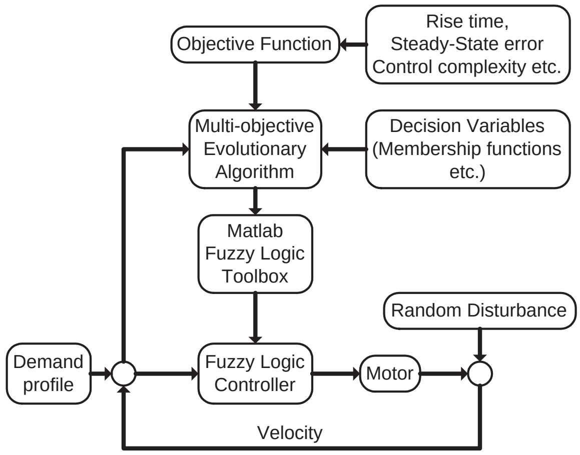

Fig. 8. On-line Fuzzy Logic design setup

figure 8 and contains a number of elements;

- Objective function

The objective function contains the elements of performance and design to be minimised, including rise-time, steady-state error, power utilisation and control complexity.

- Decision variables

The decision variable vector contains the elements of controller design which are implemented in each individual during the evolutionary process. The decision variables include the number of inputs, number of membership functions for each input and output, number of rules in the rule base, and-or-ignore conjugates in each rule, and finally the defuzzification algorithm. The selected values in the decision variables vector are passed to the Matlab Fuzzy Logic Toolbox to be constructed into a controller file. In order to reduce the necessary execution time to converge to a satisfactory conclusion the decision variable vector is bounded as follows

- number of inputs: 1-2
- number of membership functions for each input 3-5
- membership functions limited to triangular, with 2 base and one peak co-ordinate
- number of rules: 3-5
- conjugates: and, or, none
- defuzzification: centre of maximum

In addition, a random $+/-0.2Nm$ disturbance is injected during each experimental run to introduce an element of robustness into the design procedure. For each iteration of the design, the fuzzy controller was run on the motor rig and its performance ranked. It was found that the selected controller appeared early on in the procedure (generation 17 in a population of 10), in an initial run of 50 generations. The Pareto-Optimal set of solutions included several configurations and combinations of membership functions, including one which was markedly similar to the solution defined by the off-line fuzzy design with on-line tuning. The solution chosen for presentation here however, exhibits the required dynamic and steady-state performance but is coupled with a minimal set of membership functions (comprising an additional objective) and rules which presents computational advantages.

3.1 *Results of on-line fuzzy logic controller design*

The first results to present are those which show the dynamic and steady state performance of the velocity controller. The undisturbed case is shown in figure 9, and the disturbed case in figure 10 In

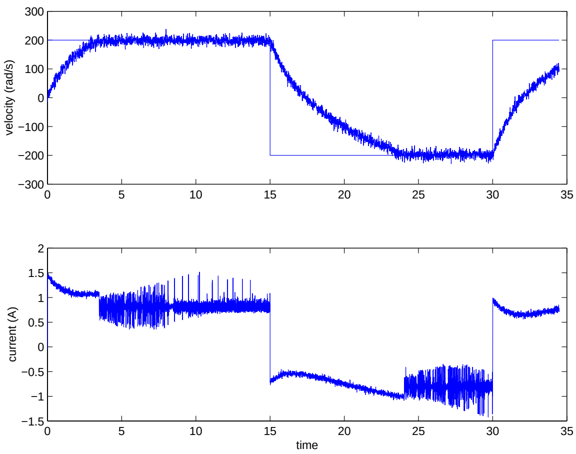

Fig. 9. On-line designed Fuzzy Logic velocity controller performance

both cases, the velocity tracking response of the system is comparable with earlier designs achieved by off-line fuzzy logic control design. One difference of particular interest is the current waveform in both cases which exhibits high frequency components. This effect has been commented upon [20] in the context of fuzzy logic control design, concluding that some off-line or on-line tuning is necessary to eliminate or effectively reduce the harmonics. In the case of the off-line fuzzy logic controller described earlier in this paper, the harmonics were reduced by on-line tuning. For future work in this case, the addition of frequency analysis to the objective function to minimise the unwanted harmonics would be a beneficial area of research. Hardware and computational constraints precluded the implementation of this analysis online at this time, but it is intended that the investigation of this phenomenon on an upgraded rig be performed at some future time. Although

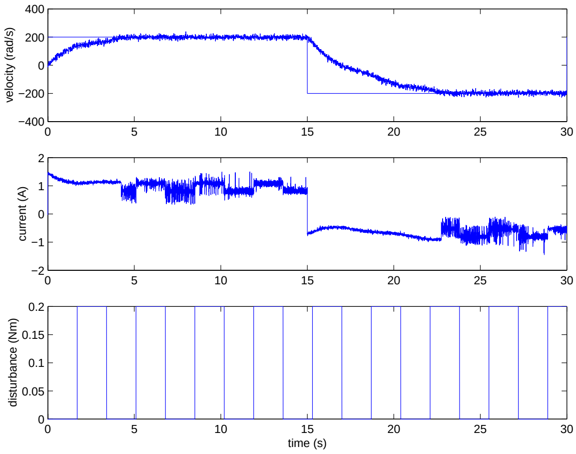

Fig. 10. On-line designed Fuzzy Logic velocity controller performance with disturbance

the performances of the various controllers are very similar, the structure of the on-line and off-line designed controllers are very different. Both have similar rule bases, but whereas the off-line design has inputs of both error and change-of-error, the automatically designed controller solely acts on error input. The membership functions

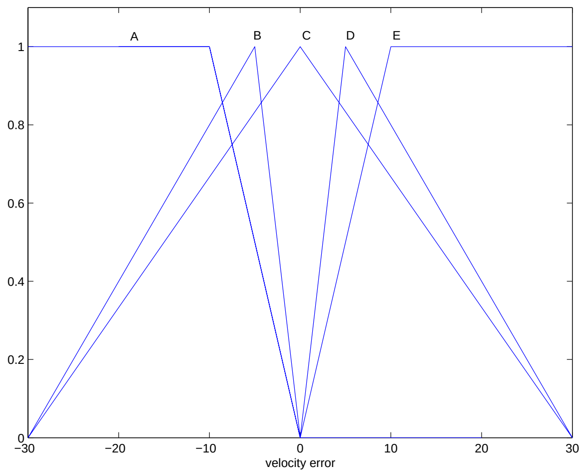

Fig. 11. On-line designed Fuzzy Logic velocity controller input membership functions. A:negbig, B:negsmall, C:zero, D:possmall, E:posbig.

which make up the input set are shown in figure 11, being the same number (5) as in the off-line designed case, but are far more closely clustered around the zero set. The membership functions which make up the output set are shown in figure 12 and are linked to the input set by the rule base;

- if velocity error is *negbig* THEN current demand is *negbig*
- if velocity error is *negsmall* THEN current demand is *negsmall*

- if velocity error is *zero* THEN current demand is *zero*
- if velocity error is *posbig* THEN current demand is *posbig*
- if velocity error is *possmall* THEN current demand is *possmall*

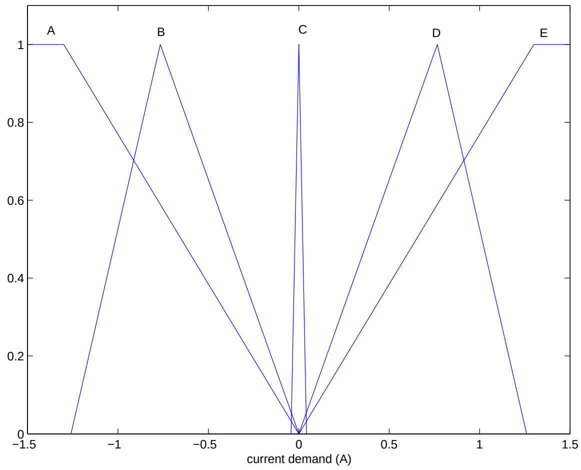

Fig. 12. On-line designed Fuzzy Logic velocity controller output membership functions. A:negbig, B:negsmall, C:zero, D:possmall, E:posbig.

The methods attached to the fuzzy logic controller were as follows;

- and:min
- or:max
- implication:min
- aggregation:max
- defuzzification:mom

4. CONCLUSIONS

The primary objective of this work, to assess the feasibility of automatically designing fuzzy logic controllers on-line with hardware in the loop has been demonstrated. A hardware platform previously intended for fuzzy logic design, formed the hardware in the loop since it was well characterised. The design of a fuzzy logic controller by traditional off-line methods had required manual tuning on line to maximise performance, and in particular, to reduce current harmonics introduced by the control action. It has been shown experimentally that on-line fuzzy logic controller design is feasible, and also that excellent dynamic and steady-state performance can be achieved. The design was optimised without the solicitation of knowledge because of the stochastic nature of the evolutionary optimisation algorithm which searches the multidimensional space of membership functions and rules for combinations which can achieve the performance specified in the objective function. Controller design based around models and simulation is often limited by the veracity of the model under consideration. For example, electromagnetic actuators may be approximated by relatively simple expressions. However under certain circumstances, dynamic effects such as eddy currents, which are extremely difficult to model, need to be included in dynamic simulation. In this case, the differences between actual and simulated plant can make a significant difference to the controller performance. It appears that the on-line fuzzy controller design offers considerable advantages, and is worthy of serious consideration, also the possibility of injecting random disturbances during the design phase resulting in a controller capable of rejecting at least bounded disturbances shows particular promise. This topic together with consideration of the effects of controller dynamics on the harmonic content of the current waveforms will form part of a further investigation.

References

[1] Mamdani E.H., Application of fuzzy algorithms for control of simple dynamic plant., *Proceedings of the IEE*, 121(12), (1974), 1585-1588.

[2] Zadeh L.A., Outline of a new approach to the analysis of complex systems and decision processes, *IEEE Transactions on Systems, Man and Cybernetics*, 3(1973), 28-44.

[3] Betin F., Pinchon D. and Capolino G.A., Control of electrical drives subject to large variations of load: a fuzzy logic approach., *Electromotion*, 8(3), July-September 2001, 155-168.

[4] Guillemin P., Universal motor control with fuzzy logic, *Fuzzy Sets and Systems*, 63(3), May 1994, 339-348.

[5] Hong T.P. and Lee C.Y., Induction of fuzzy rules and membership functions from training examples, *Fuzzy Sets and Systems*, 84(1), 1996, 33-47.

[6] Ishibuchi H. and Tanaka H., Neural networks that learn from if-then fuzzy rules, *IEEE Transactions on Fuzzy Systems*, vol.1, 1993, 85-97.

[7] Grauel A. and Mackenberg H., Mathematical analysis of the Sugeno controller leading to general design rules, *Fuzzy Sets and Systems*, 85(2), 1997, 165-175.

[8] Chang C.H. and Wu Y.C., The genetic algorithm-based tuning method for symmetric membership functions of fuzzy logic control systems, *Proceedings of the International IEEE/IAS Conference on Industrial Automation and Control: Emerging Technologies.*, 1995, 421-428.

[9] Homaifar A. and McCormick E., Simultaneous design of membership functions and rule sets for fuzzy controller using genetic algorithms, *IEEE Transactions on Fuzzy Systems*, 3(2), 1995, 129-139.

[10] Thrift P., Fuzzy logic synthesis with genetic algorithms, *Proceedings of the 4th International Conference on Genetic Algorithms*, 1996, 279-283.

[11] Wu C.J. and Liu G.Y. A genetic approach for simultaneous design of membership functions and fuzzy control rules, *Journal of Intelligent and Robotic Systems*, vol.28, 2000, 195-211.

[12] Fonseca C.M. and Fleming P.J., An overview of evolutionary algorithms in multiobjective optimisation, *Evolutionary Computation*, 3(1), 1995, 1-16.

[13] Fonseca C.M. and Fleming P.J., Multiobjective optimisation and multiple constraint handling with evolutionary algorithms - Part 1: A unified formulation and Part 2: Application example, *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 28(1), 1998, 26-37 and 38-47.

[14] Tzes A., Peng P.Y. and Guthy J., Genetic-based fuzzy clustering for DC-motor friction identification and identification, *IEEE Transactions on Control Systems Technology*, 6(4), 1998, 462-472.

[15] Kim J., Moon Y. and Ziegler B.P., Designing fuzzy net controllers using genetic algorithms, *IEEE Control Systems Magazine*, 15(3), 1995, 62-72.

[16] Kuo B.C. and Hanselman D.C., Matlab tools for control system analysis and design, *Prentice-Hall International, New Jersey*, 1994, ISBN 0-13-099946-6.

[17] Schroder P., Green B., Grum N. and Fleming P.J., On-line evolution of robust control systems: an industrial active magnetic bearing application., *IFAC Journal of Control Engineering Practice*, 9, 2001, 37-49.

[18] Chipperfield A.J., Fleming P.J. and Pohlheim H.P., A genetic algorithm toolbox for MATLAB, *Proceedings of the International Conference on Systems Engineering*, 1994, pp.200-207, Coventry U.K.

[19] Driankov D., Hellendoorn H. and Reinfrank M., An introduction to fuzzy control, *Springer-Verlag Berlin*, 1993, ISBN 3-540-56362-8.

[20]Zhu Z.Q., Shen Z.X. and Howe D., Comparative study of alternative fuzzy logic control strategies of Permanent Magnet Brushless AC drive, *Proceedings of the 2002 International conference on Control Applications*, pp.42-47, September 18-20, 2002. Glasgow, Scotland, U.K.