Proceedings of the American Control Conference
San Diego, California • June 1999

A Genetic Algorithm Approach to Designing Finite-Precision PID Controller Structures

J.F. Whidborne
Division of Engineering
Kings College London
Strand, London WC2R 2LS, UK
email james.whidborne@kcl.ac.uk

Abstract

The parameters of a digital control design need to be rounded if the controller is implemented with finite precision arithmetic. This often results in degradation of the closed loop performance and reduced stability margins. This paper presents a multi-objective genetic algorithm based approach to designing the structure of a finite-precision PID controller implementation to minimize both the performance degradation and the memory requirements of the implementation. The approach provides a set of solutions that are near Pareto-optimal, and so allows the designer to trade-off the performance degradation against the memory requirements. The method is applied to the PID controller structure for the IFAC93 benchmark problem.

1 Introduction

It is well known that controller implementations with fixed-point arithmetic offer advantages of speed, memory space, cost and simplicity when compared to floating-point arithmetic [1]. Thus, to date, fixed-point processors are still the dominant architecture in many modern digital control engineering applications found in automotive, consumer, military and medical applications. However, a closed-loop control system will suffer a performance degradation and may even become unstable when the designed infinite-precision controller is implemented with a fixed-point digital processor due to the finite precision of the parameter representation resulting from the Finite Word-Length (FWL). This so-called FWL effect is in fact strongly dependent upon the parameterization of the controller. Thus, over the years, many results have been reported in the literature dealing with FWL implementation and their relevant parameterization issues, for example, [2, 3].

Consider the discrete time system shown in Figure 1 with the plant $G(z)$. Let (A_k, B_k, C_k, D_k) be a state-space description of the state space controller, $K(z) = C_k(zI - A_k)^{-1}B_k + D_k$. In this paper, (A_k, B_k, C_k, D_k) is also called a realization of $K(z)$. The realizations of

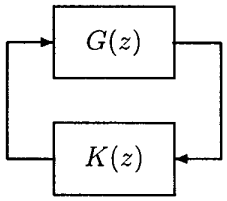

Figure 1: Discrete time feedback system

$K(z)$ are not unique, if $(A_k^0, B_k^0, C_k^0, D_k^0)$ is a realization of $K(z)$, then $(T^{-1}A_k^0T, T^{-1}B_k^0, C_k^0T, D_k^0)$ is an equivalent realization for any non-singular similarity transformation T. A common approach (for example, [3]) to the FWL problem for state space controllers is to to find equivalent controller realizations (or similarity transformations T) such that the closed loop system is insensitive to perturbations in the controller parameters.

A more direct approach is used in this paper (which is similar to the method proposed for [4]). Basically, the approach is to find an FWL controller that is near to the originally designed controller such that the robust stability and closed loop performance degradation and the FWL implementation cost are simultaneously minimized. The approach is based on the generation of near-equivalent finite word-length controller representations by means of the solution to a linear system equivalence completion problem followed by a rounding operation. A Genetic Algorithm (GA) is then used to find sets of (near) optimal near-equivalent FWL controllers. The designed can then simply choose the most appropriate design from this set. Two features of GA's make them very attractive for solving this problem. Firstly, GA's require the solution space to be encoded in binary strings, and since controllers implemented with FWL are also coded in binary, a one-to-one relationship between the genotype and the phenotype within the GA can be defined. Secondly, GA's allow the optimal structure of the solution to be found (e.g. [5]), this means that the implementation word-length

does not need to be defined *a priori*, but the GA will select the best from a predefined set, and so the implementation cost in the form of the memory requirement can also be minimized.

In this paper, a multi-objective genetic algorithm is used. This allows the designer to trade-off FWL implementation robust stability and performance measures against the memory and other requirements of the implementation. For reasons of simplicity, in this paper, a measure of the robust stability/performance degradation is simply taken as the H_∞-norm of the difference between the FWL closed-loop transfer function and the original closed-loop transfer function. The implementation cost is taken as the parameter memory storage requirement. However, the method is entirely generic, and any other set of stability, performance and implementation measures could be used, such as in [4]. The developed approach is applied to the problem of the implementation of a PID controller designed for the IFAC93 benchmark problem [6, 7].

The paper is organized as follows. In the next section, some of the concepts underpinning the method are introduced, namely, multi-objective optimization, multi-objective genetic algorithms and FWL representation. The method also requires a linear system equivalence completion problem to be solved, this also presented in the next section. In Section 3, details of the developed method are presented. The IFAC93 benchmark problem design is presented in Section 4, and in the final section, some conclusions are drawn.

2 Preliminary concepts and theory

2.1 Multi-objective optimization

The majority of engineering design problems are multi-objective, in that there are several conflicting design aims which need to be simultaneously achieved. If these design aims are expressed quantitatively as a set of n design objective functions $\phi_i(p) : i = 1 \ldots n$, where p denotes the design parameters chosen by the designer, the design problem could be formulated as a multi-objective optimization problem:

$$\min_{p \in \mathcal{P}} \{\phi_i(p), \text{ for } i = 1 \ldots n\}. \qquad (1)$$

where $\mathcal{P}$ denotes the set of possible design parameters p. In most cases, the objective functions are in conflict, so the reduction of one objective function leads to the increase in another. Subsequently, the result of the multi-objective optimization is known as a Pareto-optimal solution. A Pareto-optimal solution has the property that it is not possible to reduce any of the objective functions without increasing at least one of the other objective functions.

2.2 Genetic algorithms

Genetic algorithms are search procedures based on the evolutionary process in nature. The idea is that the GA operates on a population of individuals, each individual representing a potential solution to the problem, and applies the principle of survival of the fittest on the population, so that the individuals evolve towards better solutions to the problem.

The individuals are given a chromosomal representation, which corresponds to the genotype of an individual in nature. Three operations can be performed on individuals in the population, selection, cross-over and mutation. These correspond to the selection of individuals in nature for breeding, where the fitter members of a population breed and so pass-on their genetic material. The cross-over corresponds to the combination of genes by mating, and mutation to genetic mutation in nature. The selection is weighted so that the 'fittest' individuals are more likely to be selected for cross-over, the fitness being a function of the function which is being minimized. By means of these operations, the population will evolve towards a near-optimal solution.

GA's are very well-suited for multi-objective optimization problems. The approach used here is the Multi-Objective Genetic Algorithm (MOGA) [8, 9], which is an extension on an idea by [10]. The idea behind the MOGA is to develop a population of Pareto-optimal or near Pareto-optimal solutions. This is achieved by finding a set of solutions which are non-dominated. An individual j with a set of objective functions $\phi^j = (\phi_1^j, \ldots, \phi_n^j)$ is said to be non-dominated if for a population of N individuals, there are no other individuals $k = 1, \ldots, N, k \neq j$ such that $\phi_i^k \leq \phi_i^j \forall i = 1, \ldots, n$ and $\phi_i^k < \phi_i^j$ for at least one i. With the MOGA, non-dominated individuals are given the greatest fitness, and individuals that are dominated by many other individuals are given a small fitness. Using this mechanism, the population evolves towards a set of non-dominated, near Pareto-optimal individuals. Details of this mechanism are given in [8].

In addition to finding a set of near Pareto-optimal individuals, it is desirable that the sample of the whole Pareto-optimal set given by the set of non-dominated individuals is fairly uniform. A common mechanism to ensure this is fitness sharing [8], which works by reducing the fitness of individuals that are genetically close to each other. However, as will be seen in Section 3.3, not all the bits of a candidate solution bit string are necessarily active. Thus, 2 individuals may have the same genotype, but different gene strings. Thus it difficult to measure the difference between 2 genotypes in order to implement fitness sharing, so, for the sake of simplicity in this paper, multiple copies of genotypes are simply removed from the population.

2.3 FWL representation

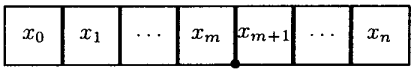

Figure 2: FWL fixed-point representation

A typical 2's complement FWL fixed-point representation of a number $q(x)$ is shown in Figure 2. The number $q(x)$ is represented by a $m+n+1$ binary string x where $x = [x_0, x_1 \ldots, x_m, x_{m+1}, \ldots, x_{m+n}]$, $x_i \in \{0,1\}$, $m \in \{0,1,2,\ldots\}$, $n \in \{0,1,2,\ldots\}$. The value $q(x)$ is given by

$$q(x) = -x_0 2^m + \sum_{i=1}^{m} x_i 2^{i-1} + \sum_{i=m+1}^{m+n} x_i 2^{m-i}. \quad (2)$$

The set of possible values which can be taken by an FWL variable represented by a $m+n+1$ binary string x is defined as $\mathcal{Q}_{n,m}$ given by $\mathcal{Q}_{n,m} = \{q : q = q(x), x_i \in \{0,1\} \forall i\}$.

2.4 A linear system equivalence completion problem

The following theorem is required.

Theorem 1 *Given a 2 state SISO LTI system $F(z)$, where*

$$F \stackrel{s}{=} \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \quad (3)$$

$$\stackrel{s}{=} \left[\begin{array}{cc|c} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ \hline c_1 & c_2 & d \end{array}\right], \quad (4)$$

then (provided certain rank and singularity conditions are satisfied) given $(\tilde{a}_{11}, \tilde{a}_{12} \neq 0, \tilde{c}_1, \tilde{c}_2)$ there exists an equivalent state-space representation $\tilde{F}$ where

$$\tilde{F} \stackrel{s}{=} \left[\begin{array}{c|c} \tilde{A} & \tilde{B} \\ \hline \tilde{C} & \tilde{D} \end{array}\right] \quad (5)$$

$$\stackrel{s}{=} \left[\begin{array}{cc|c} \tilde{a}_{11} & \tilde{a}_{12} & \tilde{b}_1 \\ \tilde{a}_{21} & \tilde{a}_{22} & \tilde{b}_2 \\ \hline \tilde{c}_1 & \tilde{c}_2 & \tilde{d} \end{array}\right], \quad (6)$$

such that $\tilde{F}(z) = F(z)$ where

$$F(z) = C\left[zI - A\right]^{-1} B + D \quad (7)$$

and

$$\tilde{F}(z) = CT\left[zI - TAT^{-1}\right]^{-1} T^{-1} B + D, \quad (8)$$

where

$$T = \left[\begin{array}{cc} t_{11} & t_{12} \\ t_{21} & t_{22} \end{array}\right] \quad (9)$$

is non-singular.

Proof: The proof is by construction. From (8),

$$\tilde{A} = TAT^{-1}, \quad (10)$$

so by a property of similar matrices [11, p.115], $\mathrm{tr}\,\tilde{A} = \mathrm{tr}\,A$. Hence, given $\tilde{a}_{11}, \tilde{a}_{12}$, $\tilde{a}_{22}$ is given by $\tilde{a}_{22} = a_{11} + a_{22} - \tilde{a}_{11}$. Similarly, from the property of similar matrices that $\det\tilde{A} = \det A$ (by definition), given $\tilde{a}_{12} \neq 0$, $\tilde{a}_{21}$ is given by $\tilde{a}_{21} = (\tilde{a}_{11}\tilde{a}_{22} - \det A)/\tilde{a}_{12}$. From (10), $AT - T\tilde{A} = 0$, which gives [12, p. 255]

$$\left[I \otimes A - \tilde{A}^T \otimes I\right]\mathbf{t} = 0 \quad (11)$$

where $\mathbf{t} = [t_{11}, t_{21}, t_{12}, t_{22}]^T$ and $[I \otimes A - \tilde{A}^T \otimes I]$ is rank 2. Now, from (8), $CT = \tilde{C}$. Hence, given $\tilde{c}_1, \tilde{c}_1$,

$$\left[\begin{array}{cccc} c_1 & c_2 & 0 & 0 \\ 0 & 0 & c_1 & c_2 \end{array}\right]\mathbf{t} = \left[\begin{array}{c} \tilde{c}_1 \\ \tilde{c}_1 \end{array}\right]. \quad (12)$$

A $Y \in \mathbb{R}^{4\times 4}$ can be constructed from (11) and (12) where $Y\mathbf{t} = z$, and where $z \in \mathbb{R}^4$ is a column vector with 2 elements of $\tilde{C}$ and two zero elements. If Y is non-singular rank 4, T and hence $\tilde{B}$ can be calculated. From (8), $\tilde{d} = d$. ■■

3 MOGA for optimal FWL controller structures

3.1 Procedure outline

The problem can be summarized as follows. Given a discrete time nominal plant $G(z)$ and designed PID (or other second order) controller $K(z)$, find an FWL controller $K_q(z)$ and state-space parameterization such that the difference (in some sense) between the closed loop system and the original closed loop system and the implementation costs are simultaneously minimized.

For this paper, two indices are defined for the MOGA. The first index, ϕ_1, is the closed loop measure of the implementation accuracy of the FWL controller compared to the original controller and is taken as a robust stability/performance degradation measure in terms of an H_∞-norm. The second index, ϕ_2, is a measure of the total number of bits required to store the FWL controller parameters. These indices are defined in Section 3.2.

The main elements of the approach are now summarized.

i) Generate partially filled random FWL parameterizations of the 2 state controller i.e.

$$\tilde{K} \stackrel{s}{=} \left[\begin{array}{cc|c} q_1 & q_2 & ? \\ ? & ? & ? \\ \hline q_3 & q_4 & D_K \end{array}\right] \quad (13)$$

where $q_j \in \mathcal{Q}_{n,m}, j = 1, \ldots, 4$, i.e. each q_j is FWL.

ii) By Theorem 1, solve a linear system equivalence completion problem such that $\tilde{K}(z) = K(z)$.

iii) Obtain $K_q \approx \tilde{K}$ by rounding the non-FWL parameters in $\tilde{K}$ so that they are FWL, i.e.

$$K_q \stackrel{s}{=} \left[\begin{array}{cc|c} q_1 & q_2 & q_7 \\ q_5 & q_6 & q_8 \\ \hline q_3 & q_4 & q_9 \end{array}\right] \tag{14}$$

where $q_j \in \mathcal{Q}_{n,m}, j = 1, \ldots, 9$.

iv) Calculate the robust stability/performance degradation index, ϕ_1, and the memory requirement index, ϕ_2, for K_q.

v) Use the MOGA to evolve a set of near Pareto-optimal solutions.

3.2 Performance indices

A measure of the difference between the FWL implemented closed loop system and the original closed loop system is taken as the H_∞ norm of the difference between the closed loop pole transfer functions of the systems. If both the implemented controller and closed loop system are stable, ϕ_1 is defined as

$$\phi_1 = \| R - R_q \|_\infty \tag{15}$$

where $R = \frac{GK}{1+GK}$ and $R_q = \frac{GK_q}{1+GK_q}$. This is also a measure of the robust stability/performance degradation. An implementation memory function, ϕ_2, is defined as the total number of bits used to implement K_q. This function is calculated bearing in mind that parameters of K_q that are $1, 0, -1$ or that are a power of 2 require less memory requirement than the $m + n + 1$ bits from (2).

3.3 Encoding of solution space

In order to generate the partially filled parameterizations of K given by (13), the genotype of each individual consists of a 71 bit binary string $[x_1, \ldots, x_{71}], x_i \in \{0, 1\}$. The bit length of the integer part of the parameters' representation $m \in 0, \ldots, 7$ is represented by $[x_1, x_2, x_3]$, and is given by $m = \sum_{i=1}^{3} x_i 2^{i-1}$. The bit length of the fractional part of the parameters' representation $n \in 0, \ldots, 15$ s represented by $[x_4, \ldots, x_7]$, and is given by $n = \max(\sum_{i=4}^{7} x_i 2^{i-4}, 15 - m)$. The four $m + n + 1$ word-length parameters $q_j \in \mathcal{Q}_{n,m}, j = 1, \ldots, 4$, where $(q_1, q_2, q_3, q_4) = (\tilde{a}_{11}, \tilde{a}_{12}, \tilde{c}_1, \tilde{c}_2)$ respectively, are represented by $(x_{8+16(j-1)}, \ldots, x_{8+m+n+16(j-1)})$. Thus not all the bits in x are necessarily active. The values of q_j are calculated by (2).

4 Application to an IFAC93 benchmark problem design

The proposed approach is applied to a PID controller [7] designed for the IFAC93 benchmark problem [6]. The continuous time nominal plant, $G(s)$, is given as

$$G(s) = \frac{25(1 - 0.4s)}{(s^2 + 3s + 25)(5s + 1)}.$$

The plant, $G(s)$, is discretized with sampling period of $t_s = 0.05$ seconds. A PID controller is designed as [7]

$$K(s) = 1.311 + 0.431/s + 1.048s/(1 + 12.92s),$$

and discretized using the bilinear transform. The initial realization K^0 is set to

$$K_q \stackrel{s}{=} \left[\begin{array}{ccc} 1 & -0.99614 & 1 \\ 0 & 1.99614 & 0 \\ \hline 0.049578 & 0.049469 & 1.41693 \end{array}\right].$$

The MOGA is implemented in MATLAB using the GA Toolbox [13]. An elitism strategy is used whereby all non-dominated individuals propagate through to the next population. Selection is performed using stochastic universal sampling with a fitness determined by the number of dominating individuals. Single point crossover is used with a probability of 0.7. Each bit has a probability of mutation of 0.00933.

The MOGA is run with a population of 120. After 800 generations (which takes about 3 hours on a 450 MHz Pentium II), a set of non-dominated solutions is obtained. This set is shown in Figure 3. The figure also shows the stable dominated solutions along with FWL implementations of the initial realization, K_q^0, for various word-lengths. In addition, the diagonal equivalent balanced realization of $K(z)$ is calculated using the `ssbal` function from the MATLAB Control System Toolbox [14, pp 9/200–201], and FWL implementations of the realization for various word-lengths are shown. Note that the axis for $\| R - R_q \|_\infty$ is shown to log scale for clarity. The figure clearly shows that improved FWL state-space realizations for the controller can be obtained using the approach.

The 45 bit solution labelled (1) in Figure 3 is selected and is the controller

$$K_q \stackrel{s}{=} \left[\begin{array}{ccc} 0.99609375 & 2^{-7} & -1.5859375 \\ 0 & 1 & -0.3515625 \\ \hline -2^{-5} & 0 & 1.41796875 \end{array}\right],$$

which requires $m = 1$ and $n = 8$ for the FWL representation given by (2). In addition, one of the parameters is zero, another is 1, and 2 others are powers of 2, and can hence be implemented by register shifts. Figure 4 shows the frequency response of the original closed loop transfer function $|R(e^{j\omega t_s})|$ and the difference between the systems $|R(e^{j\omega t_s}) - R_q(e^{j\omega t_s})|$.

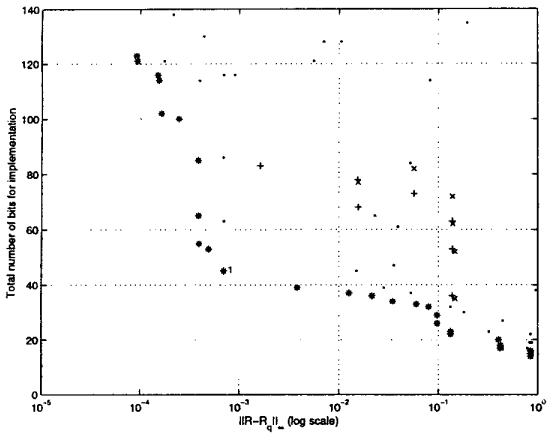

Figure 3: Solution set showing trade-off with MOGA non-dominated set (∗), MOGA dominated set (·), K_q^0 realizations (×) and equivalent balanced realizations (+). The selected controller is marked 1.

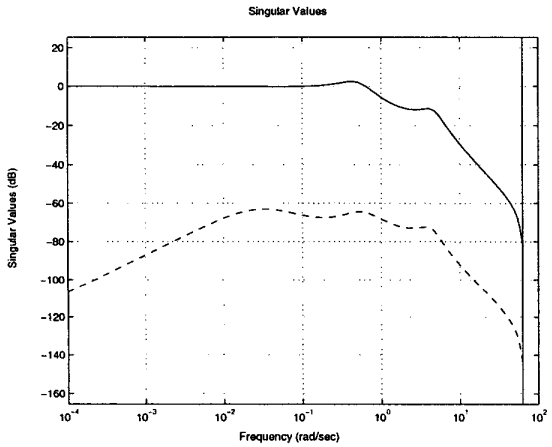

Figure 4: Singular values of R (—) and $R - R_q$ (- - -)

5 Conclusion

A GA-based method to determine optimal FWL structures for PID digital controllers has been presented. The method is illustrated by an example. The method exploits the fact that the implementation of FWL controllers is by means of binary numbers, as is the representation in genetic algorithms.

The method requires the solution of a linear system equivalence completion problem. A solution to the linear system equivalence completion problem for 2 state SISO systems has been presented. This solution needs to be extended to general linear systems for the methodology to be extended to higher order controllers. Otherwise, the method is entirely generic, in that any set of computable stability and performance measures can be included. In addition, other implementation measures could be included, such as computation time, as well as measures of other important quantization effects, namely quantization noise and scaling requirements.

References

[1] M.K. Masten and I. Panahi. Digital signal processors for modern control systems. *Control Engg Practice*, 5(4):449–458, 1997.

[2] J.B. Knowles and R. Edwards. Effect of a finite-word-length computer in a sampled-data feedback system. *Proc. IEE*, 112(6):1197–1207, 1965.

[3] M. Gevers and G. Li. *Parametrizations in Control, Estimations and Filtering Problems: Accuracy Aspects.* Springer-Verlag, Berlin, 1993.

[4] J.F. Whidborne and R.S.H. Istepanian. Optimal finite-precision PID controller structures using genetic algorithms. In *Proc. 14th IFAC World Congress*, Beijing, 1999. To appear.

[5] N.V. Dakev, J.F. Whidborne, A.J. Chipperfield, and P.J. Fleming. H_∞ design of an EMS control system for a maglev vehicle using evolutionary algorithms. *Proc. IMechE, Part I: J of Sys and Contr.*, 311(4):345–355, 1997.

[6] S.F. Graebe. Benchmark IFAC 93: Adaptive/robust control of unknown plant. Pre-conference communication, 1992. Centre for Industrial Control Science, University of Newcastle, Australia.

[7] J.F. Whidborne, G. Murad, D.-W. Gu, and I. Postlethwaite. Robust control of an unknown plant – the IFAC 93 benchmark. *Int. J. Control*, 61(3):589–640. 1995.

[8] C.M. Fonseca and P.J. Fleming. Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In *Genetic Algorithms: Proceeding of the Fifth International Conference*, pages 416–423, San Mateo, CA, 1993.

[9] C.M. Fonseca and P.J. Fleming. Multiobjective genetic algorithms. In *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, number 1993/130, pages 6/1–6/5, London, England, 1993.

[10] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA., 1989.

[11] E.D. Nering. *Linear Algebra and Matrix Theory.* John Wiley, New York, 2 edition, 1990.

[12] R.A. Horn and C.R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, Cambridge, U.K., 1991.

[13] A. Chipperfield, P.J. Fleming, H. Pohlheim, and C.M. Fonseca. *Genetic Algorithm Toolbox: User's Guide.* Dept. Automatic Control and Systems Engineering, University of Sheffield, U.K., 1994.

[14] The MathWorks, Inc., Natick, MA. *Control System Toolbox: User's Guide, version 4*, 1998.