Optimisation of the weighting functions of an H_∞ controller using genetic algorithms and structured genetic algorithms

E. ALFARO-CID[†]*, E.W. MCGOOKIN[‡] AND D.J. MURRAY-SMITH[✦]

[†] Instituto Tecnológico de Informática, Valencia, Spain

[‡] Dept. of Aerospace Engineering, University of Glasgow, Glasgow, UK

[✦]Centre for Systems and Control, Dept. of Electronics and Electrical Engineering, University of Glasgow, Glasgow, UK

* Corresponding author. E-mail: evalfaro@iti.upv.es

Abstract: In this paper the optimisation of the weighting functions for an H∞ controller using genetic algorithms and structured genetic algorithms is considered. The choice of the weighting functions is one of the key steps in the design of an H_∞ controller. The performance of the controller depends on these weighting functions since poorly chosen weighting functions will provide a poor controller. One approach that can solve this problem is the use of evolutionary techniques to tune the weighting parameters. The paper presents the improved performance of structured genetic algorithms over conventional genetic algorithms and how this technique can assist with the identification of appropriate weighting functions' orders.

Keywords: H_∞; optimization; genetic algorithms; structured genetic algorithms; ship control

1. Introduction

This paper presents an investigation into the design and optimisation of H_∞ controllers. In this case the H_∞ methodology is used to provide the structure for propulsion controllers and navigation controllers for an oil platform supply ship.

Traditionally, the steering of an ocean-going vessel has required the ability of a highly skilled helmsman to sail the boat along a desired path. As well as maintaining the correct heading the pilot has been in charge of sailing at the desired speed. Therefore, while sailing a boat there are two coupled tasks to perform: getting the ship to navigate in the desired direction (heading control) at the desired speed (propulsion control) (Fossen 1994, Skjetne 2003).

In the last decades, in order to keep up with the demand for oil, companies have started to drill for oil offshore. Oil platform supply ships are offshore service vessels employed in carrying supplies to drilling units of sub sea oil and gas. Their cargo consists of the equipment, food and water that an offshore platform needs to keep its production going and, therefore, these vessels are essential to the operations of the offshore oil and gas industry. Such supply boats need to be robust against environmental disturbances and manoeuvrable. They have to be able to handle adverse weather conditions (especially the ships for Northern latitudes, e.g. Atlantic Canada and the North Sea) and keep a position as steady as possible while unloading operations are carried out. Naturally, such supply vessels are essential for the smooth operation of oil platforms. This imposes the need for their navigation system to be accurate and robust against environmental disturbances, which can only be achieved through automatic controllers.

The particular application used in this paper is a scale model of an oil platform supply ship called CyberShip II (CS2) (Lindegaard and Fossen 2003). CS2 is a test vehicle developed in the Department of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU) in Trondheim. Computer-generated simulations based on a nonlinear hydrodynamic model of CS2 are used in the optimisation studies. These simulations have proven to be sufficiently representative of the full-scale manoeuvring dynamics of such a vessel.

In this study the navigation and propulsion control is achieved through the implementation of H_∞ control theory. H_∞ controllers are optimal controllers that minimise an H_∞ norm rather than the usual L_2 quadratic norm (Grimble 1986). The H_∞ norm is appropriate for specifying both the level of plant uncertainty and the signal gain from disturbance inputs to error outputs. The consequence of this is the robust stability inherent to H_∞ controllers, which is the main reason for the development of H_∞ techniques (Zhou et al. 1996).

The performance of an H_∞ controller depends on the choice of the weighting functions. This is not a trivial matter since poorly chosen weighting functions will provide poor controllers. A solution to this problem is to use optimisation techniques, such as Genetic Algorithms (GAs) (Goldberg 1989, Holland 1992) that optimise these weighting functions automatically. GAs are numerical search methods that mimic natural biological evolution. They operate on a population of potential solutions applying the Darwinian principle of survival of the fittest to produce better and better possible solutions to a given problem. At each generation, a new set of candidate solutions is created by the process of selecting individuals according to their level of fitness (the better the performance of the solution, the larger the fitness value is for the solution) and breeding them together using operators borrowed from natural genetics.

This process results in the evolution of populations of better possible solutions to a given problem. The GA solves the problem of tracking the desired path while keeping actuator usage to a minimum by evolving controller solutions that satisfy these objectives. In addition, the H_∞ controller has been optimised using not only a GA but also a genetic model called structured genetic algorithm (sGA) proposed by Dasgupta and McGregor (1993). The fact that the weighting functions to tune in the H_∞ problem are not only parameters but transfer functions (i.e. structures) make this optimisation problem very suitable for sGA, given that sGAs are very appropriate for structural optimisation.

References to H_∞ weighting functions tuning using GAs are not frequent in the literature. The fact that the choice of weighting functions involves, not only the adjustment of the parameters, but also determining the weighting structure, makes it a complicated optimisation problem. However, since the late 1990s, some genetic approaches to the H_∞ controller design problem can be found.

Most authors use a loop-shaping method combined with a GA search. Given a predetermined structure for the pre and post compensators required in the loop-shaping technique, the GA is used to optimise the parameters that define that structure (Dohna *et al.* 1997,Christiansson and Lennartson 1999, Kaitwanidvilai, and Parnichkun 2004, Lo and Khan 2004, Kim and Chung 2005). On the other hand, in Dakev *et al.* (1997) a multiobjective evolutionary algorithm is used in conjunction with an H_∞ loop-shaping design procedure to perform multiobjective search over a set of possible weighting function structures and parameter values, giving a structural dimension to the search process.

Despite of the small number of papers in the area of GA-based H_∞ controllers, the range of applications is quite varied: from flight control (Sveriduk *et al.* 1998, Dai and Mao 2002) to a magnetic levitation train (Dakev *et al.* 1997).

The control applications references using sGA are scarce. Tang *et al.* (1996) used sGA for the design of the pre compensator and post compensator in an H_∞ controller design with loop-shaping.

2. CyberShip II

The model subject used in this work is a mathematical model that describes the dynamics of CyberShip II (CS2). CS2 is a scale model (scale 1/70th aprox.) of an oil platform supply ship, which has been developed at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU) in Trondheim.

2.1. Mathematical model of CyberShip II

The optimisation and design of the controllers for CS2 have been conducted using a non-linear hydrodynamic model based on the kinetic and kinematic equations that represents the dynamics of the vessel (Fossen 1994). When these kinetic and kinematic equations are combined together the following matrix form is produced (assuming M to be invertible):

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\eta}} \end{bmatrix} = \begin{bmatrix} -\mathbf{M}^{-1}(\mathbf{C}(\mathbf{v})+\mathbf{D}) & \mathbf{0} \\ \mathbf{J}(\boldsymbol{\eta}) & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\eta} \end{bmatrix} + \begin{bmatrix} \mathbf{M}^{-1} \\ \mathbf{0} \end{bmatrix} \cdot \boldsymbol{\tau} \qquad (1)$$

Here **M** is the mass/inertia matrix, **C** is the Coriolis matrix, **D** is the damping matrix and **J** is the Euler transformation matrix. Also, $\nu = [u, v, r]^T$ is the body-fixed linear and angular velocity vector, $\eta = [x, y, \psi]^T$ denotes the position and orientation vector with coordinates in the earth-fixed frame and $\tau = [\tau_1, \tau_2, \tau_3]^T$ is the input force vector, given that τ_1, τ_2 and τ_3 are the forces and torque along the body-fixed x, y and z-axes, respectively. These are depicted on Figure 1.

[Insert Figure 1 about here]

The expression from (1) corresponds to the non-linear state space equation in (2):

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x}) \cdot \mathbf{x} + \mathbf{B} \cdot \boldsymbol{\tau} \qquad (2)$$

Thus, the corresponding states and inputs for this model are 6 states (u - surge velocity, v - sway velocity, r – yaw rate, ψ - heading angle, x_p – x-position and y_p – y-position) and 3 inputs (τ_1 – surge thrust force, τ_2 – sway thrust force and τ_3 – yaw thrust torque).

Previous work from the authors (Alfaro-Cid *et al.* 20005a, Alfaro-Cid *et al.* 2005b) includes response following results obtained with the real model in a water facility. These results prove the adequacy of the mathematical model and how the controllers designed with it can be directly implemented on the real scale ship.

2.2. Environmental disturbances

In order to evaluate the robustness of the GA optimised controllers against environmental disturbances, the optimisation process has been carried out for simulated manoeuvres in the presence of environmental disturbances.

In this research the analysis has focused on the disturbance considered to be the most relevant for surface vessels, i.e. wind-generated waves.

The model that has been used to simulate the wave's action on the vessel derives the forces and moments induced by a regular sea on a block-shaped ship and it is described in Zuidweg (1970). It forms a vector called τ_{waves} that is directly added to the input vector, τ, in (1) and (2) using the principle of superposition, i.e.

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x}) \cdot \mathbf{x} + \mathbf{B} \cdot \left(\boldsymbol{\tau} + \boldsymbol{\tau}_{\text{waves}} \right) \tag{3}$$

The individual wave forces and torque effects can be represented by the equations:

$$\begin{aligned} X_{wave}(t) &= \sum_{i=1}^{N} \rho g B L T \cos(\beta - \psi) s_i(t) \\ Y_{wave}(t) &= \sum_{i=1}^{N} - \rho g B L T \sin(\beta - \psi) s_i(t) \\ N_{wave}(t) &= \sum_{i=1}^{N} \frac{1}{24} \rho g B L (L^2 - B^2) \sin 2(\beta - \psi) s_i^2(t) \end{aligned} \tag{4}$$

Here L, B and T are the length, breadth and draft of the wetted part of a ship, considering it as a parallelepiped. ρ is the density of the water, $s_i(t)$, the wave slope and (β -ψ), the angle between the heading of the ship and the direction of the wave (in radians).

The wave slope, s_i can be related to the wave spectral density function $S(\omega_i)$. In this study a modified version of the Pierson-Moskowitz spectrum has been used (Fossen 1994):

$$S(\omega) = \frac{4\pi^3 H_s^2}{(0.710 T_o)^4 \omega^5} \exp\left(\frac{-16\pi^3}{(0.710 T_o)^4 \omega^4} \right) \tag{5}$$

Here, T_o is the modal period and H_s, the significant wave height.

The simulated waves included in some of the optimisation runs had a significant height of 3 meters (0.0429 m after scaling), which corresponds to a sea state code of 5 (rough sea). Waves of this height happen in the North Atlantic with a probability of 15.44% (Fossen 1994). The initial angle of encounter between the waves and the vessel has been chosen to be 135^o. During the manoeuvre this angle changes due to the vessel turning and presenting a different encounter angle to the wave. This choice should not affect the search process.

3. H$_\infty$ controller design

H$_\infty$ control has been used to provide the structure for propulsion controllers (for governing surge velocity) and navigation controllers (for governing heading) for CS2. The H$_\infty$ controller designed is a MIMO controller with 3 inputs (τ_1, τ_2 and τ_3) and two outputs (u and ψ).

H$_\infty$ control is based on a standard feedback structure. It consists of a plant, a controller, reference **r**, commanded input **u**, sensor noise **n,** output **y** and plant disturbance **d.** In order to include some performance objectives in the system model, the standard feedback structure is modified by adding weighting functions. The aim of the weighting functions is not only to put the emphasis on some of the components but also to make components measured in different metrics comparable. Once the weighting matrices are included the feedback configuration is as shown in Figure 2.

[Insert Figure 2 about here]

Any feedback control configuration can be expressed as a *linear fractional transformation (LFT)* [Zhou *et al.* (1996)]. The following block diagram (Figure 3) represents a *lower linear fractional transformation,* of the previous Figure 2.

[Insert Figure 3 about here]

Here **w** represents the exogenous input, consisting of commands, external disturbances and sensor noise (**r**, **n** and **d** in previous Figure 2), **y** is the measurement available to the controller, **u** is the output from the controller, and **z** is the error signal that is desired to keep small. The transfer function matrix **G** represents not only the conventional plant to be modelled but also any weighting functions included to specify the desired performance and **K** represents the controller.

One of the key steps in the design of any kind of controller is the choice of the controller parameters. The performance of the controller depends on the values of these parameters. Conventionally, these parameters are manually tuned by the designer, who attempts to find an acceptable controller solution. However, this is a very tedious and time consuming process.

In the case of H_∞ controllers, the parameters to tune are the weighting functions. As it has already been mentioned, the transfer function matrix G represents the plant plus the weighting functions. These weighting functions are added in order to include the desired performance objectives within the controller design structure. These weightings are not just constant values to tune but transfer functions whose structure needs to be optimised as well. Since poorly chosen weighting functions provide poor H_∞ controller designs, the choice of weighting functions must be the prime concern of the designer.

Figure 4 shows the configuration of the H_∞ controller for this application. As it can be seen, in this case, the exogenous input **w** consists of the reference signal to track for propulsion and heading (u_c and r_c) plus the disturbances in form of waves (τ_{waves}), the output from the controller **u** is the vector of commanded forces and torque (τ_{1com}, τ_{2com} and τ_{3com}), the measurement available to the controller **y** consist of the tracking errors (u_c-u and r_c-r) and, finally, the signal to minimise **z** consists of the tracking errors and the commanded forces and torque together with five weighting functions: three of them

(W_{t1}, W_{t2} and W_{t3}) are affecting the commanded forces while W_u and W_r are weighting the error signals.

[Insert Figure 4 about here]

The choice of **z** is given by the particular application being considered. Since it is a model of an oil platform supply ship, the objectives are to minimise the error between the current and desired dynamics (i.e. follow the desired manoeuvre at the desired speed) plus minimise the actuators usage, avoiding saturation and wear and tear of the actuators. This configuration of **z** and the weighting functions fulfils the objectives of minimising the error signals and the actuators usage.

The H∞ optimal control problem is then to design a stabilizing controller, **K**, so as to minimise the closed-loop transfer function from **w** to **z**, T_{zw}, in the H∞ norm,

$$\left\|T_{zw}\right\|_{\infty} = \sup_{\omega} \overline{\sigma}(T_{zw}(j\omega)) \tag{6}$$

$\overline{\sigma}(T_{zw}(j\omega))$ being the largest singular value of $T_{zw}(j\omega)$. Thus, the H∞ norm is the supreme of the largest singular values of $T_{zw}(j\omega)$ over all the values of ω.

Finding an optimal H∞ controller is often both numerically and theoretically complicated. However, in practice it is often not necessary to design an optimal controller. It is usually enough to obtain controllers that are very close, in the norm sense, to the optimal designs. These will be called *suboptimal controllers*.

Suboptimal H∞ control problem: given γ>0, find all admissible controllers **K**, if there are any, such that $\left\|T_{zw}\right\|_{\infty} < \gamma$. As it can be seen, the solution for the suboptimal control problem is not unique.

Designing a controller that reduces $\|T_{zw}\|_\infty$ results in a minimisation of the signal gain from disturbance inputs to error outputs in the controlled system. In addition, the H_∞ norm gives the maximum energy gain of the system, which is minimised as well.

Since the plant to be controlled is a MIMO system, we have chosen the state-space formulation described in Zhou *et al.* (1996) to design the H_∞ controller.

4. Genetic algorithms

GAs (Goldberg 1989, Holland 1992) are optimisation techniques that mimic the way species evolve in nature. Two mechanisms are the key elements in the evolution of many species: the Darwinian natural selection and sexual reproduction. The genetic operators involved in sexual mating allow the offspring to inherit the features from both its parents. GAs emulate this process by evolving a population of candidate solutions (suitably encoded), for a given problem, through a predetermined number of generations. It achieves this through the following heuristic mechanisms.

An initial population of possible solutions (individuals) is generated at random. In the controller's parameters optimisation problem context the individuals are decoded to obtain the corresponding controller's parameter values and these are implemented in the controllers. The controller's performance is evaluated through simulation of the system being controlled.

Then, there are three main operators that constitute the GA search mechanism: *selection, crossover* and *mutation*.

The selection procedure depends on the quantified evaluation of each candidate solution that is obtained from the cost function. Selection determines which solutions are chosen for mating according to the principal of survival of the fittest (i.e. the better the performance of the solution, the more likely it is to be chosen for mating and the more offspring it produces).

Once the new population has been selected, chromosomes are ready for crossover and mutation.

The crossover operator combines the features of two parents to create new solutions. Crossover allows an improvement in the species in terms of the evolution of new solutions that are fitter than any seen before. One or several crossover points are selected at random on each parent and then, complementary fractions from the two parents are spliced together to form a new chromosome.

The mutation operator alters a copy of a chromosome reintroducing values that might have been lost or creating totally new features. One or more locations are selected on the chromosome and replaced with new randomly generated values.

The three operators are implemented iteratively. Each iteration produces a new population of solutions, which is called a *generation*. The GA continues to apply the operators and evolve generations of solutions until a near optimum solution is found or a finite number of generations is produced.

5. Optimisation procedure

5.1. Genetic algorithms vs. structured genetic algorithms encoding

Each possible solution of the problem *search space* must be suitably encoded so that the GA can manipulate their parameter values. As it has already been shown in Figure 4, in our H$_\infty$ design the parameters to be optimised are five weighting functions: three of them (W_{t1}, W_{t2} and W_{t3}) are affecting the commanded forces while W_u and W_r are weighting the error signals.

The generic GA assumes a predefined 2nd order transfer function with a single zero structure for the weighting functions of the error signals. This way the GA can choose the structure of the transfer function, e.g. if a first order weighting function is more appropriate the GA can choose α to be equal to β and thus the zero cancels one of the poles. Thus, the weighting functions of the error signals are as follows:

$$W_u(s) = \frac{K_u^{'}(s+\alpha_u)}{(s+\beta_u)(s+\gamma_u)} \tag{7}$$

$$W_r(s) = \frac{K_r^{'}(s+\alpha_r)}{(s+\beta_r)(s+\gamma_r)} \tag{8}$$

Here, $K_u^{'}$ and $K_r^{'}$ are the gains of the weighting functions, $-\alpha_u$ and $-\alpha_r$ represent the position of the zero of the weighting functions and $-\beta_u$, $-\gamma_u$, $-\beta_r$ and $-\gamma_r$ give the positions of the two poles of the functions.

To avoid a very high order controller, the weightings for τ_{1com}, τ_{2com} and τ_{3com} (i.e. W_{t1}, W_{t2} and W_{t3}) have been kept constant. Thus, the vector to be minimised is $z = [W_u(s)\cdot(u_c-u), W_r(s)\cdot(r_c-r), K_{t1}\cdot\tau_{1com}, K_{t2}\cdot\tau_{2com}, K_{t3}\cdot\tau_{3com}]^T$. The GA optimises the eleven parameters shown in Table 1 that define this predetermined structure.

[Insert Table 1 about here]

Each weighting function parameter to be optimised is encoded as a string of five genes (McGookin, 1997). These genes, instead of being binary bits (as they used to be in the traditional GA), are integers included within the interval [0,9], in order to allow a wide range of possible values (from 0 to 9.999×10^3) in smaller strings.

[Insert Figure 5 about here]

All the values are encoded to be positive to ensure stability and minimum-phase characteristics. Since there are 11 parameters to optimise each possible GA solution is represented by a chromosome that is a string of 55 genes.

On the other hand, the sGA optimises not only the parameters of the weighting functions but also the structure. This is done by means of the multi-layered chromosome structure characteristic of sGA, where some control genes can activate and deactivate sets of parameter genes. The sGA differs from the normal GA in the chromosome structure. In sGA the chromosome consists of 2 types of genes: control genes and coefficient genes (Dasgupta and McGregor, 1993). The control genes define which coefficient genes are used in the decoding of the individual, therefore promoting a hierarchy in the chromosome structure. This hierarchy allows the sGA to be suited not only for parametric optimisation but also for structural optimisation.

In this work sGA has been used to optimise the weighting function in the H_∞ control problem. The way that sGA has been implemented is by adding 4 extra binary genes (the "control genes") to the GA chromosome representation for H_∞. Two of these genes specify the structure of the weighting function acting on the yaw rate error while the other two define the structure of the function weighting the surge error signal.

This representation allows 4 options: a constant gain (when the control genes are encoded as 00), a gain plus a pole (01), a gain plus a pole plus a zero (10) and a gain plus 2 poles plus a zero (11). Depending on the structure chosen the number of parameters needed to define the transfer function varies. Then the control genes activate or deactivate the parameter genes according to the weighting structure reflected by these control elements.

Tang *et al.* (1996) have published a similar application. The main difference with this approach is that Tang *et al.* (1996) used a loop-shaping technique where the sGA optimised the precompensator and postcompensator of the weighted plant. In addition, in their work each control gene decides the inclusion or not of a single pole or zero, while in our approach 2 control genes define the whole transfer function structure.

Thus, the sGA optimises the parameters shown in the previous table, and also the control genes (w_u and w_r) that define the structure of the weighting functions of the error signals (see table 2). The weighting functions for the actuator signals have been preset to be just gains as in the normal GA optimisation. Hence, the 13 parameters to optimise by the sGA are as follows:

[Insert Table 2 about here]

As in the case of the GA optimisation all the parameters are encoded to be positive values to ensure stability and minimum-phase characteristics.

The control genes w_u and w_r define the structure to be used for W_u(s) and W_r(s) as shown in Table 3, where i is u or r, depending on the weighting function being considered:

[Insert Table 3 about here]

Thus, each possible sGA solution is represented by a chromosome that is a string of 59 genes: 55 integers for encoding the coefficient genes plus 4 bines for encoding the control genes.

5.2. Cost function

Once an initial population of chromosomes is generated at random, the chromosomes are decoded to obtain the corresponding controller's parameter values and these are implemented in the controllers. The controller's performance is evaluated through simulation of the system being controlled.

The optimisation design criteria are quantified by the cost function in (9). In addition to this there is a desired response that the controller must track. Although in our approach we have used a response following criterion to optimise the controller, it could be extended to any other performance objective such as bandwidth or delay constraints just by modifying the fitness function to suit these objectives. The desired responses for propulsion and heading used throughout this study are two critically damped steps up and down. The heading reference is a 45^{o} double step manoeuvre. This zig-zag manoeuvre has been chosen due to its relevance in ship manoeuvring studies (Zuidweg 1970). The reference for the surge speed makes the vessel accelerate up to 0.7 m/s and then decelerate back to rest. Both manoeuvres are executed simultaneously as shown in Figure 6.

[Insert Figure 6 about here]

The cost function has three terms for each controller (Alfaro-Cid. 2003), which combine to give a single objective, multi-aspect criterion for evaluating both controller solutions:

$$C=\sum_{i=0}^{tot}\left[\left(\Delta\psi_i\right)^2+\lambda_1\left(\tau_{3i}\right)^2+\mu_1\left(\frac{\tau_{3i}-\tau_{3i-1}}{\Delta t}\right)^2+\left(\Delta u_i\right)^2+\lambda_2\left(\tau_{1i}\right)^2+\mu_2\left(\frac{\tau_{1i}-\tau_{1i-1}}{\Delta t}\right)^2\right] \quad (9)$$

Here *tot* is the total number of iterations, $\Delta\psi_{\mathrm{i}}$ is the *i*th heading angle error, τ_{3i} is the *i*th yaw thrust force, Δu_{i} is the *i*th surge velocity error and τ_{1i} is the *i*th surge thrust force.

The third and sixth terms introduce a measurement of the inputs increasing or decreasing rates (Alfaro-Cid 2003). These terms reduce the oscillations in the inputs, avoiding any unnecessary wear and tear on the actuators that shortens their lifespan.

As the input force and torque are always larger than the output errors near the optimum (since the error terms are tend towards zero), they dominate the cost values in this crucial area. This imbalance leads to solutions that provide very small thruster effort, but very poor tracking of the desired responses. In order to avoid this four scaling factors are introduced, so that an equally balanced trade-off between the elements is achieved and all the terms of the cost function are equally optimised.

5.3. GA and sGA schemes

The genetic model used for the GA optimisation is: tournament selection (with tournament size equal to 8); exponential and non-uniform mutation; and double point crossover with a probability of 0.8.

The non-uniform mutation technique is based on a non-uniform operator proposed by Michalewicz (1992). The idea is that when a gene is selected for mutation, the mutation jump it suffers depends on the time of the optimisation when this mutation happens. The sooner in the optimisation (i.e. the early the generation), the bigger the mutation jump. This way in the first generations exploration is encouraged while in the final generations, when convergence has occurred, small mutation jumps improve the fine local tuning.

If a gene is chosen for mutation, it will be assign one of the values shown in (10) with a 50% probability:

$$newgene = \begin{cases} gene + \Delta(t, 9 - gene) \\ gene - \Delta(t, gene) \end{cases} \tag{10}$$

The function $\Delta(t,y)$ returns a value in the range $[0,\ y]$ such that the probability of $\Delta(t,y)$ being close to 0 increases as t increases:

$$\Delta(t, y) = y\left(1 - r^{(1-t/T)^b}\right) \tag{11}$$

Here r is a random number from $[0, 1]$, T is the maximal generation number and b is a system parameter determining the degree of non-uniformity.

The only difference between the non-uniform mutation technique implemented in this research and the non-uniform operator proposed by Michalewicz (1992) is the non-randomness of the parameter r.

There are genes in the coding that have a higher weight than others, i.e. a small variation in one of these genes is reflected in a big variation in the parameter to optimise once the individual is decoded. In the coding used in this research, the genes carrying a higher weight are genes number 1 and 5, while changes in genes 3 and 4 are less relevant (see Figure 5).

The non-uniformity of the mutation in this case consists of two effects. On one hand the mutation jump varies depending on the generation number. On the other hand it also varies depending on the position in the chromosome of the gene to be mutated.

In the implementation, the number r is not chosen at random but the choice of r is made as shown in Figure 7. In the initial populations higher mutation jumps are associated with the gene positions 1 and 5, resulting in big variations in the phenotype that help in the exploration of the search space. As the generations progress the effect is inverted and the biggest amounts of mutation are associated with the genes 3 and 4, resulting in closer phenotypes that improve the fine tuning.

[Insert Figure 7 about here]

This non-uniform mutation scheme is combined with an exponential probability of mutation. It has been argued the increasing relevance of crossover in the initial generations and of mutation in the final ones. Using an exponential probability of mutation as shown in Figure 8 allows it.

[Insert Figure 8 about here]

This choice is supported by the results obtained in a comparison study of the performance of various GAs (Alfaro-Cid 2003). The population size has been 80 and the number of generations 50. This configuration has been run 15 times in order to analyse the performance of the method and any similarity in the solutions.

The implementation of the sGA is equal to that of GA: tournament selection (with tournament size equal to 8); exponential and non-uniform mutation; and double point crossover with a probability of 0.8. Again, the population size has been 80, the number of generations 50 and the number of runs 15.

The initial parameters for the tuning process are chosen at random by the genetic schemes.

6. Optimisation results

In this section the results obtained in the GA and sGA optimisations are shown. Both methods have been run with and without disturbances being included in the optimisation simulation scenario.

Each type of optimisation method has been run 15 times in order to ensure consistency in the controller results. The best and the average results obtained through the runs are shown in terms of their cost values. In addition the manoeuvring performance of the best controller is plotted. These plots are divided into 6 subplots. On the left hand side are the results obtained for the propulsion subsystem are plotted, while the results obtained for heading are plotted in the right hand side. The subplots at the top of the figure represent the desired and measured outputs, u and ψ respectively. The desired outputs are plotted in a dashed line and the actual outputs are represented in a solid line. The subplots in the middle of the figure represent the output errors, i.e. the surge error, $u_d – u$, and the heading error, $\psi_d - \psi$. Finally, the subplots at the bottom of the figure depict the control signals corresponding with the propulsion and heading subsystems, i.e. τ_1 and τ_3. Although the propulsion and heading manoeuvres are plotted in separate subplots, they are executed within the same simulation simultaneously.

Thus, Section 6 is structured as follows: Section 6.1 shows the results obtained using the GA parametric optimisation. The subsections 6.1.1 and 6.1.2 show the results when the optimisation is performed without and with disturbances included in the simulation. Section 6.2 follows the same structure as Section 6.1 but shows the results obtained using the sGA structural optimisation. Finally, Section 6.3 presents a comparison summary of both methods.

6.1. GA parametric optimisation

6.1.1. Optimisation without waves results. The best result of the 15 runs of the GA provided a cost value of 1.58 with the H_∞ gains shown in table 4. The GA has converged to this value in 32 generations.

[Insert Table 4 about here]

Averaging the cost values obtained through the 15 runs excluding the best and the worst yields the statistical values shown in Table 5:

[Insert Table 5 about here]

Figure 9 shows the performance of the H_∞ controller when using the weighting functions from Table 4.

[Insert Figure 9 about here]

These simulated results are good as they show that the tracking of the desired responses is accurate. The maximum surge error is around 0.01ms^{-1} while the heading error is kept smaller than 2^{o}. Regarding the actuators usage it is free from rippling and in the case of τ_1 the force is very smooth. The heading control shows two spikes that correspond with the beginning of the turn. They are due to the high gain of the system that reacts very quickly to the change and then it needs to overcompensate. The overall performance is very satisfactory, in that they exhibit good desired response tracking and the input signals do not cause the actuators to saturate.

6.1.2. Optimisation with waves results. The GA optimisation in the presence of simulated waves has converged in the last generation to a cost value of 33.4 with the following H_∞ gains:

[Insert Table 6 about here]

Averaging the cost values obtained through the 15 runs excluding the best and the worse gives the values from table 7.

[Insert Table 7 about here]

Figure 10 illustrates the simulation results of the H_∞ controller from Table 6.

[Insert Figure 10 about here]

The plot illustrates some degradation in the tracking performance, especially in the surge speed tracking. The surge error signal is quite noisy and shows a considerable steady-state error (just below 0.1ms^{-1}). The values from table 6 show that the inclusion of waves has resulted in both poles being moved away from the zero position, which explains the increased steady-state error. In addition, K_{t1} has been significantly reduced causing the slowness of the surge response. The heading response does not degrade as much as the surge, although it shows a large error peak at the beginning of the manoeuvre caused by the waves forcing the boat to overturn. The main effect of the waves in the actuators' signals is some superimposed rippling. Overall, the controller copes quite well with a rough sea state.

6.2. sGA structural optimisation

6.2.1. Optimisation without waves results. Once run 15 times, the GA that provides the best result has converged in 24 generations to a cost value of 1.56 with the following H∞ gains:

[Insert Table 8 about here]

The values obtained for the control genes mean that for the surge speed error the weighting function is a first-order transfer function with a gain equal to K_u' and a pole at $-\beta_u$ (the values α_u and γ_u are deactivated), while the weighting function for the yaw rate error is a second-order transfer function (with a gain of K_r' and poles at $-\beta_r$ and $-\gamma_r$) with a zero at $-\alpha_r$.

$$W_u(s)=\frac{K_u'}{(s+\beta_u)} \tag{12}$$

$$W_r(s)=\frac{K_r'(s+\alpha_r)}{(s+\beta_r)(s+\gamma_r)} \tag{13}$$

After averaging the cost values obtained through the 15 runs excluding the best and worse the following values are obtained:

[Insert Table 9 about here]

Next Figure 11 shows the simulated results obtained with the controller from Table 8 while tracking a zig-zag.

[Insert Figure 11 about here]

The above figure shows good tracking of the desired manoeuvre. The maximum surge error is just above 0.01ms^{-1}, although it presents a slight steady-state error. The tracking of the heading response is also good, kept well within the 2° interval. Regarding the actuators' usage the surge signal is very smooth, but the heading signal presents the high gain peaks already observed in Figure 5.

6.2.2. Optimisation with waves results. The best result of all runs of the GA with waves provides a cost value of 11.7 with the H_∞ gains shown in table 10. The GA has converged in 41 generations.

[Insert Table 10 about here]

Since the control genes are set to 1 for propulsion and 2 for heading, the weighting function for the surge error is a first-order transfer function (gain equal to K_u' and pole at -β_u). The weighting function for the heading is a first-order transfer function (gain equal to K_r' and pole at –β_r) with one zero at -α_r. The values γ_u, α_u and γ_r are deactivated.

$$W_u(s)=\frac{K_u^{'}}{(s+\beta_u)} \tag{14}$$

$$W_r(s)=\frac{K_r^{'}(s+\alpha_r)}{(s+\beta_r)} \tag{15}$$

The sGA has converged to solution that provides a smaller cost value than the one obtained with GA (11.7 versus 33.4) with a lower order controller.

After averaging the cost values obtained the values in Table 11 are obtained:

[Insert Table 11 about here]

Figure 12 presents the results obtained when simulating the solution from Table 10.

[Insert Figure 12 about here]

The performance of the controller is good considering the rough environmental disturbances included in the simulation. The surge response is a bit slower than in the optimisation without disturbances, due to the reduction in the weighting gains that can be observed in tables 8 and 10. The heading response is characterised by a slight overshoot. The main feature in the actuators' signals is the superimposed rippling, more striking in the heading signal.

6.3. Comparison of results

When comparing the results obtained with GA and sGA when optimising the H_∞ controller without waves, just by visual inspection of figures 9 and 11, they appear to be practically the same. Taking into account the cost function of the solutions, sGA provided an equivalent cost value (1.5653 versus 1.5807) using a more simple weighting structure.

Comparing Figures 10 and 12 it can be concluded that the performance of the H_∞ solution obtained using sGA when disturbances are included in the optimisation is better than solution obtained with the standard GA in the optimisation with waves. The heading error is reduced by half and the steady-state error in the surge response does not occur in the sGA results. The actuators' usage is also reduced, although the heading actuator signal is noisier.

The cost values obtained using sGA are better and they are obtained in fewer generations as shown in Table 12.

[Insert Table 12 about here]

7. Conclusions

When comparing the best cost values obtained with GA versus those obtained with sGA, it seems quite clear the advantage of using sGA for this kind of optimisation problem since it provides better cost values in fewer generations.

The good cost values obtained are reflected in the performance plots of the best controllers. In the optimisation without waves the performances of the GA and sGA optimised controllers are equivalent. Both controllers provide an accurate tracking of the desired responses while keeping a smooth actuator signal. In the optimisation with waves the sGA optimised controllers obtained a better tracking: they eliminate the steady-state error in the surge response and reduced by half the heading error. In addition the actuators' usage is decreased. Moreover all this is achieved with more simple weighting functions (i.e. lower order in the controller).

Therefore, the proposed sGA model has performed well as a structure optimisation technique in simulation. It has provided better results than those obtained with GA while reducing the order of the weighting functions and, as a result, the order of the controller. The sGA results prove that the method can be of assistance when identifying appropriate weighting functions orders.

References

[1] E. Alfaro-Cid, “Optimisation of time domain controllers for supply ships using genetic algorithms and genetic programming”. PhD Thesis, University Of Glasgow (2003)

[2] E. Alfaro-Cid, E.W. McGookin and D.J. Murray-Smith, “Evolution of a strategy for ship guidance using two implementations of genetic programming”, *Lecture Notes in Computer Science*, 3447, pp 250-260, 2005

[3] E. Alfaro-Cid, E.W. McGookin, D.J. Murray-Smith and T.I. Fossen, “Genetic algorithm optimisation of decoupled sliding mode controllers: simulated and real results”, *Control Engineering Practice*, 13, pp 739-748, 2005

[4] A. Christiansson and B. Lennartson, “Weight selection for H_∞ control using genetic algorithms”, in *Proc. of the IFAC 14th Triennial World Congress*, 1999, pp 25-30

[5] J. Dai and J. Mao, “Robust flight controller design for helicopters based on genetic algorithms”, in *Proc. of the IFAC 15th Triennial World Congress*, 2002, pp 73-78

[6] N.V. Dakev, J.F. Whidborne, A. Chipperfield, P. and Fleming, “Evolutionary H_∞ design of an electromagnetic suspension control system for a maglev vehicle”, *Proceedings of the Institution of Mechanical Engineers, Part I*, 211, pp 345-355, 1997

[7] D. Dasgupta and D.R. McGregor, “sGA: a structured genetic algorithm”, Technical *Report No. Ikbs-11-93.Ps.Z*, Dept. of Computer Science, University of Strathclyde, 1993

[8] D.C. Dohna, D.S. Desanj, and M.R. Katebi, M.R, “Automatic Weight Selection for H_∞ Control”, in *Proc. of the IEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA’97)*, 1997, pp 50-55

[9] T.I. Fossen, *Guidance and control of ocean vehicles,* Chichester: John Wiley& Sons Ltd, 1994

[10] D. Goldberg, *Genetic Algorithms in Searching Optimisation and Machine Learning,* Reading: Addison-Wesley, 1989

[11] M.J. Grimble, “Optimal H_∞ robustness and the relationship to LQG design problems”, *International Journal of Control*, 43, pp. 351-372, 1986

[12] J.H. Holland, *Adaptation in Natural and Artificial Systems,* Ann Arbor: University of Michigan Press, 1975

[13] S. Kaitwanidvilai, and M. Parnichkun, “Genetic-Algorithm-Based fixed-structure robust H_∞ loop-shaping control of a pneumatic servosystem”, *Journal of Robotics and Mechatronics*, 16, pp. 362-373, 2004

[14] M.S. Kim and C.S. Chung, “The robust flight control of an UAV using MIMO QFT: GA-based automatic loop-shaping method”, *Lecture Notes in Computer Science,* 3398, pp. 467-477, 2005

[15] K.P. Lindegaard and T.I. Fossen, “Fuel efficient rudder and propeller control allocation for marine craft: experiments with a model ship” *IEEE Transactions on Control Systems Technology,* 11, pp. 850-862, 2003

[16] K.L. Lo and L. Khan, “Hierarchical micro-genetic algorithm paradigm for automatic optimal weight selection in H_∞ loop-shaping robust flexible AC transmission system damping control design”, *IEE Proceedings in Generation, Transmission and Distribution*, 151, pp. 109-118, 2004

[17] E.W. McGookin, “Optimization of sliding mode controllers for marine applications: a study of methods and implementation issues”. PhD Thesis, University of Glasgow (1997)

[18] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs*, New York: Springer-Verlag, 1992

[19] R. Skjetne, “Ship Maneuvering: The Past, the Present and the Future”, *Sea Technology*. No. March, pp. 33-37, 2003

[20] G.D. Sweriduk, P.K. Menon and M.L. Steinberg, “Robust command Augmentation system design using genetic methods”, in *Proc. of the AIAA Guidance, Navigation and Control Conference*, 1998, pp. 286-294

[21] K.S. Tang, K.F. Man and D.W. Gu, “Structured genetic algorithm for robust H_∞ control systems design”, *IEEE Transactions On Industrial Electronics*, 43, pp. 575-582, 1996

[22] D.J. Walker and I. Postlethwaite, “Advance helicopter flight control using 2-degree-of-freedom H_∞ optimisation”, *Journal Of Guidance, Control And Dynamics*, 19, pp. 461-468, 1996

[23] K. Zhou, J.C. Doyle and K. Glover, *Robust And Optimal Control*, New Jersey: Prentice Hall, 1996

[24] J.K. Zuidweg, “Automatic guidance of ships as a control problem”. PhD Thesis, Delft University of Technology, The Netherlands (1970)

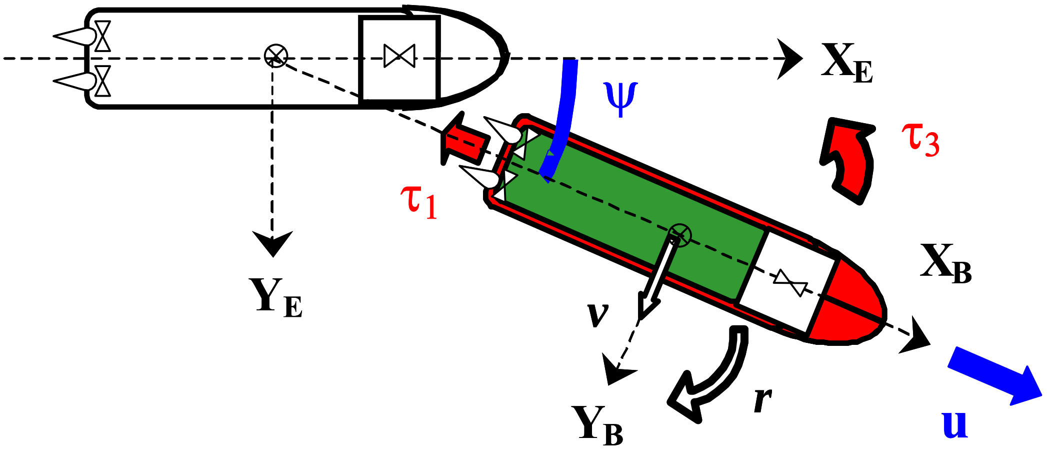


Figure 1: CS2 state variables

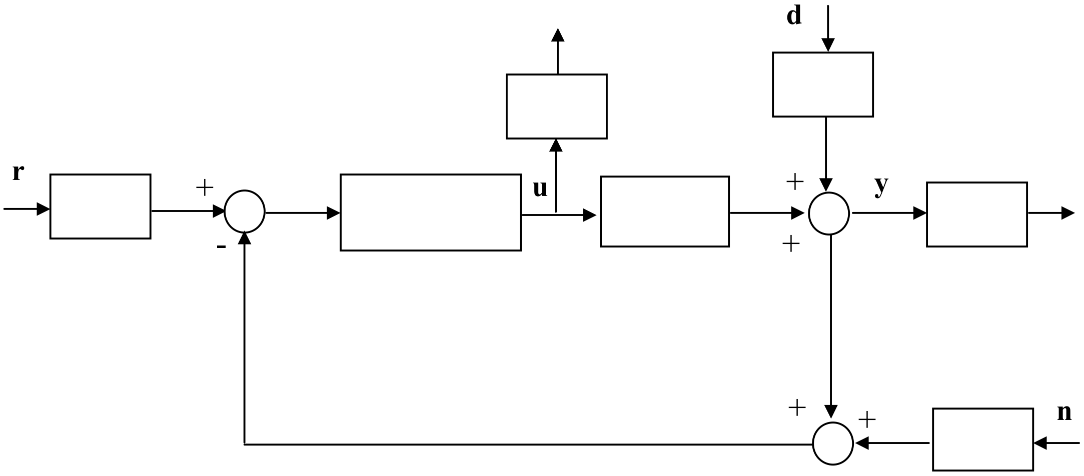


Figure 2: Standard feedback configuration with weightings

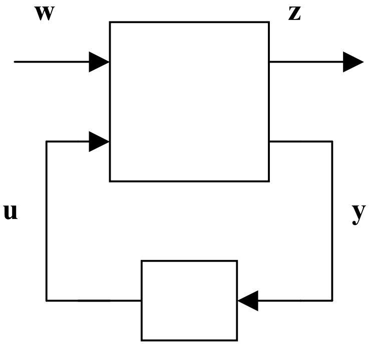


Figure 3: Linear fractional transformation configuration

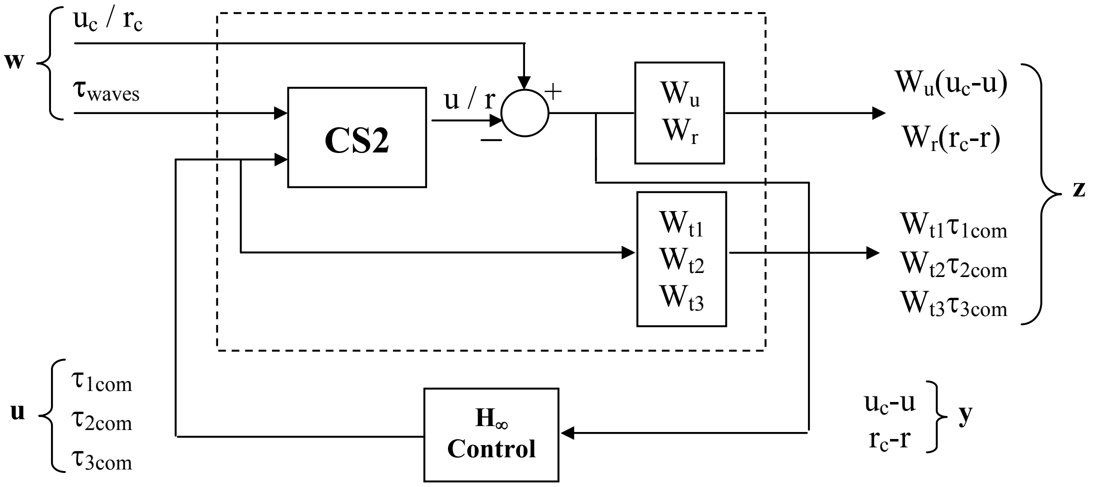
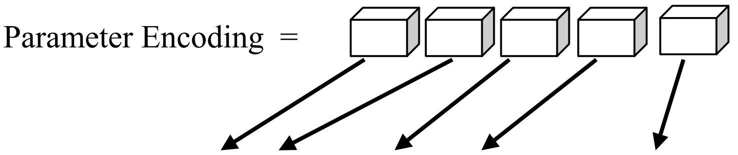


Figure 4. Configuration of Plant + Weighting Functions + H_∞ Controller



$$\text{Parameter Value} = (a + b \times 0.1 + c \times 0.01 + d \times 0.001) \times 10^{(e/2-2)}$$

Figure 5: Parameter encoding

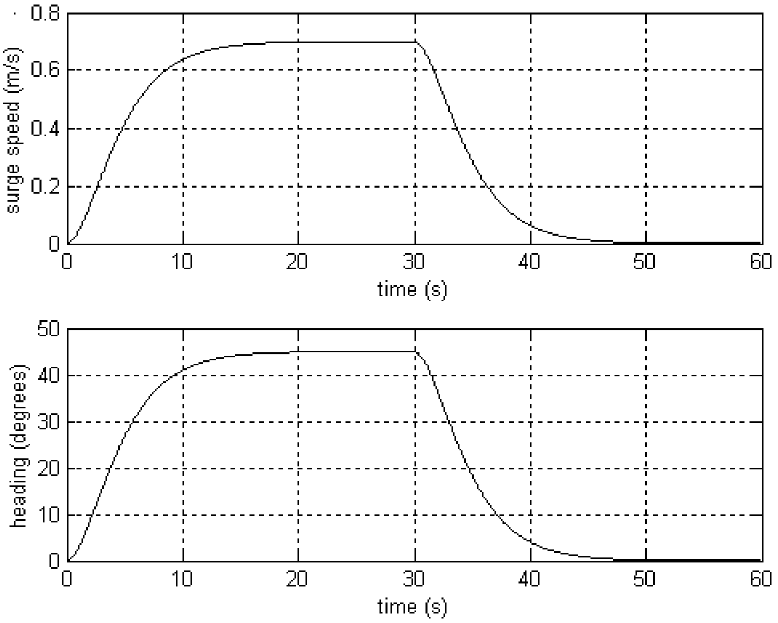


Figure 6: Desired responses to track

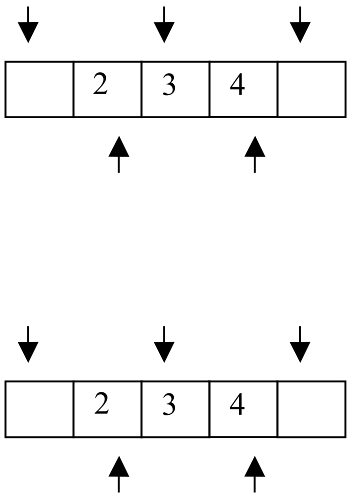


Figure 7: Assignation of values of *r* depending on the gene position

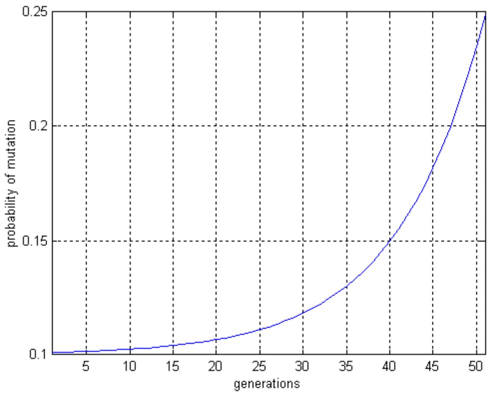


Figure 8: Exponential probability of mutation

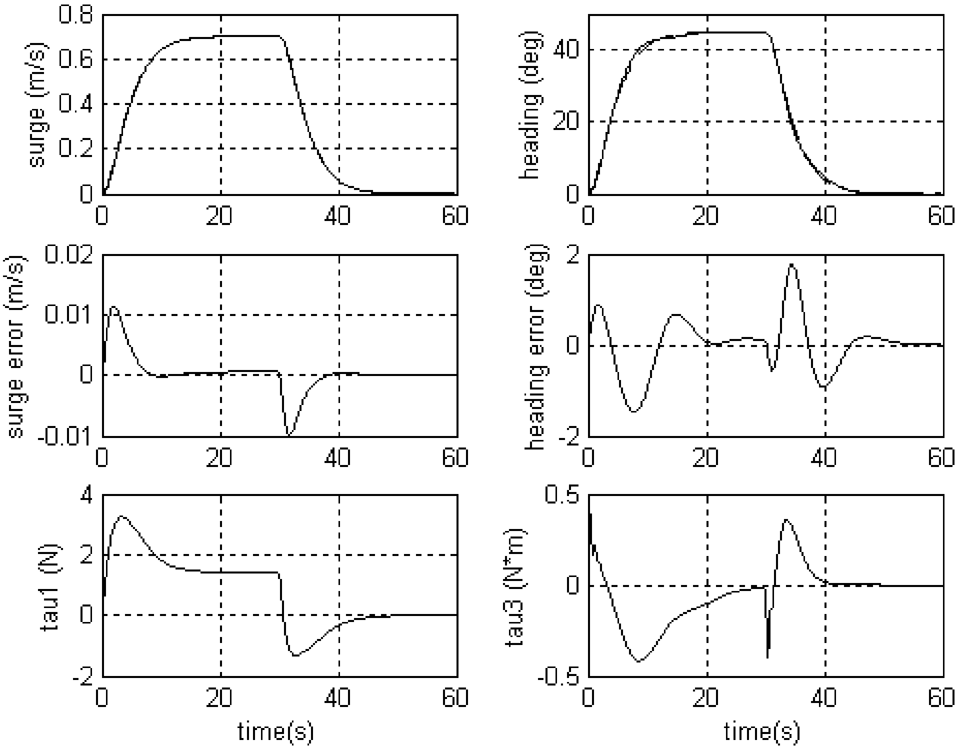


Figure 9: Simulated results of the H$_\infty$ controller GA optimised without waves

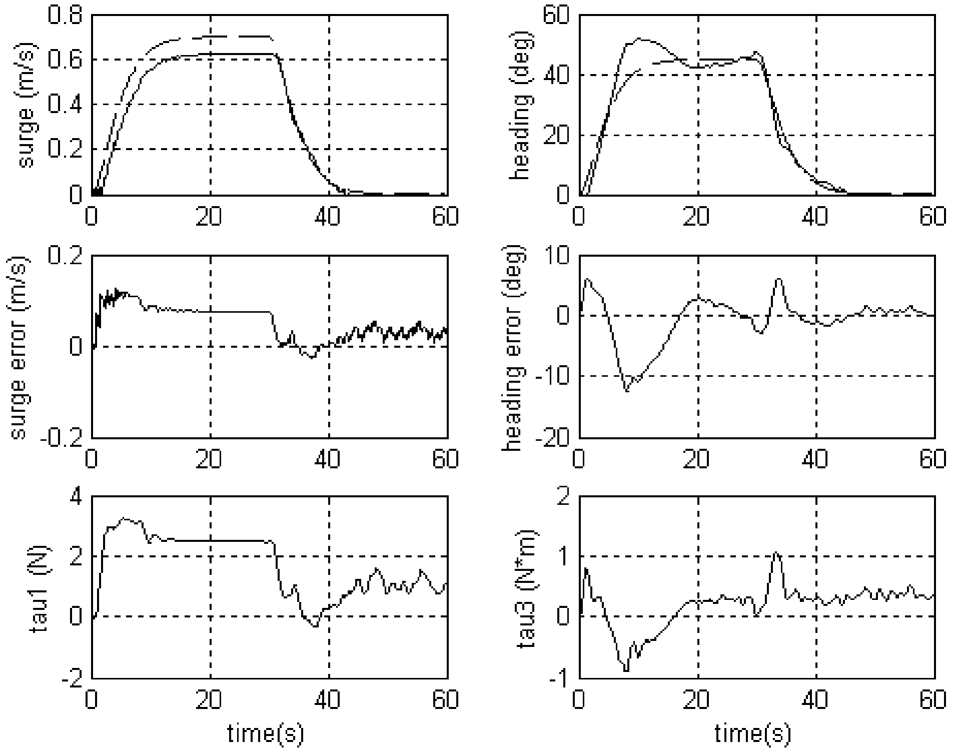


Figure 10: Simulated results of the H$_\infty$ controller GA optimised with waves

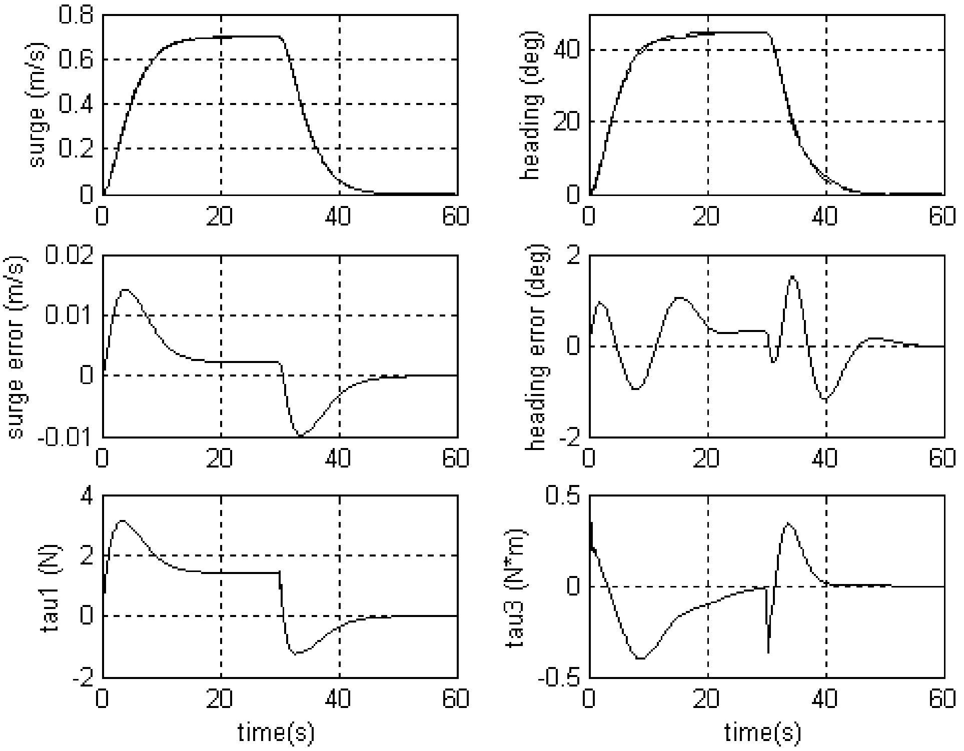


Figure 11: Simulated results of the H_∞ controller sGA optimised without waves

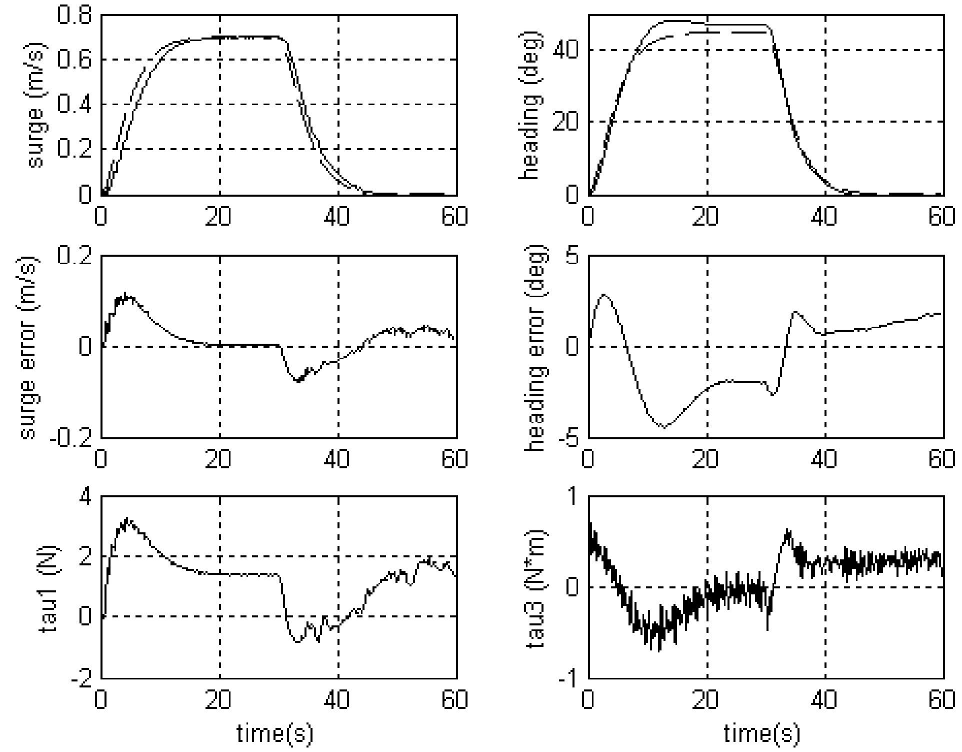


Figure 12: Simulated results of the H_∞ controller sGA optimised with waves

Table 1. Parameters to optimise for H_∞ control using GA

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r	K_{t1}	K_{t2}	K_{t3}

Table 2. Parameters to optimise for H_∞ control using sGA

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r	K_{t1}	K_{t2}	K_{t3}	w_u	w_r

Table 3. Structures represented by the control genes

w_i	W_i(s)
0	K_i’
1	$\frac{K_i^{'}}{s+\beta_i}$
2	$\frac{K_i^{'}\cdot(s+\alpha_i)}{s+\beta_i}$
3	$\frac{K_i^{'}\cdot(s+\alpha_i)}{(s+\beta_i)\cdot(s+\gamma_i)}$

Table 4. Best H_∞ results – GA optimisation without waves

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r
0.9276	98.92	0.2171	0.01999	0.2901	5.63	0.05868	0.06149

K_{t1}	K_{t2}	K_{t3}
0.03675	560.7	0.0532

Table 5. Average and standard deviation cost function results
GA optimisation without waves

Avg. cost values	StDev. cost values
2.6	2.8

Table 6. Best H_∞ results – GA optimisation with waves

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r
3.465	8.215	0.06798	16.97	0.058	0.6639	0.0587	0.01419

K_{t1}	K_{t2}	K_{t3}
0.00698	0.4741	0.0133

Table 7. Average and standard deviation cost values results
GA optimisation with waves

Avg. cost values	StDev. cost values
40.9	6.3

Table 8. Best H_∞ results – sGA optimisation without waves

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r
450	0.09	0.02749	0.09789	0.969	0.9972	0.00002	0.07061

K_{t1}	K_{t2}	K_{t3}	w_u	w_r
0.01701	0.557	0.03378	1	3

Table 9. Average and standard deviation cost value results
sGA optimisation without waves

Avg. cost values	StDev. cost values
3.1	3.4

Table 10. Best H_∞ results – sGA optimisation with waves

K_u’	α_u	β_u	γ_u	K_r’	α_r	β_r	γ_r
0.09952	3009	0.007	26.6	0.09431	3.67	0.0308	6560

K_{t1}	K_{t2}	K_{t3}	w_u	w_r
0.00398	5509	0.0402	1	2

Table 11. Average and standard deviation cost value results
sGA optimisation without waves

Avg. cost values	StDev. cost values
29.1	11.1

Table 12. GA versus sGA

	GA w/o. waves	sGA w/o. waves	GA w. waves	sGA w. waves
Cost value	1.58.	1.56	33.4	11.7
Generation of convergence	32	24	50	41