A Case Study of a Genetically Evolved Control System

Andrea Soltoggio

Prosjektoppgave

Norwegian University of Science and Technology

Faculty of Information Technology, Mathematics and Electrical Engineering

Department of Computer Science
Supervisor: Professor Keith Downing
November 2003

Acknowledgement

This paper has been written during the fall semester 2003 at the Department of Computer Science, NTNU, Trondheim.

I want to address a special thank to the people who contributed to this work with talks and discussions on the topic throughout the semester.

My supervisor at the Department of Computer Science, Professor Keith Downing, for the supervision of my work and indications on each aspect of it, from the direction of my research to the feedback on the results.
P.h.D. student Diego Federici for all the practical hints, suggestions and constant attention to the progress of my work.

Professor Thor Inge Fossen, Professor Tor Arne Johansen and Professor Tor Engebret Onshus at the Department of Engineering Cybernetics for their helpfulness and the precious advices and information about control engineering theory and practice.

I also thank my professor Pauline Haddow and all my classmates of the courses *"Modellering og representasjon av evolusjonær maskinvare"* and *"Biologisk inspirasjon - feiltoleranse og adaptivitet"* for the discussions on similar or related topics and the comparison of the results of our works.

28 November 2003
Andrea Soltoggio

Abstract

A comparison between a genetically evolved controlled system and a standard PID controlled system is carried out. In *"Method and Apparatus for Automatic Synthesis Controllers"* by John Koza et al., a method for the automatic design and synthesis of controllers by means of genetic programming is proposed; several genetically evolved controllers are presented and compared with controllers designed with existing methodologies. In this paper I illustrate a possible procedure to reproduce the results claimed by Koza with regard to the two lags plant controller. The system is compared to a PID proposed in a textbook. The control problem is analysed to identify specifications, requirements and characteristics of the controlled system. The lack of some constraints and data regarding the physics of the control problem are pointed out to be initial flaws of the method. The solution is searched in a not well defined search space: inferences are drawn in order to continue the analysis. A simulation of the genetically evolved controller and the PID is carried out with Matlab. Through the data presented, the two system are compared. A substantial difference in the problem specifications seems to be partially responsible for the manifested difference of performance in favour of the genetically evolved controller. Hence a new PID controller is tuned respecting the constraints assumed for the evolved controller and a new comparison is carried out. The similar results and the comparable or better performances obtained with newly tuned PID suggest to reject the hypothesis of absolute superiority of the genetic programming controller. Though, the topic is left open to discuss the achievement of Pareto optimal classes of solutions by the genetic programming methodology with more carefully chosen control constraints.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivations

The aim of this paper is the study of an artificially evolved control system. A genetic evolutionary computation, described in (Koza et al. 2000) and (Koza et al. 2003), designed and evolved by scratch the structure and the parameters of several controllers for different typologies of plants. The results of the experiment consist in both the automatic synthesis of a controller and the invention of a new designing tool for control system engineering.
The evolved controllers are claimed to outperform standard controllers considered the state of art in the field. In particular, a controller evolved for a second order plant is said to increase from 1.4 to 9 times the performance of an optimum controller applied to the same plant (Koza et al. 2003, col 46, 54). The controller chosen for the comparison is a PID[1] described in (Dorf, Bishop 2001, p. 597).
The outstanding performances shown by the genetically programmed controller are motivated in (Koza et al. 2003) by the use of a second derivative action, autonomously discovered and embedded in the controller by the evolutionary algorithm. In other words, the evolutionary computation rediscovered the structure of a PID controller with the addition of a second derivative and infringed the patents (Callender, Stevenson 1939) and (Jones 1942).
With respect to alternative structures proposed to improve the performance of PID control, Åström, Hägglund (1995, p. 111), state that *"PID control is sufficient for processes where the dominant dynamics are of the second order. For such processes there are no benefits gained by using a more complex controller."*
In addition, Åström, Hägglund (2001), with respect to comparisons between controllers, say: *"It is very easy to demonstrate that any controller with reasonable tuning will outperform a PID with Ziegler-Nichols tuning. Many strategies proposed can easily be eliminated if they are compared with a well-tuned PID."*
Yet, the controller used for the comparison (Dorf, Bishop 2001) is tuned to minimize an ITAE[2] performance index and it is considered optimum with respect to this index. The genetically evolved controller is said to be 2.42 times better with respect to the same performance index (Koza et al. 2003, col 46).

The previous quotations and data are clearly in deep contrast. It is

[1]PID is for Proportional, Integral and Derivative action. An insight of a a general control system and a PID controller is given in sections 1.3 and 1.6

[2]ITAE is for Integral Time-weighted Absolute Error and it is a performance index used to evaluate the performance of a controller. A description of the evaluation criteria is given in section 1.4.

evident that a better insight into the problem is needed to understand the real magnitude and consistency of the method and results claimed in (Koza et al. 2003).
Throughout this paper I analyse the control problem proposed in (Dorf, Bishop 2001) and used in (Koza et al. 2003). Specifications and constraints are analysed to give a unique interpretation of the problem. A simulation of the controllers is carried out and the data are compared. Eventually, a new PID controller is tuned to make a new comparison to the evolved controller. In the conclusion some considerations are drawn regarding the interpretation of my results and possible future work.

1.2 Evolutionary computation in control system engineering

With the increase of computational power in computers and the availability of powerful tools like parallel computer architectures, evolutionary computation has become more applicable to complex linear and nonlinear control problems.
Complex design and optimization problems, uncertainties of the plant dynamics and lack of well known procedures of synthesis, especially in nonlinear control, have been the key factors for the spread of new synthesis and tuning techniques by means of evolutionary computation.
A considerable revival of PID control in the last ten years is also caused by the introduction of automatic tuning and new procedures to automatically optimize the performance of a given system (Åström, Hägglund 2001).
Fleming, Purshouse (2002) say that *"The evolutionary algorithm is a robust search and optimization methodology that is able to cope with ill-behaved problem domains, exhibiting attributes such as multimodality, discontinuity, time-variance, randomness, and noise."*
They also explain that a control problem rarely has a single solution and it is generally apt to be suitable for a family of non-dominated solutions. *'These Pareto optimal (PO) solutions are those for which no other solution can be found which improves on a particular objective without detriment to one or more other objectives'.* (Fleming, Purshouse 2002)
Controlled systems appear to be strongly affected by classes of Pareto optimal solutions. In most of the cases one solution is preferred to an other for marginal reasons. If a control problem is specified with strict constraints, it is likely to have a unique optimum solution for the problem. Contrary, if some system specifications are left free to vary in a certain range, several solutions might be suitable for the control problem. Evolutionary computation has been proven to be an excellent tool to explore similar solutions distant from each other in the search space.

1.3 General description of a controlled system

A description of the structure and components of a controlled system is reported in the next section, in section 1.3.2 the use of Laplace transformation, the concept of transfer function in section 1.3.3 and the concept of frequency response and the use of Bode diagrams in section 1.3.4. These concepts are used throughout this paper to describe, analyse and compare the controllers.

1.3.1 Structure and components

A controlled system is composed of different elements. The plant is considered to be a part of the overall controlled system as shown in figure 1.

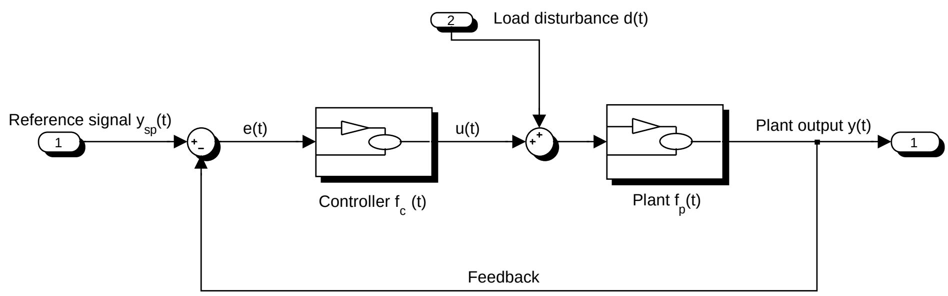

Figure 1: Simplified model of a controlled system

This system typology is defined as *feedback control* or *closed loop control.* It makes use of a feedback signal by means of which the desired value of the plant output is compared to the actual one. The calculated error is the input signal into the controller that provides the plant with an action function of the error. An alternative typology is the *open loop control* that does not have a feedback signal and offers very poor performance in comparison to the closed loop typology.
Variables commonly used to describe the dynamics of the controlled system are the reference signal or setpoint or desired value $y_{sp}(t)$, the plant output $y(t)$, the error between the plant output and the reference signal $e(t) = y_{sp}(t) - y(t)$, the control variable $u(t)$, the load disturbance $d_l(t)$ and the feedback disturbance $d_f(t)$.

The main components of a controlled system are: a *pre-filter*, a *compensator*, the *plant* and *transductors* which are divided into *actuators* and *sensors*. A drawing of a general controlled system is shown in figure 2.

The following definitions are given

> *A **transductor** is an electrical device that converts one form of energy into an other. There are two kind of transductors: actuators and sensors.*
>
> *An **actuator** is a particular transductor that transforms a signal into an action on the plant. It usually receives an input signal from the controller and provides the plant with the required action.*
>
> *A **sensor** is a particular transductor that transforms a physical measurement of a plant variable into an electrical signal. It is mainly used to obtain the feedback signal.*
>
> *A **compensator** or controller is an additional component or circuit that is inserted into a control system to compensate for a deficient performance.*
>
> *A **pre-filter** is a transfer function $G_p(s)$ that filters the input signal $Y_{sp}(s)$ prior to the calculation of the error signal.*

1.3.2 Variable representation with the Laplace transformation

Very often, as in figure 2, the variables are expressed in the frequency domain by the Laplace transformation. The operator $\mathscr{L}$ is defined by the following equation

$$\mathscr{L}[f(t)] = F(s) = \int_0^\infty f(t)e^{-st}dt \tag{1}$$

where $f(t)$ is a function of time, s a complex variable, $\mathscr{L}$ is an operational symbol and $F(s)$ is the Laplace transform of $f(t)$. The two representation are equivalent; by convention, the inverse Laplace transformation is indicated by the symbol $\mathscr{L}^{-1}$. The Laplace transform method allows to solve with algebraic equations in the complex variable s more difficult differential equations in the real variable t.

The Laplace variable s is often used as a differential operator where

$$s \equiv \frac{d}{dt} \tag{2}$$

and

$$\frac{1}{s} \equiv \int_0^t dt \tag{3}$$

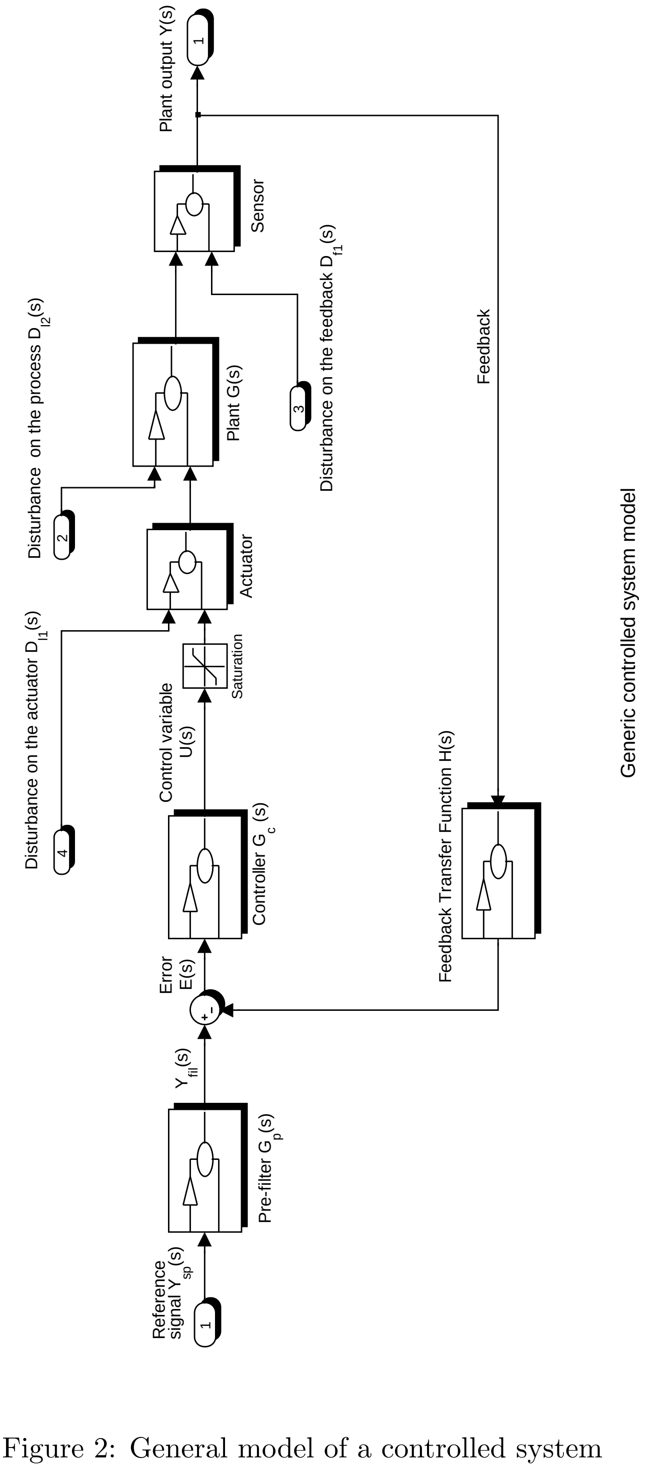

Figure 2: General model of a controlled system

1.3.3 Transfer function

Given the definition of the Laplace transformation, we can define the concept of transfer function as follows

> *The transfer function of a linear, time-invariant, differential equation system is defined as the ratio of the Laplace transform of the output (response function) to the Laplace transform of the input (driving function) under the assumption that all initial conditions are zero. (Ogata 1997, p. 55)*

As just stated, a transfer function expresses the relation between the input and the output of a system regardless of the internal dynamics. It can be defined only for a linear and time-invariant system: for this reason, if the system is nonlinear or contains a nonlinear element, the transfer function cannot be used. A generic transfer function is expressed in the form

$$G(s) = \frac{a_0 s^m + a_1 s^{m-1} + \cdots + a_{m-1} s + a_m}{b_0 s^n + b_1 s^{n-1} + \cdots + b_{n-1} s + b_n} \tag{4}$$

where m and n are the grades of the numerator and the denominator and $n \geq m$. The solutions of the numerator are called zeros of the transfer function and the solutions of the denominator are called poles. To outline the values of zeros and poles, equation 4 can be also represented in the form

$$G(s) = K \frac{(s + z_1)(s + z_2) \cdots (s + z_m)}{(s + p_1)(s + p_2) \cdots (s + p_n)} \tag{5}$$

where $z_1, \ldots, z_m$ and $p_1, \ldots, p_n$ are the zeros and poles of the transfer function and K is called *gain*.
A transfer function is also indicated as the ratio of the output signal and the input signal. For example, the transfer function from the reference signal $Y_{sp}(s)$ to the plant output $Y(s)$ is indicated by the ratio $Y(s)/Y_{sp}(s)$. In order to make the notation lighter, I use Y/Y_{sp} instead of $Y(s)/Y_{sp}(s)$. The uniqueness of the representation is guaranteed by the capital letter used for the frequency domain.

1.3.4 Frequency response and Bode diagrams

The frequency response is a representation of the system's response to sinusoidal inputs at varying frequencies. The output of a linear system to a sinusoidal input is a sinusoid of the same frequency but with a different magnitude and phase. The frequency response is defined as the magnitude and phase differences between the input and output sinusoids. Alternatively, in (Dorf, Bishop 2001, p. 407) the following definition is given

The frequency response of a system is defined as the steady-state response of the system to a sinusoidal input signal. The sinusoid is a unique input signal, and the resulting output signal for a linear system, as well as signals throughout the system, is sinusoidal in the steady state; it differs from the input waveform only in amplitude and phase angle.

The frequency response can be graphically drawn with a Bode diagram or with a Nyquist diagram. In this paper I will use Bode diagrams. The magnitude is expressed in dB where

$$dB = 20 \cdot log_{10}|G(\omega)|, \tag{6}$$

the frequency is represented on a logarithmic scale.
As an example figure 3 shows the Bode diagram of the standard PID controlled system from the input signal to the output value. The system behaves like a low-pass filter as the curves goes down for high frequencies.

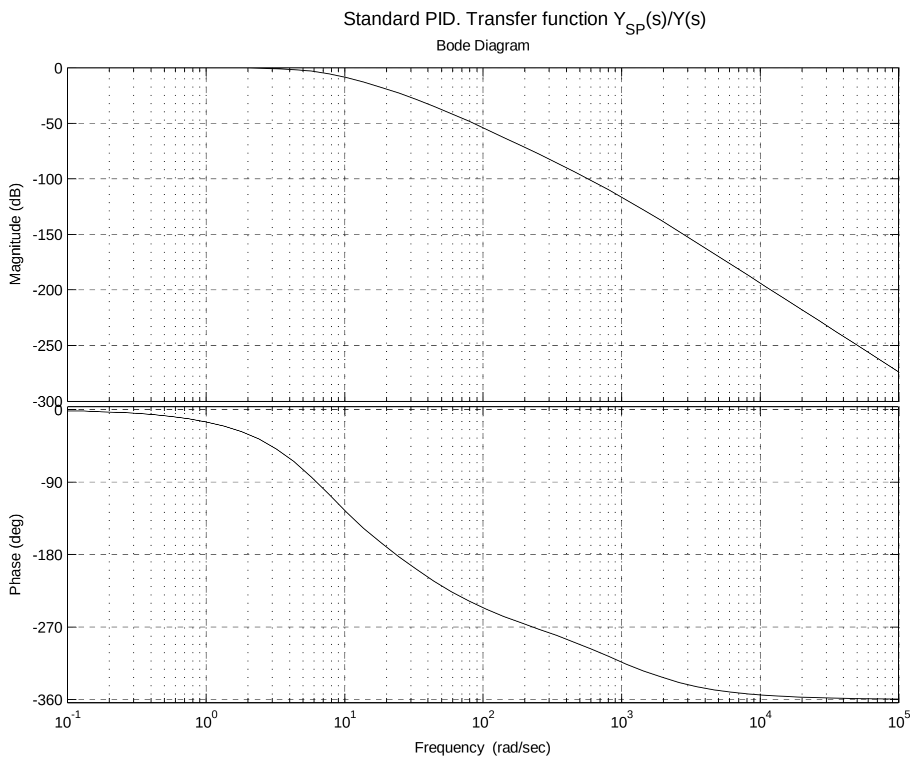

Figure 3: Bode diagram for the standard PID controlled system from Y_{sp} to Y.

1.4 Criteria for evaluating the performance of a controlled system and system constraints

The problem of the evaluation of a control system performance has always been a central topic in control system theory. Constraints and requirements play an important role in the design of a controller. With the emergence of evolutionary algorithms in control system engineering, the problem of the performance has become related to the evaluation of the fitness of a solution. For each control system, a performance index should be calculated in order to express the quality of the controller as an unique positive value. Often a performance index can become a constraint in the design of the system. Usually a performance index is a value that should be minimized in order to obtain a well tuned controller with good performance. Conversely, a constraint is a performance index which should have values within a certain range and it should not necessarily be minimized.

In (Dorf, Bishop 2001) the following definition of performance index is given

> *A performance index is a quantitative measure of the performance of a system and is chosen so that emphasis is given to the important system specification.*

The definition is generic because for different systems there might be different desired characteristics. For some systems a very fast response could be the desired target in spite of an abrupt jump of the output variable. For other systems the movement of the output variable should be as smooth as possible paying the prize of a slower response. Performance indices and requirements are expressed in the time domain or in the frequency domain.

1.4.1 Time domain indices

Several indices are proposed. I will limit the description to the indices used in this paper.
A measure of the difference between the desired value and the actual plant output, intended as the area between the two curves, is given by the IAE and ITAE indices defined as follows. The Integral of Absolute Error is expressed by

$$IAE = \int_0^T |e(t)|dt. \tag{7}$$

The Integral of Time-weighted Absolute Error is expressed by

$$ITAE = \int_0^T t|e(t)|dt. \tag{8}$$

where T is an arbitrarily chosen value of time so that the integral reaches a steady-state value. A system is considered an optimum control system when the parameters are tuned so that the index reaches a minimum value. Equation (8) expresses a measure of the value of the error between the plant output $Y(s)$ and the setpoint $Y_{sp}(s)$ weighted on time. That means that the index is penalized more heavily for errors that occur later.

To set the requirements for a controlled system, however, the IAE and ITAE performance indices just seen provide a poor description of the system dynamics. Other values can better describe the characteristics of the response to a step reference signal or disturbance. Values commonly used in the specification and verification of a controlled systems are *rising time*, *overshoot*, *settling time*, *delay time*, *steady-state error*. Given a step input to the reference signal, which is intended to bring the output value from a value of Y_1 to Y_2, we have the following definitions.

- The **delay time** is the time taken by the system output to reach 50% of Y_2-Y_1.
- The **rising time** is the time taken by the output to rise from 10% to 90% of Y_2-Y_1. Sometimes it is considered the rising time calculated from 0% to 100% of the step magnitude.
- The **overshoot** is the amount the system output response proceeds beyond the desired value (Y_2).
- The **settling time** is the time required for the system output to settle within a certain percentage of the input amplitude.
- The **steady-state error** e_∞ is the error when the time period is large and the transient response has decayed leaving the continuous response.

1.4.2 Frequency domain indices

Performance indices or system specifications can also be given in the frequency domain. A frequency domain approach to the design of control systems is also used very much. Like in the time domain, the distinction between performance indices, characteristics and system specifications is blurred. An even very limited introduction to the frequency domain approach is far beyond the aim of this paper; I will limit myself to the description of few indices and evaluation criteria.

The **bandwidth** of a system is defined as the frequency range $0 \leq \omega \leq \omega_b$ in which the magnitude of the closed loop does not drop $-3dB$ from the low frequency value. *"The bandwidth indicates the frequency where the gain*

starts to fall off from its low-frequency value" (Ogata 1997). From equation (6), a value of $-3dB$ is equivalent to an attenuation of the signal of $\sqrt{2}/2$ in the linear scale. The bandwidth gives an indication of how fast the system response is following a variation of the reference signal. A high bandwidth has the downside to amplify the noise present in the feedback line.
Noise suppression requirements are often given in the frequency domain. A noise measured on the feedback is characterized by a range of frequencies and intensities. Feedback noise is often characterized by high frequencies. A desired characteristic of the control system is a pre-specified attenuation of frequencies above a certain value.
Other important indices used in the frequency domain design are the gain margin and the phase margin. The gain margin is defined as the change in the open loop gain required to make a closed loop system unstable. The phase margin is defined as the change in open loop phase shift required to make a closed loop system unstable. For this reason a system is often designed with a minimum pre-specified phase margin and gain margin.

1.4.3 Control constraints and requirements

When designing the controller and trying to optimize the chosen performance indices, particular attention should be devoted to the respect of the constraints given for the specified problem. Each control problem has some kind of constraints due to physical limitations, technological limitations, power consumption etc. Constraints are often particular performance indices which describe a characteristic of the plant that should be maintained within a given range of values.

The most significant design constraints are described in the following list.

1. Overshoot $\leq O_{MAX}$
   Typical values of O_{MAX} are 0%, 2%, 10%. For example, the controller of an elevator should have $O_{MAX} = 0\%$ since we want to avoid that the elevator goes a bit beyond the floor and then back to the desired level. In other situation, in order to obtain a faster movement, a value of overshoot within a certain percentage can be tolerated.

2. $|u(t)| \leq u_{MAX}$
   The value of the control variable $u(t)$ is limited between $-u_{MAX}$ and u_{MAX}. This limitation is due to the limit of the actuator or the limit of the maximum strength applicable on the plant without causing damage. For example, the electrical engine of an elevator can apply a variable lifting power which has an upper limit due to the engine's characteristic. A limit could also be imposed by the strength of the steel wire that can undergo a maximum tension. Finally, a maximum

lifting power may be required in order to assure a comfortable service to the users.

3. $|\dot{u}(t)| \leq u'_{MAX}$
The change rate of the control variable is limited by the characteristic of the actuator. Using the previous example of the elevator, suppose that the target is to move the elevator as fast as possible from one floor to an other using instantly the maximum power u_{MAX} of the engine. The internal mechanical dynamics of the engine, in spite of our desire, does not allow the control variable $u(t)$ representing the applied power to go from 0 to u_{MAX} in no time. Electrical engines however can reach the maximum power in a very short time. Conversely, if the system is a room to be warmed up and the control variable is the temperature of an electric heater, it is clear that I will have to wait some time before the heater reaches its maximum temperature and the control variable its maximum value. The internal dynamic of the actuator should be carefully considered. Only after a precise study of the physics of the actuator, it is possible to decide to ignore it or not in the system model.

4. Saturated control allowed/not allowed
For most of the industrial processes, the control variable should never reach the maximum value of saturation, or if it happens, for a very short time. Several reasons justify this constraint. One is the fact that most actuators require a very high power consumption to work at top revs. Besides, the effect of *wear and tear* increases enormously when an actuator is working at the limit of its capability. The nonlinear behaviour of the controlled system caused by the saturation is also often undesired. Finally $\dot{u}(t)$ presents a discontinuity when the saturation level is reached or left. Such kind of control is called *bang-bang* control, and it is suitable only for particular control problem. However, different problems show different characteristics: for example, a heater is usually built to work properly at the maximum temperature even for a long time. In this case, this constraint is not necessary.

5. $|e_\infty| \leq e_{MAX}$
For most of the control system a steady-state error of 0 is desired in order to make the output variable get indefinitely close to the setpoint during a steady-state condition of the system. A null steady-state error is obtained using an integral action in the controller.

6. Settling time $\leq$ desired ST
A short settling time is often desired. However, a too low settling time constraint might result in an impossible solution due to the the conflict with other constraints as u_{MAX} or u'_{MAX}. If the settling time

constraint is extremely important, the problem can be modified, in order to obtain a possible solution, relaxing u_{MAX} or $u^{'}_{MAX}$. This is done using an actuator with better performance in terms of power or fast response. In some cases economical or technological reasons hinder our ambitious target and a compromise on the settling time has to be chosen.

7. $\omega_b \geq \Omega_{MIN}$
   This constraint is often used alternatively to the settling time constraint when designing the system using the frequency domain approach. In fact the bandwidth of the closed loop system gives an indication of the speed of the system. Besides, a high bandwidth allows a good compensation of the disturbance at the plant input. A lower limit to the bandwidth is imposed when a disturbance of a given intensity at the plant input has to be suppressed. The higher the bandwidth, the better is the load disturbance suppression. A high bandwidth constraint might be in contrast with u_{MAX} or $u^{'}_{MAX}$. Besides, the bandwidth has an upper limit imposed by the next constraint.

8. $\omega_b \leq \Omega_{MAX}$
   This constraint is imposed to limit the bandwidth of the closed loop. A too high bandwidth amplifies the noise on the feedback bringing the control system to poor performance. Eventually the system can become unstable or the plant be damaged. Thus, this constraint should be set after a careful consideration of the level of noise present on the feedback signal.

Other constraints used in the frequency domain design approach are the gain margin and the phase margin.

The optimization of a performance index within the constraints that might be set for a specific control problem is equivalent to find the solution of an optimization problem. The search space is a hyperspace delimited by the hyper-planes corresponding to the constraints.

The number of constraints that can be imposed by technological or environmental reasons suggests that, to design a good control system, a complete set of data should be obtained from the system to be controlled. In most of the cases measurements in situ of the noise intensity are necessary. That means, as pointed out in (Åström, Hägglund 1995), that a deep understanding of the physics behind the process is necessary in order to design a good controller, understand which phenomena should be included in the mathematical model and which phenomena can be approximated or ignored.

1.5 Introduction to the genetic programming method for synthesis of controllers

The method for the automatic synthesis of controllers proposed in (Koza et al. 2000) and fully explained in a United States Patent (Koza et al. 2003) consists in a complex setting of software, machines and choices of parameters. Given a description of the plant and the constraints of a control problem, the purpose of the method is the automatic generation of a controller that constitutes a solution to the problem. The reference documents used in this paper have the purpose to explain in details the invention proposed. Here the description will be limited to a brief overview.

The two-lag controller studied in this paper was created using a parallel computer architecture of 66 computers. Each computer contained a 533-MHz DEC Alpha microprocessor and 64 megabytes ram. The computers were connected with a 100 megabit/sec Ethernet. The processors and the host used the Linux operating system.

The method makes use of a genetic programming technique (Koza 1992) (Banzhaf et al. 1998). We provide an extremely brief background on genetic programming here. An initial population of solutions is evolved applying a Darwinian principle of natural selection. Recombination or crossover, mutation, gene duplication, gene deletions are operations inspired by biology and used to breed a population of programs. The following three steps are executed by the algorithm in order to evolve the population (Koza et al. 2000, page 131).

1. An initial population of typically random possible solutions is generated.
2. The following cycle of steps is performed until the termination criterion is met.
   (A) Each program of the population is executed to determine its value of fitness.
   (B) A new population is created by the application of the following operations. The individuals are selected with a probability function based on the fitness measure.
      - Reproduction: a copy of the selected program is inserted into the new population
      - Crossover: a new offspring program is created by recombining parts of two other selected programs
      - Mutation: a new offspring program is created by randomly mutating a randomly chosen part of the selected individual

- Architecture-altering operations: a new offspring program is created by altering the program architecture of the selected individual.

3. The individual with the best fitness is chosen to represent the solution of the current generation.

Before executing these three points of the evolutionary algorithm, the following preparatory steps have to be set (Koza et al. 2003).

1. Determine the architecture of the program trees
2. Identify the terminals
3. Identify the functions
4. Define the fitness measure
5. Choose control parameters for the run
6. Choose the termination criterion

A solution is represented by a genotype codified as a program tree. The program tree represents the block diagram of a controller. In this embodiment, the program tree is converted into a phenotype represented by a SPICE netlist. The controller is hence created as an electronic circuit. The program architecture trees makes use of automatically defined functions (ADFs)[3]. The architecture-altering operations may insert and delete ADFs to particular individual program trees in the population.

The set of terminals identifies the input and output of the blocks represented in the program tree. Some of the terminals used are REFERENCE_SIGNAL, PLANT_OUTPUT and CONTROLLER_OUTPUT.

The repertoire of functions is wide and comprehends one or more argument functions; an example set of functions is provided below, a complete list of all the functions used by the method is provided in (Koza et al. 2000, pages 134, 135) and (Koza et al. 2003, col. 48). The one-argument INVERTER negates the time domain signal represented by its argument. The one-argument DIFFERENTIATOR differentiates the input signal and represent the transfer function s (see section 1.3.3). The one-argument INTEGRATOR represents the transfer function $1/s$. The two argument LEAD

[3]An ADF is a function whose body is dynamically evolved during the run and which may be invoked by the main result-producing branch(es) or by other automatically defined function(s) (Koza et al. 2000, page 136). More information about ADFs can be found in (Koza 1992), (Koza 1994) and (Banzhaf et al. 1998)

function applies the transfer function $1+\tau s$. The two-arguments LAG function applies the transfer function $1/(1+\tau s)$. Examples of the use of these functions are clearly evident in figure 17 representing the block diagram of the GP controller.

The fitness is determined by a combination of different elements. Eight time domain based ITAE indices are calculated for different combinations of the plant parameters[4] and step reference signals. Two step reference signal of $1Volts$ and $1\mu Volts$ are used. One time domain based element measuring the controller's stability when faced with an extreme spiked reference signal and one element represented by a frequency domain based index are added. The values obtained by the different indices are added to obtain a single value. The smaller the fitness, the better. The time domain analysis was carried out in an interval of time of 9.6 seconds. The ITAE index was used in the following modified form

$$\int_{t=0}^{t=9.6} t|e(t)|A(e(t))Bdt \tag{9}$$

where B represents the inverse of the step reference signal so that the $1Volts$ and $1\mu Volts$ signal have the same weight. The function A weights all variations below the reference signal and up to 2% above the reference signal by a factor or 1.0 and heavily penalizes overshoots over 2% by a factor 10.0. A discrete approximation to the integral was used by considering 120 80-millisecond time steps between $= 0$ to $t = 9.6$ seconds.

The control parameters for the run comprehend the population size, the maximum size of the program tree and the percentage of the different genetic operations applied. The population size was 66.000, each node having 1000 individuals. Generations are run asynchronous on each node. For each generation four boatloads of emigrants are dispatched to each of the four adjacent processing nodes. The boatloads have 20 emigrants chosen on a fitness basis. The maximum size of the program tree was 150 points and 100 points for each automatically defined function. The percentages for the genetic operations on each generation up to and including generation 5 were 78% one-offspring crossover, 10% reproductions, 1% mutations, 5% subroutine creations, 5% subroutine duplications and 1% subroutine deletions. After generation 5 the percentages were 86% one-offspring crossover, 10% reproductions, 1% mutations, 1% subroutine creations, 1% subroutine duplications and 1% subroutine deletions.

[4]Section 2.2.1 will explain that for a robust control design the plant parameters are assumed to be varying in a specified range.

The termination criterion was not set since the run was manually monitored and manually terminated when the fitness of many successive best-of-generation individuals appeared to have reached a plateau.

Each individual of the population required an average of 4.8 seconds to be evaluated. The best individual from generation 32, represented in figure 17, was produced after evaluating $2.57 \cdot 10^6$ individuals (66,000 times 33). The necessary time for the computation was 44.5 hours with an expenditure of $5.6 \cdot 10^{15}$ computer cycles.

1.6 PID control systems: introduction and representation

A general closed loop controller (figure 1) implements the control variable $u(t)$ as a function of the error $e(t)$. A PID controller bases its action intensity on the sum of three values derived from the error: a proportional action, a derivative action and an integral action. The weights of the three different actions are the parameters of a PID controller. The tuning of a PID controller consists in the search of the parameters which can optimize a pre-specified performance index within some particular constraints.
The typology of PID controller is widely spread in industry where more than 90% of all control loops are PID (Åström, Hägglund 2001).

The action taken by a PID controller on the plant can be expressed in the time domain as

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \tag{10}$$

In the frequency domain, a PID controller can be expressed by the transfer function

$$G_{PID}(s) = K\left(1 + \frac{1}{sT_i} + sT_d\right) \tag{11}$$

Equation (11) is called *standard* or *non-interacting form* (Åström, Hägglund 1995).
Equation (11) can also be written to highlight the three PID parameters as

$$G_{PID}(s) = \frac{K_3 s^2 + K_1 s + K_2}{s} \tag{12}$$

where K_1, K_2 and K_3 are the weights of the proportional, integral and derivative actions.
An alternative representation is expressed by

$$G'_{PID}(s) = K'\Big(1 + \frac{1}{sT'_i}\Big)(1 + sT'_d) \tag{13}$$

and is called *serial* or *interacting form*. The reason is that the controller (11) has the different components assembled with a parallel structure. Thus, the proportional, integral and derivative actions do not influence each other. In the interacting controller (13) the derivative action influences the proportional and integrative actions through a serial connection. Every interacting representation has an equivalent non-interacting representation, but not vice versa. For this reason the non-interacting form is considered more general (Åström, Hägglund 1995). I implemented the standard PID controller using a non-interacting model form.

2 Methods and problem analysis

2.1 Methods

The genetically evolved controller that I have studied is proposed in (Koza et al. 2003) and has been evolved to control a second order plant. The controller representation makes use of two transfer functions for the pre-filter and the compensator.
For the model analysis and simulation I used Matlab, the Matlab Control Toolbox and Simulink.
The choice of the MathWorks Inc. software is due to the completeness of the available tools for control system analysis. The control system expressed with transfer functions or block diagrams in Matlab is also independent of the particular implementation. The SPICE netlist used in (Koza et al. 2003) to represent the controller and the plant makes use of specific electronic components that imply a pre-specified form of implementation. Besides, the SPICE netlist of the controller is not given in (Koza et al. 2003).

The analysis is based both on the time domain and the frequency domain while in (Koza et al. 2003) is base only on the time domain.
The sampling time for the system was set to $1msec$ after the verification that a smaller sampling time was not increasing the accuracy of the simulation. In some cases though, where the system is particularly sensitive to the numerical approximation, the sampling time has been decreased to $100\mu sec$.
The performance index to be minimized is the ITAE index for the step reference signal. With respect to the disturbance, I measured the output signal reporting the maximum value and an IAE index. The choice of IAE instead of ITAE has the aim of measuring the error without weighting it on time. While it is reasonable to consider more relevant a later error than an initial error when applying a reference step signal, I considered the time weight not significant with a noise disturbance, especially if the measurement is done on an interval of time with a continuous disturbance applied to the plant.
The intensity of the control variable is reported as a main characteristic of the controller.
The frequency domain analysis makes use of Bode diagrams and bandwidth measurements. The system is verified with respect to the transfer function from the reference signal to the output, the transfer function from the load disturbance to the output and the transfer function from the feedback to the output. The three Bode diagrams related to the functions represent respectively the system response from the reference signal to the output, the system attenuation of load disturbance and the system sensitivity to the feedback noise. A mathematical description of the plant and controllers in sections 2.2.1, 2.2.2 and 2.2.3 provides the data for the simulation.
An additional component, a filter on the derivative had to be employed. A

detailed explanation is provided in section 2.4. The analysis and simulation are presented initially for the standard PID controller. The results are explained with details and graphics. The successive analysis and simulation of the GP controller follows the same steps: for this reason the data and graphics are reported with more concise explanations without losing level of accuracy.
In the comparison that follows, the most significant data are fetched from the two previous sections and pointed out with considerations and graphical representations.
Finally, a new controller with a PID structure and respecting the same constraints in (Koza et al. 2003) is tuned and compared with the GP controller.

2.2 Mathematical description of the plant and controllers

2.2.1 The plant chosen for the comparison

Dorf, Bishop (2001, example 12.9, p. 699) show a design method for a robust PID-controlled system. The plant to be controlled is expressed by the transfer function

$$G(s) = \frac{K}{(\tau s + 1)^2} \tag{14}$$

For a robust system design the two parameters K and τ are considered variable in the intervals $1 \leq K \leq 2$ and $0.5 \leq \tau \leq 1$.
The constraints for the design will be described in section 2.3. This plant is used for both the standard PID and the GP evolved control systems.

2.2.2 The standard PID control system and tuning method

The controller is designed through several steps in order to determine the three parameters of a PID controller and the pre-filter. The design method uses the ITAE performance index. For a general closed-loop transfer function as

$$T(s) = \frac{Y(s)}{R(s)} = \frac{b_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0} \tag{15}$$

Dorf, Bishop (2001, p. 252) give the coefficients that minimize the ITAE performance criterion for a step input. The parameter ω_n in table 1 is set depending on the required settling time and the available power to control the system. A rapid dynamic and a short settling time will require a large ω_n and stronger control action on the system. Since the ITAE performance index is minimized, the control is considered optimum with respect to that criterion. The controller composed by a pre-filter and a compensator is expressed by the following transfer functions

$s+\omega_n$
$s^2+1.4\omega_n s+\omega_n^2$
$s^3+1.75\omega_n s^2+2.15\omega_n^2+\omega_n^3$
$s^4+2.1\omega_n s^3+3.4\omega_n^2 s^2+\omega_n^3 s+\omega_n^4$
$s^5+2.8\omega_n s^4+5.0\omega_n^2 s^3+\omega_n^3 s^2+\omega_n^4 s+\omega_n^5$
$s^6+3.25\omega_n s^5+6.60\omega_n^2 s^4+\omega_n^3 s^3+\omega_n^4 s^2+\omega_n^5 s+\omega_n^6$

Table 1: The Optimum Coefficients of T(s) Based on the ITAE Criterion for a Step input.

$$G_p(s) = \frac{42.67}{s^2+11.38s+42.67} \tag{16}$$

$$G_c(s) = \frac{12\cdot(s^2+11.38s+42.67)}{s} \tag{17}$$

The tuning method through which equations (16) and (17) are obtained starts with the choice of the parameter ω_n of table 1. The value of ω_n that can be chosen will be limited by considering the maximum allowable $u(t)$, where $u(t)$ is the output of the controller. Table 2 shows the effect of ω_n on the intensity of the control variable and settling time (Dorf, Bishop 2001).

A generic PID controller is expressed as

$$G_c(s) = \frac{K_3 s^2+K_1 s+K_2}{s}. \tag{18}$$

Hence, the transfer function of the system (for $K=1$, $\tau=1$) without pre-filtering is[5]

$$\begin{aligned} T_1(s) &= \frac{Y(s)}{Y_{sp}(s)} = \frac{G_c G(s)}{1+G_c G(s)} = \\ &= \frac{K_3 s^2+K_1 s+K_2}{s^3+(2+K_3)s^2+(1+K_1)s+K_2} \end{aligned} \tag{19}$$

From equation (19) and table 1 we have

[5] A closed loop transfer function $G(s)$ is related to the open loop transfer function $L(s)$ by the relation $G(s)=\frac{L(s)}{1+Ls}$.

ω_n	10	20	40
$u(t)$ maximum for $R(s)=1/s$	35	135	550
Settling time (sec)	0.9	0.5	0.3

Table 2: Maximum Value of Plant Input

$$\begin{aligned} K_3 &= 1.75\omega_n - 2 \\ K_1 &= 2.15\omega_n^2 - 1 \\ K_2 &= \omega_n^3 \end{aligned} \tag{20}$$

From the parameters (20) and equation (18), the transfer function of the compensator (17) is obtained. Finally the equation of the pre-filter is obtain so that the overall transfer function has the same form as equation (15).

The overall controlled system is expressed by the transfer function

$$\frac{Y(s)}{R(s)} = T(s) = \frac{512}{s^3 + 14s^2 + 137.6s + 512} \tag{21}$$

Equation (21) shows $T(s)$ for the values of the parameters $K = 1, \tau = 1$. The system can be specified by a Matlab script that, given the parameter K and τ, calculates the transfer function T(s). Appendix B.1.1 reports the script used for the purpose. $T(s)$ is expressed as $N(s)/D(s)$ where $N(s)$ and $D(s)$ are vectors representing the numerator and denumerator of the transfer function.

An alternative way to represent and simulate a PID controlled system is using *Simulink*.

2.2.3 The genetically evolved control system

In (Koza et al. 2003) the plant to be controlled is the one just described in section 2.2.1. The authors of the invention have chosen that plant on purpose to be able to compare their GP controller to the textbook controller. The pre-filter proposed is[6]

$$\begin{aligned} &G_p(s) = \\ &= \frac{1(1 + 0.1262s)(1 + 0.2029s)}{(1 + 0.03851s)(1 + 0.05146s)(1 + 0.08375s)(1 + 0.1561s)(1 + 0.1680s)} \end{aligned} \tag{22}$$

In order to emphasize the gain and the zeros/poles values, the transfer function (22) can be rewritten as follows

$$\begin{aligned} &G_p(s) = \\ &= 5883.01 \cdot \frac{(s + 7.9239)(s + 4.9285)}{(s + 25.9673)(s + 19.4326)(s + 11.9403)(s + 6.4061)(s + 5.9524)} \end{aligned} \tag{23}$$

[6]The equation is reported modified according to the presumed printing errors found in the original paper. See appendix A.

The transfer function for the proposed compensator is

$$G_c(s) = \frac{7497.05 + 1300.63s + 71.2511s^2 + 1.2426s^3}{s} \tag{24}$$

The values of the numerator of the previous function correspond to the integral, proportional, derivative and second derivative actions.
The Matlab script reported in section B.1.2 is used to calculate, from equation (22), (24) and the values for K and τ, the transfer functions $Y(s)/Y_{sp}(s)$ and $Y(s)/D_f(s)$.
Contrary to the standard PID, for the GP controller the transfer function of the overall system does not provide a correct representation of the system in all the conditions. In fact, for the plant parameters $K = 1$ and $\tau = 1$, the saturation at the plant input introduces a strong nonlinearity in the system dynamics. In this situation, as explained in section 1.3.3, the transfer function can not be used to represent the relation between the input and the output. The Simulink model provides a valid alternative to test the system.

2.3 Control constraints and requirements for the analysed controlled systems

The controller described in (Dorf, Bishop 2001, p. 697) is supposed to control a temperature. The performance index to be minimized is the ITAE index. The constraints given are an overshoot less than 4% and a settling time less than 2 seconds. Several other constraints like u_{MAX}, u'_{MAX} and Ω_{MAX} are implicitly considered and discernible from the final system. As it will be shown from the model simulation, for example, $u_{MAX} = 24.6Volts$ and there is not any nonlinear effect due to saturation. Other constraints like the noise suppression on the feedback are simply not mentioned.
Hence, the problem is not completely specified and the controller is not completely described.
However, it is important to notice that the purpose of the exercise in (Dorf, Bishop 2001, p. 697) is to show a tuning method that is far from the implementation of a real and complete control system. In other words, the method explain how to tune the PID parameters in order to minimize the ITAE index and gives an extremely simplified example of a controller implementation. Thus, the results that can be obtained from the simulation of the system are qualitative and do not provide a good base for a comparison.

The target of the design process in (Koza et al. 2003) is more ambitious. The genetic computation evolved a controller by means of a simulation-based fitness calculation. The target is not the description of a tuning system, but the real design of a complete controller, from the structure to the choice of the parameters. The constraints imposed for the simulation are (Koza et al. 2003, col. 47)

- Overshoot: $O_{max} \leq 2\%$
- Control variable: $|u(t)| \leq u_{MAX} = 40Volts$.
- Limited bandwidth for the closed loop Y/Y_{sp}.
  "The second constraint is that the closed loop frequency response of the system is below a 40dB per decade low-pass curve whose corner is at 100Hz". (Koza et al. 2003, col. 47). That means that the maximum bandwidth allowed Ω_{MAX} is $401 rad/sec$ and after this value the attenuation is $40dB$ per decade[7]. In (Koza et al. 2003) it is also said that *"This bandwidth limitation reflects the desirability of limiting the effect of high frequency noise in the reference input".*

From this list of constraints, some important considerations have to be done.

The bandwidth of the closed loop from the reference signal Y_{sp} to the plant output (Y/Y_{sp}) is different from the bandwidth of the closed loop from the filtered signal to the output (Y/Y_{fil}). This is due to the presence of a pre-filter between the reference signal and the point of the error calculation. If we consider figure 1 the two transfer functions just mentioned are the same function. A limit on the bandwidth does not only limit the noise at the reference signal, but even the noise on the feedback signal, because the point of application is the same.
If we now consider figure 2, when a pre-filter is introduced, the point where the reference signal is applied differs from the point where a feedback noise is applied. As a consequence, the limitation of the bandwidth for Y/Y_{fil} becomes largely independent from the limitation of the bandwidth on the reference signal.
That means that the constraint regarding the bandwidth limit for the closed loop $Y(s)/Y_{fil}(s)$ is missing.
The consequence is that the bandwidth can be as high as desired and the load disturbance can be suppressed up to any desired level. Thus, even before the simulation we can draw a hypothesis on the 9 times better performance claimed for the GP in (Koza et al. 2003). By simply increasing the value of the bandwidth, the disturbance suppression can be increased up to any desired level without any theoretical limit.
On the other hand, this implies the necessity of a feedback signal free of noise. Unfortunately, as a matter of fact, there is not any physical measurement tha can be considered free of noise or uncertainty. This makes any controller designed without a limit constraint on the bandwidth a theoretical controller without chances to be implemented.
The reason why the GP controller shows a disturbance attenuation only

[7]Using the conversion 1Hz = 6.28rad/sec, the reference low-pass filter transfer function is expressed by $G(s) = \frac{628^2}{(s+628)^2}$

9 times better than the PID is probably due to an implicit and unknown noise introduced by the numerical approximation: the digital simulation of an analog system, however precise, implies a numerical approximation which eventually can be seen as noise in the signal. This point will be better explained later in section 2.4.
Another important constraint missing is the maximum value of $\dot{u}(t)$. This allows for situation where the dynamic of the actuator is extremely fast in comparison to the dynamic of the system. In these cases $\dot{u}(t)$ can vary up to any extreme largely value.
The last important characteristic not specified is the use of saturated control. As it will be shown in the simulation, the controller brings the control variable to the saturation in order to reach a faster step response. This characteristic makes the controller unsuitable for most industrial applications where the saturation value should never be reached during the normal control activity.

2.4 Implementing the derivative function

The transfer function of a derivative is often expressed as $G(s) = s$, as defined with equation (2) and used in the previous two sections. However, a transfer function can not have more zeros than poles. This is due to the impossibility for the output to predict the future signal to the input. Thus, the mathematical operator s is used as an approximation of the real function

$$G(s) = \frac{s}{\tau_d s + 1}. \tag{25}$$

In some text books a PID controller is expressed directly using equation (25) (Nachtigal 1990). Equation (25) can be written as the series of a derivative and a low-pass filter

$$G(s) = s \cdot \frac{1}{\tau_d s + 1} \tag{26}$$

When the constant time τ_d is very small in comparison to the constant time of the process, the function (25) can be considered a good approximation of a derivative function for frequencies below $\tau^{-1} rad/sec$ (Nachtigal 1990).

The choice of τ_d is the result of an accurate compromise: a very small τ_d implies a large bandwidth for the low-pass filter and it takes to a very good derivative function with a small loss in phase margin. It has the downside to be very sensitive to noise, though. For this reason, this approach is actually

used very seldom.
Increasing the value of τ_d gives a more narrow bandwidth with the capacity of suppressing high frequency noise. The downside is that the phase margin decreases and so the performance of the system. Eventually a too narrow low-pass filter would decrease the phase margin to a dangerous level of the stability threshold. Figure 4 shows the effect on the stability of the system when a too narrow low-pass filter is applied ($\tau_d = 10^{-1}$). A further increment of τ_d makes the system unstable with oscillations that are getting wider with time: for this reason a robust controlled system is often designed with a minimum phase margin requirement.

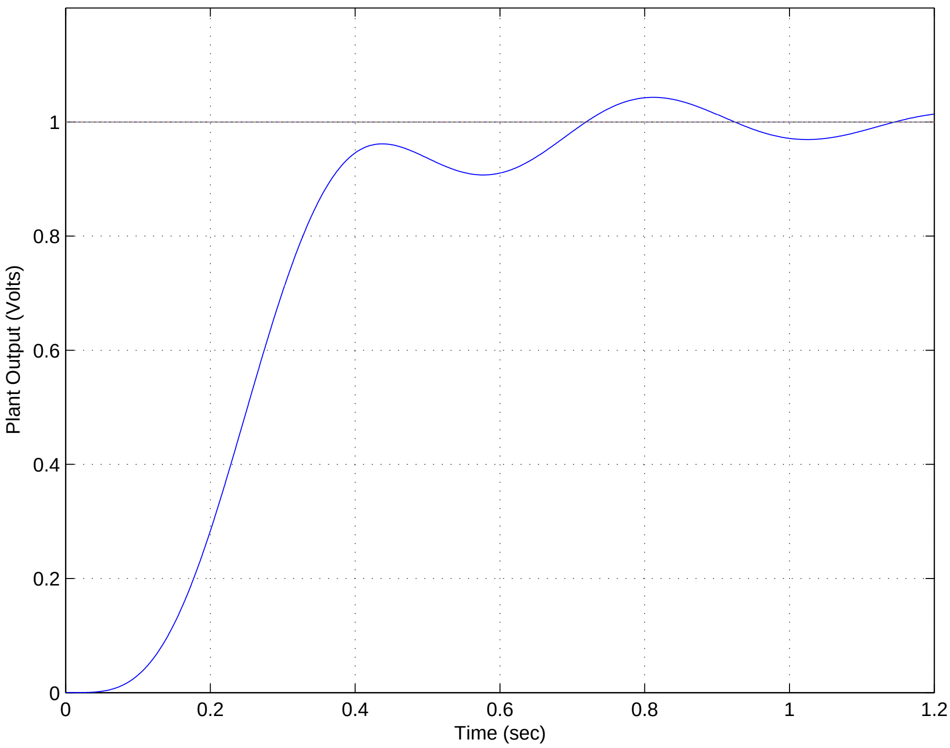

Figure 4: Effect of a too narrow low-pass filter on the PID controlled plant stability

Both systems that I am analysing are supposed to be free of noise on the feedback (see section 2.3). For this reason there is no need to implement a narrow low-pass filter. However, the numerical simulation of a continuous system implies approximation. This has the very same effect of noise. The derivative control variable and especially the double derivative control variable of the GP controller feel the effect of such noise. Figure 5 shows the effect on the derivative control variable when $\tau_d = 10^{-4}$. A derivative operation applied on signal of figure 5, as the GP controller actually does, is simply not feasible.

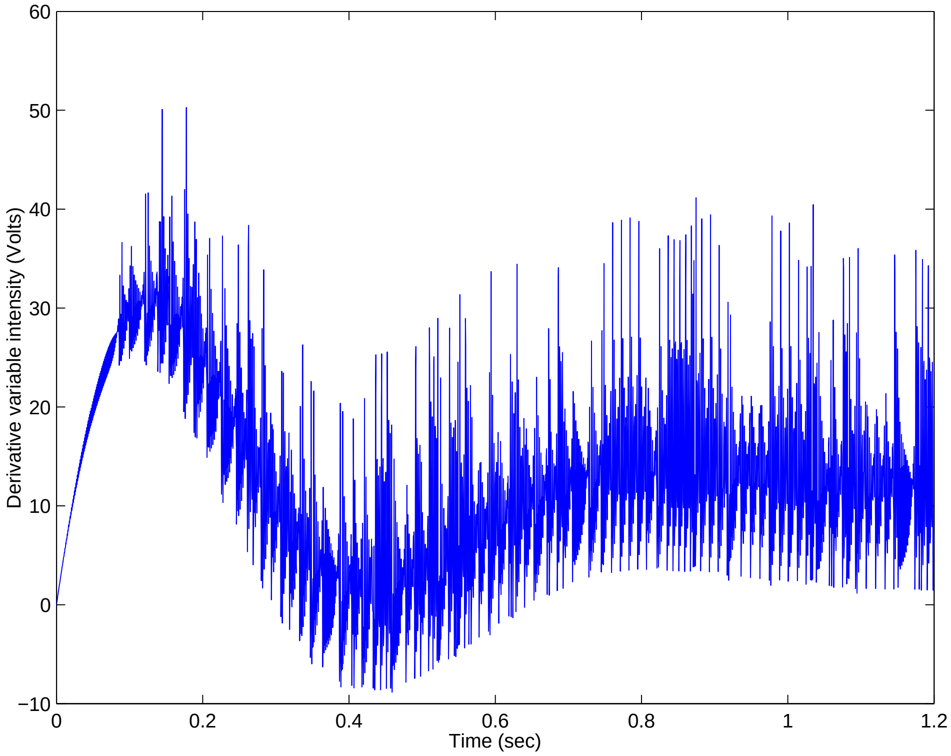

Figure 5: Effect of numerical approximation on the derivative control variable. Sampling step: $0.001 sec$; Solver: $ode45(Dormand - Prince)$

A solution to the problem can be found increasing the sampling of the simulation or using a stiff solver as $ode15s(stiff/NDF)$. Yet, this effort does not help to make the simulation realistic: in real problems the feedback signal is affected by disturbance. For this reason, while implementing the controller, the intensity and characteristic of the noise have to be taken into account. Given the specification of free-of-noise feedback signal and therefore the requirement for an accurate derivative function combined with the necessity of filtering the numerical approximation, I chose the following compromise:

I implemented the derivative function as

$$G_d(s) = \frac{\tau_d^{-1}}{s + \tau_d^{-1}} \tag{27}$$

where $\tau_d^{-1} = 1000$. This value offers a very good approximation of a derivative function[8] and it is not affected by the numerical approximation. Figure 6 shows the Bode and phase diagram of equation (27).

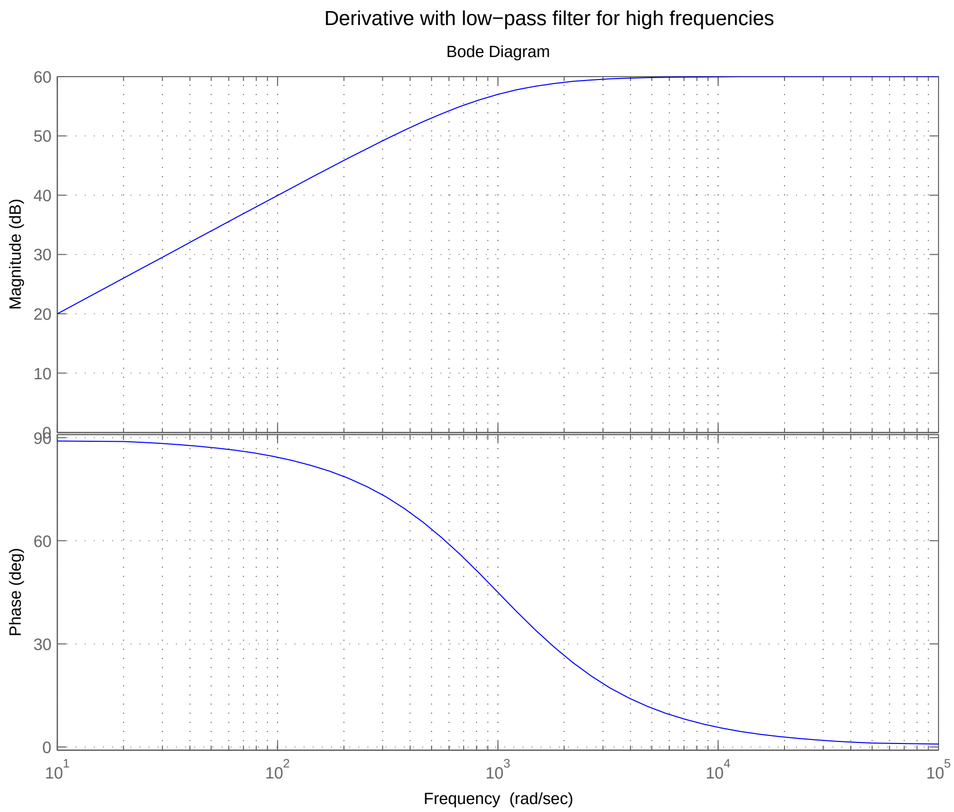

Figure 6: Bode and phase diagram of the derivative plus low-pass filter transfer function (27)

Simulink offers a derivative block du/dt which is intended to provide the derivative of the input at the output. During the simulation of both the models with the derivative blocks, I encountered the same simulation problems described above due to numerical approximation behaving like noise. For the derivative block, unlike for other blocks, the solver does not take smaller steps when the input changes rapidly (The MathWorks Inc. 2002, p. 2-69). The simulation was only possible with a stiff resolution method (ode15s) and produced in some cases wrong or impossible results.

[8] A simulation of a controller with a double derivative as shown later in figure 18 has provided a very accurate second derivative function with zero values few μseconds delayed from the peak of the first derivative.

3 Simulation and comparison

3.1 PID representation and simulation

Figure 7 shows the model used for the PID simulation. The saturation block present in the model is a nonlinear element and cannot be described by a transfer function. For this reason, the transfer function $T(s)$ is not equivalent to the system of figure 11. However, as the simulation will show, the control variable $u(t)$ of the PID controller never reaches the saturation value and the Simulink model operates within a linear dynamic.

3.1.1 Response to a unitary step reference signal

The response to a step reference signal is obtained with the command *step* of the Matlab Control Toolbox. A second Matlab function was created for the analysis of the data. The function reported in appendix B.2.1 analyses the result of the step function and gives the values of overshoot, delay time, rise time, settling time and ITAE.
The data obtained by the step command and the Simulink simulation are reported in table 3.

The simulations with the two different methods have resulted in the same data. This result confirms the three following considerations. The system behaves as linear in spite of the saturation at the plant input. The low-pass filter, introduced in the passage from a mathematical to a physical description of the model, does not affect the system's performance[9]. The equal results confirm the correctness of both s models. Figure 8 shows the results of the simulation for two configurations of the parameters.

3.1.2 Response to a unitary step disturbance

A disturbance $D_l(s)$ (see figure 1) might affect the plant dynamic and move away the plant output from the desired value. A good controller is able to minimize the effect of such disturbance.

[9]The step command has been tried on the original transfer function $T(s)$ and on the modified one with the filter.

	$K=1$ $\tau=1$	$K=1$ $\tau=0.5$	$K=2$ $\tau=1$	$K=2$ $\tau=0.5$
Overshoot	2%	0.3%	0%	0.35%
Delay time (ms)	261	243	245	241
Rise time (ms)	290	391	352	407
Settling time (ms)	943	629	629	661
ITAE ($mVolts \cdot sec^2$)	48.6	49.0	46.2	49.9

Table 3: Response to the unitary step reference signal for the standard PID.

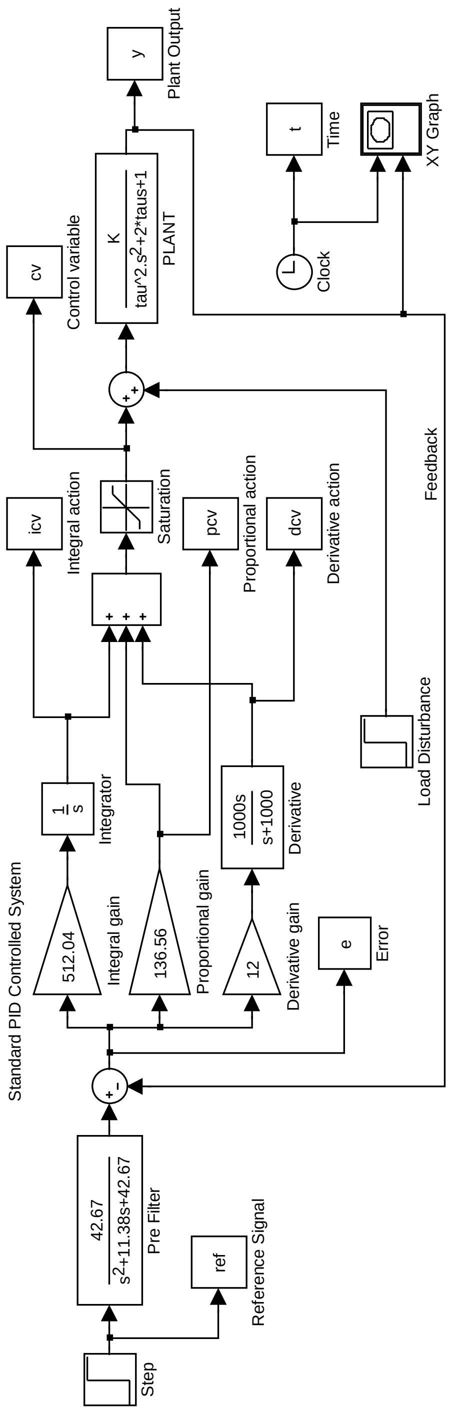

Figure 7: Simulink model for the standard PID

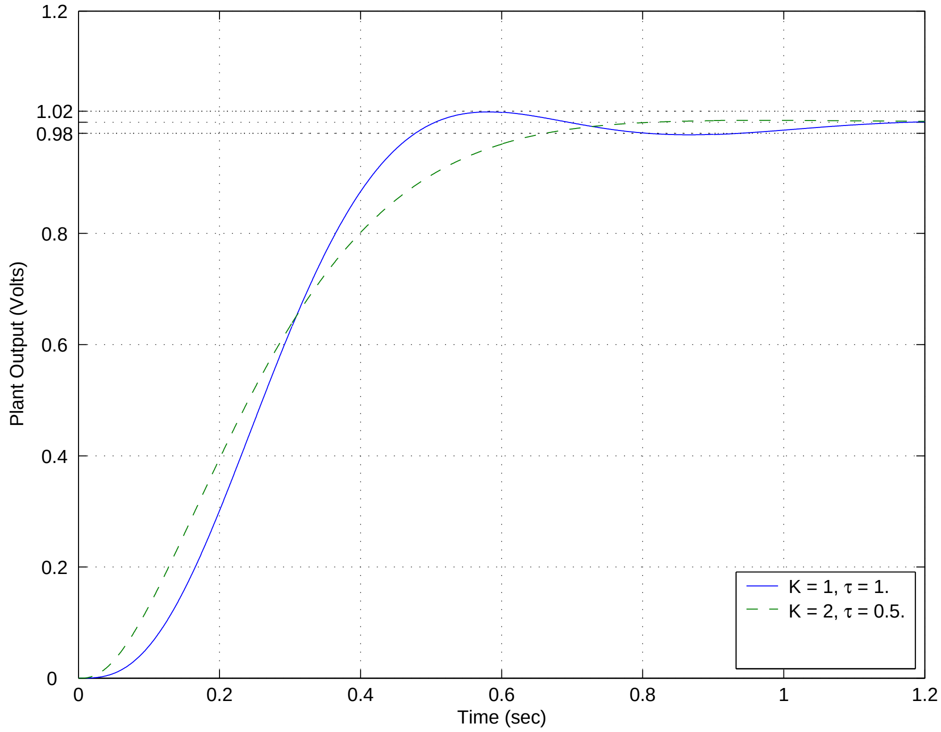

Figure 8: PID response to a unitary step reference signal. $K = 1, \tau = 1$ and $K = 2, \tau = 0.5$.

Given the transfer function from the disturbance $D_l(s)$ to the output $Y(s)$, the command *step* measures the response to a unitary step disturbance. The disturbance response is also simulated with the Simulink model of the PID controller introducing a step signal at the plant input (see figure 7).
Table 4 reports the data obtained.

	$K = 1$ $\tau = 1$	$K = 1$ $\tau = 0.5$	$K = 2$ $\tau = 1$	$K = 2$ $\tau = 0.5$
Maximum deviation $(mVolts)$	6.4	6.0	6.0	5.8
IAE $(\mu Volts \cdot sec)$	197.0	218.1	270	316.4

Table 4: Response to a unitary step disturbance for the standard PID control system

Figure 9 shows the plant output when a unitary step disturbance is applied at the plant input. From table 4 and figure 9 it is evident how the feedback provides a powerful way to suppress a disturbance. The capacity to suppress a load disturbance is directly related to the bandwidth of the closed loop from the point of the error calculation to the output.

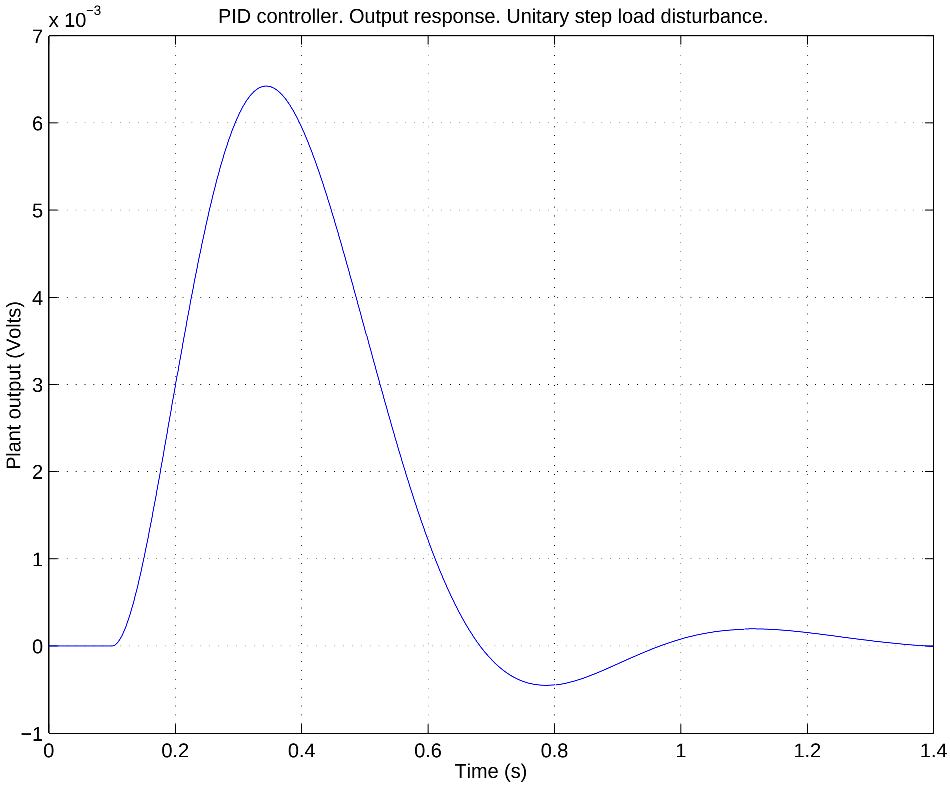

Figure 9: PID response to a unitary step load disturbance. $K = 1, \tau = 1$.

3.1.3 Internal variables dynamics

The Simulink model has the advantage to show the structure of the controller that, in this case, it is implemented using a non-interacting form (see section 1.6). Through the model, it is possible to verify and analyse the value of the control variables for a better understanding of the system dynamics. Figure 10 shows the behaviour of the internal variables when a step is applied to the reference signal.

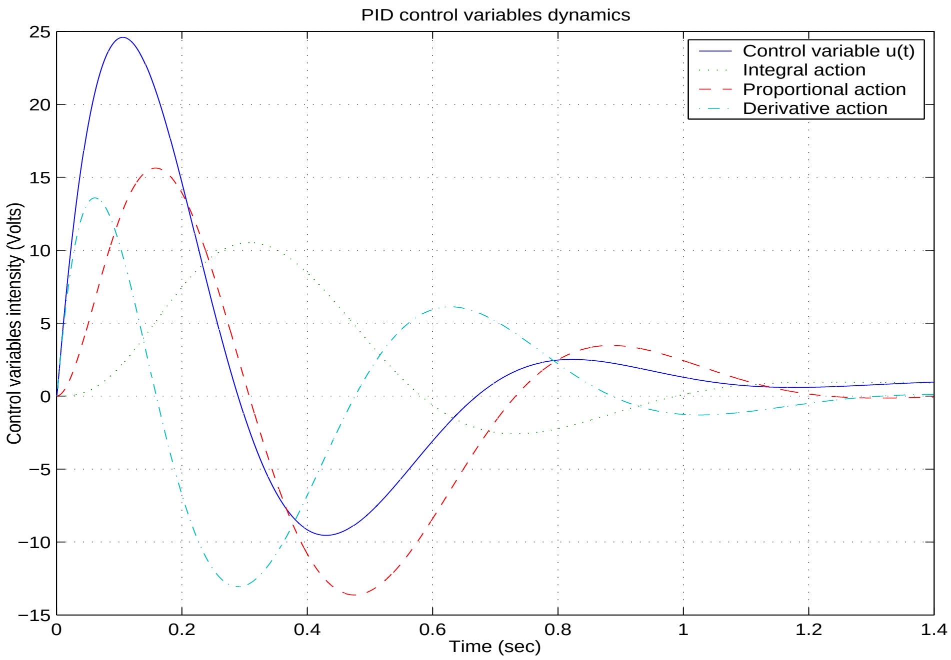

Figure 10: Internal variables of the PID controller. Response to a step reference signal. Plant parameters: $K = 1, \tau = 1$.

For the different values of the parameters K and τ, table 5 shows the maximum value of the control variable, the maximum value of its derivative and an indication of the actuator usage in percentage[10].

[10]Being not known the nature of the actuator is not possible to compute a power consumption. The actuator usage percentage gives the ratio between the actual intensity of the control variable and the maximum possible intensity calculated from the step time to the the settling time.

	$K=1$ $\tau=1$	$K=1$ $\tau=0.5$	$K=2$ $\tau=1$	$K=2$ $\tau=0.5$
Maximum value of $u(t)$ $(Volts)$	24.6	8.6	14.5	4.65
Maximum value of $\dot{u}(t)$ $(Volts/sec)$	495	460	480	430
Actuator usage	11.24%	2.61%	6.02%	1.27%

Table 5: Control variable values and power consumption

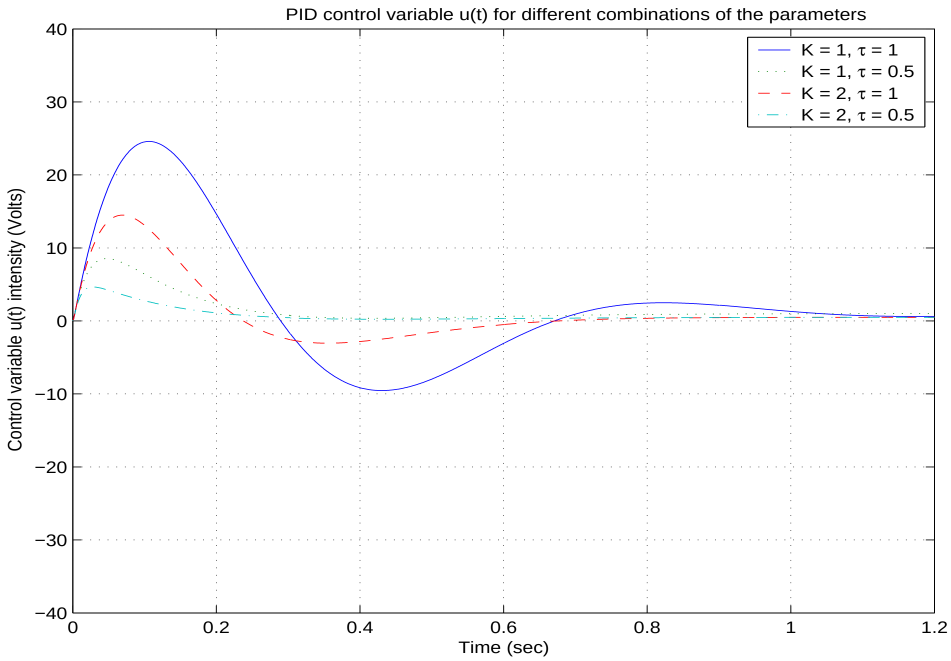

Figure 11: Control variable $u(t)$ of the PID controller. Response to a step reference signal.

3.1.4 Frequency analysis

Table 6 presents the values of the bandwidth for the closed loops from the reference signal and the filtered reference signal to the output.
A significant result discernible from figure 12 is that the filter introduced to approximate the derivative function does not change significantly the bandwidth. The added pole with the derivative filter has a minimum effect on the frequency response. That means that the introduction of the filter is legitimate. The simulation and analysis of the controller are therefore valid and coherent with the original design in (Dorf, Bishop 2001).

A Bode diagram with a comparison of the two transfer function with and without filter is shown in figure 12. The difference between the transfer functions becomes evident for frequencies greater than $10^3 rad/sec$ when the signal in the original function is attenuated by 100 ($-40bB$) and is not influential on the system dynamic.

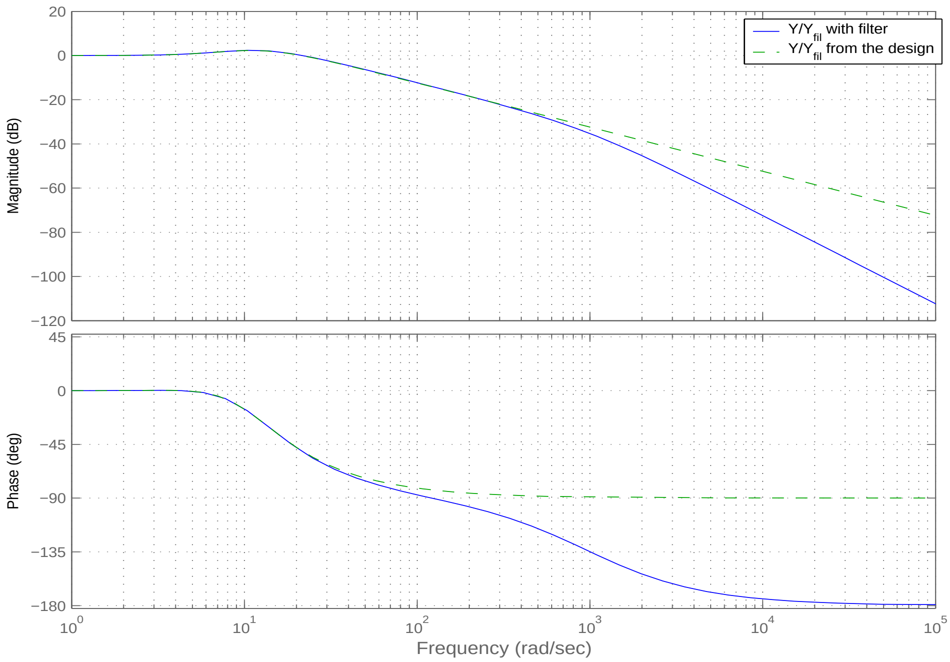

Figure 12: Bode diagram of the PID closed loop transfer functions Y/Y_{fil} with and without filter.

The transfer function Y/Y_{sp} (figure 13) has a narrower bandwidth than Y/Y_{fil} because of the desired limitation of $u(t)$. The bandwidth of Y/Y_{sp} is an indication of how fast the system is in term of settling time. The bandwidth of the transfer function Y/Y_{fil} is higher in order to have a better disturbance suppression and it is shown in figure 14. Figure 15 shows the load disturbance attenuation measured with the transfer function Y/D_l.

From the plot we can see that at the frequency of about $10 rad/sec$ the disturbance has the weakest attenuation that is however around 100 times (-40bB).

	$K = 1$ $\tau = 1$	$K = 1$ $\tau = 0.5$	$K = 2$ $\tau = 1$	$K = 2$ $\tau = 0.5$
$\omega_b(Y/Y_{sp})$ (rad/sec)	8.2	5.4	6	5.2
$\omega_b(Y/Y_{fil})$ (rad/sec)	19.9	57.6	33.2	113

Table 6: PID bandwidth of the closed loops

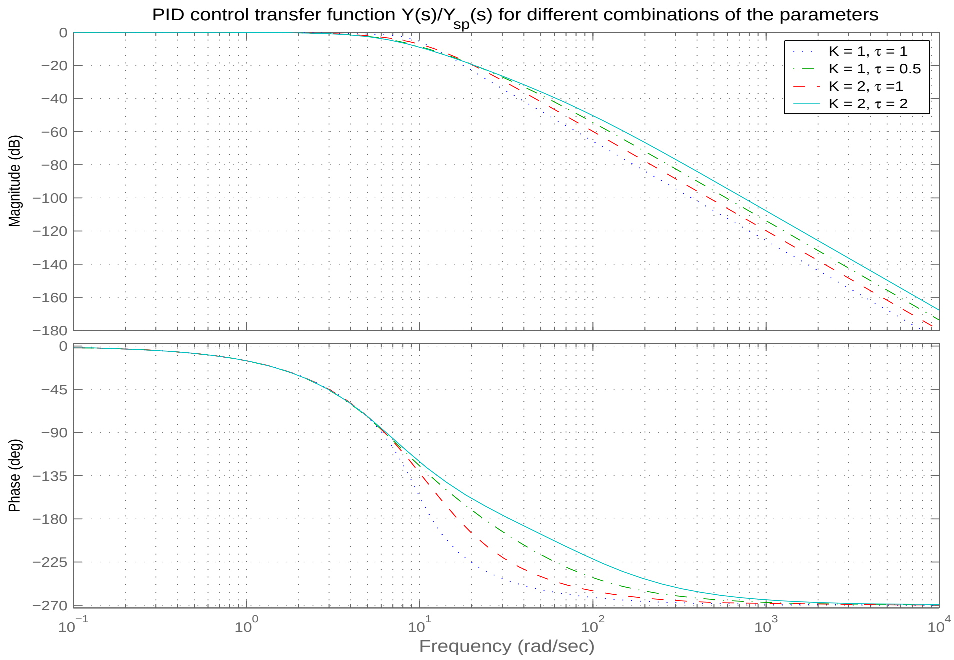

Figure 13: Bode diagram of the PID closed loop transfer function Y/Y_{sp} for the different combinations of the parameters.

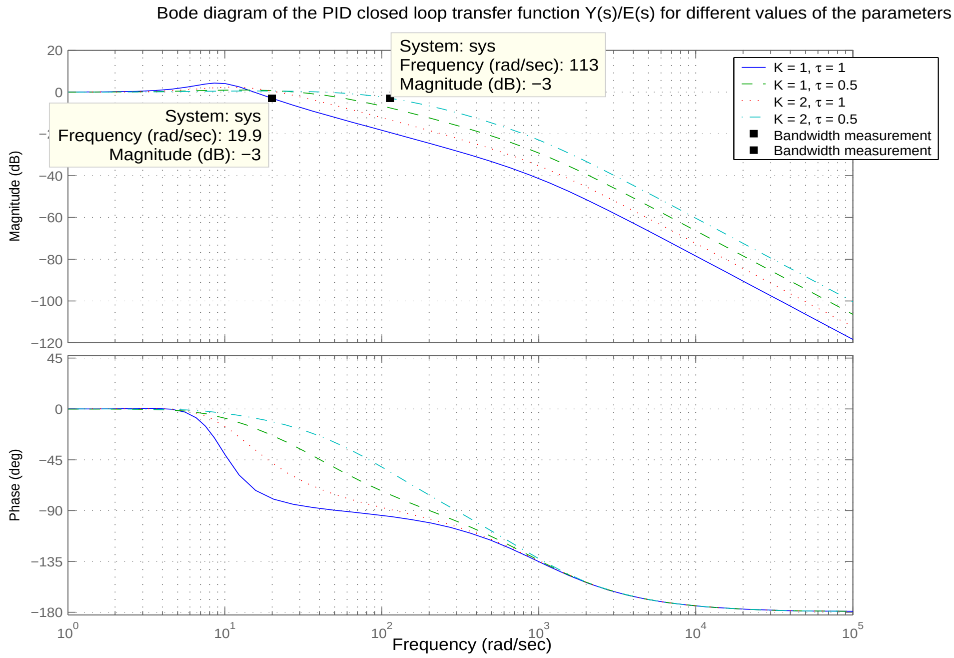

Figure 14: Bode diagram of the PID closed loop transfer function Y/Y_{fil} for the different combinations of the parameters.

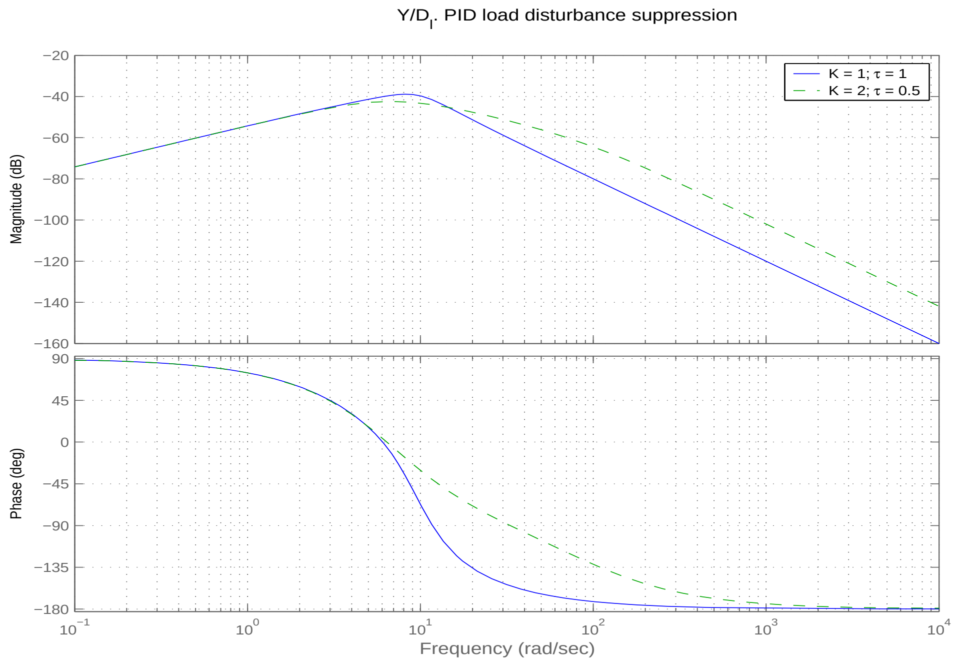

Figure 15: Bode diagram of Y/D_l for the different combinations of the parameters.

3.1.5 Local minimum verification

A verification that the PID parameters are a local optimum for the ITAE index can be obtained by calculating the ITAE index for several combinations of the parameters. This is done by the Matlab script reported in appendix B.2.3. Two of the three parameters of the PID are changed for values near the solution proposed. Figure 16 plots the result of the computation with the inverse of the ITAE index[11]. It is evident that the chosen values of the derivative and proportional parameters ($K_1 = 136.6$ and $K_3 = 12$) give a minimum ITAE. This analysis is not reliable because is limited to a restricted search space and is made by varying only two parameters. However it gives an indication of the presence of a local optimum.

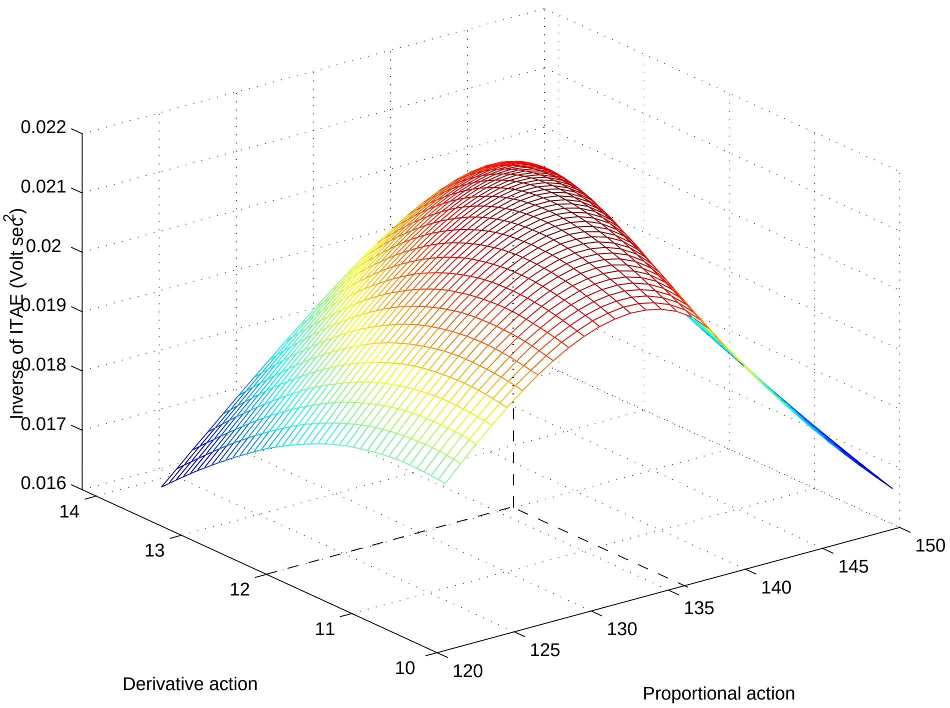

Figure 16: Plot of ITAE measurements for variation of the proportional and derivative gains of the PID controller.

[11]The inverse of the ITAE index has been chosen only for graphical reasons

3.2 Genetic programmed controller representation and simulation

The Simulink model for the GP controller is shown in figure 17 and it has been reproduced as in (Koza et al. 2003, FIG. 11)[12].

The simulation of GP controller presents two peculiarities.
The first is that the step response of the transfer function differs from the step response obtained with the Simulink model for the parameters values of $K = 1$ and $\tau = 1$. The reason is that for these values of the parameters the control variable is saturated and the model is no longer linear. Thus, in the next section, the data reported are obtained only by the model simulation with Simulink. This kind of nonlinearity due to the saturation of the control variable is usually considered while designing a controller: it does not imply that the controller is not linear, but simply that in some particular cases it might show a nonlinear behavior.
The second characteristic is that the transfer functions proposed in (Koza et al. 2003, col. 56) cannot be implemented with a non-interacting structure of the integral, proportional, derivative and second derivative actions. It has to be implemented using the structure drawn in (Koza et al. 2003, FIG. 11) or figure 17. Equation (22) and (24), if implemented with the non-interacting structure shown in figure 18, produce a controller which suffers from the windup effect[13] when the control variable reaches the saturation value for $K = 1, \tau = 1$ with a unitary step reference signal. Instead, the controller structure evolved by the genetic programming computation clearly shows an anti windup effect during the simulation.

3.2.1 Response to a unitary step reference signal

Table 7 shows the data of the simulation of the GP control system with a unitary step reference signal. Figure 19 represents the plant output for two combinations of the parameters values.

3.2.2 Response to a unitary step disturbance

Table 8 reports the recorded output values for a unitary step load disturbance. The GP controller appears to be approximately 9-10 times better than the PID controller in term of noise suppression.

[12] The figure is reported with two variations of what I presumed to be printing errors. For details see section A.

[13] The windup is a nonlinear effect that arises when the error remains of the same sign for a long time like when the control variable reaches the saturation value. If the controller uses an integrating action, the error will continue to be integrated and the integral term might become very large causing large overshoot and long settling time while unloading (Åström, Hägglund 1995).

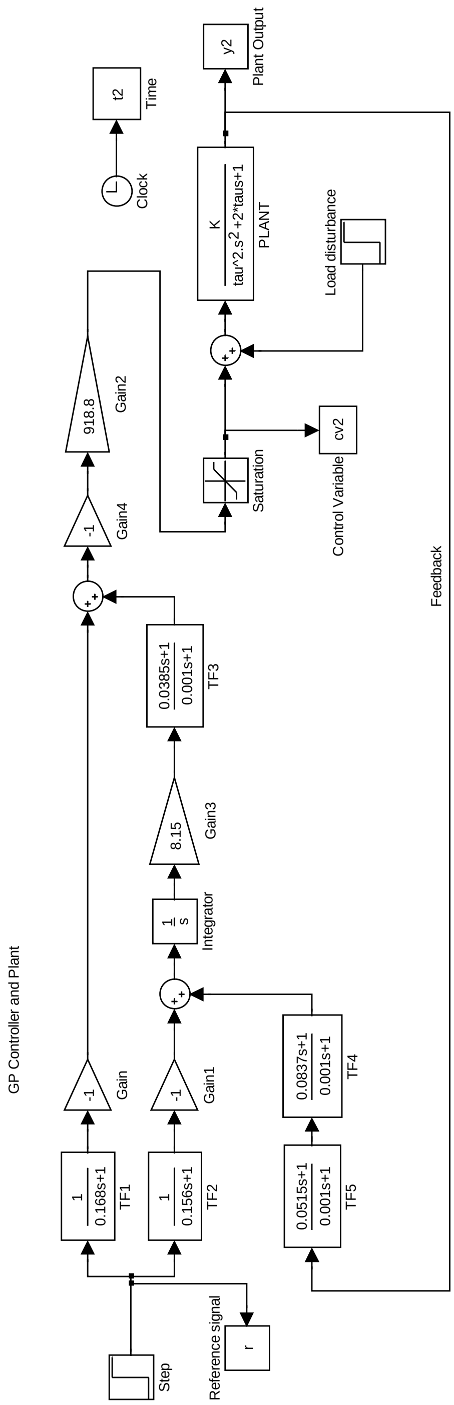

Figure 17: Simulink model for the GP controller

	$K=1$ $\tau=1$	$K=1$ $\tau=0.5$	$K=2$ $\tau=1$	$K=2$ $\tau=0.5$
Overshoot	1.5%	0.4%	0.4%	0.5%
Delay time (ms)	174	154	154	154
Rise time (ms)	181	239	233	243
Settling (ms)	300	417	418	419
ITAE ($mVolts \cdot sec^2$)	18.5	19.8	19.7	20.0

Table 7: Performance results of the GP evolved controller

	$K=1$ $\tau=1$	$K=1$ $\tau=0.5$	$K=2$ $\tau=1$	$K=2$ $\tau=0.5$
Maximum deviation ($mVolts$)	0.670	0.636	0.650	0.631
IAE ($\mu Volts \cdot sec$)	22.0	25.8	23.8	26.7

Table 8: Response to a unitary step disturbance for the GP evolved control system

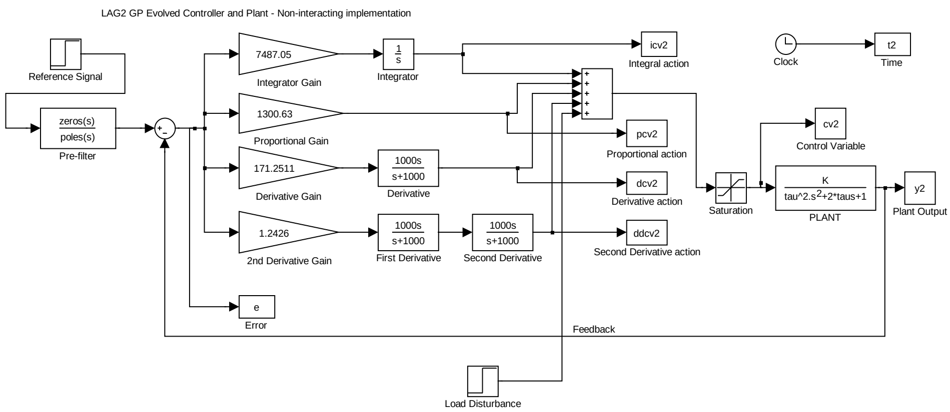

Figure 18: Non-interacting model form of the GP controller.

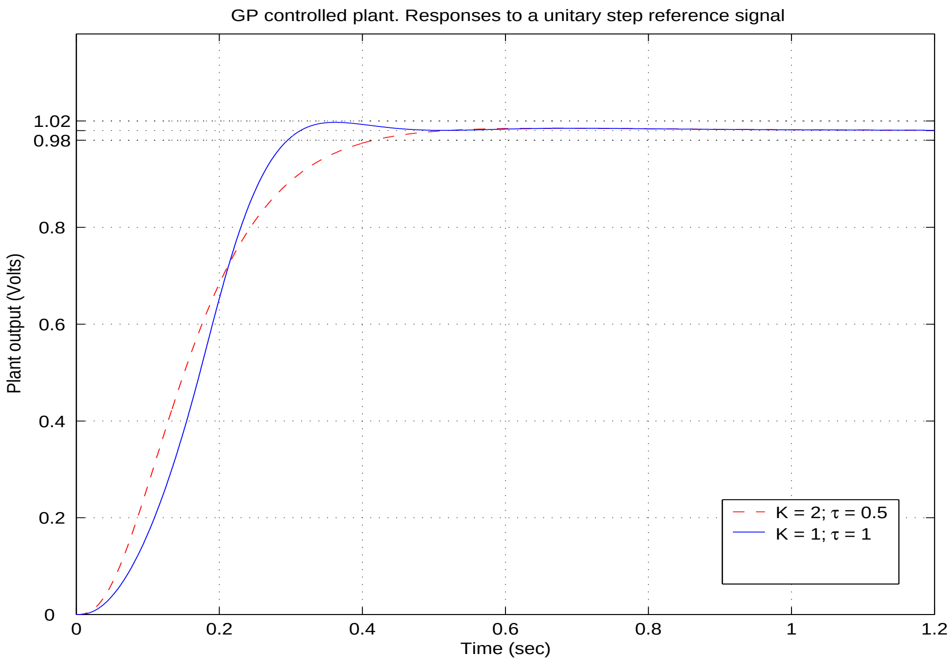

Figure 19: Responses to a unitary step reference signal.

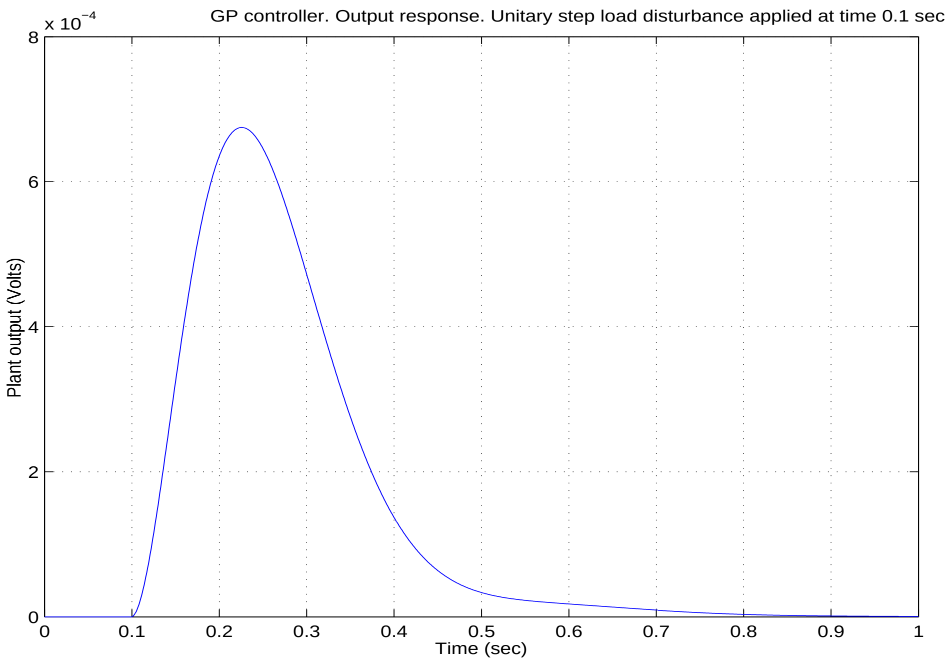

Figure 20: Responses to a unitary step disturbance.

3.2.3 Control variable dynamic

Within the GP controller structure it is not possible to identify the different actions of the integral, proportional, derivative and second derivative part. However it is possible to measure the controller output $u(t)$. Figure 21 show the results of the simulation. For the value of $K = 2, \tau = 0.5$ the bandwidth of the system is particularly high as shown in the next section: the increase in the bandwidth is such that the numerical approximation noise is fetched by the double derivative and amplified. The effect is shown in figure 22. To obtain a smooth control variable it has been necessary to decrease the sampling time of the simulation from $1ms$ to $100\mu s$.

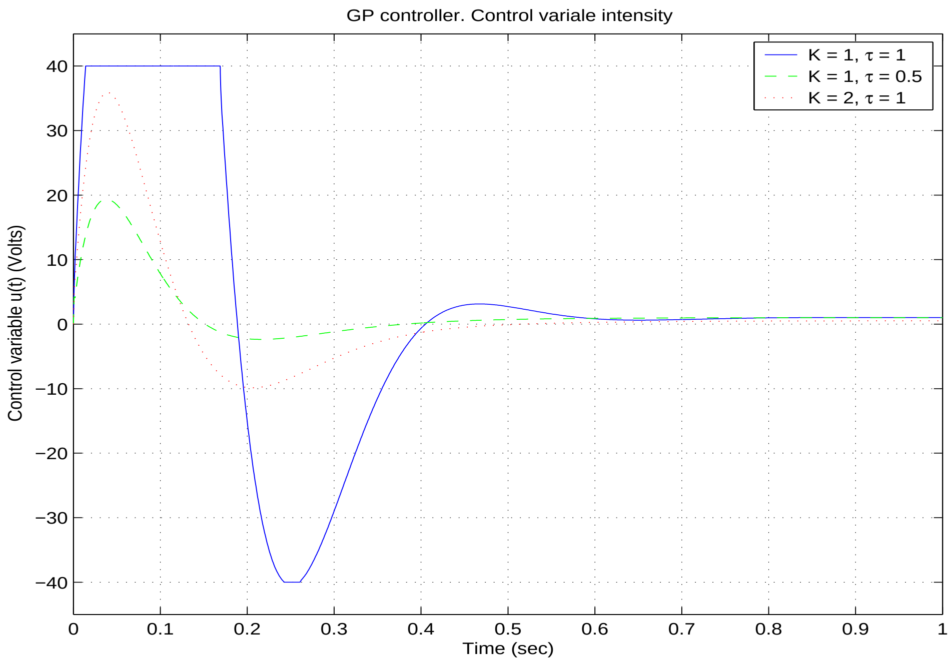

Figure 21: Control variable $u(t)$ of the GP controller. Response to a step reference signal.

For the different values of the parameters K and τ, table 9 shows the maximum value of the control variable, the maximum value of its derivative and an indication of the actuator usage from the step signal to the settling time. It is evident as the control variable reaches the saturation value for $K = 1, \tau = 1$. With this configuration of parameter the the controller makes use of saturated control.

	$K = 1$ $\tau = 1$	$K = 1$ $\tau = 0.5$	$K = 2$ $\tau = 1$	$K = 2$ $\tau = 0.5$
Maximum value of $u(t)$ (Volts)	40	19.35	35.9	9.81
Max $\dot{u}(t)$ (Volts/sec)	3087	1927	2518	2766
Actuator usage	85.45%	4.86%	15.36%	2.30%

Table 9: GP control variable values and actuator usage

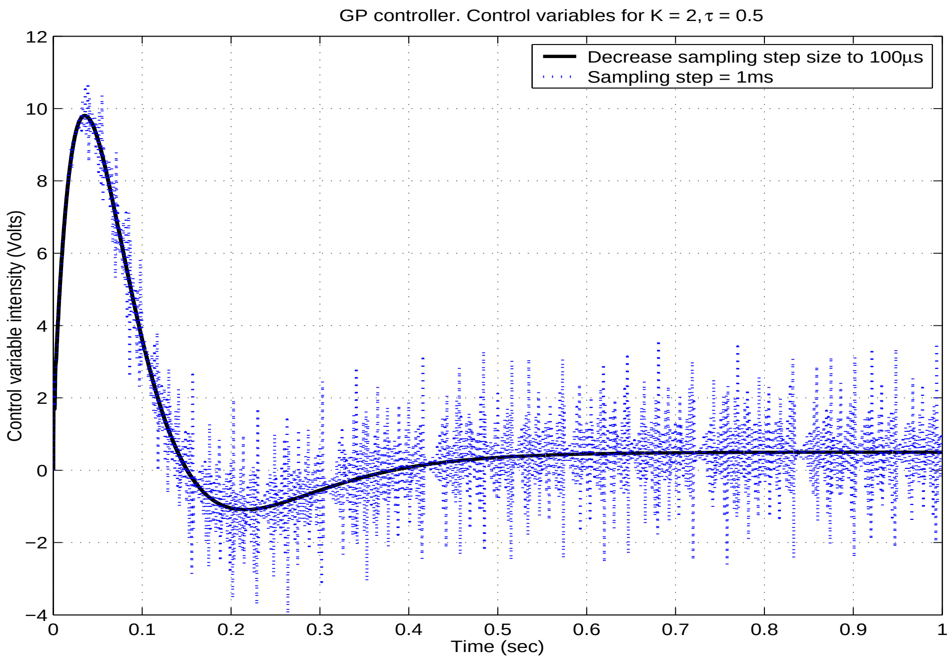

Figure 22: Control variable $u(t)$ of the GP controller for $K = 2, \tau = 0.5$.

3.2.4 Frequency analysis

Table 10 presents the data of the bandwidth of the closed loops from the reference signal to the output and from the filter reference signal to the output. The first line presents the values of the bandwidth from the reference signal to the output for the different values of the plant parameter. The second line reports the bandwidth from the filtered reference signal to the output with the filters on the derivatives. The third line reports the bandwidth of the system as described in (Koza et al. 2003) without the addition of the filters on the derivatives. Figures 23, 24 and 25 show the Bode diagrams of the transfer functions Y/Y_{sp}, Y/Y_{fil} and Y/D_l.

	$K=1$ $\tau=1$	$K=1$ $\tau=0.5$	$K=2$ $\tau=1$	$K=2$ $\tau=0.5$
ω_b Y/Y_{sp} (rad/sec)	9.6	8.8	9	8.6
ω_b Y/Y_{fil} (rad/sec)	51.7[14]	3070	1720	4640
ω_b Y/Y_{fil} (rad/sec) as in (Koza et al. 2003)	∞[15]	∞	∞	∞

Table 10: Bandwidth for the GP controller closed loops

3.2.5 An infinite bandwidth

An important point arises from the previous section. It is important to notice that in figure 24 the magnitude diagram lowers with a slope of 40b/decade for high frequencies because of the introduction of two filters on the derivatives[16]. Otherwise, the transfer function as presented in (Koza et al. 2003, col. 56) and (Koza et al. 2003, FIG. 11) shows a constant diagram with an infinite bandwidth as reported in table 10 and shown in figure 26. A consequence of this is that the model, in the way it is implemented and simulated, does not have the required characteristics of the controller in (Koza et al. 2003). As explained in section 3.1.4, the introduction of a filter on the derivative function is made under the constraint that the addition does not modify the dynamic of the system. On the other hand, it is evident that, however large we choose the low-pass filter and wide the bandwidth, the difference between a finite bandwidth and an infinite bandwidth will always be infinite. A closed loop with infinite bandwidth has an infinite grade of load disturbance suppression, it requires an infinitely precise and

[14]However the frequency response does not go under $-5.4dB$ before $\omega = 1000rad/sec$ as shown in figure 24

[15]An attenuation of $-3bB$ is reached for $\omega = 48.2rad/sec$, yet the frequency response never goes under -5dB, that is equivalent to have an infinite bandwidth.

[16]Each filter is represented by a pole. A pole increases the slope of the function of 20dB/decade.

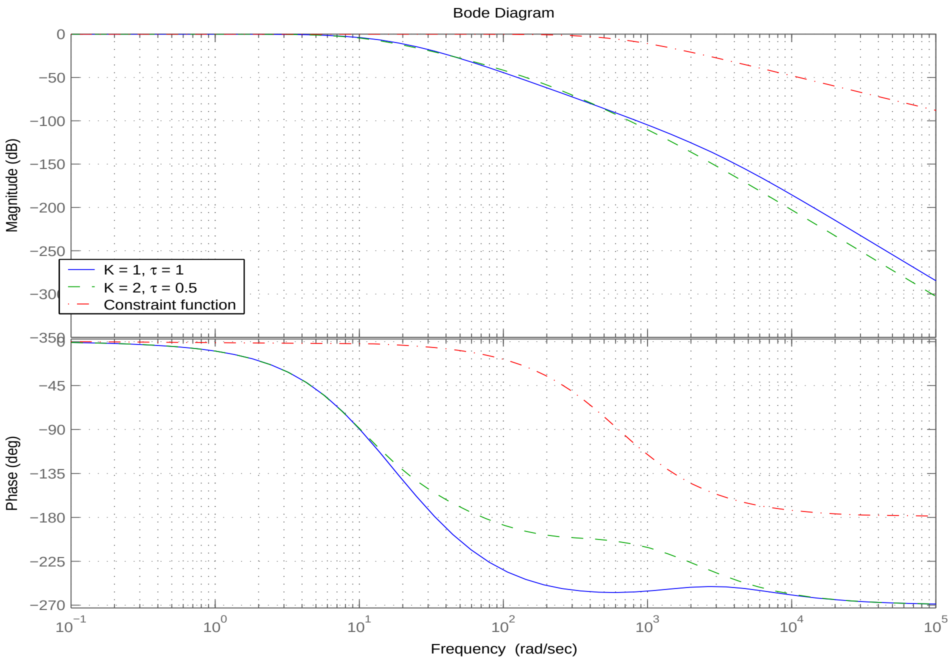

Figure 23: Bode diagram of Y/Y_{sp} of the GP controlled system.

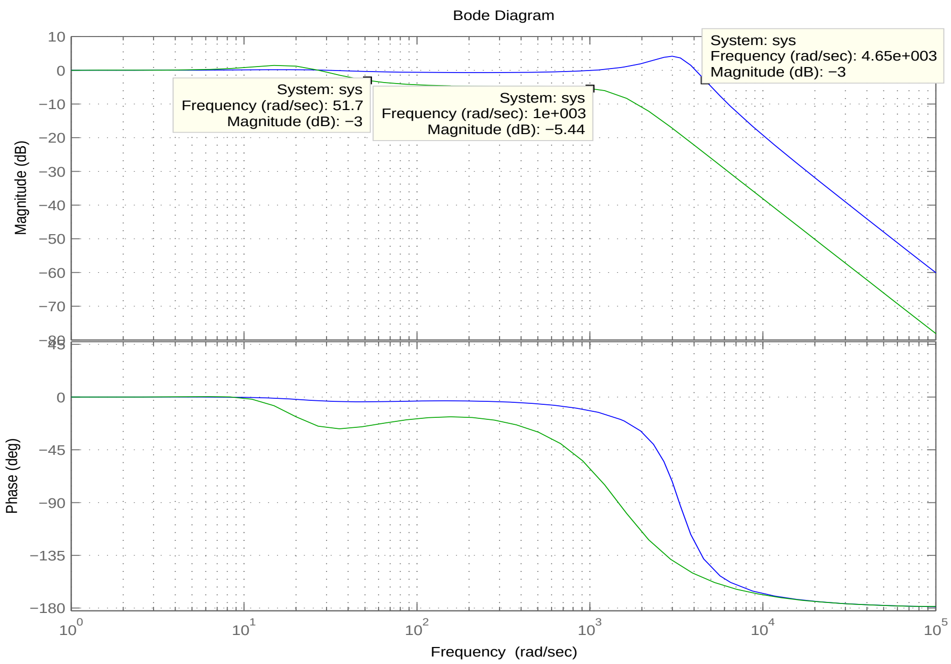

Figure 24: Bode diagram of Y/Y_{fil} of the GP controlled system.

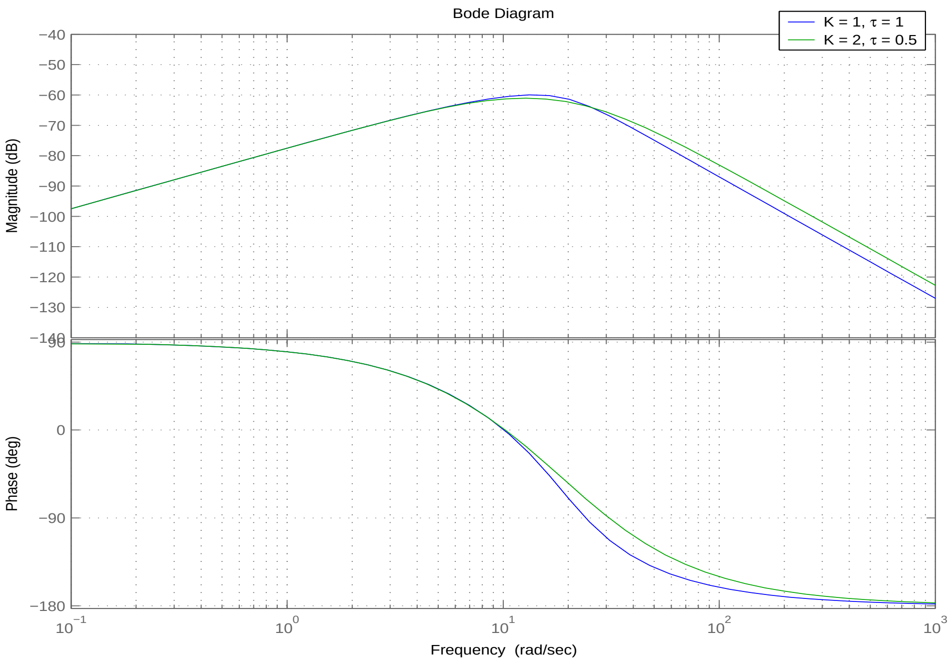

Figure 25: Bode diagram of Y/D_l of the GP controlled system.

free of error feedback and it is not possible to build it or simulate it. Hence, the model proposed in (Koza et al. 2003) is not implementable, except by introducing substantial modification on the system characteristics.

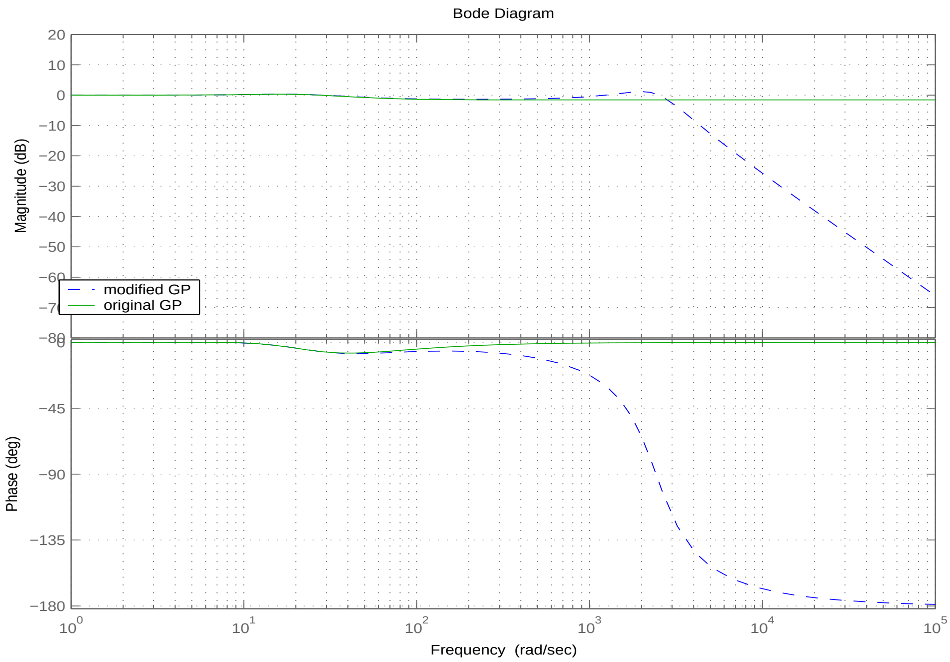

Figure 26: Bode diagram of Y/Y_{fil} with and without filters on the derivatives.

3.3 Comparison

Table 11 and 12 summarize the significant data for the chosen values of the plant parameters of $K = 1, \tau = 1$ and $K = 2, \tau = 0.5$.

	PID	GP	Characteristic
Overshoot	2%	1.5%	limited[17]
Delay time (ms)	261	174	to minimize
Rise time (ms)	290	181	to minimize
Settling time (ms)	943	300	to minimize
ITAE ($mVolts \cdot sec^2$)	48.6	18.5	to minimize
Load disturbance deviation ($mVolts$)	6.4	0.670	to minimize
IAE of disturbance ($\mu Volts \cdot sec^2$)	197.0	22.0	to minimize
Maximum $u(t)$ ($Volts$)	24.6	40	limited[18]
Maximum $\dot{u}(t)$ ($Volts/sec$)	495	3087	unspecified/free
Saturated control	No	Yes	unspecified/free
Actuator usage	11.24%	86.45%	unspecified/free
Bandwidth Y/Y_{sp} (rad/sec)	8.2	9.6	limited[19]
Bandwidth Y/Y_{fil} (rad/sec)	19.9	51.7	unspecified/free

Table 11: Summary of data for the PID and GP controller for $K = 1, \tau = 1$

	PID	GP	Characteristic[20]
Overshoot	0.35%	0.5%	limited
Delay time (ms)	241	154	to minimize
Rise time (ms)	407	243	to minimize
Settling time (ms)	661	419	to minimize
ITAE ($mVolts \cdot sec^2$)	49.9	20.0	to minimize
Load disturbance deviation ($mVolts$)	5.8	0.631	to minimize
IAE of disturbance ($\mu Volts \cdot sec^2$)	316.4	26.7	to minimize
Maximum $u(t)$ ($Volts$)	4.65	9.8	limited
Maximum $\dot{u}(t)$ ($Volts/sec$)	430	2766	unspecified/free
Saturated control	No	No	unspecified/free
Actuator usage	1.27%	2.30%	unspecified/free
Bandwidth Y/Y_{sp} (rad/sec)	5.2	8.6	limited
Bandwidth Y/Y_{fil} (rad/sec)	113	4640	unspecified/free

Table 12: Summary of data for the PID and GP controller for $K = 2, \tau = 0.5$

All the constraints (i.e. the limited characteristics) are respected. The characteristics to be minimized are considerably improved by the GP con-

[17] Overshoot is limited to 2%
[18] $|u(t)|$ is limited to $40 Volts$
[19] The bandwidth for Y/Y_{sp} is limited to $401 rad/sec$
[20] With the same limitations as above

troller. The unspecified characteristics, though, are brought to extreme values. In other words, the gain in performance is reached to the detriment of feasibility and usability of the GP controller. Saturated controller is used, while the PID is far under the threshold of $40Volts$. $\dot{u}(t)$ is about 6 times greater and the actuator usage, which is an indicator of the power consumption, is more than 7 times greater. Thus, the GP controller would probably be unsuitable for the majority of the control problems or alternatively judged badly tuned.
On the other hand, the fact that the evolutionary computation explored the unspecified constraints in order to obtain a better solution is indicative of the effectiveness of the method. Since some parameter like $\dot{u}(t)$ are unlimited, it is not surprising that the GP makes use of very high values. Besides, the synthesis of a saturated control, though usually avoided, gives a proof of the capability of the evolutionary computation of dealing with nonlinear effects. Had it not been like that, the search method would have been less effective or shown some kind of limitations.
This inference is not enough yet to consider the proposed genetic programming method a good design method for control systems.
A further comparison should be done. Possibly with a controller using the available resources left free by unspecified constraints. In particular, since the GP controller makes use of saturated control (see table 11 and figure 21), a fair comparison should consider the use of saturated control for the standard controller as well. In the next section I propose a new comparison following this principle.

3.4 Adapting the PID controller to the GP controller constraints

Given the relevant differences in the constraints and characteristics of the two controllers just compared, I tuned the PID given by equations (16) and (17) in order to make it comparable to the GP controller. In the next sections it is shown how to tune it, a frequency analysis and the result of the simulation directly compared with the GP controller.

3.4.1 Tuning the PID parameters and embedding an anti-windup system

From equations (20), the new parameters can easily be obtained using $\omega_n = 16$ instead of $\omega_n = 8$ as in (Dorf, Bishop 2001).
The new PID parameters are:

$$\begin{aligned} K_1 &= 2.15\omega_n^2 - 1 = 549.4 \\ K_2 &= \omega_n^3 = 4096 \\ K_3 &= 1.75\omega_n - 2 = 27 \end{aligned} \tag{28}$$

In order to outdo the performance of the GP with respect to the load disturbance suppression it is sufficient to increase the bandwidth of the closed loop Y/Y_{fil} by inserting a gain of 3 on the feedback line.
Finally, since the new controller has to respect the same constraints as the GP controller, it is necessary to introduce an anti windup device to avoid the integral windup when the control variable reaches the saturation values. Several anti windup systems are described in detail in (Åström, Hägglund 1995). However, since the purpose is not a study of the anti windup system, I implemented the integrator part of the controller with a limitation function included in the integrator block of Simulink. The limits for the integrator block have been set to -8 and $+8$ Volts. The controller is completely described in figure 27. The Matlab script for the frequency analysis is reported in appendix B.1.3.

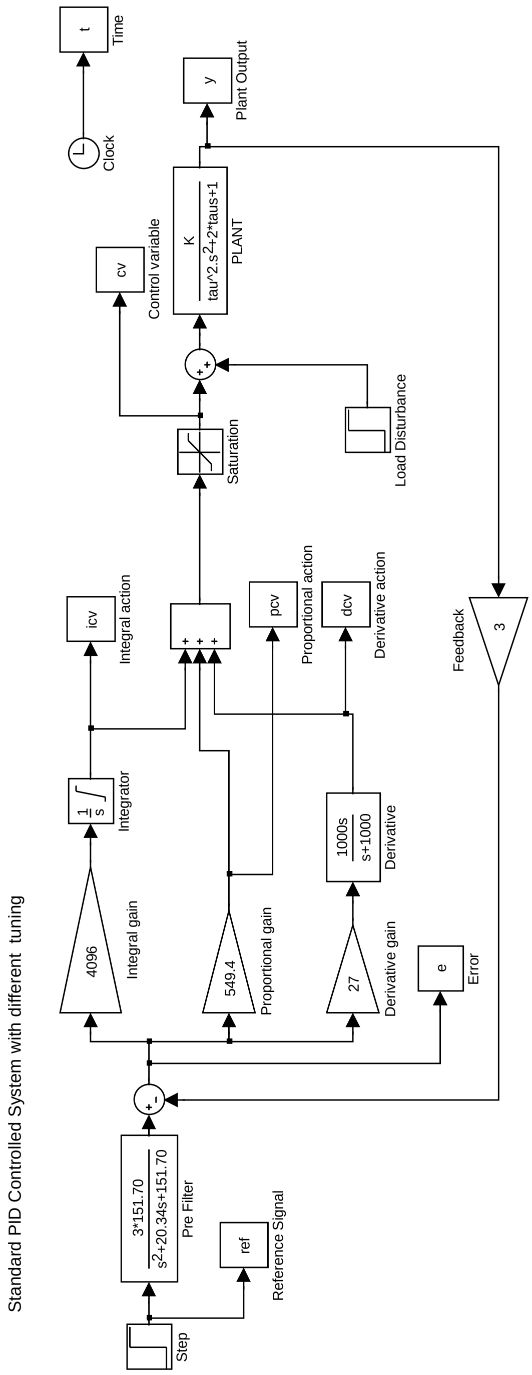

Figure 27: Simulink model for the new PID

3.4.2 Simulation results and comparison

Table 13 reports the data of the analysis and simulation grouped in a single table. The characteristics verified are the same tested for the PID and GP in the previous sections.

	$K = 1$ $\tau = 1$	$K = 1$ $\tau = 0.5$	$K = 2$ $\tau = 1$	$K = 2$ $\tau = 0.5$
Overshoot	1.2%	1%	0.9%	1%
Delay time (ms)	170	130	138	128
Rise time (ms)	176	210	197	208
Settling (ms)	285	326	325	324
ITAE ($mVolts \cdot sec^2$)	16.7	13.5	13.6	13.3
Load disturbance deviation ($mVolts$)	0.460	0.420	0.435	0.418
IAE for disturbance ($\mu Volts \cdot sec$)	19.0	19.0	19.0	19.0
Maximum $u(t)$ ($Volts$)	40	36	40	20.1
Maximum $\dot{u}(t)$ ($Volts/sec$)	10654	8761	9805	7573
Actuator usage	97%	7.2%	25.7%	3.48%
Bandwidth Y/Y_{sp} (rad/sec)	11.2	10.4	10.7	10.3
Bandwidth Y/Y_{fil} (rad/sec)	105	435	196	897

Table 13: Performance results of the new PID

	PID	GP	new PID	Characteristic
Overshoot	0.3%	0.4%	1%	limited
Delay time (ms)	243	154	**130**	to minimize
Rise time (ms)	391	239	**210**	to minimize
Settling time (ms)	629	417	**326**	to minimize
ITAE ($mVolts \cdot sec^2$)	49.0	19.8	**13.5**	to minimize
Load disturbance deviation ($mVolts$)	6.0	0.64	**0.42**	to minimize
IAE of disturbance ($\mu Volts \cdot sec^2$)	218.1	25.8	**19.0**	to minimize
Maximum $u(t)$ ($Volts$)	8.6	19.4	36.0	limited
Maximum $\dot{u}(t)$ ($Volts/sec$)	460	1927	8761	unspecified/free
Saturated control	No	No	No	unspecified/free
Actuator usage	2.61%	4.86%	7.2%	unspecified/free
Bandwidth Y/Y_{sp} (rad/sec)	5.4	8.8	10.4	limited
Bandwidth Y/Y_{fil} (rad/sec)	57.6	3070	435	unspecified/free

Table 14: Summary of data for the PID and GP controller for $K = 1, \tau = 0.5$

Table 14 reports a comparison of the modified PID, the GP controller and the standard PID for values of the parameters not considered in table 11 and 12.

It is evident that all the values to be minimized are smaller for the modified PID and all the constraints are respected. Thus, the new PID is better than the GP controller with respect of every performance index considered for the analysis. However, the unspecified constraints are brought to even more extreme, with one exception. The bandwidth for Y/Y_{fil} (last line of table 14) is still much lower for the new PID even if the load disturbance suppression is improved. That means that the GP controller use of bandwidth is badly optimized. Probably, the use of the second derivative is the main cause of such a large bandwidth. It seems that in spite of this considerable disadvantage[21] the use of the second derivative does not improve the performance of the system.

Figures 28, 29, 30, 31 and 32 reports different plots for the compared systems.

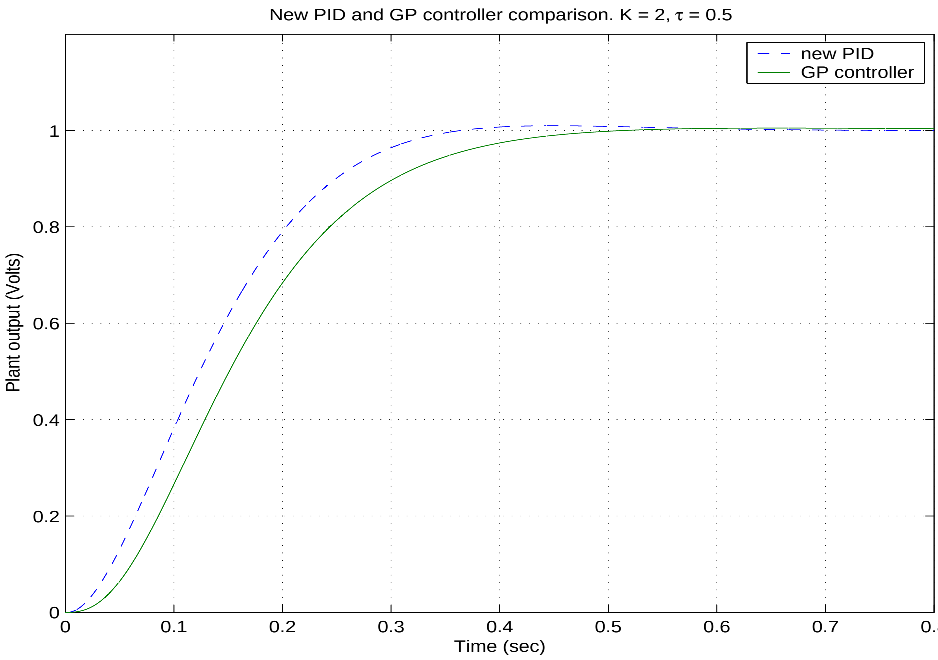

Figure 28: Step response comparison between the new PID and the GP controller. $K = 2$,$\tau = 0.5$.

[21] From the research I made in the field, it seems that there is no any relevant application in which a controller with a second derivative is applied.

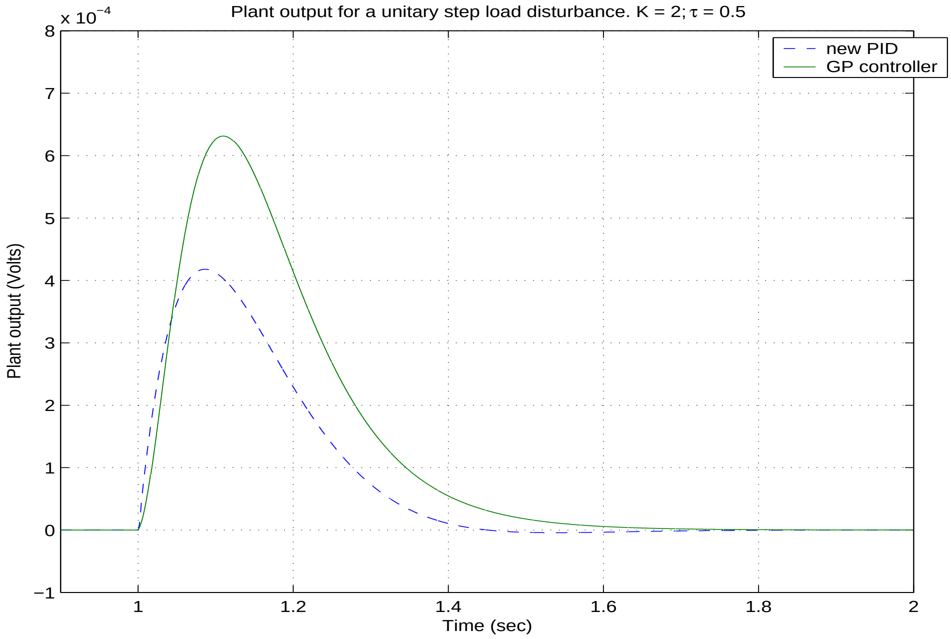

Figure 29: Plant output for a unitary step load disturbance. $K = 2, \tau = 0.5$.

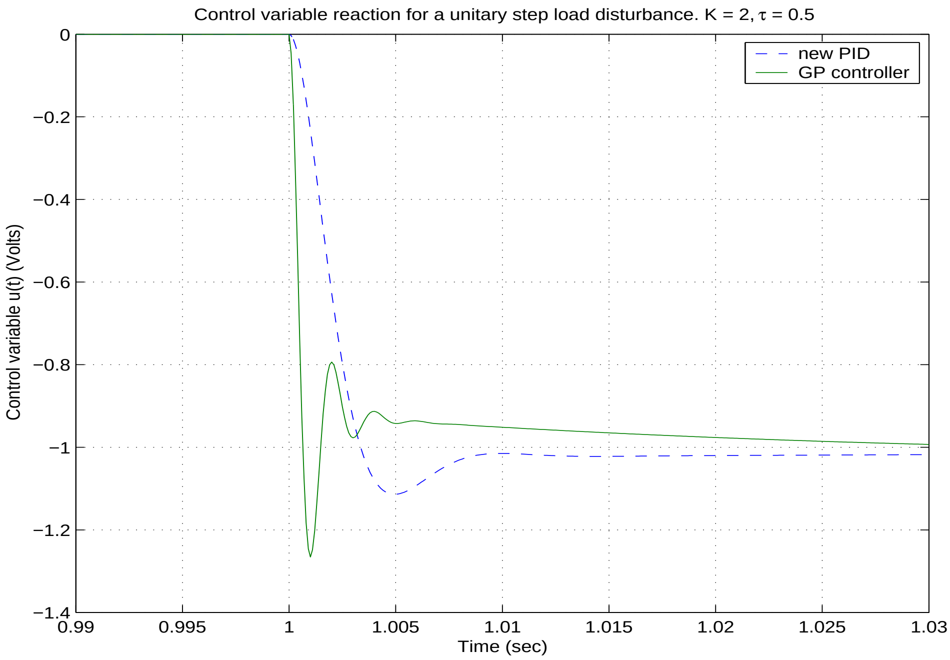

Figure 30: Control variables reaction to a step load disturbance. $K = 2, \tau = 0.5$.

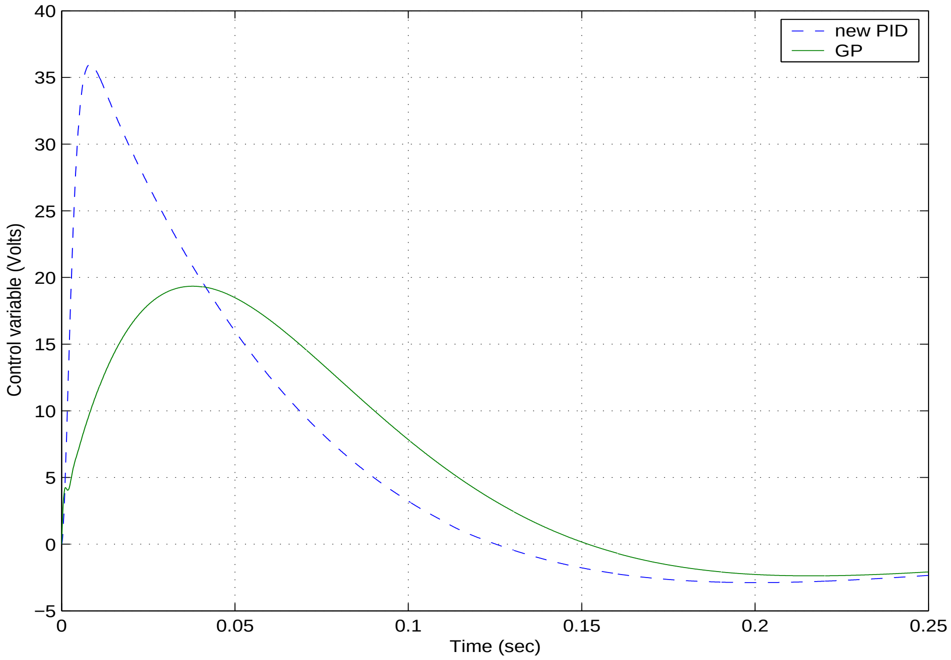

Figure 31: Control variables reaction to a step reference signal. $K = 1$,$\tau = 0.5$.

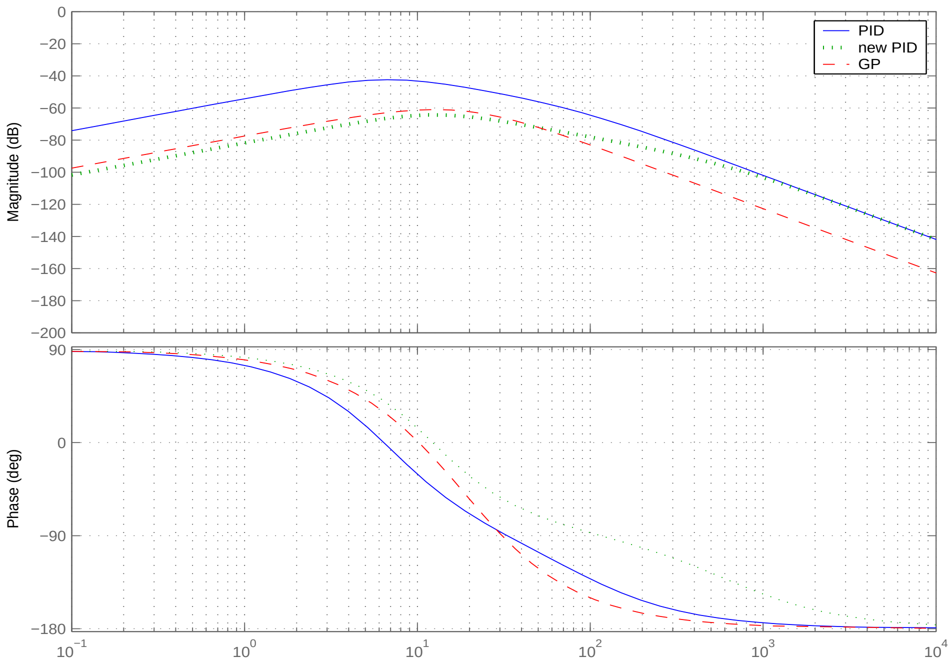

Figure 32: Bode diagram of the function Y/D_l for the three controllers considered. $K = 2, \tau = 0.5$.

4 Conclusion

The study of the genetically evolved controller has outlined several aspects of the design of a control system. The results from the requirements and specifications analysis suggest that, independently of the methodology of synthesis used, the early stages of the design should be deeply concerned with the physics and the characteristics of the plant. Experts in the field of control engineering suggest that no good solution can be found to a poorly specified problem.

The extreme difference in the performance of the GP controller compared to the PID has to be attributed to the exploration of unspecified constraints. For example, the 9 times improvement in the disturbance suppression is reached by increasing the bandwidth up to 50 times. The fast settling time is obtained by means of saturated control, higher bandwidth, higher actuator usage and faster control variable ($\dot{u}(t)$). For these reasons, we are brought to reject the claims of absolute superiority of the GP controller. While the PID is adaptable to several control problems, the GP controller shows extreme characteristics and it is suitable for very specific systems only. The proof of the existence of such systems, for which the GP controller might work properly, is not given in (Koza et al. 2003) and is left as a doubt.

The design methods are also very different. The GP controller is not tunable because the tuning procedure is integrated in the automatic synthesis. It requires the run of the evolutionary computation for each specific control problem. The challenge of the evolutionary computation method does not only lay in the required computational power, the choice of the parameters, as described in section 1.5, has to be carefully considered as well. The importance of this choice is outlined by Fleming, Purshouse (2002, page 1225): *"Even if a good choice of parameters can be found for a particular problem, this set will be sub-optimal for many other problem. Making the wrong choice of EA parameters can produce exceedingly poor results"*. For the PID controller, the tuning system appears to be extremely less expensive and simpler as shown and demonstrated in section 3.4.1. However, the tuning procedure of the PID controller does not allow any different structure of the controller. The genetic programming method is free to evolve every kind of structure, linear or even nonlinear. The extremely different typology and potentiality of the two methods do not allow an easy comparison.

A considerable achievement of the evolutionary computation that is not mentioned in (Koza et al. 2003) is, in my opinion, the synthesis of an anti-windup structure to avoid overshoot when operating under saturated control condition. The anti windup structure is not explicitly synthesized in a particular point of the controller. However, the simulation under saturated control condition shows that there is no integral windup (compare figures 3.2.1 and 21). The anti windup device is nowadays implemented in every PID and widely studied, but in the past has been well known only in the industrial field and considered an industrial secret (Åström, Hägglund 1995). The autonomous decision of use of saturated control, on one hand makes the comparison biased, on the other hand shows the potential of the evolutionary computation in the solution's search.

From the comparison with the modified PID, we have proved that a roughly tuned PID controller (section 3.4.1) can beat the GP controller under every considered performance index given the same constraints. A further element arises from the new comparison: the unspecified constraint of the bandwidth limitation for the feedback noise brings the evolutionary computation to evolve a controller that "wastes" a large amount of bandwidth. That might work in a simulation, but in a real system it would certainly have some undesired effects due to feedback noise. As shown in table 14, the modified PID controller can obtain a much better suppression of noise using a bandwidth up to 7 times narrower.

From the analysis of the results, I am brought to think that the limitations of the method lay essentially at the simulation level. If the fitness measurement were obtained through a trial of the controller on a real plant, a controller with a bandwidth of 4640 rad/sec would have probably damaged the plant and obtained a bad value of fitness. It is possible that by means of this reliable measurement of fitness, the evolutionary computation would have found a very good control system, at the price of several damaged plants though. This is one of the main reason that limits the use of online genetic algorithms in some control fields.
Literature and recent studies show that evolutionary computation can be a powerful tool that has many applications in the control field (Fleming, Purshouse 2002). In order to obtain applicable and implementable results and gain the trust of the experts in the field, a deep study of the specific area and problem in which evolutionary computation is going to be applied, is an essential and basic requirement.

The results illustrated in this paper are liable to my personal interpretation of the source documents. Several difficulties in reproducing the results claimed in (Koza et al. 2003) have been overcome by assuming the presence of printing errors. Appendix A provides a list of the doubtful parts or printing errors. As a consequence, my results are dependent on the explained assumptions.

4.1 Future work

The fidelity of a simulation, which is considered a weak point of the evolutionary search, can be improved by a better formulation of the model. Koza et al. (2003) propose the use of several criteria to calculate the fitness, including the consideration of the effect of noise in the feedback. We do not know why this constraint was not originally introduced in the problem setting, but we believe that in this way a more feasible and reasonable solution can be found. Thus, several directions of possible future investigations are left open. Setting the constraints with more consideration of the physical control problem or giving it a stricter formulation could bring the method to find good and realistic solutions. The result of such computation could be considered significant in a comparison or evaluation of a similar solution obtained with a standard method. Hence, the study of a generic control problem suggests that a possible future application of the method with more properly chosen constraints might produce more significant results.

The controllers created for the third order plant and the third order with delay plant (Koza et al. 2003, col. 56 and 64) can be objects of future investigations. Although we have reason to believe that the same flaws in the constraints setting might have produced controllers that suffer the same problems of the two lag controller, the increased complexity of the classical design method makes an automatically generated solution more attractive. If the control problem is known to be potentially well controlled by a prespecified controller structure, genetic algorithms can be chosen instead of genetic programming.

Eventually, the general method and knowledge introduced to study this specific control problem, which has shown to be complex and interdisciplinary, constitutes a solid base for the study of several and different application of evolutionary computation in control engineering. Fleming, Purshouse (2002) suggest a variety of control problems considered by evolutionary techniques: online applications, nonlinear control, system identification and fault diagnosis to mention some of them.

A Doubtful parts or printing errors in the source documents

- The transfer function (22) has been reported modified according to the presumed printing errors found in (Koza et al. 2003). In the original paper the function is written as follows

$$G_{p32}(s) = \frac{1(1+.1262s)(1+.2029s)}{(1+.03851s)\boxed{(1+.05146)}\boxed{(1+.08375)}(1+.1561s)(1+.1680s)} \tag{29}$$

In (Koza et al. 2000), the same transfer function is written as follows

$$G_{p32}(s) = \frac{1(1+.1262s)(1+.2029s)}{(1+.03851s)(1+.05146s)\boxed{(1+.08375)}(1+.1561s)(1+.1680s)} \tag{30}$$

I modified (30) and (31) into (22). I assumed that the second and third factor in (30) miss the operator 's'. In (31) the operator 's' is missing in the third factor.

- Figure 11 in (Koza et al. 2003) shows a block with the transfer function $(1+0.515s)$: I assumed that it is actually referring to the factor $(1+0.05146s)$ as correctly written in (Koza et al. 2000).

- All the blocks of (Koza et al. 2003, Figure 11) expressed as

$$[(1+\tau_1 s)] \tag{31}$$

have been considered an approximation of the legitimate transfer functions

$$\left[\frac{(1+\tau_1 s)}{(1+\tau_2 s)}\right] \tag{32}$$

where $\tau_2 \ll \tau_1$ and therefore $(1+\tau_2 s) \approx 1$. No indications are given though and the value of τ_2 has been chosen from the considerations made in section 2.4.

- Figure 11 in (Koza et al. 2003), as the corresponding figure in (Koza et al. 2000), shows a sum block from the feedback signal without the sign indication. The other summing blocks the figure show a plus '+' operator. While it is usually assumed that the feedback signal is subtracted to the reference signal, in this case the system shows an unstable behaviour. Hence, the simulation has been carried out assuming the presence of a plus operator even where it is missing.

- The last line of Table 2 in (Koza et al. 2003, col. 55) reports a settling time of 944ms for the GP controller and 304ms for the PID controller. I assume that it was intended the other way around.

- Table 6 in (Koza et al. 2003, col. 62) it is said in the title to report the characteristics for the two lag plant. However the data do not seem to belong to the two lag plant and I therefore ignored the content.

B Matlab source codes

B.1 System data specification

B.1.1 System data specification for the standard PID

```
% Standard PID controller data

k = 2;        % from 1 to 2
tau = 0.5;     % from 0.5 to 1

Pn = [k];
Pd = [tau^2 tau*2 1];

% pre-filter
PreN = [0 0 42.67];
PreD = [1 11.38 42.67];

% compensator
Cn = [12*1 12*11.38 12*42.67];
Cd = [0 1 0];
%printsys(compNum, compDen)

% open look L(s)
% compensator-plant
[CPn, CPd] = series(Cn, Cd, Pn, Pd);
[FilterN, FilterD] = zp2tf([],[-1000],1000);
[CPn, CPd] = series(CPn, CPd, FilterN, FilterD);

% Y/D
[dydn, dydd] = feedback(Pn, Pd, Cn, Cd);

% T1 = Y/Yfil
[n, d] = feedback(CPn, CPd, [1] , [1]);

% T(S)
[PIDn, PIDd] = series(PreN, PreD, n, d);

printsys(PIDn, PIDd)
```

B.1.2 System data specification for the GP controller

```
% GP controlled system

% SYSTEM DEFINITION
% LAG2 Plant transfer function

k = 1;
tau = 1;
Pn = [k];
Pd = [tau^2 2*tau 1];

% Compensator
Cn = [1.242 71.2511 1300.63 7487.04];
Cd = [0 0 1 0];

% Series of Controller-Plant L(s) open loop
CPn = conv(Cn, Pn);
CPd = conv(Cd, Pd);
[FilterN, FilterD] = zp2tf([],[-1000 -1000], 1000000);
[CPn, CPd] = series(CPn, CPd, FilterN, FilterD);

% Y/D disturbance to output
[ydn, ydd] = feedback(Pn, Pd, Cn, Cd);


% controller-plant feedback L(s) closed loop
[n, d] = feedback(CPn, CPd, 1, 1);


% pre-filter
z = [-0.1262^(-1) -0.2029^(-1)];
p = [-0.03851^(-1) -0.05146^(-1) -0.08375^(-1) -0.1561^(-1) -0.1680^(-1)];
gain = 5883;
[PreN, PreD] = ZP2TF(z', p', gain);


% T(s) = Y(s)/Ysp(s)
[GPn, GPd] = series(PreN, PreD, n, d);
printsys(GPn,GPd)
```

B.1.3 System data specification for the modified PID

```
% Standard PID controller data
% omega = 16

k = 1;         % from 1 to 2
tau = 1;     % from 0.5 to 1

Pn = [k];
Pd = [tau^2 tau*2 1];

% pre-filter
PreN = [0 0 3*151.7];
PreD = [1 20.34 151.7];

% compensator
Cn = [27*1 27*20.34 27*151.7];
Cd = [0 1 0];
%printsys(compNum, compDen)

% compensator-plant
[CPn, CPd] = series(Cn, Cd, Pn, Pd);
[FilterN, FilterD] = zp2tf([]',[-1000]',1000);
[CPn, CPd] = series(CPn, CPd, FilterN, FilterD);


% Y(s)/D(s)
[dydn2, dydd2] = feedback(Pn, Pd, 3*Cn, Cd);

% T(s) = Y(s)/R(s)
[n, d] = feedback(CPn, CPd, [3] , [1]);
[PIDn, PIDd] = series(PreN, PreD, n, d);

printsys(PIDn, PIDd)
```

B.2 Functions for the system performance evaluation

B.2.1 System performance given the transfer function

I used the following function to evaluate the system performance given the transfer function $T(s)$. The function simulates the sistem using the command *step* from the *Control System Toolbox*.

```
% Function for evaluating system performance (Overshoot, Delay, Rise,
% Settling) in response to a step input
% num = numerator of the transfer function
% den = denumerator of the transfer function
% Ft = Final time of computation
% Tstep = Time step for the vector of time

function [Overshoot, DelayT, RiseT, SettlingT, y] = Sys_Perf(num, den, Ft, Tstep)
t = 0:Tstep:Ft;

[y, z, t] = step(num,den,t);
plot(t,y,t,0.98,'--',t,1.02,'--');
grid;

Overshoot = (max(y) - 1) * 100;

r = 1; while y(r) < 0.5; r = r + 1; end;
DelayT = (r - 1) * Tstep;

r = 1; while y(r) < 0.1; r = r + 1; end;
p = r; while y(p) <= 0.9; p = p + 1; end;
RiseT = ((p - 1) - (r - 1)) * Tstep;


r = length(t);
while (y(r) >= 0.98) & (y(r) <= 1.02);
    r = r - 1;
end;
SettlingT = r * Tstep;
```

B.2.2 System performance given the result vector of the simulation

I used the following function to evaluate the performance of a system given the the result vector of the simulation. The result vector can be obtained through different ways. I simultated the system with *Simulink* and reported the results to the workspace.

```
\% Function for evaluating system performance (Overshoot, Delay, R
\% Setteling)
\% The input vector y is supposed to be the response to a step reference
\% signal.
\% Tstep = Time step for the vector of time

function [Overshoot, DelayT, RiseT, SettlingT] = Sys_Perf(t, y, Tstep)

Overshoot = (max(y) - 1) * 100

r = 1; while y(r) < 0.5; r = r + 1; end;
DelayT = (r - 1) * Tstep

r = 1; while y(r) < 0.1; r = r + 1; end;
p = r; while y(p) <= 0.9; p = p + 1; end;
RiseT = ((p - 1) - (r - 1)) * Tstep

r = length(t);
while (y(r) >= 0.98) & (y(r) <= 1.02);
    r = r - 1;
end;
SettlingT = r * Tstep
```

B.2.3 ITAE computation for variation of the PID parameters

```
% Minimizing ITEA

K1MIN = 120;
K1MAX = 150;

K2 = 512;

K3MIN = 10;
K3MAX = 14;

t = 0:0.01:1;
x = 0;
```

```
y = 0;
K3 = K3MIN;

while K3 < K3MAX;
    y = 0;
    K1 = K1MIN;
    x = x + 1;
    K3 = K3 + 0.1;

while  K1 < K1MAX;
    y = y + 1;
       K1 = K1 + 1;
    [numd, dend] = PPID(K1, K2, K3);
    [res, z, t] = step(numd, dend, t);
    ITAE(x,y) = t * abs(1-res);
end;
end;


% Standard PID controller data

function [numd, dend] = PPID(K1, K2, K3)

k = 1;        % from 1 to 2
tau =1;     % from 0.5 to 1

PlantNum = [k];
PlantDen = [tau^2 tau*2 1];

% pre-filter
preNum = [0 0 K2/K3];
preDen = [1 K1/K3 K2/K3];

% compensator
compNum = [K3 K1 K2];
compDen = [0 1 0];
%printsys(compNum, compDen)

% plant
poli = [-1/tau -1/tau];
zeri = [];

[numPlant, denPlant] = zp2tf(zeri', poli', k);
```

```
% compensator-plant
[numC_P, denC_P] = series(compNum, compDen, numPlant, denPlant);

% T(s) = Y(s)/R(s)
[a, b] = feedback(numC_P, denC_P, [1] , [1]);
[numd, dend] = series(preNum, preDen, a, b);

%printsys(numd, dend)
```

References

W. Banzhaf, P. Nordin, R. E. Keller, F. D. Francone (1998) *Genetic Programming - An Introduction*, Morgan Kaufmann, San Francisco, CA and dpunkt, Heidelberg.

A. Callender, A. B. Stevenson (1939) *Automatic Control of Variable Physical Characteristic.* U.S. Patent 2,175,985. Filed February 17, 1936 in United States. Filed February 1935 in Great Britain. Issued October 10, 1939 in United States.

Richard C. Dorf, Robert H. Bishop (1997) *Modern Control System Analysis & Design Using Matlab & Simulink*, Seventh Edition, Addison Wesley Longman, Inc. Menlo Park Californa.

Richard C. Dorf, Robert H. Bishop.(2001) *Modern Control Systems*, Ninth Edition, Upper Saddle River. N.J.: Prentice Hall.

P. J. Fleming, R. C. Purshouse (2002) *Evolutionary algorithms in control system engineering: a survey.* Control Engineering Practice, Vol 10. 2002, p.1223-1241.

Gary J. Gray, David J. Murray-Smith, Yun ¡Li, Ken C. Sharman, Thomas Weinbrenner (1997) *Nonlinear model structure identification using genetic programming.* Control Engineering Practice, Vol 6. 1998. p.1341-1352.

Mo Jamshidi, Leandro dos Santos Coelho, Renato A. Krohling, Peter J. Fleminig (2002). *Robust Control System with Genetic Algorithms*, CRC Press.

H. S. Jones (1942) *Control Apparatus.* United States Patent 2,282,726. Filed October 25, 1939. Issued May 12, 1942.

J. R. Koza (1992) *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press: Cambrigde, MA, 1992.

J. R. Koza (1994) Introduction to genetic programming. In Kinner, Kenneth E. Jr (editor).*Advances in Genetic Programming*, Cambridge, MA, The MIT Press. Pages 21-45. Chapter 2.

J. R. Koza (1995) *Survey of Genetic Algorithms and Genetic Programming* Procedings of 1995 WESCON Conference. Piscataway, NJ: IEEE. 589 - 594

J. R. Koza, Forrest H. Bennet, David Andre (1999) *Method and Apparatus for Automated Design of Complex Structures Using Genetic Programming* U.S. Patent 5,867,397. Issued Feb 2. 1999.

J. R. Koza, Martin A. Keane, Jessen Yu, Forrest H. Bennett III, William Mydlowec (2000) *Automatic Creation of Human-Competitive Programs and Controllers by Means of Genetic Programming.* Genetic Programming and Evolvable Machines, 1, p121-164.

J. R. Koza, Martin A. Keane, Jessen Yu, Forrest H. Bennet, William Mydlowec (2003) *U.S. Patent 6,564,194. Method and Apparatus for Automatic Systesis Controllers.* Issued May 13. 2003.

G. P. Liu, S. Daley (2001) *Optimal-tuning PID control for industrial systems.* Control Engineering Practice, Vol 9, 2001. p.1185-1194.

The MathWorks Inc. (2002) *Using the Control System Toolbox* Version 5, Online only, Revised for Version 5.2 (Release 13). July 2002.

The MathWorks Inc. (2002) *Using Simulink* Version 5, Fifth Printing, Revised for Simulink 5 (Release 13). July 2002.

The MathWorks Inc. (2002) *Simulink Reference* Version 5, Fifth Printing, Revised for Simulink 5 (Release 13). July 2002.

T. Morimoto, Y. Hashimoto (2000) *AI approaches to identification and control of total plant production systems.* Control Engineering Practice, Vol 8. 2000. p.555-567.

Chester L. Nachtigal (1990) *Instrumentation and Control, Fundamentals and applications* Wiley Interscience Publication, John Wiley & Sons, Inc.

Katsuhito Ogata (1997) *Modern Control Engineering*, Third Edition, Upper Saddle River. N.J.:Prentice Hall.

C. Vlachos, D. Williams, J.B. Gomm (2001) *Solution to the Shell standard cotrol problem using genetically tuned PID controllers.* Control Engineering Practice, Vol 10. 2002, p.151-163.

K. J. Åström, T. Hägglund (2001) *The future of PID control.* Control Engineering Practice, Vol 9, 2001. p.1163-1175.

K. J. Åström, T. Hägglund (1995) *PID Controllers: Theory, Design, and Tuning* Second Edition. Research Triangle Park, N.C.