The Internal Performance of Iterative Feedback Tuning

By

Jaston Sikaundi

July 2008

A thesis submitted in fulfilment of the requirements for the degree of Master of Science in Electrical Engineering at the University of Cape Town.

The financial assistance of the Department of Labour (DST) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the DST.

Declaration

I, Jaston Sikaundi, declare that this dissertation is my own unaided work apart from those sections that have been acknowledged. It is being submitted for the degree of Master of Science in Electrical Engineering at the University of Cape Town. It has not been submitted for the award of any degree or examination at this or any other university.

Signature Signed by candidate

Date 07/10/08

Acknowledgements

I would firstly like to thank my supervisor Professor Martin Braae for suggesting this thesis topic and for his continuous support throughout the project. I would like to thank my friends for being there for me when times were hard and providing the stress relief that was needed.

I would like to thank my mother for believing in me when I doubted myself. Most importantly I would like to thank my father, the late Dr Martin Ben Sikaundi for insisting that I complete a Masters degree and giving me hope for the future.

Finally I would like to thank God my saviour for bringing me back up every time I fell down.

Abstract

Under certain conditions Iterative Feedback Tuning (IFT) may produce a controller that cancels the poles of the process and as a result can give a closed loop that has poor internal performance. The disadvantage of this is that the closed loop will have poor input disturbance rejection. A solution for ensuring that IFT does not have poor internal performance is to make sure that the disturbance rejection is adequate.

However an adequate input disturbance may lead to other undesirable dynamics in the closed loop performance. These are such as overshoot in the response for setpoint tracking and that for output disturbance rejection. On the other hand the advantage of pole shifting is that for a one degree of freedom control structure all the characteristic equations of the loop transfer functions will be the same.

Four methods are proposed for avoiding pole-zero cancellation by concentrating on the input disturbance. These methods are using: a model for input disturbance rejection, time-weighted IFT for disturbance rejection, a setpoint-tracking model with overshoot and approximate pole placement IFT. Approximate pole placement IFT was chosen as the best method. The reason is that the dynamics of the closed loop can be specified with the choice of characteristic equation. This method was then investigated further to establish its feasibility on a physical system.

After the evaluation of this method, it was applied on a DC motor for speed control to show that it is viable in practice. Multiple experiments were done to show that this method does not produce a controller that cancels the process poles, confirming it as a good solution to prevent poor internal performance.

Table of Contents

List of Figures

List of Tables

List of Symbols

#Samples	The maximum number of samples in an experiment interval
APPIFT	Approximate Pole Placement Iterative Feedback Tuning
APPVRFT	Approximate pole placement virtual reference feedback tuning
ASM	Algorithmic state machine
b_n / bn	Coefficient of s^n in the controller numerator
d(s) / d(t)	Output disturbance
e(s) / e	Error between setpoint and output in closed loop
em	Error between process output and model output
em(iftN)	The error between the closed loop output and the model output at sample iftN.
exp#	The experiment number
f(s) / f	Feedback element
g(s) / g	Process/ Plant
IFT	Iterative Feedback Tuning
IFT experiment one	The first IFT experiment
IFT experiment two	The second IFT experiment
iftN	The sample number
k(s) / k	Controller
m(s) / m	Model
N	Number of samples in one experiment
n(s) / n(t)	Sensor noise
p(s) / p	Prefilter
PI	Proportional plus integral
PID	Proportional plus integral plus differential
r / c(t)	closed loop setpoint
r(iftN)	The IFT setpoint at sample iftN during the ***IFT experiment two***
rIft	The IFT excitation at the closed loop input
rRef	The setpoint operating point of the algorithm
T	Time constant
u(s) / u	Process input
v(iftN)	The IFT input disturbance at sample iftN during the ***IFT experiment two***
V(s) / V	Transfer function from the input disturbance to the closed loop output
v(s) / v	Input disturbance
vIft	The IFT excitation at the process input.

VRFT	Virtual Reference Tuning
y(s) / y	Process output
ym(s) / ym	Model output
γ	Controller parameter update constant
ρ	Controller parameters
σ/η	Model parameters

Chapter 1

Introduction

1.1 SUBJECT OF THIS THESIS

This thesis investigates ways of improving the internal performance of an Iterative Feedback Tuning control loop, by avoiding pole-zero cancellation.

1.2 BACKGROUND TO THESIS

This thesis originated from previous research (Wei et al, 2006), where Iterative Feedback Tuning (IFT) was applied on a one degree of freedom control structure. The configuration that was investigated is that of a first order process $g(s) = \frac{A}{1+sT}$, a PI controller $k(s) = \frac{b_1 s + b_0}{s}$ and a first order model $m(s) = \frac{1}{1+sT_m}$. When IFT was applied to optimize the parameters of the controller, the resulting controller cancelled the process poles for all the investigated processes.

The consequence of cancelling process poles is that the closed loop may not meet the design requirements. If the process poles are unstable the closed loop will be ***internally unstable*** (Maciejowski, 1989). If the poles are stable yet slow and/or oscillatory the closed loop will exhibit poor ***internal performance*** (Laplant, 1999).

In the previous work an attempt was made to remove the adverse effects of pole-zero cancellation by applying a step input disturbance during the first IFT

experiment. The performance criterion was to minimise the effect that this input disturbance had on the output. The outcome was that the gain of the controller increased exponentially even though the effect of pole shifting was observed. It was also noted that the final controller parameters were dependent on the initial controller parameters.

As a result this project was called to investigate ways of preventing pole-zero cancellation and thus improving on the internal performance of an Iterative Feedback Tuning control loop.

1.3 OBJECTIVES OF THIS THESIS

The objectives of this thesis are to:

- Verify the extent to which pole-zero cancellation can occur with IFT.
- Find methods that prevent pole-zero cancellation and optimise the controller to shift the poles instead.
- Simulate the methods and apply the best method on a DC motor system to prove the results practically.
- Draw conclusions on the applied method and make recommendations for future work.

1.4 LIMITATIONS AND SCOPE OF THESIS

This project was to use the current method of performing Iterative Feedback Tuning written in Visual Basic 6. The investigation was to be done through researching articles found from the University of Cape Town library, through inter library loans and those accessible on the internet using search engines such as scholar.google.com.

1.5 PLAN OF DEVELOPMENT

This thesis begins with an overview of the relevant literature that was reviewed at the start of this project. The next chapter then describes the method and program used for the investigations. The document goes on to show that pole-zero cancellation does in fact occur in IFT under some conditions and to demonstrate problems that may occur because of this. Following this, four methods are proposed for preventing pole-zero cancellation. These methods are compared and the best method is chosen. The chosen method is then analysed in more depth and the results of its application on a DC motor are presented. Finally conclusions are made and future work is recommended.

Chapter 2

Literature Review

This chapter begins with the basics of iterative feedback tuning and an explanation of what is meant by the internal performance of a system. This is done as an essential introduction to the review.

2.1 BRIEF OVERVIEW OF ITERATIVE FEEDBACK TUNING

Most of the information used in this chapter was obtained from (Hjalmarsson et al., 1998 & Hjalmarsson, 2002). Iterative Feedback Tuning (IFT) is a closed loop automatic controller tuning technique. It was first proposed by Hjarlmarsson et al (1994), (Gevers, 2002). However the algorithm is thoroughly explained in (Hjalmarsson et al., 1998). The novelty of this algorithm is that the gradient of the error squared cost function for a limited complexity controller can be obtained through experiments on the closed loop as will be shown later.

IFT can be applied to both a one and a two degree of freedom control structure. The structure investigated in this thesis is a one degree of freedom control structure so as to create an industrial situation where there is only one controller in the loop. The one degree of freedom control structure is shown fig 2.1.1.

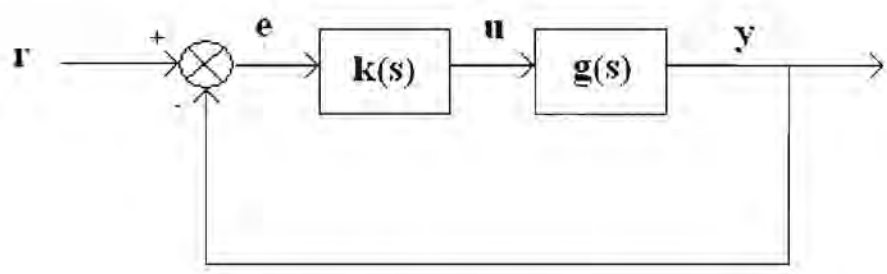


Fig 2.1.1- A one degree of freedom control structure

The output of this loop is $y = \frac{kg}{1+kg} r$.

The IFT process begins with the selection of a desired model that when perturbed with the same signal as the closed loop gives the desired response, *ym*. The difference between the model output and the actual closed loop is the model following error defined by *em* =*y*-*ym*. This arrangement is shown in fig 2.1.2.

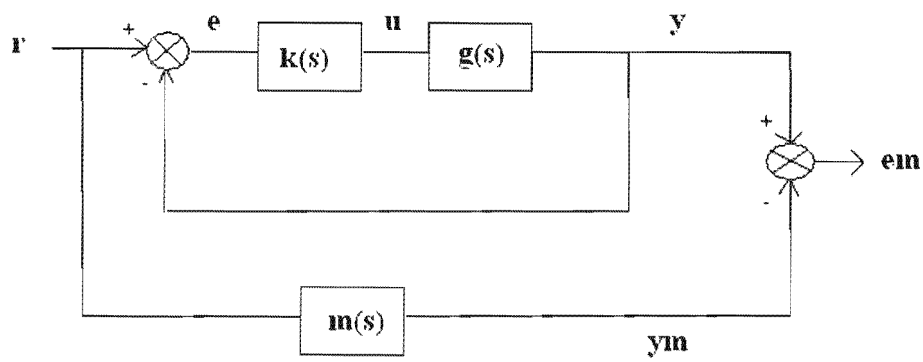


Fig 2.1.2- IFT Block diagram loop structure

After obtaining a model the next objective is to choose a differentiable, signal based cost function, J. The commonly used cost function is:

$$J = \frac{1}{2N}\left[\sum_{i=1}^{N}(L_y em_i)^2 + \lambda\sum_{i=1}^{N}(L_u u_i)^2\right] \qquad (2.1.1)$$

The objective of IFT is to minimize this cost function. By minimising this cost function, the error on the output is minimized while there is still a penalty on the control effort. The constant λ defines the weighting on this penalty. The number of samples taken in an experiment interval is N. The filters L_y and L_u are frequency weightings on *em* and *u* respectively. For this analysis these filters are set to one. The closed loop controller, k(s) has parameters that are tuned until the cost function is minimized.

To obtain the parameters of a controller that would minimize this cost function, the solution to the derivative of the cost function in equation (2.2.1) with respect to the controller parameters is needed. Thus the solution to the following equation is needed:

$$\frac{\partial J}{\partial \rho} = \frac{1}{N}\left[\sum_{i=1}^{N}(L_y em_i(\rho))\frac{\partial em_i}{\partial \rho}(\rho) + \lambda\sum_{i=1}^{N}(L_u u_i(\rho))\frac{\partial u_i}{\partial \rho}(\rho)\right] = 0 \qquad (2.1.2)$$

where ρ represents the controller parameters.

Using the gradient, $\frac{\partial J}{\partial \rho}$, the solution to the above equation can be obtained from the following controller update algorithm:

$$\rho_{i+1} = \rho_i - \gamma_i R_i^{-1}\frac{\partial J}{\partial \rho}(\rho_i) \qquad (2.1.3)$$

where R is a positive definite matrix such as a unit matrix or an approximation of the Hessian of J. The sequence γ_i is a positive scalar that determines the step size of a parameter update.

The signals needed to obtain $\frac{\partial J}{\partial \rho}$ are: $em\frac{\partial em}{\partial \rho}$ and $u\frac{\partial u}{\partial \rho}$. These can be obtained as follows:

$$\frac{\partial em}{\partial \rho} = \frac{\partial(y - ym)}{\partial \rho} = \frac{\partial y}{\partial \rho}$$

$$\frac{\partial y}{\partial \rho} = \frac{1}{k}\frac{\partial k}{\partial \rho}\frac{kg}{1+kg}(r - y) \qquad (2.1.4)$$

The differential $\frac{\partial k}{\partial \rho}$ is easily obtained because the controller is known. The signal $\frac{kg}{1+kg}(r - y)$ is obtained by passing the error signal as a setpoint into the

closed loop during the second experiment. Therefore the IFT procedure involves two experiments as follows:

IFT experiment one

The setpoint is set to r. During this experiment the control error signal $e = r\text{-}y$ and the closed loop output error, *em,* are collected.

IFT experiment two

After all transients have decayed the error signal, *e,* which was collected in the first experiment, is then used as the setpoint in the second experiment so as to obtain $\frac{kg}{1+kg}(r-y) = y_2$. This will be the output during the second experiment. This output y_2 is then filtered through $\frac{1}{k}\frac{\partial k}{\partial \rho}$ (which is known) so as to obtain $\frac{\partial y}{\partial \rho} = \frac{1}{k}\frac{\partial k}{\partial \rho} y_2$.

The second term of the gradient equation needs $\frac{\partial u}{\partial \rho}$ that is calculated as follows:

$$\frac{\partial u}{\partial \rho} = \frac{1}{k}\frac{\partial k}{\partial \rho}\frac{k}{1+kg}(r-y) \tag{2.1.5}$$

During the second experiment stated above the process input is:

$$u_2 = \frac{k}{1+kg}(r-y) \tag{2.1.6}$$

Therefore:

$$\frac{\partial u}{\partial \rho} = \frac{1}{k}\frac{\partial k}{\partial \rho} u_2. \tag{2.1.7}$$

Using all these signals the gradient of the cost function (2.1.2) is obtained as shown below.

$$\frac{\partial J}{\partial \rho} = \frac{1}{N}\left[\sum_{i=1}^{N} em_{1i}\frac{1}{k}\frac{\partial k}{\partial \rho}y_{2i} + \lambda\sum_{i=1}^{N} u_{1i}\frac{1}{k}\frac{\partial k}{\partial \rho}u_{2i}\right] \tag{2.1.8}$$

This is used to update the controller parameters.

2.2 INTERNAL PERFORMANCE OF A CONTROL SYSTEM

The criterion for the ***internal stability*** of a control loop is taken from (Maciejowski, 1989). The idea is that all the transfer functions from inputs to outputs must be stable for the closed loop to be internally stable. Consider the configuration in fig 2.2.1.

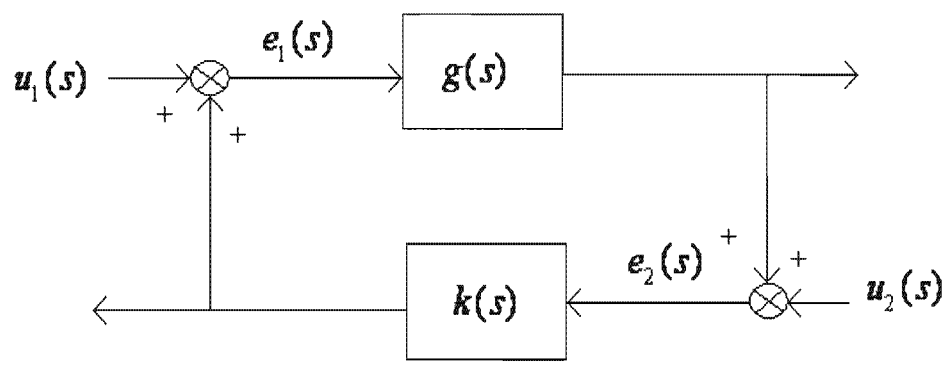


Fig 2.2.1- Control loop external and internal signals

If the loop transfer functions are defined such that:

$$\begin{bmatrix} e_1(s) \\ e_2(s) \end{bmatrix} = \begin{bmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{bmatrix}\begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix} \tag{2.2.1}$$

Then for the above loop to be internally stable, the transfer functions $H_{11}(s)$, $H_{12}(s)$, $H_{21}(s)$ and $H_{22}(s)$ must all be stable i.e. they must not contain any poles on the right half of the s-plane. Therefore even if the k(s) zeros cancelled the unstable poles of g(s) the loop would not be internally stable.

At the same time, good ***internal performance*** is not guaranteed for a system that is internally stable. The controller zero can cancel slow and/or oscillatory process poles thus making the response to an input disturbance sluggish or undesirable and thus exhibiting poor internal performance (Laplant, 1999).

Consequently a closed loop obtained using IFT that leads to poor ***internal performance*** is one that is achieved through pole-zero cancellation and thus retains a poor response to certain loop inputs. The internal performance of a one degree of freedom control structure can be tested by adding an input disturbance (Wei et al., 2006). The reason is that the transfer function from the input disturbance to output will retain these undesirable open loop process poles. This will be shown in section 4.1.

2.3 ROBUST ITERATIVE FEEDBACK TUNING

IFT is a well studied automatic controller tuning technique and has been applied on numerous processes (Hjalmarsson and Gevers, 2003). There have been many attempts to improve the performance of the system. The main issues of concern have been the speed of convergence (Hildebrand et al, 2005a & Solari, 2005) and robustness of IFT (Prochazka et al, 2005 & Lecchini et al, 2006). This suggests that IFT sometimes leads to a system that is not robust.

In (Solari, 2005) IFT is compared to pole placement design. Pole placement is referred to as the "robust approach" to loop design. IFT design was only better than the pole placement design when the complexity of the controller was increased. Therefore the performance of the IFT loop depends on the complexity of the controller.

In (Lechinni et al., 2006) it is shown that the H_2 criterion used for IFT does not necessarily give a robust closed loop. This paper uses Loop Transfer Recovery and LQG (Linear Quadratic Gaussian) synthesis techniques that are designed to

give a robust feedback loop. IFT for loop transfer recovery involves putting synthetic white noise as an input and output disturbance when performing IFT.

In (Prochazka et al., 2005) a new criterion is chosen that has both robustness and performance of the system included. This paper studies the one degree of freedom controller that is the focus in this thesis. It involves the use of a new cost function that is the sum of two cost functions, one is the regular cost function and the other is the term added for robustness. It also involves shaping the sensitivity function. The second term is weighted by the user according to the design requirements. However this will only change the minimum point of the cost function which will depend on the weighting chosen by the user.

2.4 POLE SHIFTING, POLE CANCELLING AND POLE PLACEMENT

Pole shifting is a technique used to shift the closed loop poles away from the positions of the open loop to place them in different, better positions. When poles are not cancelled in the open loop they are shifted in the closed loop. Therefore shifting poles is the alternative to cancelling the poles. The disadvantage of cancelling poles is that the slow poles remain the closed loop response to an input disturbance. In the paper by Wei (Wei et al., 2006) it was shown that IFT resulted in a controller that cancelled the process poles. The paper discusses a pole shifting method; however the gain of the controller could not be limited.

Middleton and Graebe (1999) state that cancelling slow stable poles is a design trade off between good input disturbance rejection and robustness. It is shown that when poles are shifted, there is overshoot in the step response for setpoint tracking because there is always a dominant zero. Thus a pole-shifting design is stated to have a negative impact on robustness.

Pole placement on the other hand relies on shifting the loop poles to the desired location. Solari (2005) states that pole placement is the robust approach to design and is in disagreement with the above conclusion. With pole placement the zero positions are neglected and the pole location is chosen according to a Diophantine equation. A final desired characteristic equation is chosen by the user. The zeros and gain of the loop controller are then adjusted in such a manner that the poles are located in that specific position. Lecchini and Gevers (2002) give a method intended for non-minimum phase processes, which does approximate pole placement for IFT. It does this by not having a fixed model but a model with adjustable zeros. These zeros are tuned at the same time as the controller parameters. This procedure would have been the right approach to avoid pole-zero cancellation, however the controller that is obtained in the simulation in (Lecchini and Gevers, 2002) has a zero that cancels a process pole. A summary of their method is found in Appendix A.3.

Middleton and Graebe (1999) affirm that it is a well known fact that when doing pole shifting a two degree of freedom controller is needed to prevent excessive overshoot. Chen and Seborg (2002) in agreement state that the overshoot can be removed by using a filter in the path of the setpoint. This filter can be seen as transforming the closed loop into a two degree of freedom control structure.

2.5 DISTURBANCE REJECTION IN IFT

In process control disturbance rejection is more important than setpoint tracking (Chen and Seborg, 2002). It is also one of the main applications of IFT (Hildebrand et al., 2005a). There have been cases where IFT focuses on output disturbance rejection as reported in (Hildebrand, 2005a, 2005b & Wahlberg, 2005) and on the input disturbance rejection in (Hjalmarsson et al., 1998, Wei et al., 2006 & Sobota and Schlegal, unknown).

In this project the focus will be on the cases for input disturbance rejection because the poor internal performance of a loop results in bad input disturbance rejection (Laplant, 1999). In (Wei et al., 2006), when the controller was tuned to

reject input disturbances, the gain of the closed loop increased exponentially. References (Hjalmarsson et al., 1998 & Sobota and Schlegal, unknown) managed to improve on input disturbance and have an optimal controller. The reason is that the cost functions were similar in the sense that they included the penalty on the control effort. This limited the gain of the controller.

The cost function in (Wei et al., 2006) did not include the second term in equation (2.1.1) i.e. the penalty on the control effort. The reason was that it is intrinsically defined in the choice of the model. This is useful because the literature gave no means of determining a value for the constant λ that is used in equation (2.1.1). Therefore λ can only be found using trial and error and knowledge of the process. This makes the procedure less automatic and thus less desirable.

The version of IFT performed by Wei et al (2006) had both IFT experiments happen by injecting the signals through the input disturbance. The resultant transfer function, V(s), from the input disturbance, v(s), to the output response, y(s) is:

$$y = \frac{g}{1+kg} v = V(s)v(s) \tag{2.5.1}$$

The cost function gradient then becomes:

$$\frac{\partial J}{\partial \rho} = \frac{1}{N}\left[\sum_{i=1}^{N} Vv \frac{\partial k}{\partial \rho} V^2 v\right] = \frac{1}{N}\left[\sum_{i=1}^{N} y_{1i} \frac{\partial k}{\partial \rho} y_{2i}\right] \tag{2.5.2}$$

This version of IFT rejects input disturbances. However as mentioned previously the gain of the controller increases exponentially.

2.6 USING A PI/PID CONTROLLER

One of the main applications for IFT is with the use of a PI/PID controller. The reason is IFT is designed for a limited complexity controller and the majority of closed loops have PID controllers (Lequin et al., 2002 & Sobota and Schlegal, unknown). Since the situation that led to pole-zero cancellation in (Wei et al., 2006) used a PI controller it is discussed in this section.

There has not been any literature that focuses on the fact that in IFT the controller zero may cancel the process pole, apart from (Wei et al., 2006). There is a different type of pole-zero cancellation documented in (Hjalmarsson et al., 1998 & Lechinni and Gevers, 2002). This is when the controller zero cancels its integrating pole meant for setpoint tracking. In (Lechinni and Gevers, 2002) the controller has a zero that cancels the process pole however this is not highlighted. Wei et al (2006) performed IFT using PI controller on a first order process. In this case the optimum PI controller was one with a zero that cancelled the process pole.

Chen and Seborg (2002) tune PI and PID controllers based on synthesis and disturbance rejection. The controllers are tuned for disturbance rejection in a closed loop for different types of processes. It is stated that the setpoint trajectory is usually acceptable but that it can be separately tuned using a weighting factor on the setpoint. When tuning a PI controller for optimisation it is suggested that the desired closed loop model is defined as one with two poles. When tuning for input disturbance rejection, the input disturbance rejection model has two poles and this model can be achieved with the use of a PI controller. When tuning a second order system the closed loop model can be defined as one with three poles and the controller is a PID controller to achieve that model. The desired model must also contain the dead time of the process. For example for a first order process the disturbance rejection model must be of the form:

$$m(s) = \frac{k_d s e^{-\theta s}}{(t_c s + 1)^2} \tag{2.6.1}$$

where k_d is a constant, t_c a time constant and θ is the delay. However it should be noted that the resultant model will have a time constant longer that t_c because the two poles are located on the same place. For a second order process a disturbance rejection model is:

$$m(s) = \frac{k_d s e^{-\theta s}}{(t_c s + 1)^3} \tag{2.6.2}$$

where the coefficients are the same as stated above.

Lequin et al (2002) compare the tuning of a PID controller using IFT with that of other classical tuning methods used in industry. IFT is not done using a model but by having a mask period where there is no weighting on the cost function. This mask period is started large and is decreased with iteration until the desired mask period is achieved if the closed loop does not oscillate. This is compared to the response obtained by the other classical tuning methods. IFT was found to either give a resultant response that either matched that of the classical tuning methods or one that bettered them. The weighting, λ, on the penalty on the control effort was kept at zero for the majority of the experiments. The authors also mention that the user of IFT is not really concerned about the transient but that the setpoint is reached with overshoot but no oscillations. However having a model on a system is a way of limiting the control action therefore the need for a penalty on the control effort is not needed (Wei et al., 2006).

2.7 VIRTUAL REFERENCE FEEDBACK TUNING

One of the main problems with IFT is that it optimizes the cost function locally (Solari, 2005 & Hildebrand et al., 2004). Therefore the initial values of the controller parameters may dictate if the system will ever settle to the global optimum. The settling time of the algorithm depends on many things but mainly

on the parameter update constant. If this constant is too large then the process may never settle to a minimum. If on the other hand it is too small it will take a long time to reach the optimum. Therefore it is beneficial to ensure that the initial parameters are close to the optimal before the IFT procedure begins.

Virtual reference feedback tuning (VRFT) is a technique that has one global minimum. It has the advantage that it is a one shot technique therefore it gives a final controller after one iteration. However it is not always accurate for a closed loop model that cannot be followed exactly with the class of loop controllers being used. However IFT can converge to an optimum therefore the two techniques are complimentary. VRFT can be used to get the initial values of the closed loop and IFT can be used to "fine tune" or "polish" these parameter values (Campi et al., 2002).

Nonetheless there is the difficulty of performing the VRFT technique when working with transfer functions in the s-plane. The procedure requires the inverse of the desired closed loop model. Majority of the models used do not contain a zero therefore when they are inverted the result is a dynamic model that is non-causal. The solution to this may be to add a non-dominant filter to simulate the inverted model output, however the locations of the poles of this non-dominant filter are limited by the sampling time of the program. VRFT is discussed and the equations of a case study are developed in Appendix A.1.

From the literature review it was evident that there was a gap in the literature in terms of IFT performing pole-zero cancellation and ways of avoiding it. The next chapter discusses the programs that will be used to first verify that pole-zero cancellation does actually occur with IFT in view of the fact that it is not mentioned in any literature apart from (Wei et al., 2006).

Chapter 3

Iterative Feedback Tuning Program

The IFT program used in this thesis was written in Visual Basic 6. It was originally written by Wei et al (2006). The programs presented in this chapter will be used in the subsequent chapters to investigate pole-zero cancellation in IFT.

3.1 IFT PROGRAM

Alterations were made to the program allowing it to reach the predicted results. Such as the optimal closed loop controller is no longer dependent on the initial controller values and IFT can be done at any reference setpoint. Modifications were also made on the IFT program done through perturbation of the input disturbance.

3.2 SIMULATIONS

For simulations the output of the transfer function of the desired closed loop model, the output of the various closed loop filters as well as of the IFT calculations was done within a Runge-Kutta (RK4) algorithm. The program was originally written to work on real time using a timer interval. However for simulation purposes this was changed to calling the IFT function repetitively so as to speed up the simulation process.

3.3 IFT PROGRAM FOR SETPOINT TRACKING

This thesis mainly focuses on systems consisting of a first order process $g(s)=\frac{A}{1+sT}$ and a PI controller $k(s)=\frac{b_1 s+b_0}{s}$ and closed loop desired model $m(s)=\frac{1}{1+sT_m}$. Therefore the controller parameter updates using equation (2.1.3) are:

$$\rho_n = \rho_{n-1} - \gamma R^{-1} em \times \frac{\partial J}{\partial \rho}$$

And so:

$$b_{0n} = b_{0(n-1)} - \gamma R^{-1} em \times \frac{1}{b_1 s + b_0} y_2 \qquad (3.3.1)$$

$$b_{1n} = b_{1(n-1)} - \gamma R^{-1} em \times \frac{s}{b_1 s + b_0} y_2$$

Figure 3.3.1 is an Algorithmic State Machine (ASM) diagram describing how the IFT program works for setpoint tracking. The explanation of this ASM is given below.

The abbreviations are:

- **rRef** is the setpoint operating point of the algorithm.
- **rIft** is the IFT excitation at the closed loop input.
- **iftN** is the sample number.
- **exp#** is the experiment number
- **#Samples** is the maximum number of samples in an experiment interval
- **em(iftN)** is the error between the closed loop output and the model output at sample iftN.
- **r(iftN)** is the IFT setpoint at sample iftN during the second experiment.

The program is idle and waits for the timer interval to elapse. When the timer interrupt occurs the sample number is increased. Depending on the sample number there are two options that program can follow.

Suppose that the IFT sample number is less than the maximum sample number and the experiment number is one. ***IFT experiment one*** is performed. During this experiment the setpoint is the sum of reference setpoint (**rRef**) and the IFT setpoint (**rIft**). The setpoint error (**r(iftN)** = r – y) and the error between the model output and closed loop output are collected. This is performed at each timer interrupt until the IFT sample number (**iftN**) exceeds the maximum sample number (**#Samples**).

When the IFT sample number exceeds the maximum sample number, the experiment number is increased to two. During this experiment number two (a dummy experiment used for the purpose of returning the loop to its initial status) the IFT setpoint is removed and the setpoint is the reference setpoint rRef. This is kept constant for the duration of one experiment to allow the closed loop to settle in preparation for experiment number three which is ***IFT experiment two***.

Once again when the maximum IFT sample number is exceeded then the experiment number is increased to three. The states of the filters used to calculate the gradients are then cleared and ***IFT experiment two*** is performed. During this experiment the reference signal is now a sum of the reference signal and the error collected during the first experiment i.e. r = rRef + r(iftN). The closed loop output is now collected and subtracted from the reference signal. The reason is that it is the change in output that is required. The gradient of the cost function is calculated using this difference. This occurs until the maximum number of samples is exceeded.

When this maximum number of samples is exceeded the controller parameters are updated and the experiment number is set to one and the cycle repeats until the optimal parameters have been obtained.

This program is executed in Visual Basic in a number of slightly different forms in the investigations in this thesis.

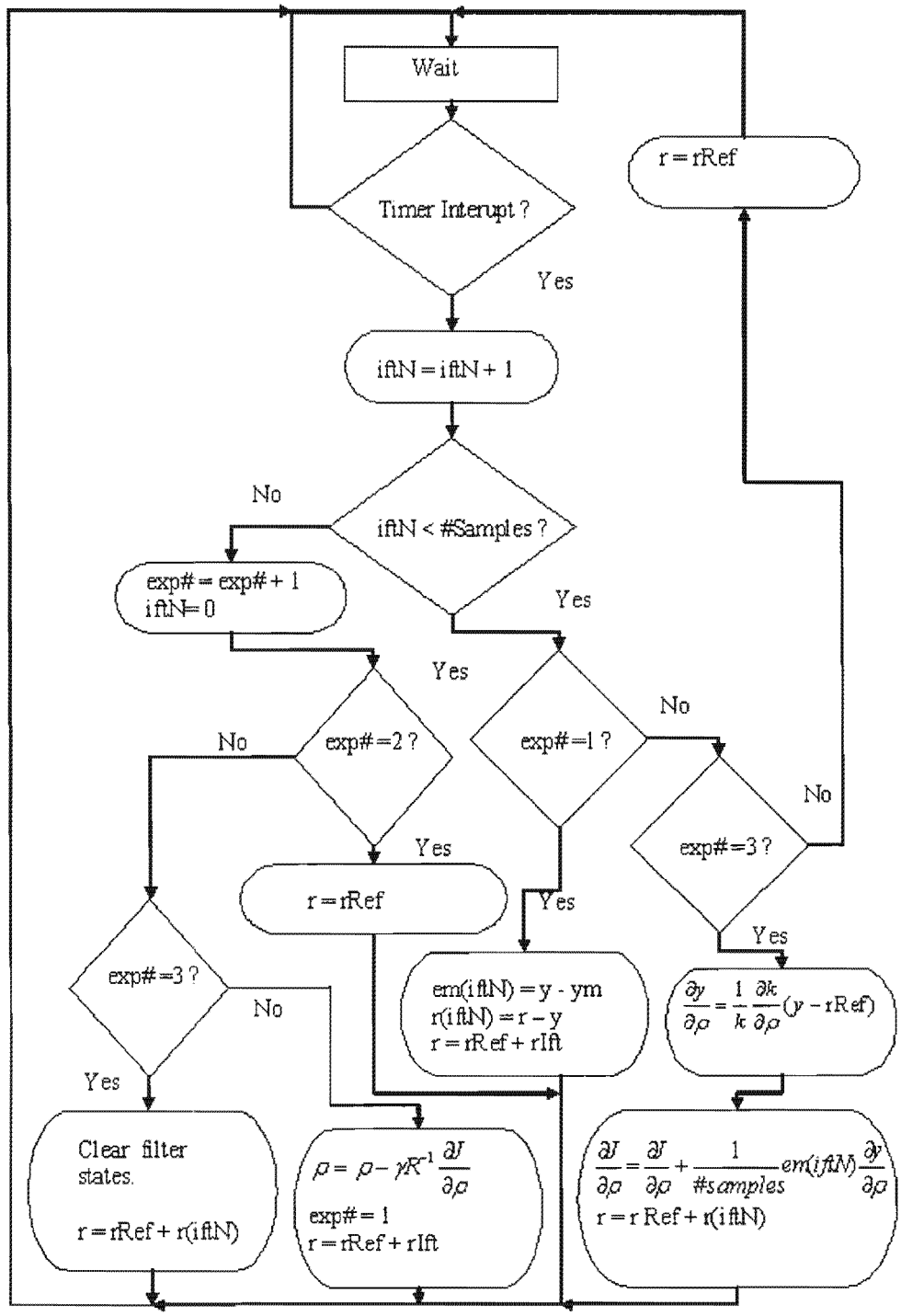


Fig 3.3.1 - IFT program for setpoint tracking ASM

3.4 IFT PROGRAM FOR DISTURBANCE REJECTION

Using the same case study as in section 3.3, the program algorithm has a slight modification. This was necessary to take care of the differences in both the transfer functions and the calculations. The controller parameter updates for the PI controller occur in the following manner.

$$\rho_n = \rho_{n-1} - \gamma R^{-1} em \times \frac{\partial J}{\partial \rho},$$

Therefore:

$$b_{0n} = b_{0(n-1)} + \gamma R^{-1} em \times \frac{1}{s} y_2 \qquad (3.4.1)$$

$$b_{1n} = b_{1(n-1)} + \gamma R^{-1} em \times y_2$$

For this process there are only two experiments because at the end of every experiment the output returns back to the reference point. This method is shown in the ASM diagram in fig 3.4.1. The explanation is below.

The additional abbreviations used in this figure are stated below:

- **vIft** is the IFT excitation at the process input.
- **v(iftN)** is the IFT input disturbance at sample iftN during the second IFT experiment.

A brief explanation of the ASM follows.

The program is idle and waits for the timer interval to elapse. When the timer interrupt occurs the sample number is increased. Depending on the sample number there are two options that program can follow.

Suppose that the IFT sample number is less than the maximum sample number (**#Samples**) and the experiment number is one. ***IFT experiment one*** is

performed. During this experiment the setpoint is reference setpoint (**rRef**) and the input disturbance is the IFT input disturbance (**vIft**). The change in the closed loop output (y-rRef) and the error between the model output and closed loop output (y-ym) are collected. This is performed at each timer interval until the IFT sample number (**iftN**) exceeds the maximum sample number (**#Samples**).

When the IFT sample number exceeds the maximum sample number, the experiment number is increased to two. The states of the filters used to calculate the gradients are then cleared and ***IFT experiment two*** is performed. During this experiment the setpoint is kept at the reference setpoint, **rRef** and the input disturbance is now a combination of the IFT input disturbance (that has been rejected) and the output collected during the first experiment i.e. v = vIft +v(iftN). The difference in the closed loop output (y- rRef) is collected and used to calculate the gradient of the cost function. Again this process occurs until the maximum number of samples is exceeded.

When this maximum number of samples is exceeded the controller parameters are updated and the experiment number is set to one. The input disturbance is removed making it equivalent to a negative input disturbance and like before the whole cycle repeats until the optimal parameters have been obtained.

This program is used in a number of variations in the investigations for this thesis.

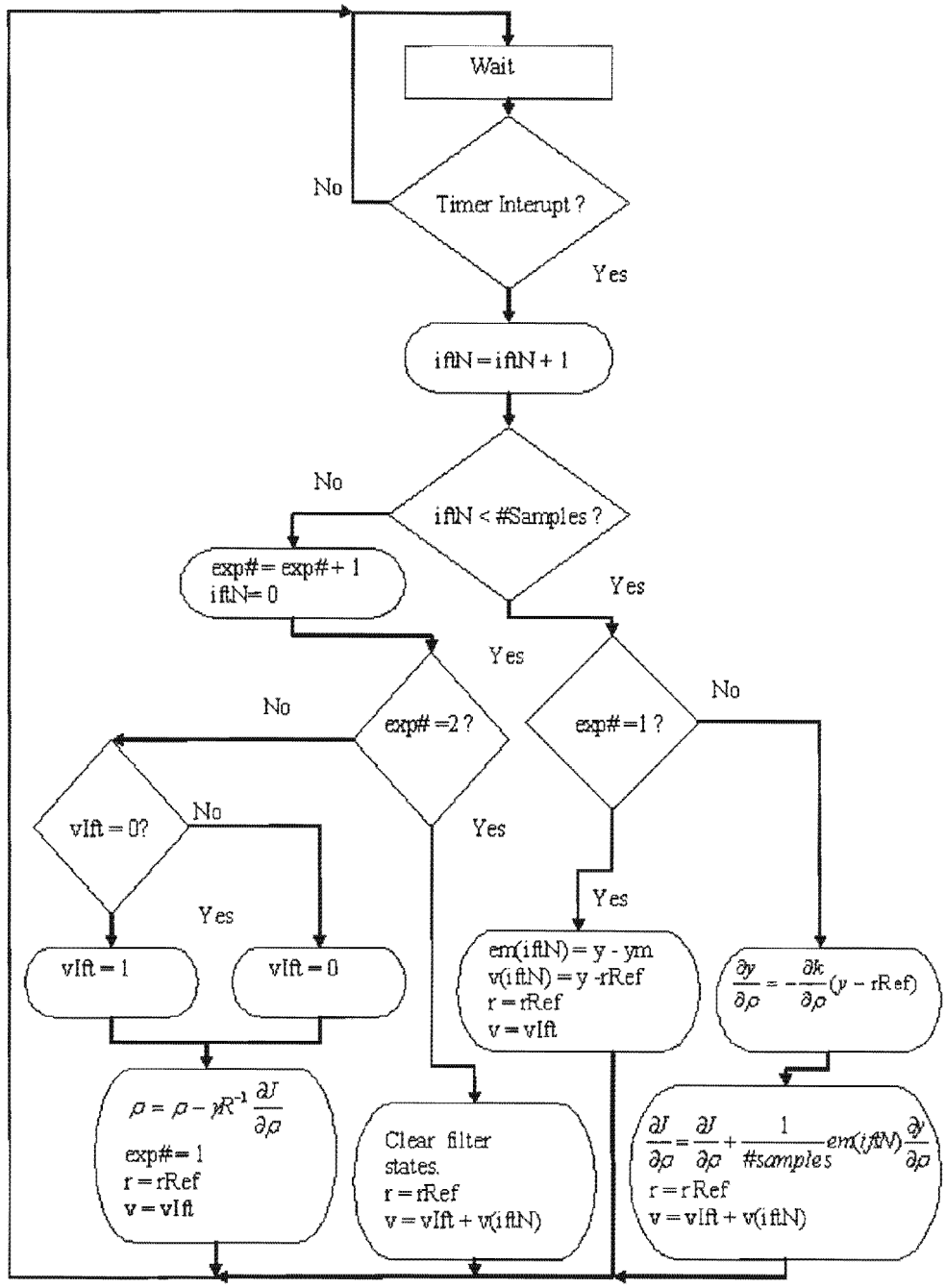


Fig 3.4.1- IFT for input disturbance rejection ASM

The IFT investigations for the next chapters are performed using variants of the programs presented in this chapter.

Chapter 4

Pole-Zero Cancellation in IFT

Pole-zero cancellation can occur in some systems and may not be desirable for plant performance. This chapter analyses the effect that such cancellation has on the loop transfer functions. It also verifies that this can also occur in IFT systems because the program used previously (Wei et al., 2006) had some errors. This was also ascribed to gaps in the literature, as discovered through the literature survey. The verification is done through a theoretical demonstration, a simulation demonstration and a practical implementation.

4.1 EFFECT OF POLE-ZERO CANCELLATION ON LOOP TRANSFER FUNCTIONS

Consider a well-known general control loop as shown below.

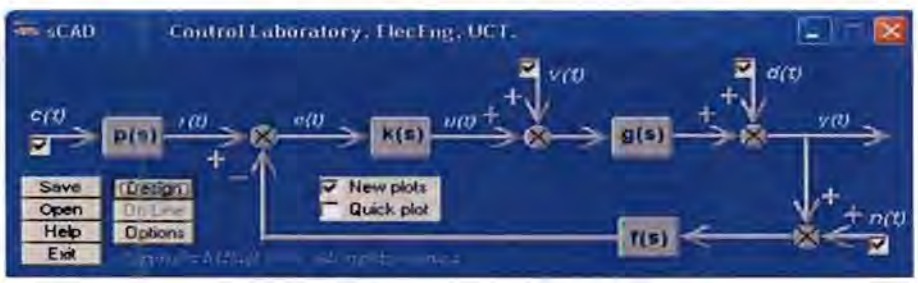


Fig 4.1.1- IFT for input disturbance rejection

The process is the transfer function g(s), the prefilter is p(s), the feedback element is f(s) and the loop controller is k(s). The command signal is c(t), the input disturbance is v(t) and the output disturbance is d(t). Signal y(t) is the process output, e(t) is the loop error and n(t) is the sensor noise.

There are twelve transfer functions that map the external signals to the closed loop responses. They are shown in the equation below and are structured as ratios of polynomials, with subscripts denoting the block. Thus the process model is $g(s) = \frac{n_g}{d_g}$.

$$\begin{bmatrix} y \\ e \\ u \end{bmatrix} = \begin{bmatrix} \frac{n_p n_g n_k d_f}{d_p(d_g d_k d_f + n_g n_k n_f)} & \frac{d_g d_k d_f}{d_g d_k d_f + n_g n_k n_f} & \frac{d_k d_f n_g}{d_g d_k d_f + n_g n_k n_f} & \frac{-n_g n_k n_f}{d_g d_k d_f + n_g n_k n_f} \\ \frac{n_p d_f d_g d_k}{d_p(d_g d_k d_f + n_g n_k n_f)} & \frac{-n_f d_g d_k}{d_g d_k d_f + n_g n_k n_f} & \frac{-n_f d_k n_g}{d_g d_k d_f + n_g n_k n_f} & \frac{-n_f d_g d_k}{d_g d_k d_f + n_g n_k n_f} \\ \frac{n_p n_k d_f d_g}{d_p(d_g d_k d_f + n_g n_k n_f)} & \frac{-n_k n_f d_g}{d_g d_k d_f + n_g n_k n_f} & \frac{-n_k n_f n_g}{d_g d_k d_f + n_g n_k n_f} & \frac{-n_k d_g n_f}{d_g d_k d_f + n_g n_k n_f} \end{bmatrix} \begin{bmatrix} c \\ d \\ v \\ n \end{bmatrix}$$

For ease of reference, the above transfer functions can be represented as the matrix equation:

$$\begin{bmatrix} y \\ e \\ u \end{bmatrix} = \begin{bmatrix} h_{yc} & h_{yd} & h_{yv} & h_{yn} \\ h_{ec} & h_{ed} & h_{ev} & h_{en} \\ h_{uc} & h_{nd} & h_{uv} & h_{un} \end{bmatrix} \begin{bmatrix} c \\ d \\ v \\ n \end{bmatrix} \qquad (4.1.1)$$

It can be seen that some of these transfer functions are identical as is shown with the different colour loops. As a result an engineer will have nine of the twelve transfer functions to optimize. These are h_{yc}, h_{yd}, h_{yv}, h_{yn} (or h_{uv}), h_{ec}, h_{ed} (or h_{en}), h_{ev}, h_{uc}, h_{ud} (or h_{un}).

Looking at the nine transfer functions it can be seen that transfer functions h_{yv} and h_{ev} will retain the poles of the process model g(s) when pole-zero cancellation occurs i.e. when $n_k = d_g$. If these poles of the plant are slow then there will be a slow input disturbance rejection. If need be, the slow poles can be cancelled from transfer function h_{ev} using $n_f = d_g$. Note however that the poles of g(s) cannot be cancelled using p(s), f(s) or k(s) in the case of transfer function h_{yv}. Therefore the characteristic equation for the rejection of input disturbances, v(t), can only be improved using pole-shifting.

The output response of a closed loop system with pole-zero cancellation is shown in the fig 4.1.2. The process model is $g(s) = \frac{1}{1+10s}$. The controller chosen is $k(s) = \frac{0.5(1+10s)}{s}$. p(s) and f(s) are both one.

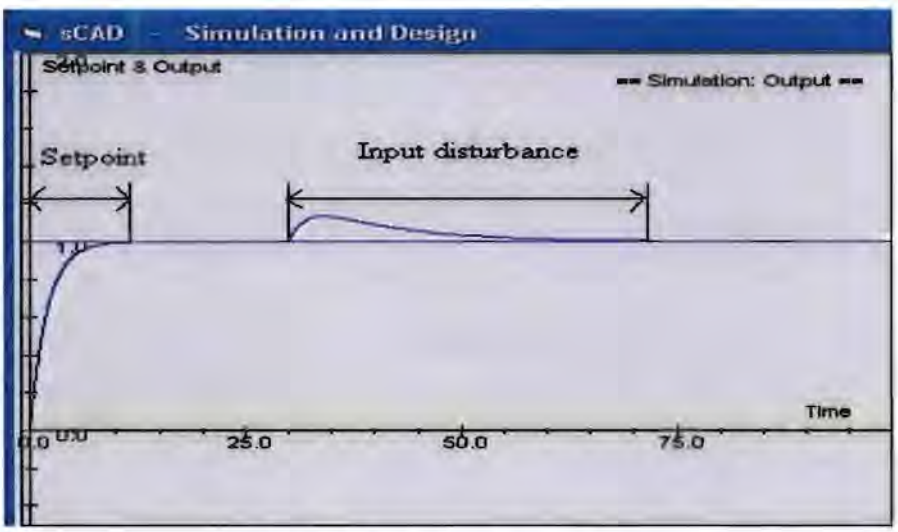


Fig 4.1.2- Poor internal performance

Clearly the input disturbance imposed at time, t = 30 seconds, takes a longer time to be rejected than the output takes to reach the setpoint, changed at time t=0 seconds. This figure confirms that the slower process pole remains in the transfer function h_{yv}.

These observations motivate the current research and emphasize the significance of studying input disturbance rejection.

4.2 DEMONSTRATION OF POLE-ZERO CANCELLATION IN IFT

Consider the case where f(s) and p(s) are set to 1 so that the focus of analysis is entirely on the controller, k(s). A common industrial situation is considered in

which the dominant process dynamics are first order, $g(s) = \frac{A}{1+sT_p}$, and the controller takes the form of a traditional PI controller, $k(s) = \frac{b_1 s + b_0}{s}$ with two tuneable parameters. The desired model for the closed loop is $m(s) = \frac{1}{1+sT_m}$. The reasons for choosing a desired first order closed loop model, m(s), is that it is simple and it reduces the need to specify an explicit limit on the control law because the plant input u is implicitly defined by the model since: $u = \frac{m}{g} r$, (Wei et al., 2006).

4.2.1 Theoretical Demonstration

The closed loop transfer function equals that of the model for setpoint tracking

when $h_{yr} = \frac{kg}{1+kg} = m$ (4.2.1)

Rearranging this equation gives:

$$k = \frac{1}{g} \times \frac{m}{1-m}$$

$$= \frac{d_g}{n_g} \times \frac{n_m}{d_m - n_m} \quad (4.2.2)$$

The optimal controller for (4.2.1) can only be obtained if it belongs to the class of controllers being tuned.

For the above case study (4.2.2) gives:

$$k = \frac{1}{AT_m} \times \frac{1+sT_p}{s} \quad (4.2.3)$$

It is shown below that the controller that gives a minimum of this cost function (with $\lambda = 0$) is one that has zeros at the same place as the poles of the process.

This was confirmed by simulation in (Wei et al., 2006). This controller belongs to the class of PI controllers. Clearly IFT performed on the case study will result in pole-zero cancellation given that the loop controller being tuned is also PI.

4.2.2 Simulation Demonstration

IFT is operated on a plant with process model $g(s)=\frac{1}{1+5s}$, the desired closed loop model is $m(s)=\frac{1}{1+2s}$ and the initial controller is $k(s)=\frac{1+1s}{s}$. The update constant is $\gamma=\frac{0.01}{|\text{Cost function Gradient}|}$ thus preventing large changes in the controller parameter values and is small enough to settle close to the optimum. The initial output signals are shown in Fig 4.2.1 below.

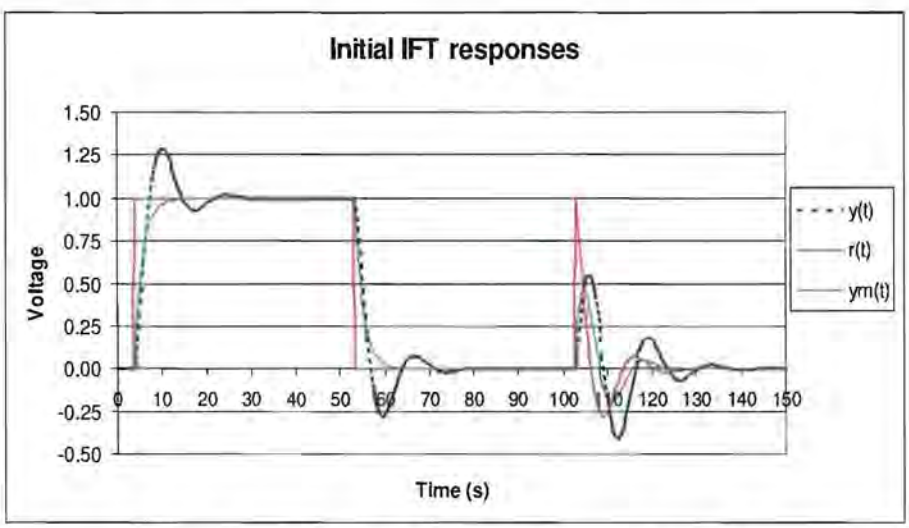


Fig 4.2.1- Simulation of IFT signals with the original controller

The above graph shows the initial IFT signals in simulation. The closed loop y(t) is different from that of the model ym(t). The IFT algorithm aims to minimise the difference between these two responses. ***IFT experiment one*** occurs from 4 seconds till 54 seconds. The reference signal is taken back to zero in preparation for ***IFT experiment two*** beginning at 104 seconds. The final results after the controller parameters have reached the optimum are shown in the fig 4.2.2 below.

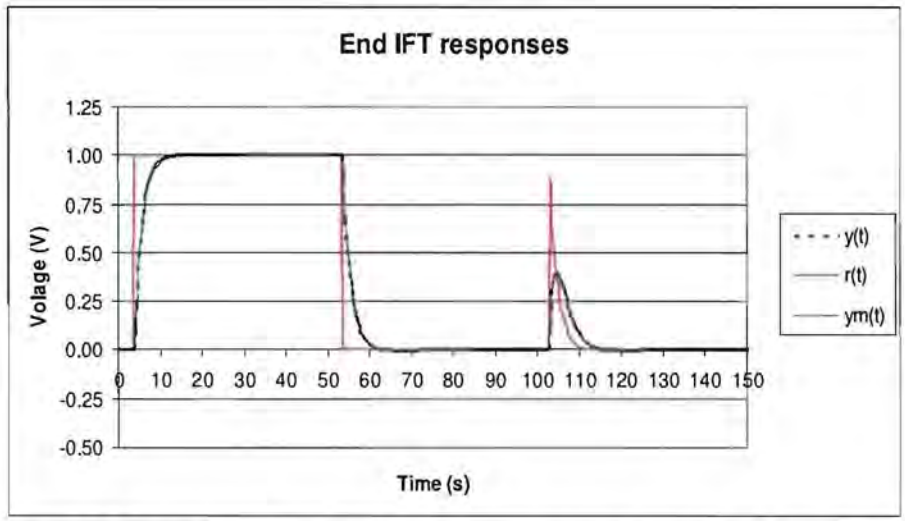


Fig 4.2.2- Simulation of IFT after the controller parameters have reached the optimal

The above graph shows the signals after the cost function has been optimized. The IFT process has achieved its goal of minimizing the difference between the model output and the closed loop output. The figure below shows how the controller zero varies with the IFT cycles.

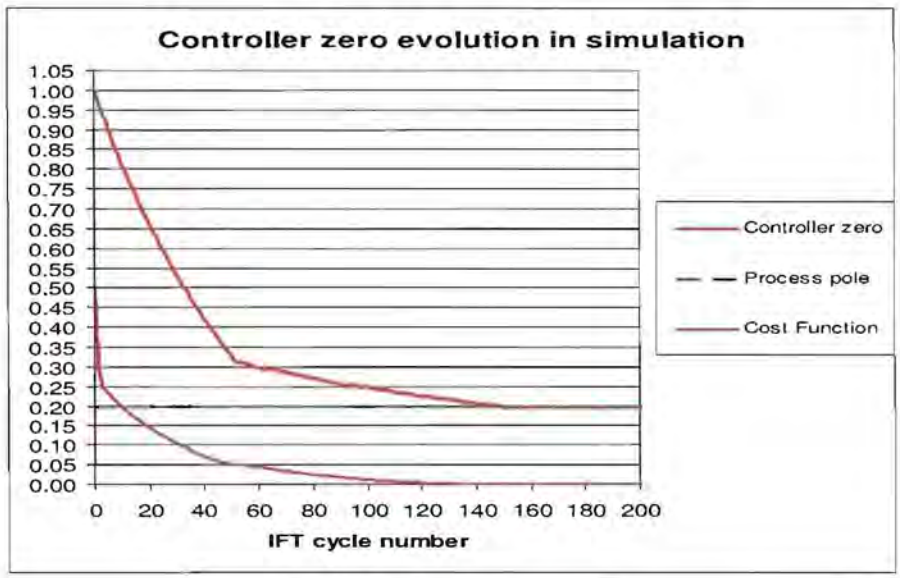


Fig 4.2.3- Controller zero evolution

The standard IFT algorithm tuned the controller parameters to $k(s)=\frac{0.5(1+5s)}{s}$ as expected from in equation (4.2.3). The cost function is minimal when the controller zero cancels the process pole. A correction to the program meant that the final controller is no longer dependent on the initial controller and was the same as that obtained theoretically.

4.2.3 Physical Implementation

Consider a DC motor that is set up for speed control. Step tests give a first order transfer function of $g(s)=\frac{3.0\pm0.10}{1+(7.3\pm0.34)s}$ when speeding up from approximately 5V to 8V and $g(s)=\frac{3.0\pm0.10}{1+(10.3\pm0.4)s}$ when slowing down from approximately 8V to 5V. The reason for choosing this voltage range is to avoid the non-operational voltage range of the motor around zero volts and to be around a velocity that is fast. This DC Motor and conditions of this experiment are discussed in section 8.1.

The controller to be tuned is once again a PI controller with initial controller as $k(s)=\frac{1+1s}{s}$ and the model chosen is $m(s)=\frac{1}{1+2.5s}$. The update constant gamma is $\gamma=\frac{0.01}{|\text{CostfunctionGradient}|}$. The chosen update constant was the same as that used in section 4.2.2 to allow the parameters to reach to within 0.01 of the optimum values. This gave an update vector that was constant in magnitude therefore allowing an update even when the cost function gradient was small and the parameters had not reached the optimum values. This model was to be tracked when the motor was speeding up. Figure 4.2.4 shows the initial closed loop and that of the model to the IFT signals.

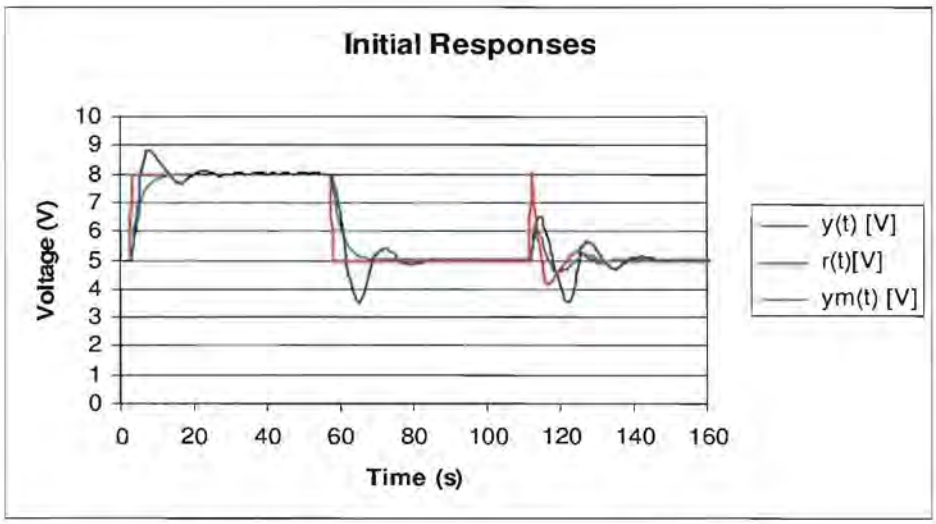


Fig 4.2.4- Initial IFT responses for upward tracking model

The closed loop output above, y(t), is oscillatory and takes a longer period of time to settle. As before the initial step up around t= 4 seconds is the IFT setpoint used to collect data for the second experiment. The setpoint is then stepped down around 54 seconds to prepare for ***IFT experiment two*** that begins at 104 seconds. The motor has different transfer function when speeding up and decelerating. Therefore in this attempt the closed loop output is recorded when the motor is speeding up from 5V to 8V is to track the model. The graph below shows the response of the system after 200 IFT cycles.

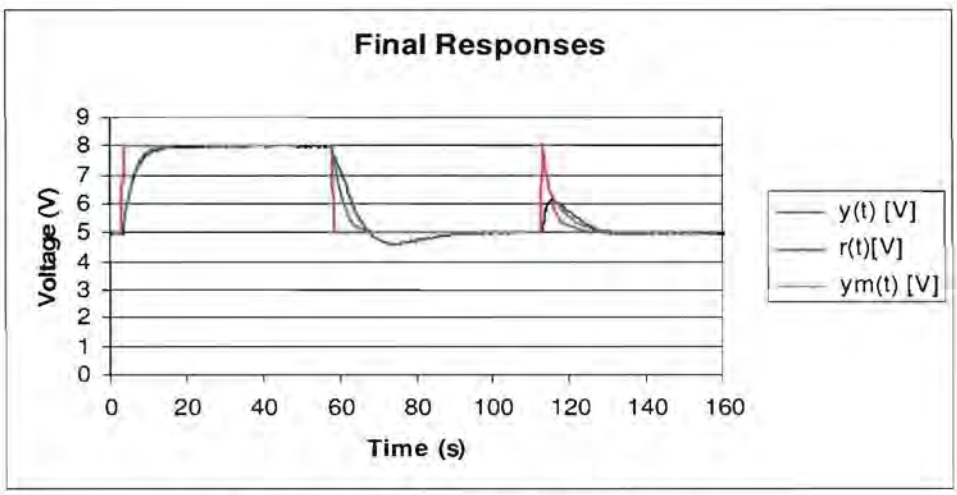


Fig 4.2.5- IFT responses after 300 IFT cycles

It is obvious from the difference in the two graphs that IFT algorithm minimised the difference between the output of the closed loop and that of the model. The decelerating response differed however and can be explained by the fact that the downwards model for the motor is significantly different from the upwards one. The focus was to track the upward motion of the motor and which the algorithm was successful in doing. The variation of the controller with IFT cycle is shown in fig 4.2.6.

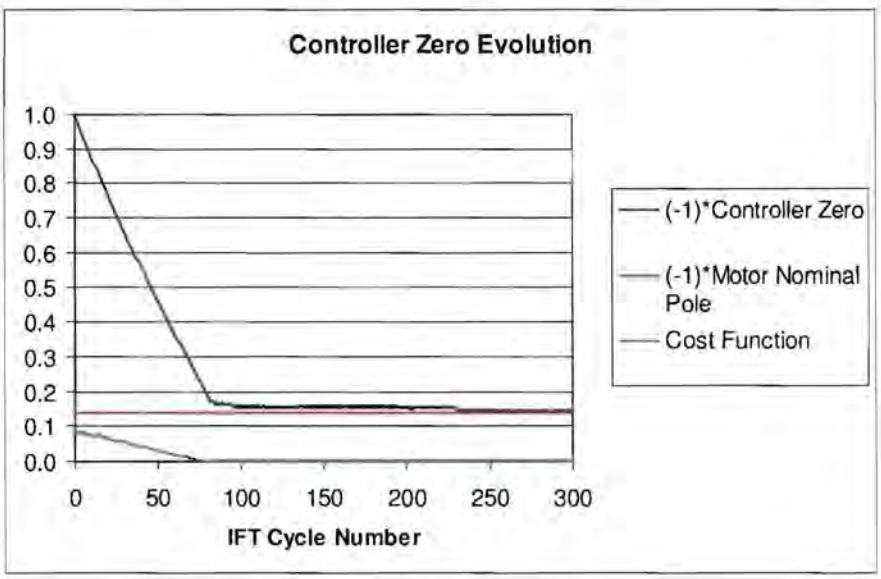


Fig 4.2.6- Controller Zero Evolution when using model $m(s) = \frac{1}{1+2.5s}$

Clearly the cost function gradient is at a minimum at the point when the controller zero enters the region of the process pole. This shows that even in a real world situation under certain conditions there may be pole-zero cancellation. The above experiment was repeated for a model of $m(s) = \frac{1}{1+3s}$ to see if it also leads to pole-zero cancellation for a different model as the equations show. The controller zero evolution is shown below.

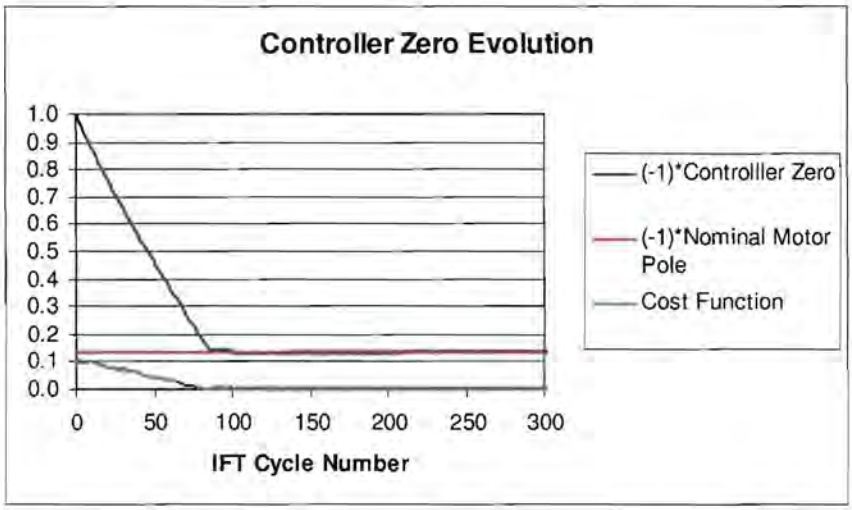


Fig 4.2.7- Controller zero evolution when using model $m(s) = \frac{1}{1+3s}$

Figure 4.2.7 confirms that under the experimental conditions pole-zero cancellation will occur even when the model is changed. To see if pole-zero cancellation would occur on a process with a different model IFT was done to track the response of the motor when decelerating. The initial response is shown in fig 4.2.8

.

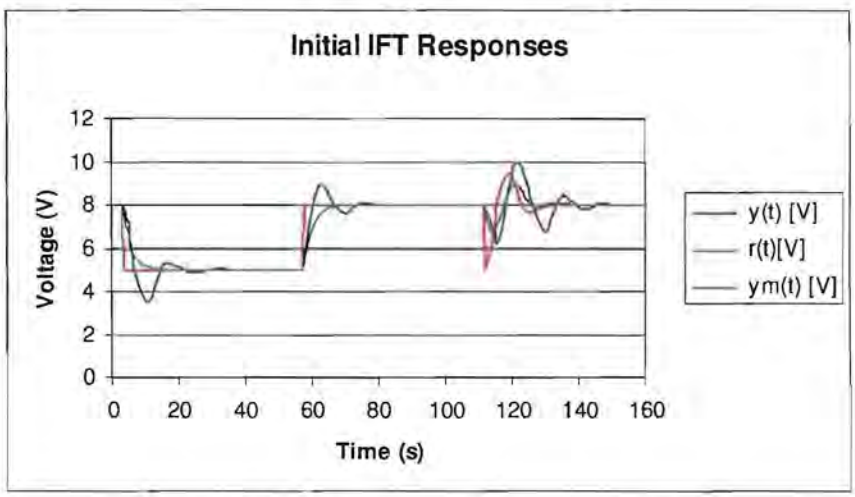


Fig 4.2.8- Initial IFT responses for downward tracking

The above response used the initial controller $k(s)=\frac{s+1}{s}$. It is clear that the process output and the model are very different. Figure 4.2.9 below shows that after IFT has been performed the model and the output are more similar.

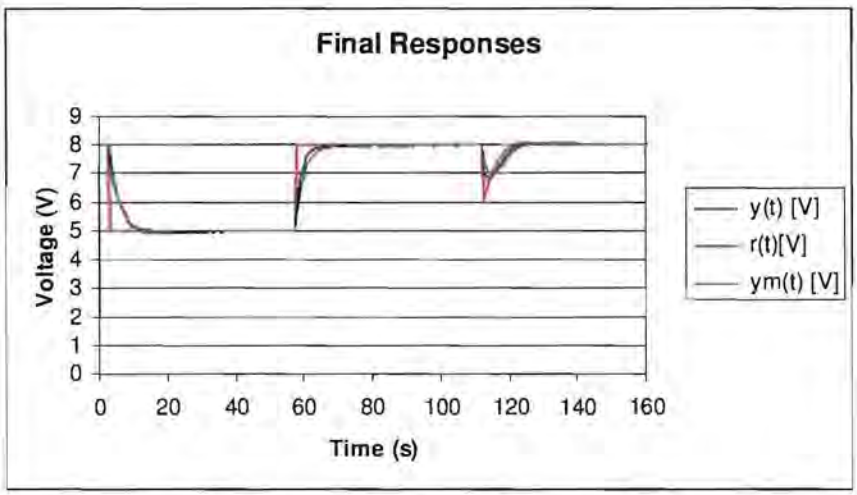


Fig 4.2.9- Response after IFT has been performed

The response above was obtained by performing IFT over two days because the optimum parameters had not been attained on the first day. On the second day IFT was performed with the update constant gamma reduced to half its original magnitude to see if the limit cycling of the controller zero position will be reduced. The response in fig 4.2.10 shows the controller zero evolution for the 200 IFT cycles on the first day.

In this figure the controller zero appears to oscillate about the process pole. The controller zero evolution for the continuation on the second day is shown in fig 4.2.11. It can be seen that the controller zero oscillates about the region of the nominal motor pole; however by a reduced amount. The reason for not settling was because the magnitude of the update vector was too large compared to the controller parameters. However it is clear that the process pole attracts the controller zero.

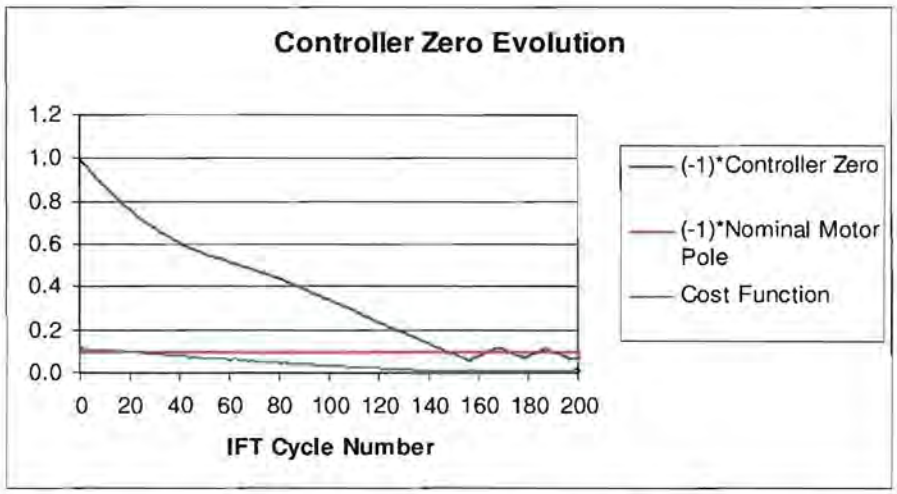


Fig 4.2.10- Controller zero evolution on day one

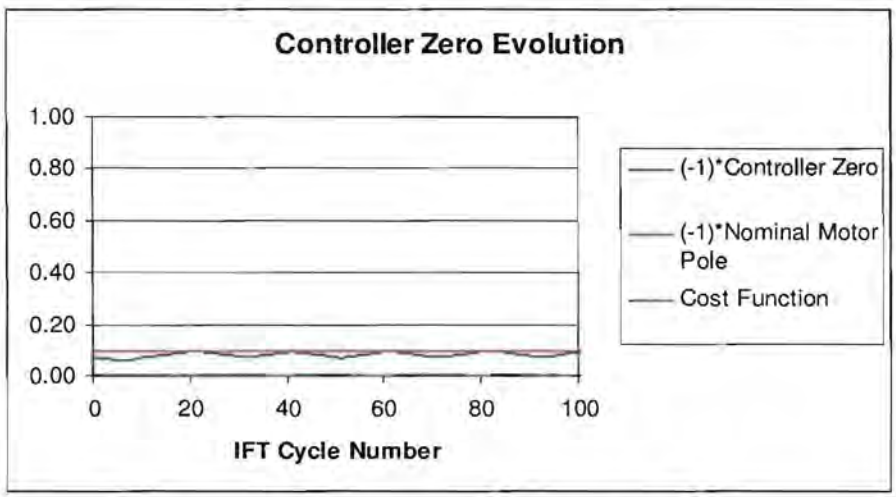


Fig 4.2.11- Controller zero evolution on day two

This chapter motivated the research into preventing pole-zero cancellation. The methods investigated are presented in the following chapter.

Chapter 5

Preventing Pole-Zero Cancellation in IFT

In this chapter four methods to prevent pole-zero cancellation are proposed. The methods are assessed, and the best method is chosen. These methods are explored using simulation in Microsoft Visual Basic 6 on the case study presented earlier i.e. A first order process with a PI controller.

The four methods proposed and investigated in this chapter are as follows:

1. Using an input disturbance rejection model
2. Using time-weighted input disturbance rejection
3. Using an setpoint tracking model with overshoot since it is shown in (Middleton and Graebe, 1999) that whenever there is pole-shifting there is overshoot, and
4. Using approximate pole-placement Iterative feedback tuning.

Unless otherwise stated, the experiments in this chapter are done using the process model $g(s) = \frac{1}{1+5s}$.

5.1 USING AN INPUT DISTURBANCE REJECTION MODEL

This method originated as a result of (Wei et al., 2006) where there is no model for input disturbance rejection (i.e. m(s) = 0). It is shown that pole-shifting occurred in this circumstance. The result was however accompanied by an exponential increase in the magnitude of the controller parameters with every IFT cycle. The reason is that for the output not to be affected at all by an input disturbance the transfer function from the process input to output, $h_{yv} = \frac{g}{1+kg}$ must be zero at all frequencies. Since g is fixed, this transfer function can only

be zero at all frequencies if the controller k has an infinite gain at all frequencies.

A method that can prevent this gain from rising to infinity was shown in (Hjalmarsson et al., 1998 & Sobota and Schlegal, unknown) where IFT was done through the setpoint but with an input disturbance during the first experiment. It involves choosing a non-zero value for λ in equation (2.1.1). This penalises the control effort and thus prevents the gain from rising. However different values of λ will give different minimum points for the cost function and thus different values for the controller parameters. Some that may cancel the process poles. Since a system is needed that automatically tunes itself, choosing λ may be difficult.

To avoid this, the suggested method in this section is to use a model for input disturbance as shown in fig 5.1.1 (p(s) = f(s) = 1 and n = d = r =0). This will also involve setting λ in the equation (2.1.1) to 0. Recall that for setpoint tracking $u = \frac{m}{g} r$ (Wei et al., 2006). For input disturbance rejection this becomes:

$$u = \frac{m-g}{g} v \tag{5.1.1}$$

Therefore there is still no need to penalise the control effort as it is done implicitly with the choice of the input disturbance rejection model.

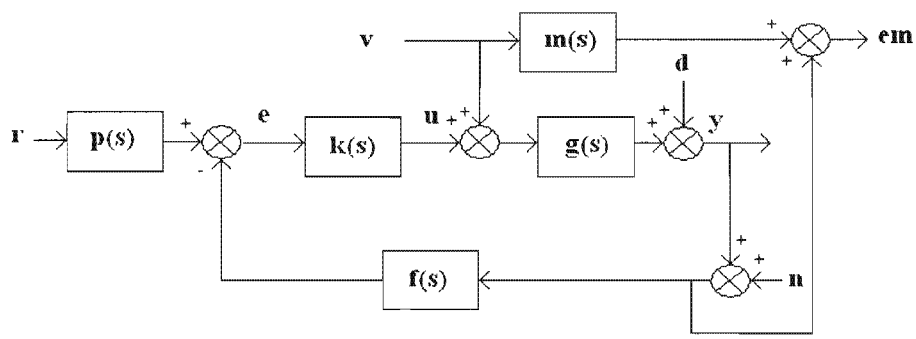


Fig 5.1.1- Control loop with input disturbance rejection model

The problem now becomes one of choosing a model for the rejection of input disturbances. Instead of choosing a setpoint tracking model, an equivalent input disturbance rejection model can be used. As shown earlier in equation (4.1.1), if

pole-zero cancellation does not occur, all the loop transfer functions (for a one degree of freedom control structure) will have the same characteristic equations. Since pole-zero cancellation was shown to occur for the case study in chapter 4, a first order input disturbance rejection model of the form:

$$m(s) = \frac{ds}{1 + T_m s} \tag{5.1.2}$$

is attempted. T_m will determine the time constant of the input disturbance rejection while. The ratio $\frac{d}{T_m}$ will determine the initial value of the model output when it is subject to a step in its input disturbance. The design requirements are in the form of a model that has a high frequency gain of one, a damping factor of one and to settle to within 0.1% of its initial value in 8 seconds (i.e. $7T_m=8$). The model used for experimentation is $m(s) = \frac{1.143s}{1+1.143s}$.

Table 5.1.1 shows where this version of IFT places the closed loop poles for different processes. The update parameter γ was chosen to be $\frac{0.01}{|\text{Cost function gradient}|}$ since this allows a parameter update even for a small cost function gradient and gives a reasonable error even though the magnitude of the update vector was constant. It also prevents excessive changes in the controller parameters.

TABLE 5.1.1- Effect of 1st order model on closed loop pole positions

g(s)	Closed loop pole positions	Damping factor	Settling Time (7T)
$\frac{1}{1+2s}$	-0.534+/-j0.783	0.563	13.1
$\frac{1}{1+3s}$	-0.418+/-j0.726	0.499	16.7
$\frac{1}{1+4s}$	-0.352+/-j 0.685	0.457	19.9
$\frac{1}{1+5s}$	-0.307+/-j0.653	0.425	22.8

None of the processes resulted in a closed loop that satisfies the design requirements. The slower the process pole the slower and the less damped the resultant closed loop was. This led to a change in the type of model used.

Having a first order model puts a strain on the initial value of the output when there is a step input disturbance. For example in the above situation when there is a unit step into the model, there is a unit height in the output before decay. This can be seen by looking at $y_v = \frac{g}{1+kg} v$ and using the initial value theorem.

When there is a step input to the input disturbance in a closed loop having a PI controller and a first order plant the output becomes:

$$y_v = \frac{\frac{A}{T}s}{s^2 + \frac{(Ab_1+1)}{T}s + \frac{Ab_0}{T}} \times \frac{B}{s} \qquad (5.1.3)$$

The initial value of this system is always zero. This means that the model is unachievable. Consequently what is needed is a second order model or higher. Therefore the second order model chosen is one with the same settling time as the previous first order model and one that has a high damping factor, e.g. $m(s) = \frac{cs}{s^2 + 1.75s + 1}$ the poles are at s= -0.875+/-j0.4841 giving a damping factor of 0.875. The variable c can be adjusted to limit the maximum value of the output response. The process chosen for experimentation is $g(s) = \frac{1}{1+5s}$. Its pole is at s = - 0.2. IFT through the input disturbance is then applied for different maximum changes of the plant output as a fraction of the input disturbance. This is shown in Table 5.1.2.

This method avoids pole-zero cancellation. As can be seen as c is decreased so does the settling time and hence the damping factor of the final system. It is suggested that IFT is started with a conservative (large) value of c and if this does not meet the speed or damping factor requirement, c be decreased until it

does. This method of defining input disturbance rejection may result in a good system. However the user cannot directly specify the closed loop damping factor and speed through the choice of the model.

TABLE 5.1.2- The effect of *c* on the closed loop pole positions

Fractional change	c	Controller Zero	Closed loop poles	Damping factor	Settling time (7T)
0.25	0.625	-0.988	-0.348+/-j0.608	0.497	20.1
0.20	0.500	-0.902	-0.407+/-j0.623	0.547	17.2
0.15	0.375	-0.807	-0.506+/-j0.632	0.625	13.8
0.10	0.250	-0.693	-0.716+/-j0.584	0.775	9.8
0.05	0.125	-0.801	-0.962+/-j0.675	0.819	7.3

This method avoids pole-zero cancellation. As can be seen as c is decreased so does the settling time and hence the damping factor of the final system. It is suggested that IFT is started with a conservative (large) value of c and if this does not meet the speed or damping factor requirement, c be decreased until it does. This method of defining input disturbance rejection may result in a good system. However the user cannot directly specify the closed loop damping factor and speed through the choice of the model.

The value of c is chosen through trial and error using simulation and as can be seen it is difficult to find a model that will give an output with the combination of damping factor and speed requirements that are specified in the characteristic equation. However an estimate value of A and T can be obtained through step test before IFT is performed. The data from these tests can produce better results because $c = \frac{A}{T}$. In this situation $c = \frac{1}{5} = 0.2$ would give the desired settling time and damping factor.

5.2 USING TIME WEIGHTED INPUT DISTURBANCE REJECTION

This method is based on the normal time-weighted IFT presented in (Hjalmarsson et al., 1998 & Gevers, 2002). The advantage of time weighting is that it takes the focus off the transient phase of the system response to a perturbation. The main disadvantage is that it may lead to an oscillatory system (Gevers, 2002) that may not be acceptable for most processes. Time-weighted IFT with respect to input disturbance rejection was not found in the literature. This section of the thesis attempts to do a time-weighted rejection of input disturbances. IFT is again done through the input disturbance as it is to reject input disturbances.

Gevers (2002) states that the simplest way to do time-weighted IFT is by making the weighting on the penalty on control effort (λu) one throughout the experiment and that on the model tracking component (y-ym) zero during the desired transient time and then one afterwards. In this situation the cost function that is to be optimised is as follows:

$$J = \frac{1}{2N}\sum_{i=1}^{N}(w_{ei}em_i^2 + w_{ui}\lambda u_i^2) \tag{5.2.1}$$

where w_e and w_u are the time weightings on u and e respectively. The process output, y and the process input, u are related to the input disturbance through the equations $y = \frac{g}{1+kg}v = Vv$ and $u = \frac{-kg}{1+kg}v$. The derivative of this cost function with respect to the controller parameters then becomes:

$$\begin{aligned}\frac{\partial J}{\partial \rho} &= \frac{1}{2N}\sum_{i=1}^{N} w_{ei}2em(\frac{\partial y}{\partial \rho} - \frac{\partial ym}{\partial \rho}) + w_{ui}2\lambda u\frac{\partial u}{\partial \rho} \\ &= \frac{1}{2N}\sum_{i=1}^{N} w_{ei}(-2em\frac{\partial k}{\partial \rho}(V^2v)) + w_{ui}(-2\lambda u_1(\frac{\partial k}{\partial \rho}(Vv) - k\frac{\partial k}{\partial \rho}(V^2v))) \\ &= \frac{1}{2N}\sum_{i=1}^{N} w_{ei}(-2em\frac{\partial k}{\partial \rho}(y_2)) + w_{ui}(-2\lambda u_1(\frac{\partial k}{\partial \rho}(y_1) - k\frac{\partial k}{\partial \rho}(y_2)))\end{aligned} \tag{5.2.2}$$

The above IFT algorithm is then applied on the process $g(s)=\frac{1}{1+5s}$. The disturbance rejection requirement once again is for the system to be within 0.1% of its final value (zero) in 8 seconds. This ensures that this method can be compared the previous other methods.

For this version of time-weighted IFT the setpoint is set to zero and there is a unit step input disturbance that is applied to the system during the first experiment. The gradient calculation uses only the data obtained after the desired settling time. When the output value is within 0.1% of the final value (0.001 because there is a unit step signal and the final value is zero) then the system is considered as having settled and there is no further calculation of the cost function because the error *em* and λ are set to zero. This ensures that it can be compared to the other methods where the required settling time too is to be at 0.1% of the final value within 8 seconds.

The final closed loop pole positions for different values of λ are shown in the table 5.2.1 below.

TABLE 5.2.1- Effect of λ on closed loop pole positions

λ	Closed loop poles	Damping factor	Controller Zero	Settling time (7T)
0.000	-0.568+/-j0.367	0.840	-0.488	12.32
0.001	-0.399+/-j0.314	0.786	-0.431	17.54
0.002	-0.370+/-j0.302	0.775	-0.422	18.92
0.005	-0.336+/-j0.279	0.769	-0.404	20.83
0.010	-0.317+/-j0.248	0.787	-0.373	22.08
0.020	-0.338+/-j0.146	0.918	-0.285	20.71

The closed loop damping factor initially reduces after there is a non-zero penalty on the control effort because the system is already damped when λ = 0. It can be seen that after λ = 0.005 any increase in λ results in an increase in the damping factor. The damping factor of this system can also depend on the required

settling time (Gevers, 2002). Using a non-zero value for λ reduces the gain of the controller and thus the real component of the pole and consequently the speed of response of the closed loop. However none of the closed loops above had the desired settling time. This leads to a re-evaluation of how the settling time is defined.

The manner of calculating settling time to within 0.1% applies to the setpoint tracking transfer function and not one for disturbance rejection. Therefore the theory of settling time applies to a second order rising process output and not one that is decaying. The relation of the input disturbance to output is $y = \frac{\frac{Ab1s}{T}}{s^2 + \frac{(Ab1+1)s}{T} + \frac{Ab0}{T}} v$ where T is the time constant of the process g(s).

Therefore to apply the rules of settling time this transfer function must be integrated to obtain $y_{new} = \frac{1}{s} y = \frac{\frac{Ab1}{T}}{s^2 + \frac{(Ab1+1)s}{T} + \frac{Ab0}{T}} v$. If this new transfer function has complex poles the output in the time domain will be of the form $y_{new} = D(1 - e^{-\frac{t}{T_c}} \sin(wt + \phi)$ where T_c is the time constant of the closed loop. Therefore when t = $7T_c$ then the integrated output is $y_{new} = D(1 - 0.000912\sin(w7T_c + \phi))$. Thus the process output is within +/- 0.1% of the final value at time t = $7T_c$. This transfer function can then behave in a manner that will allow for the settling time rules to be used. Where the output can be regarded as settled in $7T_c$.

This was programmed in the following manner: After 8 seconds have passed the cost function gradient is calculated as before. The settled value is obtained from experiment one and is assumed to be the last value of y_{new} after the whole sampling time. The performance requirement was for the output to settle in 8 seconds. Therefore in experiment two the cost function gradient was prevented from accumulating when y_{new} is within +/-0.1% of the final value. This is done by setting λ and *em* to zero. Figure 5.2.1 shows how this method is performed.

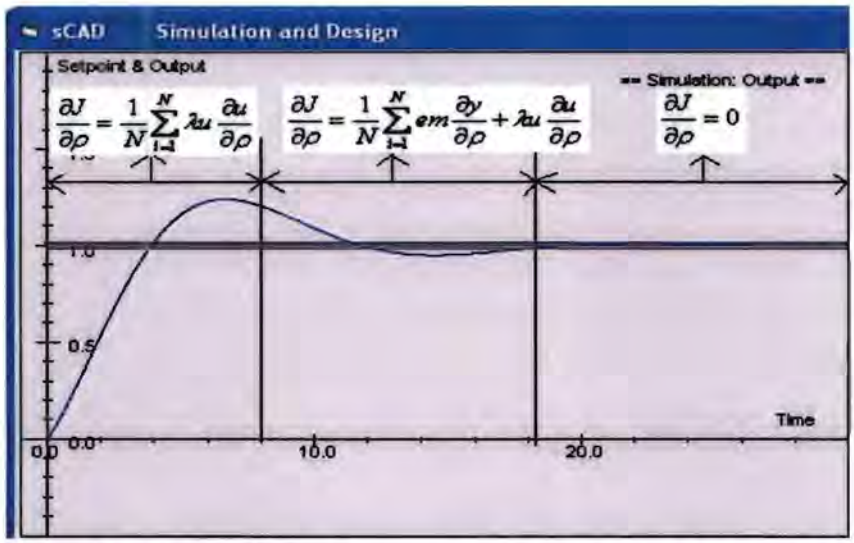


Fig 5.2.1- Diagram showing the manner in which time-weighted IFT is performed

The results of using this procedure on the process $g(s) = \frac{1}{1+5s}$ are shown in the table 5.2.2. Each experiment length is 30 seconds for λ = 0 – 0.02 and 40 seconds after that. The reason for this increase is that the closed loop becomes slower as λ increases. Once again the aim was to have the closed loop settle to +/- 0.1% of its final value in 8 seconds.

TABLE 5.2.2- Showing variation of settling time and damping factor with λ.

λ	Closed loop poles	Damping factor	Controller Zero	Settling time (7T)
0.000	-0.832+/-j0.368	0.915	-0.565	8.41
0.001	-0.403+/-j0.312	0.791	-0.429	17.37
0.002	-0.370+/-j0.299	0.777	-0.420	18.92
0.005	-0.339+/-j0.276	0.775	-0.400	20.65
0.010	-0.320+/-j0.246	0.793	-0.370	21.88
0.020	-0.331+/-j0.162	0.898	-0.294	21.15
0.030	-0.371, -0.289	1.000	-0.233	24.22
0.031	-0.494, -0.218	1.000	-0.210	32.11
0.032	-0.504, -0.205	1.000	-0.203	34.14

As expected when there is no penalty on the control effort the settling time of the input disturbance rejection is near the expected settling time of 8 seconds (with a 5% error). The error may be attributed to factors such as the way that IFT is performed. The reason is that the data obtained when y_{new} is in the +/- 0.1% band is neglected even if the output is not permanently in that band. The way that γ was defined also prevents it from reaching the exact optimum because of the constant update vector magnitude. Other sources of error may be numerical errors of the program and the transfer function output. However the acquired settling time is reasonable.

As λ is increased the second term in the criterion, equation (5.2.1), begins to dominate the cost function gradient. Thus the controller zero moves closer to the integration pole at s = 0. This is expected given that the integrator is responsible for the high gain of the controller at low frequency. The second term of the criterion is minimised when the gain is zero at all frequencies i.e. k(s) = 0. This can be seen by looking at the transfer function from the input disturbance to controller output $u = \frac{-kg}{1+kg}v$. Therefore u is smallest when k is zero at all frequencies. Figure 5.2.2 shows how the closed loop characteristics change with an increase in λ.

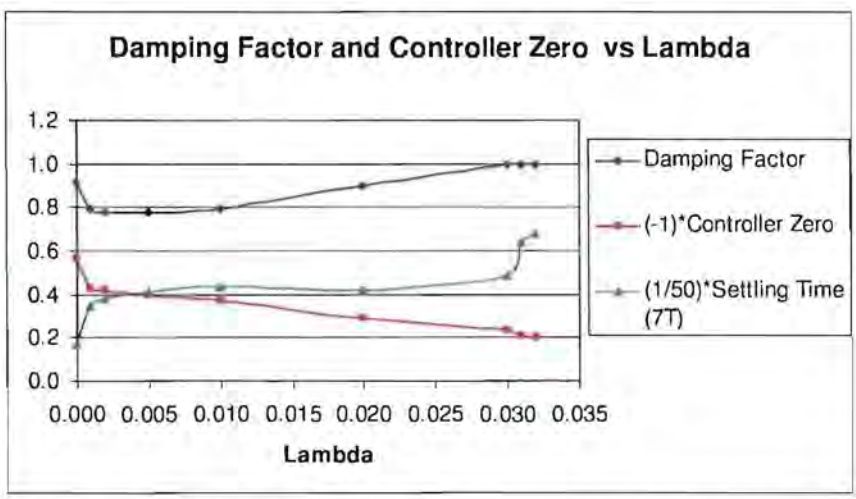


Fig 5.2.2- The effect of λ on the closed loop characteristics when doing time-weighted IFT

As can be seen in fig 5.4.2 the controller zero is dependent on the value of λ and when λ = 0.032 it cancels the process pole at s = -0.2. The damping factor is also non-linearly related to λ. Some values of λ can worsen the damping factor. However after a certain point the damping factor increases with an increase in λ as it did before the manner of measuring settling time was re-evaluated. Unfortunately another consequence is that the settling time of the closed loop increases with λ. This is expected since the controller magnitude decreases as λ increases and the controller zero approaches s = 0. The reason is that the criterion becomes one of minimising u as stated earlier.

Nonetheless it is shown that there is a value for λ that will result in pole-zero cancellation. Therefore time-weighted IFT may be a solution for preventing pole-zero cancellation but not at all values of λ. As mentioned earlier the selection of λ is not automatic but through trial and error. Therefore the user may select a value for λ that results in pole-zero cancellation. The alternative would be to have no penalty on the control effort i.e. λ = 0. However as mentioned earlier this may result in a lightly damped system or a high controller gain.

In practice this time-weighted IFT would have to be performed differently. The reason is that there is the presence of noise in some systems. Therefore the input signals would have to be larger so that the signal to noise ratio is high. A change in the definition of settling time to a value such +/-2% of final value instead would be needed to avoid the effects of noise. However this bigger band can also result in a bigger error as the gradient does not accumulate when the output is within this band. It is up to the user to determine what value of λ to use. This maybe influenced by the desired settling time since the desired settling time may result in a closed loop that is not damped. It is then concluded that if the time-weighted IFT can be applied with λ = 0 and the resultant system is non-oscillatory then time-weighted input disturbance rejection may be a good approach to avoiding pole-zero cancellation.

On the other hand if the initial closed loop is already settled in this time period there will be no change in the controller parameters no matter how unacceptable

the trajectory is. Using a penalty on the control effort has the following disadvantages. Firstly the transfer function from the input disturbance v to the controller output u contains the controller in the numerator so there is still a chance of pole-zero cancellation for some values of λ as shown. Secondly the selection of λ is chosen through trial and error. It is not an automatic selection. As shown previously the use of a model may eliminate the need for a penalty on the control effort.

5.3 USING A SETPOINT TRACKING MODEL WITH OVERSHOOT

Middleton and Graebe (1999) show that for a one degree of freedom closed loop arrangement, pole-shifting will give a closed loop that overshoots when there is a setpoint change. This led to the idea proposed in this section that having a setpoint tracking model with some overshoot might produce a controller that will shift rather than cancel the process poles. This method of deliberately using a model with overshoot was not found in the literature.

This model must have less than the maximum overshoot permitted by the system and will need to be second order or higher (for overshoot). The setpoint-tracking model that is chosen is $m(s) = \frac{1}{s^2 + 1.75s + 1}$. This gives the same characteristic equation as the previous models and satisfies the design requirements. Classical setpoint tracking IFT was attempted with this model with λ set to zero. Table 5.3.1 shows the results of this.

As can be seen, this technique resulted in pole-shifting as opposed to pole-zero cancellation. This meant that the characteristic equations of all the transfer functions are the same as shown in section 4.1. However the results were all different from that of the closed loop speed requirements. This method of specifying the closed loop performance is not ideal.

TABLE 5.3.1- Effect of model with a small overshoot on different processes

g(s)	Closed loop poles	Damping factor	Settling time (7T)
$\frac{1}{1+2s}$	-0.430+/-j0.362	0.765	16.279
$\frac{1}{1+3s}$	-0.375+/-j0.278	0.803	18.667
$\frac{1}{1+4s}$	-0.340+/-j0.212	0.849	20.588
$\frac{1}{1+5s}$	-0.332+/-j0.155	0.906	21.084

The previous model only had a small overshoot that is hardly noticeable therefore it is very damped. The alternative was to use a model with a damping factor of 0.707 that corresponds to overshoot with no oscillations. The transfer function was thus $m(s) = \frac{1.531}{s^2 + 1.75s + 1.531}$ with poles at 0.875+/-j0.875. The effects of this change are shown in the table 5.3.2.

TABLE 5.3.2- Effect of model with maximum overshoot on different processes

g(s)	Controller Zero	Closed loop poles	Damping factor	Settling time (7T)
$\frac{1}{1+2s}$	-1.030	-0.505+j0.520	0.697	13.86
$\frac{1}{1+3s}$	-0.621	-0.464+/-j0.392	0.764	15.09
$\frac{1}{1+4s}$	-0.445	-0.445+/-j0.295	0.833	15.73
$\frac{1}{1+5s}$	-0.348	-0.435+/-j0.209	0.901	16.09

The results are similar although the settling times are shorter. And once again none of the processes achieved the performance defined by the model.

When there is a dominant zero in the s-plane, there will be overshoot in the step response of the system. Therefore the next attempt was to investigate the use of

a zero in the model to provide overshoot. This was done by moving the zero in steps from a position that it is very dominant to one that is less dominant. The results are shown in the table 5.3.3 below. The characteristic equation was chosen as the latter with a high damping factor i.e. $s^2+1.75s+1$. This ensures the zero and not the complex poles cause the overshoot.

The settling time is approximated using the characteristic equation therefore the effect of the controller zero is not taken into account because the transfer function from the input disturbance to output does not contain the controller zero and thus would not be influenced by it. It can be seen that the controller zero does not cancel the process pole at s = -0.2. The data is plotted in the fig 5.3.1.

TABLE 5.3.3- Effect of a zero in the closed loop model

Model Zero	Closed loop poles	Damping factor	Controller Zero	Settling time (7T)
- 0.1	-0.436+/-j1.887	0.225	-5.583	16.06
- 0.2	-0.644+/-j1.768	0.342	-3.256	10.87
- 0.3	-0.784+/-j1.561	0.449	-2.231	8.93
- 0.4	-0.867+/-j1.300	0.555	-1.591	8.07
- 0.5	-0.898+/-j0.995	0.700	-1.125	7.80
- 0.6	-0.886+/-j0.645	0.808	-0.764	7.90
- 0.7	-0.844+/-j0.169	0.981	-0.498	8.29
- 0.8	-1.147, -0.414	1.000	-0.349	16.91
- 0.9	-1.112, -0.316	1.000	-0.286	22.15
- 1.0	-1.034, -0.281	1.000	-0.261	24.91
-1.5	-0.751,-0.263	1.000	-0.243	26.62
-2.0	-0.619, -0.280	1.000	-0.248	25.00
-3.0	-0.479, -0.325	1.000	-0.258	21.54
No Zero	-0.332+/-j0.155	0.906	-0.289	21.08

Figure 5.3.1 shows that the settling time is non-linearly related to the model zero. As the model zero becomes less dominant, the settling time becomes

smaller however after a certain stage this begins to increase when the damping factor becomes unity and hence the system contains two real poles. The reason is that the zero is responsible for the overshoot thus making the model appear less damped. This break in of the poles into the real axis means that one pole becomes slower and the other faster. Thus the closed loop becomes slower as the closed loop performance is dominated by the slower pole that is attracted by the more dominant controller zero as can be seen in table 5.3.3.

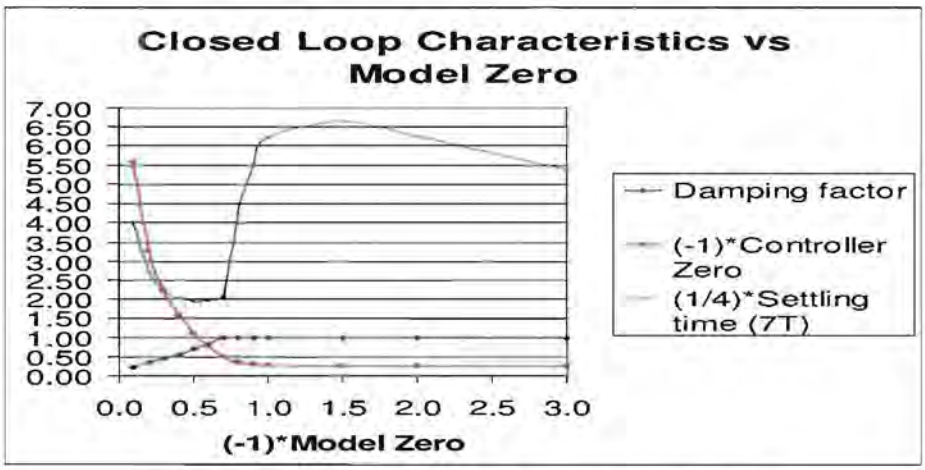


Fig 5.3.1- Effect of the closed loop as the model zero is varied

However this method is not ideal because the model zero position needs to be chosen through trial and error. Having a zero in the model can help prevent pole-zero cancellation in the final loop. As the controller zero will help achieve the desired model that contains a zero. Clearly a zero placed between s = -0.6 and s = -0.7 would result in the desired settling time and damping factor. However thus far there is no way of predicting this zero position automatically.

The above attempts led to the design of the next method of preventing pole-zero cancellation.

5.4 USING APPROXIMATE POLE PLACEMENT IFT

The concept of this method is based on (Lecchini and Gevers, 2002) however with variations. It is also used for a different cause and is simpler. The aim of

this method is to combat the effects of the other techniques where the resultant settling time and damping factor was different for all attempted processes. This method is also applied to the case study stated previously i.e. A first order process and a PI controller.

Approximate pole placement is done for non-minimum phase processes in (Lecchini and Gevers, 2002) where the numerator of the model has adjustable parameters; hence the zeros of the model can change. This means that there is a cost function derivative with respect to the model parameters. The reason for having a changing model is that for non-minimum phase plants, the plant zero is unknown. When model-following, the delay and the non-minimum phase zeros must be present in the model. However because these are not known as the process is unknown, the positions of the model zeros are adjustable so that the focus is on pole placement.

Due to the limit in the variable parameters in the controller, this pole placement cannot be called exact but is approximate pole placement. The free parameters in the model are optimised so that the process can deal with non-minimum phase-zeros that usually cause the IFT controller to cancel the integrating pole added for setpoint tracking. Unfortunately as mentioned in the literature review, the controller zero in (Lecchini and Gevers, 2002) cancels the slow process pole of which this thesis tries to avoid. A summary of this method is in Appendix A.3.

Approximate pole placement iterative feedback tuning (APPIFT) in this section is applied in a different manner. In this version of approximate pole placement the model poles are not limited to be on the same location as they are in (Lecchini and Gevers, 2002) and thus the settling time and trajectory can be directly specified through the chosen pole positions. Like the previous methods for avoiding pole-zero cancellation presented in this chapter, this method is analysed in the s-plane.

The main advantage of pole placement is that it ensures that all the characteristic equations of the transfer functions are the same (for a one degree of freedom

control structure) and in doing so avoids pole-zero cancellation. Looking at the transfer function from the input disturbance to the output it can be seen that if the process does not have any zeros the only closed loop zero is that which belongs to the controller. The basic idea is that the zero of the PI controller is inserted into the model whenever there is a controller parameter update.

Using the case study the closed look setpoint tracking transfer function is:

$$h_{yc} = \frac{A(b_1 s + b_0)}{Ts^2 + (Ab1 + 1)s + Ab_0}$$

$$= \frac{\frac{Ab_1 s}{T} + \frac{Ab_0}{T}}{s^2 + \frac{(Ab_1 + 1)s}{T} + \frac{Ab_0}{T}} \qquad (5.4.1)$$

A pole placement design will place the poles of the closed loop at the location defined by a characteristic equation such s^2+cs+d chosen by the user. The problem that arises is that A and T are unknown. If they were known b_1 and b_0 could be chosen in such a way so that the poles are placed at the required position.

Comparing the closed loop characteristic equation with that of the desired characteristic equations, we obtain the following two equations:

$c = \frac{(Ab_1 + 1)}{T}$ and $d = \frac{Ab_0}{T}$.

Using the later and rearranging the result becomes:

$$\frac{A}{T} = \frac{d}{b_0} \qquad (5.4.2)$$

The desired setpoint tracking model will then be:

$$m(s) = \frac{d}{b_0} \times \frac{b_1 s + b_0}{s^2 + cs + d} \qquad (5.4.3)$$

As can be seen the numerator has the zeros of the PI controller. This model always ensures that the final gain of the model is always one. No pole-zero

cancellation can occur now because the zero of the controller is always present in the model. If pole-zero cancellation were to occur then the closed loop will have no zero thus making it even more different from the desired model. Pole-placement in this situation will occur because the only difference between the closed loop system and its desired model is the characteristic equation. The characteristic equation of the model is second order like that of the closed loop so it can be achieved. The cost function is thus at its global minimum when there is pole placement.

One of the consequences of this method is that the controller variables are now present in the model. This means that the derivative of the model can no longer be omitted as it was previously when calculating the derivative of the cost function. The IFT equations for the cost function derivatives now become:

$$\frac{\partial J}{\partial \rho} = \frac{1}{N}\left[\sum_{i=1}^{N}(em_i(\rho))\frac{\partial em_i}{\partial \rho}(\rho)\right] \tag{5.4.4}$$

where $\frac{\partial em}{\partial \rho} = \frac{\partial (y - ym)}{\partial \rho} = \frac{\partial y}{\partial \rho} - \frac{\partial y_m}{\partial \rho}$. Note that $\frac{\partial y_m}{\partial \rho} \neq 0$ making it different from what it was previously. The derivatives with respect to the controller parameters are then:

$$\frac{\partial y}{\partial b_0} = \frac{1}{b_1 s + b_0} \times \frac{\frac{Ab_1 s}{T} + \frac{Ab_0}{T}}{s^2 + \frac{(Ab_1 + 1)s}{T} + \frac{Ab_0}{T}}(r - y)$$

$$\frac{\partial y}{\partial b_1} = \frac{s}{b_1 s + b_0} \times \frac{\frac{Ab_1 s}{T} + \frac{Ab_0}{T}}{s^2 + \frac{(Ab_1 + 1)s}{T} + \frac{Ab_0}{T}}(r - y) \tag{5.4.5}$$

As normal and,

$$\frac{\partial ym}{\partial b_0} = -\frac{d \times b_1}{{b_0}^2} \times \frac{s}{s^2 + cs + d} \times r$$

$$\frac{\partial ym}{\partial b_1} = \frac{d}{b_0} \times \frac{s}{s^2 + cs + d} \times r \tag{5.4.6}$$

Therefore there are two new derivatives to calculate.

To make the calculation of the model derivative simpler, APPIFT can be done through the input disturbance. This is done by obtaining an equivalent input disturbance rejection model. $h_{ym} = \frac{h_{yr}}{k} \Rightarrow m = \frac{d}{b_0} \times \frac{s}{s^2 + cs + d}$. Gradient $\frac{\partial y}{\partial \rho}$ is as defined earlier and the resulting extra derivatives needed are then:

$$\frac{\partial ym}{\partial b_0} = -\frac{d}{b_0^{\,2}} \frac{s}{s^2 + cs + d} \times v \qquad (5.4.7)$$

$$\frac{\partial ym}{\partial b_1} = 0$$

The derivative with respect to b_1 is zero and that with respect to b_0 is simply the model output divided by the parameter b_0. This means that the only calculation that needs to be made to find this derivative is that of division.

This method was applied to both stable and unstable plants. The characteristic equation chosen for pole placement is $s^2+2s+1.25$, which has closed loop poles located at $s = -1 \pm j0.5$. This will give a settling time of 7 seconds (7T =7). The initial controller chosen was $k(s) = \frac{1+s}{s}$ for stable plants and $k(s) = \frac{-3-3s}{s}$ for unstable plants. This reason is that IFT is meant to be applied on a stable closed loop (Hjalmarsson et al., 1998). The results are shown in Table 5.4.1.

The final closed loop is relatively consistent regardless of the choice of the plant. The pole-placement and model characteristic equations are similar. The results are comparable to the requirements given by the characteristic equation. i.e. 7T = 7 seconds and damping factor of 0.894. The design requirements are met. This is shown in Fig 5.4.1.

Figure 5.4.1 shows that there is no point at which there is pole-zero cancellation and the closed loop characteristics are consistent. Therefore the closed loops had good internal performance.

TABLE 5.4.1- APPIFT for first order processes

Process g(s)	g(s) poles	Final k(s) zeros	Closed loop h(s) poles	Damping Factor	Settling Time (7T)
$\frac{1}{1-6s}$	0.167	-0.571	- 1.03 +/-j0.46	0.913	6.8
$\frac{1}{1-5s}$	0.200	-0.563	- 1.03 +/-j0.46	0.913	6.8
$\frac{1}{1-4s}$	0.250	-0.550	- 1.03 +/-j0.46	0.913	6.8
$\frac{1}{1-3s}$	0.333	-0.533	- 1.03 +/-j0.46	0.913	6.8
$\frac{1}{1-2s}$	0.500	-0.497	- 1.03 +/-j0.46	0.913	6.8
$\frac{1}{1+2s}$	-0.500	-0.811	- 1.04+/-j0.45	0.918	6.7
$\frac{1}{1+3s}$	-0.333	-0.736	- 1.04+/-j0.45	0.918	6.7
$\frac{1}{1+4s}$	-0.250	-0.703	- 1.04+/- j0.45	0.918	6.7
$\frac{1}{1+5s}$	-0.200	-0.682	- 1.04+/- j0.45	0.918	6.7
$\frac{1}{1+6s}$	-0.167	-0.671	- 1.04+/- j0.45	0.918	6.7

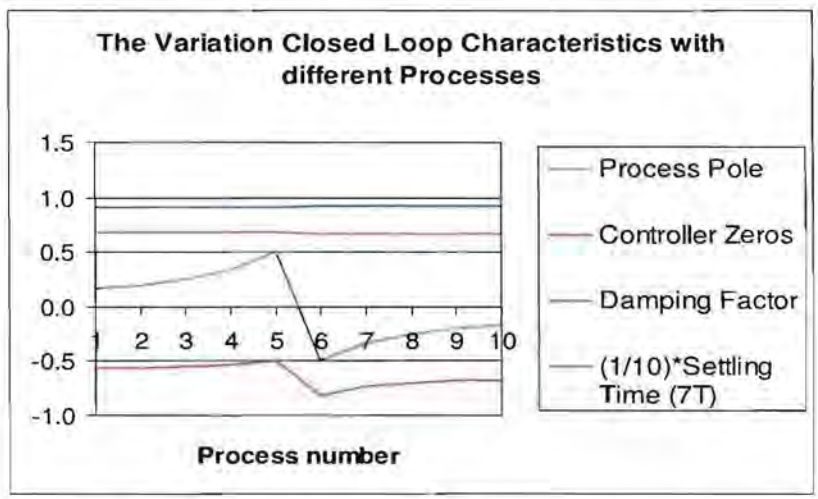


Fig 5.4.1- Closed loop characteristics for different processes

5.5 SUMMARY OF THE INVESTIGATED METHODS

This section summarises the four investigated methods. The best method is then chosen.

5.5.1 Using an Input Disturbance Rejection Model

This method is dependent on selection a value of the variable c in the equation $m(s) = \frac{cs}{s^2 + ds + e}$ where d and e are constants chosen according to the loop requirements in terms of speed and damping. The variable c can be chosen depending on the maximum height of output allowed as a fraction of the input disturbance. However it does not always result in the desired closed loop dynamics therefore the value c can be adjusted until the desired closed loop is achieved.

A value for c can be obtained by doing open loop tests on the process and obtaining estimates for the gain of the process, A, and time constant of the process, T (provided there is a first order process and the model is second order). If A and T are known then the desired performance maybe achieved. However this approach takes away the benefit of IFT in which the process model does not need to be known.

5.5.2 Using Time-Weighted IFT

The second method investigated was using Time-weighting IFT. When using time-weightings the control effort may need to be penalised. However the choice of the value of λ influences the resultant closed loop and the final zero of the controller. It was also shown that there is a value for λ that will give a controller zero that will cancel the process poles. The ideal speed of response is achieved when λ = 0. This gives the desired speed of disturbance rejection; on the other hand it gives no control on the damping factor of the final closed loop. Once there is a non-zero value for λ the speed of response is compromised.

5.5.3 Using a Setpoint Tracking Model with Overshoot

The third method that was investigated was one of using a setpoint-tracking model with overshoot. This method prevented pole-zero cancellation. However the settling time changed with the place of this model zero. The damping factor of this closed loop was also dependent on the zero position. The zero position that will result in the desired closed loop performance can be obtained through trial and error. However like the method of using a disturbance rejection model, an estimate of the zero position can be obtained from doing tests on the process and obtaining A and T.

5.5.4 Using Approximate Pole Placement IFT

The fourth and final method was approximate pole placement IFT. This method provided a consistent closed loop characteristic equation regardless of the process being used. It does not depend on trial and error, and the design requirements are closely met every time.

5.5.5 Chosen Method for Preventing Pole-Zero Cancellation

It is clear that approximate pole placement IFT is the ideal method for preventing pole-zero cancellation. The reason is that it gives a closed loop that satisfies the design requirements every time. In the literature review pole placement was stated to be the robust approach to design. The advantage is that it is automated thus is not prone to process modelling errors. For these reasons this is the chosen method to prevent pole-zero cancellation. This method is investigated further in the next chapter.

Chapter 6

Analysis of Approximate Pole Placement Iterative Feedback Tuning

This chapter analyses the approximate pole placement IFT (APPIFT) algorithm. In the previous chapter this method was chosen as the preferred means for preventing pole-zero cancellation and thus improving on internal performance of the closed loop. The aim of this chapter is to investigate its feasibility on physical processes that are not ideal.

The investigations mainly focus on:

- The effect on closed loop performance when the model output derivative ($\frac{\partial y_m}{\partial \rho}$) is not used in the cost function gradient calculation for first order and second order pole placement.
- The performance when the process is a damped second order process and hence appears to be first order.
- Developing approximate pole placement IFT for second order processes.
- The effect of the sampling time changes when using approximate pole placement IFT.

All the investigations in this chapter were performed through simulation using programs written in Visual Basic 6.

6.1 APPIFT FOR FIRST ORDER PROCESSES

This algorithm was developed in section 5.4. Through multiple experiments, it was discovered that when doing pole placement IFT on a first order process, the model output derivative, $\frac{\partial y_m}{\partial \rho}$, is not needed in the cost function gradient calculation. Results for pole placement to s= -1+/-j0.5 without the model output derivative are shown in table 6.1.1 for a number of process models, g(s). As previously the update constant was chosen to be $\gamma = \frac{0.01}{|\text{cost function gradient}|}$ and the sampling time was 20ms.

TABLE 6.1.1- APPIFT without model derivative in cost function gradient calculation

g(s)	g(s) poles	K(s) zeros	Closed loop h(s) poles	Damping Factor	Settling Time (7T)
$\frac{1}{1+2s}$	-0.500	-0.811	-1.03+/-j0.46	0.913	6.8
$\frac{1}{1+3s}$	-0.333	-0.736	-1.03+/-j0.46	0.913	6.8
$\frac{1}{1+4s}$	-0.250	-0.703	-1.03+/-j0.46	0.913	6.8
$\frac{1}{1+5s}$	-0.200	-0.682	-1.03+/-j0.46	0.913	6.8
$\frac{1}{1+6s}$	-0.167	-0.671	-1.03 +/j0.46	0.913	6.8

These results are similar to those obtained in section 5.4 when there was a model derivative in the cost function gradient calculation. For this reason if the process is first order and the experimental conditions are the same as used for the above, the model derivative is not needed.

The argument for the controller parameters obtained as giving the global minimum point of the cost function is aided by the fact that no model derivative is needed. The controller parameters will continue to be updated unless the cost

function derivative with respect to the controller parameters is zero. The cost function gradient equation was used to analyse why this derivative is zero at the optimum points whether or not the model derivative is included. This was done as follows:

$$\frac{\partial J}{\partial \rho} = \frac{1}{N}\sum_{i=1}^{N} em_i\left(\frac{\partial y_i}{\partial \rho} - \frac{\partial ym_i}{\partial \rho}\right)$$

$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - ym_i)\left(\frac{\partial y_i}{\partial \rho} - \frac{\partial ym_i}{\partial \rho}\right) \tag{6.1.1}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left[\frac{\frac{A}{T}s}{s^2 + \frac{Ah_1+1}{T}s + \frac{A}{T}b_0}v - \frac{\frac{d}{b_0}s}{s^2+cs+d}v\right]\left[-\frac{\left(\frac{A}{T}\right)^2 s}{\left(s^2 + \frac{Ah_1+1}{T}s + \frac{A}{T}b_0\right)^2}v + \frac{\frac{d}{b_0^2}s}{s^2+cs+d}v\right]$$

Therefore since there is only one derivative for approximate pole placement done through the input disturbance, this can be displayed as the sum of the terms shown below:

$$\frac{\partial J}{\partial b_0} = \frac{1}{N}\sum_{i=1}^{N}\begin{pmatrix} y\frac{\partial y}{\partial b_0} & -y\frac{\partial ym}{\partial b_0} \\ -ym\frac{\partial y}{\partial b_0} & ym\frac{\partial ym}{\partial b_0} \end{pmatrix}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\left[\begin{array}{l} -\frac{\left(\frac{A}{T}\right)^3 s^2}{\left(s^2 + \frac{Ab_1+1}{T}s + \frac{A}{T}b_0\right)^3}v^2 \\ +\frac{\frac{d}{b_0}\left(\frac{A}{T}\right)^2 s^2}{\left(s^2 + \frac{Ab_1+1}{T}s + \frac{A}{T}b_0\right)^2\left(s^2+cs+d\right)}v^2 \end{array}\right.$$

$$\left.\begin{array}{l} +\frac{\frac{A}{T}\frac{d}{b_0^2}s^2}{\left(s^2 + \frac{Ab_1+1}{T}s + \frac{A}{T}b_0\right)\left(s^2+cs+d\right)}v^2 \\ -\frac{\frac{d^2}{b_0^3}s^2}{\left(s^2+cs+d\right)^2}v^2 \end{array}\right]$$

Looking at the equation above it can be seen that the sum of the terms on the left circled in red are zero only when the model and the process are equal. These terms are obtained by having only the process output derivative i.e.

$\frac{\partial J}{\partial \rho} = \frac{1}{N}\sum_{i=1}^{N}\left(em_i(\frac{\partial y_i}{\partial \rho} - 0) \right) = \frac{1}{N}\sum_{i=1}^{N}\left(y(\frac{\partial y_i}{\partial \rho}) - ym(\frac{\partial y_i}{\partial \rho}) \right)$. The sum of the terms on the right circled in blue are also only zero when the model and the process are the same. These terms are due to the model output derivative being present in the cost function derivative calculation i.e. $\frac{\partial J}{\partial \rho} = \frac{1}{N}\sum_{i=1}^{N}\left(em_i(\frac{\partial y_i}{\partial \rho} - \frac{\partial ym_i}{\partial \rho}) \right)$

$= \frac{1}{N}\sum_{i=1}^{N}\left(y(\frac{\partial y_i}{\partial \rho}) - y(\frac{\partial ym_i}{\partial \rho}) - ym(\frac{\partial y_i}{\partial \rho}) + ym(\frac{\partial ym_i}{\partial \rho}) \right)$. Therefore whether or not the model derivative is included, the cost function gradient is zero at the same point i.e. when the process output follows the output of the model. For that reason whether these terms are included or not the process will settle at the same point i.e. the only time that this cost function derivative is zero is when the two terms are equal. Consequently instead of $\frac{\partial ym_i}{\partial \rho}$ the process derivative $(\frac{\partial y_i}{\partial \rho})$ can be left out of the equation and the controller parameters would get to the same optimum values.

However in the situation that the model cannot be tracked exactly this will not give the same optimum parameters. Therefore the model output derivative will be needed in order to obtain the true mathematical derivative and achieve the minimum point of the cost function. This will occur under circumstances such as the process not being first order, but of a higher order. Therefore in practice it is better to include the model derivative term. The calculation for this term is simply one of division therefore its exclusion it will not reduce computation time by a significant amount.

It should also be noted that in the case of doing pole placement by using an input disturbance there are two global minimums i.e. when the cost function is zero. The first, when the gain is infinite (when both y and ym are zero) and the other when the model and closed loop are the same (y = ym).

6.2 APPIFT FOR FIRST ORDER PROCESSES APPLIED ON SECOND ORDER PROCESSES THAT APPEAR TO BE FIRST ORDER

This section investigates the effect that pole placement for first order processes has on processes that are second order, but have damped poles. In the experiments the non-dominant (faster) pole was progressively shifted closer to the more dominant pole. The reason was that there are many processes like this in practice. These processes however appear to be first order. It was expected that the performance will decay as the process became less first order dominant. However the aim of the investigation was to see whether the performance may still be acceptable and not result in an unstable or unacceptable closed loop.

The results of first order pole placement when the model is $g(s) = \frac{1}{1+2s}$ (pole at s = -0.5) are shown in fig 6.2.1 and fig 6.2.2. The pole placement algorithm will attempt to place the poles at s = -1+/-j0.5 and the model update constant was $\gamma = \frac{0.01}{|\text{cost function gradient}|}$ as used previously. The characteristic equation and the update constant remained the same for all the trials in this section. There was a unit step input disturbance at time t = 0 seconds for ***IFT experiment one*** and ***IFT experiment two*** occurs after 20 seconds.

The responses in fig 6.2.1 were obtained with the closed loop output response using the initial controller $k(s) = \frac{s+1}{s}$ and the initial model $m(s) = \frac{1.25s}{s^2 + 2s + 1.25}$. The closed loop disturbance rejection time takes 12 seconds which is twice that of the model which takes 6 seconds.

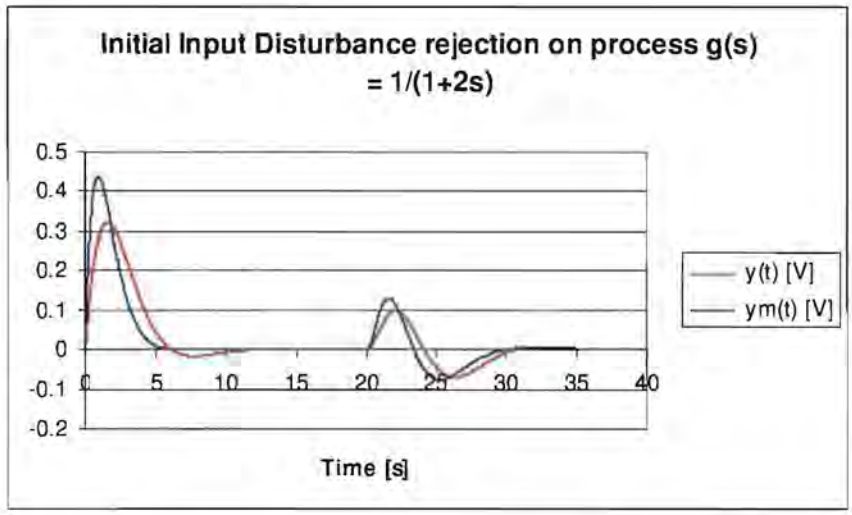


Fig 6.2.1- Initial responses to step input disturbance for $g(s) = \frac{1}{1+2s}$

Figure 6.2.2 was the final output response after the controller parameters had settled.

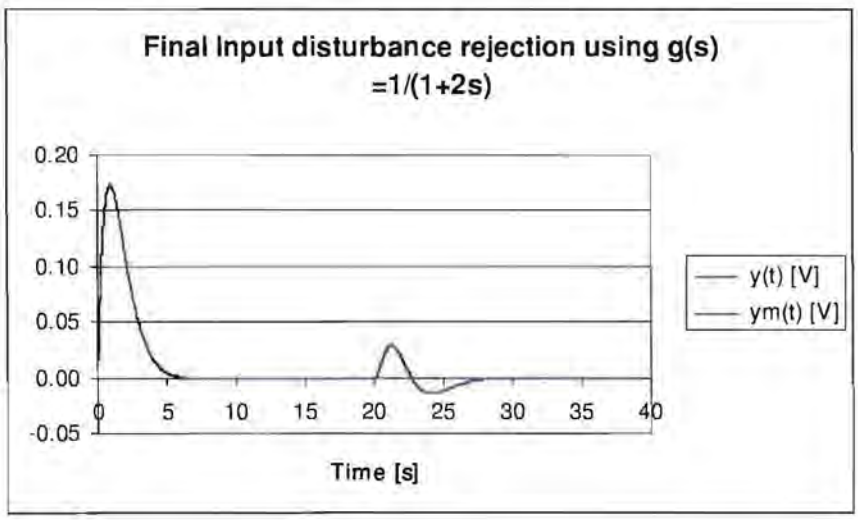


Fig 6.2.2- Final responses to step input disturbance for $g(s) = \frac{1}{1+2s}$

It can be seen that the closed loop output tracks that of the model output for input disturbance rejection. The final average controller was

$k(s) = \frac{3.16s + 2.56}{s}$ which places the closed poles at s = -1.04+/-j0.45. Therefore approximate pole placement had taken place.

The first second order process to be used was $g(s) = \frac{5}{s^2 + 10.5s + 5}$. This has a pole at s = -10 and one at s = -0.5. The dominant pole is located at the same place as that of the previous first order process. When step tests were used to calculate the transfer function of this process it appears to be that of the first order process $g(s) = \frac{1}{1 + 2s}$. The initial and final rejection of a unit input disturbance is shown in the fig 6.2.3 and fig 6.2.4 below.

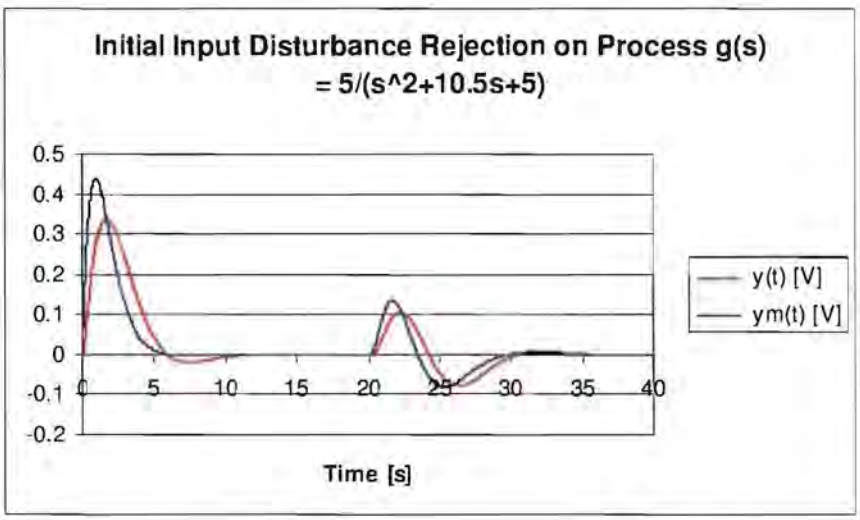


Fig 6.2.3- Initial responses to step input disturbance for $g(s) = \frac{5}{s^2 + 10.5s + 5}$

It can be seen above that the closed loop disturbance rejection is slower than the open loop, the controller used is $k(s) = \frac{s+1}{s}$. The closed loop output response looks similar to the output when the first order process was used. The final responses after the controller parameters had been optimised are shown in fig 6.2.4.

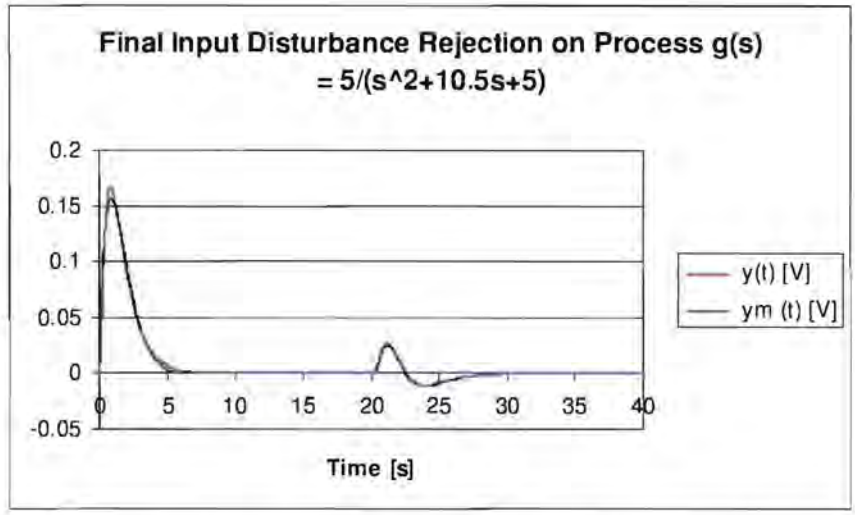


Fig 6.2.4- Final response to step input disturbance for $g(s) = \frac{5}{s^2 + 10.5s + 5}$

The optimum controller is $k(s) = \frac{2.82 + 3.88s}{s}$ with a zero at s = -0.73 the closed loop poles are calculated to be at s = - 7.496, s = -2.115 and s = -0.890. The final closed loop output is similar to the model in many places however it takes a longer time to be completely rejected. This is a slight deviation from when the first order process is used. The closed loop output did not resemble the model output completely however there was a significant improvement from the original controller. The final pole positions are different from the desired closed loop pole position. However the closed loop response is still acceptable.

The above pole placement was applied under the same conditions on processes: $g(s) = \frac{4}{s^2 + 8.5s + 4}$, $g(s) = \frac{3}{s^2 + 6.5s + 3}$ and $g(s) = \frac{2}{s^2 + 2.5s + 2}$. The responses are shown in Appendix B and results are tabulated in table 6.2.1.

Looking at the results it can be seen that as the faster open loop pole approaches the slower open loop pole, the slower closed loop pole moves closer to the origin. The closed loop also becomes less damped. Hence the closed loop transient decay becomes slower and it is also more oscillatory. None of the above closed loop tracked the trajectory of the model output. Therefore pole

placement IFT worked best for processes that are very first order dominant and the performance deteriorated as the process became more second order dominant as expected.

TABLE 6.2.1- APPIFT using a PI controller applied on damped second order processes

Process g(s)	Open loop Pole positions	Closed loop pole positions
$\frac{5}{s^2+10.5s+5}$	s = -10 and s = -0.5	s = - 7.496, s = -2.115 and s = -0.890
$\frac{4}{s^2+8.5s+4}$	s = -8 and s= -0.5	s = -4.592, s = -3.086 and s = -0.821
$\frac{3}{s^2+6.5s+3}$	s = -6 and s = -0.5	s = -2.877+/-j2.050 and s = -0.745
$\frac{2}{s^2+4.5s+2}$	s = -4 and s = - 0.5	s = - 1.931+/-j2.808 and s = -0.639

When the above experiment was attempted with the faster process pole at s = -2 and anything slower, the gain kept on rising. On the process $g(s)=\frac{1}{s^2+2.5s+1}$ with poles at s = - 0.5 and s = -2 the gain of the system kept on rising and it did not reach an optimum value after 4500 IFT cycles. This can be seen below.

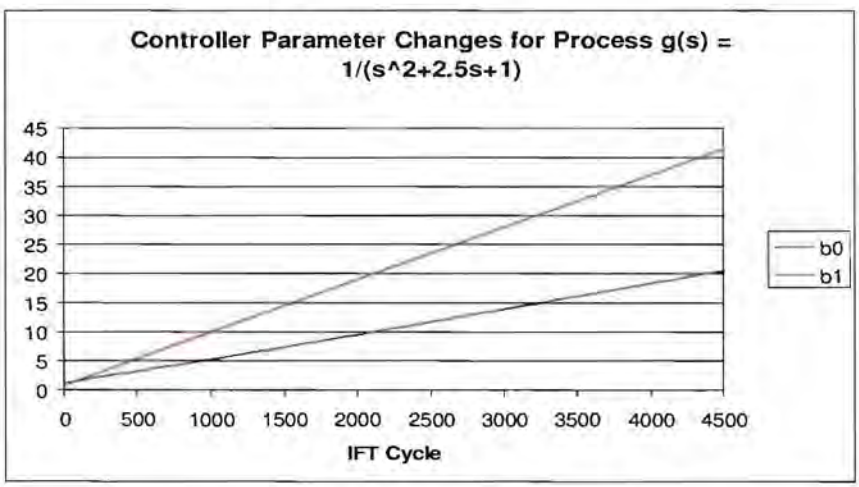


Fig 6.2.5- Controller parameter changes for APPIFT on $g(s)=\frac{1}{s^2+2.5s+1}$

The controller parameter values increased by a constant amount but at a different rate. It appears that the gain of the controller was headed towards infinity. This is where the global minimum is. This means that when using pole placement IFT through the input disturbance designed for first order process, on a process that does not have enough first order dominance the results may be similar to that of using no model at all i.e. the gain of the controller keeps on rising. It can also result in a closed loop characteristic equation that gives dynamics different from that which is desired.

As a result of the information obtained in this section, the next section develops a pole placement algorithm for second order systems.

6.3 APPIFT FOR SECOND ORDER PROCESSES

When doing APPIFT for a second order process, two zeros are needed in the controller. This is different from APPIFT for a first order system where a PI controller has sufficient variables for pole placement. For the second order process $g(s)=\frac{ad}{s^2+cs+d}$ in closed loop with the PI controller $k(s)=\frac{b_1 s+b_0}{s}$, the closed loop setpoint tracking transfer function is $h_{yr}(s)=\frac{ad(b_1 s+b_0)}{s^3+cs^2+(adb_1+d)s+adb_0}$. The value c is unknown so this will inhibit pole placement. If c was known the location for pole placement would be limited. Therefore two zeros are needed.

6.3.1 Requirements for pole placement of a second order process

The required controller is one with three variables and is of the form:

$$k(s)=\frac{b_2 s^2+b_1 s+b_0}{s} \tag{6.3.1}$$

This will give the closed loop transfer function:

$$h_{yr}(s)=\frac{ad(b_2 s^2+b_1 s+b_0)}{s^3+(adb_2+c)s^2+(adb_1+d)s+adb_0} \tag{6.3.2}$$

Therefore for pole placement to a closed loop with characteristic equation

$$\varphi_c = s^3 + Bs^2 + Cs + D \tag{6.3.3}$$

With this characteristic equation the optimum controller parameters are:

$$b_0 = \frac{D}{ad} \tag{6.3.4}$$

$$b_1 = \frac{C-d}{ad} \tag{6.3.5}$$

$$b_2 = \frac{B-c}{ad} \tag{6.3.6}$$

However the controller above is not causal therefore the controller needs to be of the form:

$$k(s) = \frac{b_2 s^2 + b_1 s + b_0}{s(T_c s + 1)} \tag{6.3.7}$$

Where the pole at $s = -\frac{1}{T_c}$ is added for causality. The closed loop transfer function is then:

$$h_{yr}(s) = \frac{ad(b_2 s^2 + b_1 s + b_0)}{T_c s^4 + (1 + T_c c)s^3 + (adb_2 + c + T_c d)s^2 + (adb_1 + d)s + adb_0} \tag{6.3.8}$$

For the above transfer function to be similar to the optimal in equation (6.3.2), T_c must be small. For the experiments in this section T_c is chosen to be 0.01 so as to make the pole added for causality less dominant. Therefore the equation of the closed loop becomes:

$$h_{yr}(s) = \frac{ad(b_2 s^2 + b_1 s + b_0)}{0.01 s^4 + (1 + 0.01c)s^3 + (adb_2 + c + 0.01d)s^2 + (adb_1 + d)s + adb_0} \tag{6.3.9}$$

However, since the process is generally unknown the h(s) in equation (6.3.2) will be used, which has a characteristic equation that is an approximation of that in equation (6.3.8) for small values T_c.

The error terms are therefore the $0.01c$ and $0.01d$ that can be assumed to be small. The less dominant the extra pole that is added for causality is, the more accurate the pole placement will be. The $0.01s^4$ will also give a small error especially because a square wave does not provide a lot of high frequency excitation.

Like in the previous chapter the equivalent input disturbance model will be used. The reason is shown in section 5.4, that there is only one model derivative that is needed and it is a scalar multiple of the model output. This is:

$$h_{yv}(s) = \frac{ads(0.01s+1)}{s^3 + (adb_2 + c)s^2 + (adb_1 + d)s + adb_0} \tag{6.3.10}$$

6.3.2 Proof that the closed loop can be achieved

The first stage was to do IFT through the input disturbance with an input disturbance model that the closed loop should be able to track. This was to confirm that the controller parameters could in fact reach the desired values (with a constant model). The process that was used is $g(s) = \frac{0.4}{s^2 + 0.4s + 0.4}$ which has poles located at s = -0.2+/-j0.6 which is oscillatory and clearly second order.

The chosen closed loop pole locations were s = -0.7+/-j0.7 and s = -0.5. Therefore the fixed input disturbance rejection model to be used was $m(s) = \frac{0.4s(0.01s+1)}{s^3 + 1.9s^2 + 1.68s + 0.49}$ that was obtained by substituting the values in the equation (6.3.10). This was to place the poles at s = -0.7+/-j0.7 and s = -0.5. Using equations (6.3.4), (6.3.5) and (6.3.6), the optimum values were calculated to be $b_2 = 3.75$, $b_1 = 3.20$ and $b_0 = 1.23$ (without including the pole added for causality).

For the following experiments, the step size was once again defined to be $\gamma = \frac{0.01}{|\text{cost function gradient}|}$ as before. The sample time, dt = 0.001, so as to make it one tenth of the time constant of the fastest open loop dynamic. The initial controller is $k(s) = \frac{2s^2 + 2s + 1}{s(0.01s + 1)}$ for all the experiments involving the controller with three variables in this thesis.

The results of the controller parameter changes are shown in the fig 6.3.1 below.

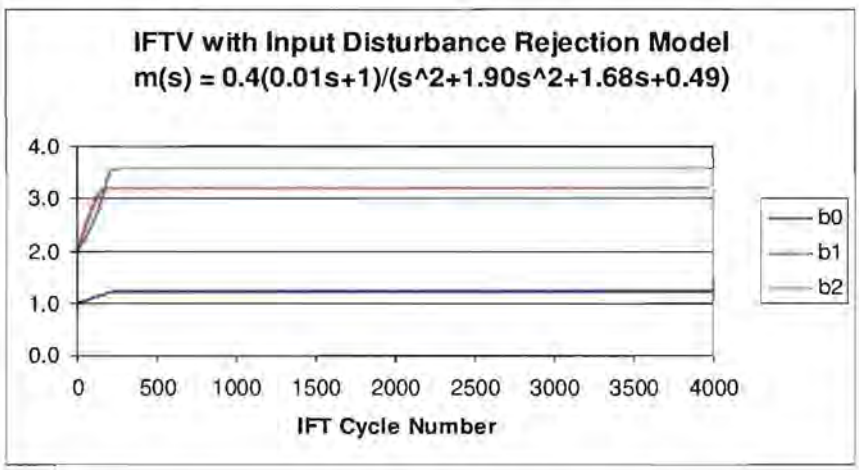


Fig 6.3.1- Controller parameter evolution for IFT on $g(s) = \frac{0.4}{s^2 + 0.4s + 0.4}$ part 1

The average final controller parameters values b_2 = 3.58, b_1 = 3.20 and b_0 = 1.23. The difference between the calculated values and those obtained could be because of the errors mentioned above in section 6.3.1 i.e. the pole added for causality has been ignored. The parameter b_2 has the largest error. This could be because a step function does not provide enough high frequency excitation, the sampling time was not fast enough (as will be explored later) or that the constant step size of γ did not allow the final optimum values to be reached exactly. A constant vector magnitude step size for γ was used so the parameters still update by a reasonable amount when the derivatives were small. This allowed it to reach its optimum region faster.

Given that the parameter values obtained were not exactly what were calculated, the above experiment was redone using the strictly equivalent input disturbance rejection closed loop of equation (6.3.8) as the model. This is to see if the approximation is good enough. This is disturbance rejection model is $m(s) = \frac{0.4s(0.01s+1)}{0.01s^4+1.004s^3+1.904s^2+1.68s+0.49}$. These results are shown in fig 6.3.2.

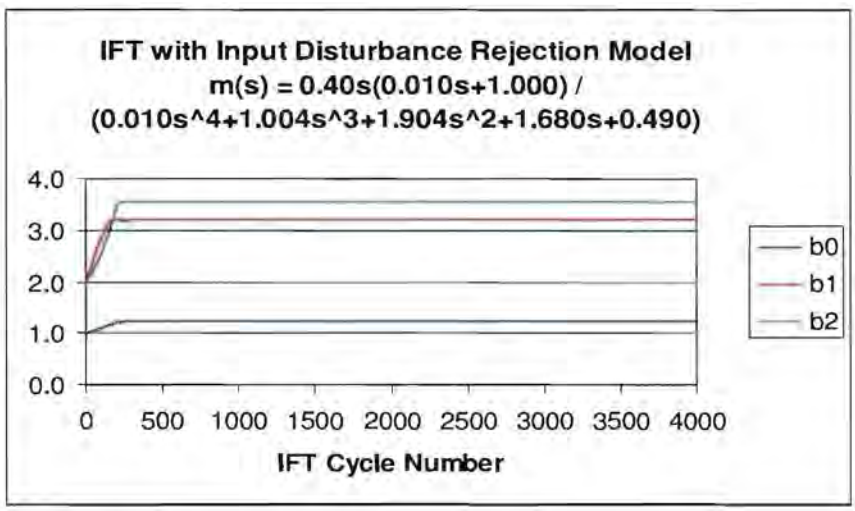


Fig 6.3.2- Controller parameter evolution for IFT on $g(s) = \frac{0.4}{s^2+0.4s+0.4}$ part 2

The final values are almost equivalent to those obtained with the previous model, with $b_2 = 3.57$, $b_1 = 3.20$ and $b_0 = 1.23$. Therefore the extra error terms do not affect the final controller much with the chosen location of the non-dominant pole of the controller.

6.3.3 Second order pole placement

Since it was confirmed that the parameters could get close to the calculated parameters, approximate pole placement IFT was then attempted on the same process above. The results are then shown in fig 6.3.3.

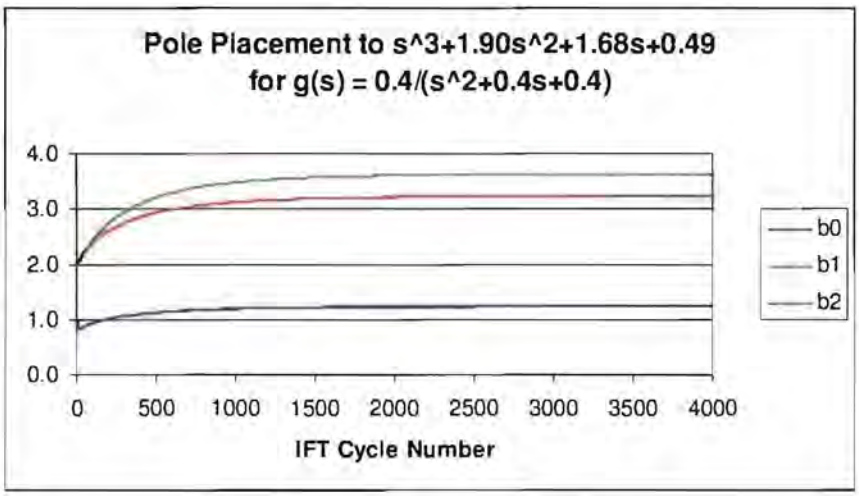


Fig 6.3.3- Controller Parameter evolution for APPIFT on $g(s) = \dfrac{0.4}{s^2 + 0.4s + 0.4}$

The controller parameters settle at b_2 = 3.62, b_1= 3.23 and b_0 =1.24. These are close to the values obtained when using IFT with a constant disturbance rejection model. The parameters give pole placement to s = -98.54, s = -0.69+/-j0.76 and s = -0.49. Therefore, approximate pole placement had occurred without actually knowing the process model. The algorithm was been successful. It can be seen that the algorithm took a significantly longer period of time to settle than that with a constant model. The direction of the change in parameter b_0 is first in the wrong direction before turning to the right direction.

This pole placement IFT algorithm was also applied on a different second order process to confirm that it can work on a different process. The process chosen was $g(s) = \dfrac{0.45}{s^2 + 0.6s + 0.45}$ with poles at s = -0.3+/-j0.6. The results are shown in the fig 6.3.4.

The result is b_2 = 2.81, b_1 = 2.77 and b_0 = 1.10 this places the closed loop poles at s = -98.72, s = -0.70+/-j0.75 and s = -0.48. Confirming that approximate pole placement works on a different process.

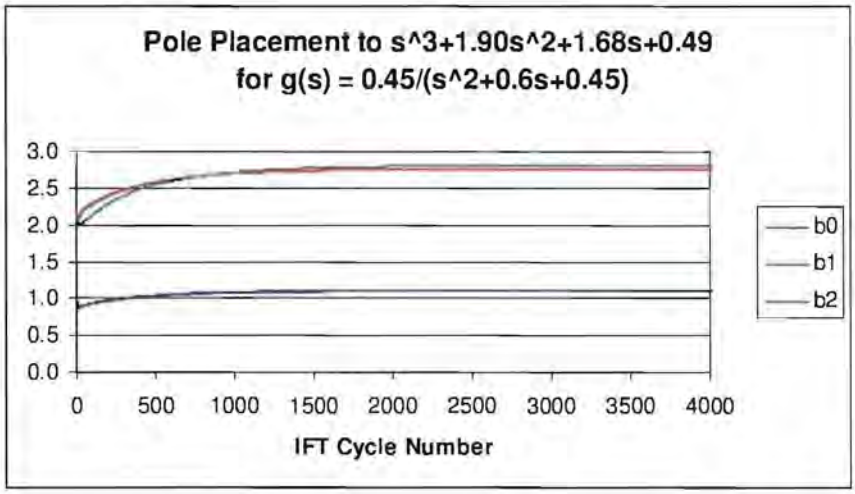


Fig 6.3.4- Controller parameter evolution for APPIFT on $g(s) = \frac{0.45}{s^2 + 0.6s + 0.45}$

6.3.4 Second order Pole placement without the model derivative

The next objective was to see whether the model derivative is needed for pole placement for second order processes. The characteristic equation used for pole placement was the same as the one above i.e. $s^3 + 1.9s^2 + 1.68s + 0.49$. Figure 6.3.5 below shows the controller parameter evolution without the model derivative.

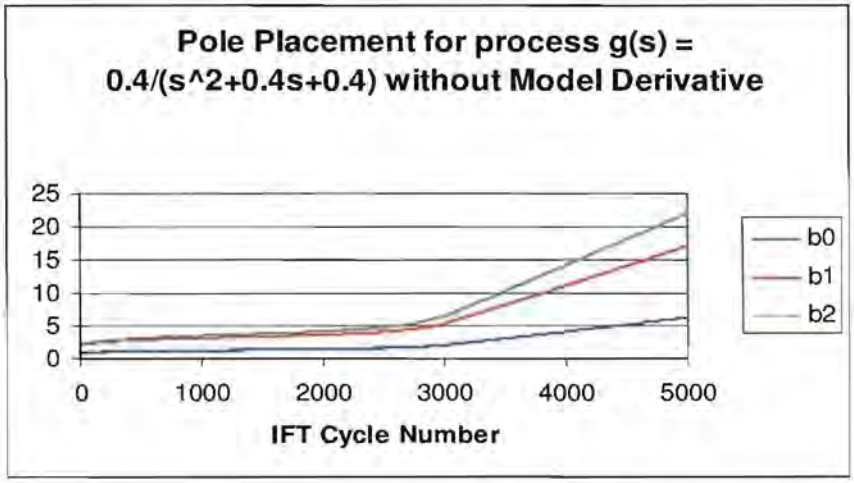


Fig 6.3.5 - Controller parameter evolution for APPIFT without model derivative

It can be seen that pole placement does not occur without the model derivative in this situation because the actual closed loop and model can never be exactly the same because of the non-dominant pole added for causality. When this does not happen, a model derivative is needed. Therefore as suggested earlier it is safer to have a model derivative. Figure 6.3.6 shows how the cost function varies with IFT cycle.

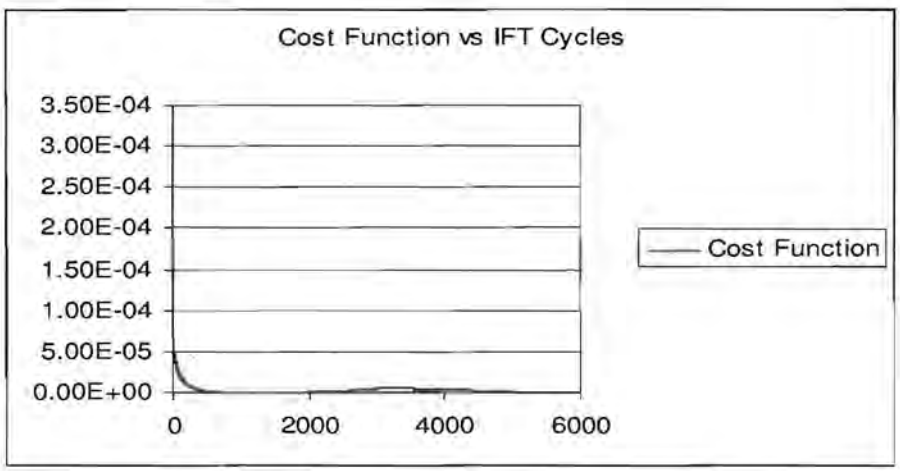


Fig 6.3.6- Cost function evolution for APPIFT without model derivative

The above figure shows that the cost function did in fact pass its local minimum. Hence the need to have the model derivative present in the cost function gradient calculation.

6.4 EFFECT OF SAMPLING TIME ON CONTROLLER PARAMETERS

Second order pole placement was implemented on the process $g(s)=\frac{0.4}{s^2+0.4s+0.4}$. The final result is shown in fig 6.4.1 with a sampling time of 0.02s as was used for the first order processes. The reason is that this is the sampling time used for pole placement for first order processes and it was of interest to see how a limited sampling time would affect pole placement for

second order processes. This sampling time is larger than the time constant of the non-dominant open loop pole of the controller i.e. 0.01s.

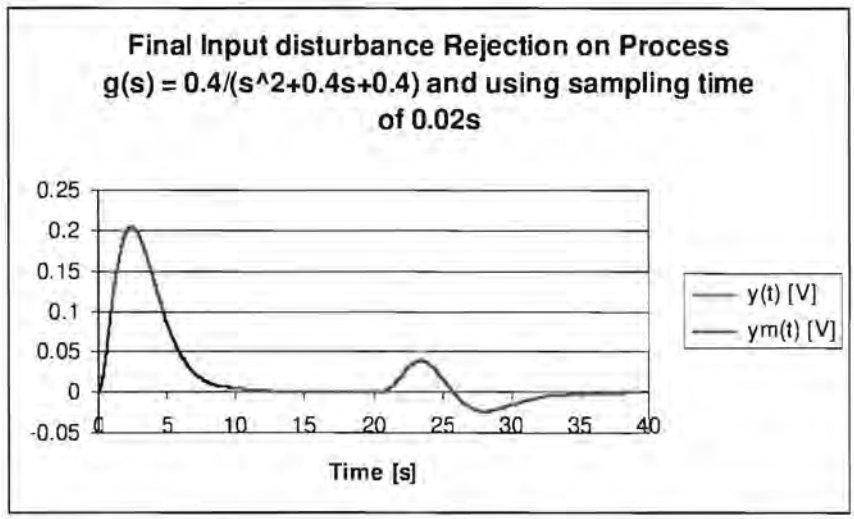


Fig 6.4.1- Results for APPIFT for second order processes with a sampling time of 0.02s

Final controller was $k(s)=\frac{1.21s^2+3.19s+1.21}{s(0.01s+1)}$. The model matches the closed loop. With this controller the closed loop poles are located at s = -99.525, s = -0.276+/-j1.194 and s = -0.324 with the resultant closed loop fastest dynamic having a time constant smaller than the sampling time. Pole placement was also attempted for systems of sampling times 0.01s, 0.005s, 0.002s and 0.001s. The results are given in Appendix B.2. For all of these sampling times the resultant closed loop output tracked the model output.

Figure 6.4.2 shows how the controller parameters vary as the sampling time of the program is changed. As the sampling time decreased the parameter b_2 was closer to the optimum calculated values. The parameters b_1 and b_2 seem not to be affected much by the sampling time.

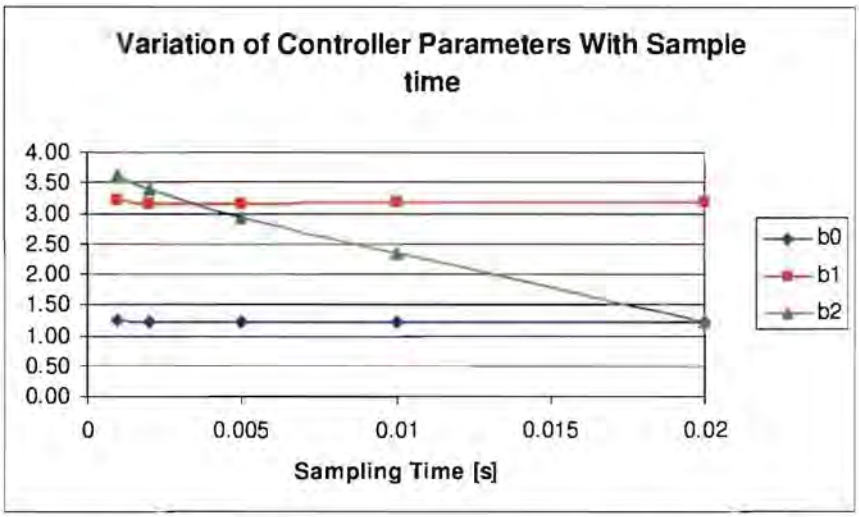


Fig 6.4.2- Effect of sampling time changes on APPIFT for second order processes

Figure 6.4.3 shows a trend line passed through the obtained values in attempt to predict the parameter values had the sampling time been infinitely small.

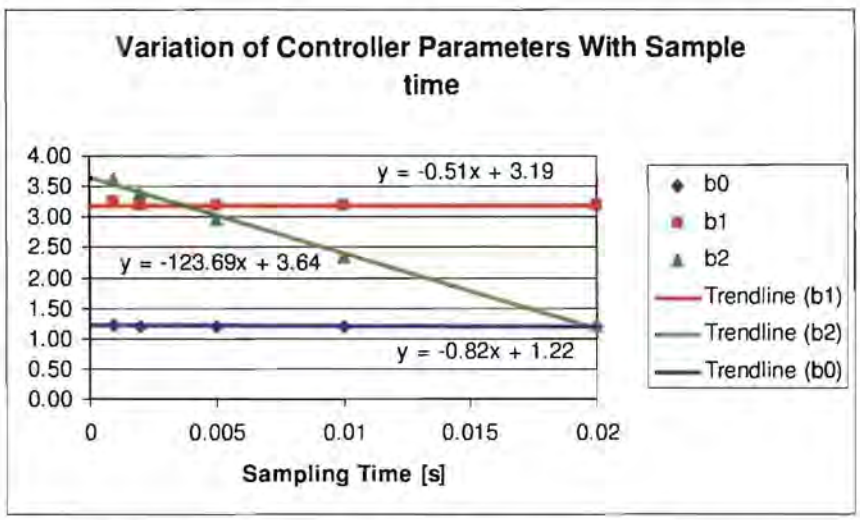


Fig 6.4.3– Trend of sampling time changes on APPIFT for second order processes

The values the parameters for a continuous sampling are predicted to be b_2 = 3.64, b_1 = 3.19 and b_0 = 1.22. Therefore the smaller the sampling time the more accurate was the approximate pole placement algorithm for second order processes. Parameter b_2 is the high frequency coefficient therefore it was most affected by the sampling time.

Nevertheless, looking at all the closed loop responses it can be noted that they all tracked the model. The reason for this is that some of the sampling times may have been too long to capture the high frequency responses. Thus the response matched the model well since the parameter b_2 does not influence the shape of response a lot (even though the value of b_2 is very dependent on sampling time). The explanation was mentioned earlier i.e. a step input does not provide a lot of high frequency excitation to tune the parameter b_2 and the sampling time was too slow to catch the high frequency response.

6.5 APPIFT FOR SECOND ORDER PROCESSES APPLIED ON FIRST ORDER APPEARING SECOND ORDER PROCESSES

After developing a pole placement algorithm for second order processes, pole placement for the damped second order processes in section 6.2 was attempted. The characteristic equation was chosen to position the poles at s = - 7 and s = -1+/-j0.5. This was to make it comparable to the characteristic equation for pole placement for first order systems, that aimed to place the poles at s = -1+/-j0.5. The damped pole was to be positioned at s = -7 because it puts it in the middle of the four non-dominant poles of the processes used previously and is not too fast for the sample time of 20ms. This was to determine if it makes a difference whether the damped pole was slower or faster than the non-dominant pole. It was expected that pole placement will not be exact because the sampling time is too large as shown in section 6.4; however the aim was to see if with the limited sampling time the final closed loop would be better than when APPIFT for first order processes was used in section 6.2.

6.5.1 Second order pole placement on second order plants that appear first order

The results on the process $g(s) = \dfrac{5}{s^2 + 10.5s + 5}$ are shown in fig 6.5.1 simulated at 20ms to make them comparable to the results of section 6.2. Once again there

is a unit step input disturbance at time t = 0 for ***IFT experiment one*** and ***IFT experiment two*** begins at t = 20.

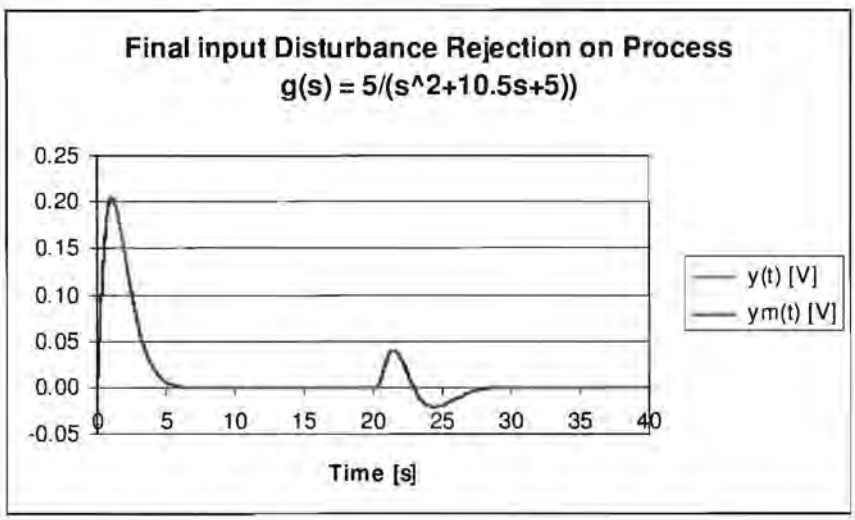


Fig 6.5.1- APPIFT with sampling time of 0.02s applied on damped second order process

The output matches the model in Fig 6.5.1. The final controller is $k(s) = \frac{0.04s^2 + 2.72s + 2.13}{s(0.01s + 1)}$. The final closed loop pole positions are located at: s = -99.93, s = -8.56 and s = -1.01+/-j0.48. Approximate pole-placement appears to have occurred for the complex conjugate poles.

The above experiment was also attempted on processes: $g(s) = \frac{4}{s^2 + 8.5s + 4}$, $g(s) = \frac{3}{s^2 + 6.5s + 3}$ and $g(s) = \frac{2}{s^2 + 2.5s + 2}$ as was done in section 6.2. The responses are given in Appendix B and results are summarised in table 6.5.1.

All the processes resulted in a closed loop that matched that of the model as shown in the Appendix B.3. The closest values to the desired pole placement were obtained with the process with the most non-dominant pole i.e. $g(s) = \frac{5}{s^2 + 10.5s + 5}$ where the imaginary poles were placed close to the desired location. The rest of the resultant closed loop poles differed from the desired

pole positions. Therefore model following was achieved, making it better than the results in section 6.2. However pole placement did not occur. Therefore the setpoint tracking dynamics may not follow that of a model with that characteristic equation because of the pole position. The combination of pole positions allowed for the model to be followed at that specific sampling rate.

TABLE 6.5.1- APPIFT for first order looking second order processes

Process	Open loop Pole positions	Closed loop pole positions
$\frac{5}{s^2+10.5s+5}$	s = -10 and s = - 0.5	s = -99.927, s = -8.555 and s = -1.009+/-j0.478
$\frac{4}{s^2+8.5s+4}$	s = -8 and s= - 0.5	s = -99.359, s = -6.724 and s = -1.208+/-j0.246.
$\frac{3}{s^2+6.5s+3}$	s = -6 and s = - 0.5	s = -98.820, s = -4.136, s = - 2.654 and s = -0.891
$\frac{2}{s^2+4.5s+2}$	s = -4 and s = - 0.5	s = -98.288, s = -2.719+/j2.155 and s = -0.774

As was mentioned earlier, when the sampling time approaches zero, the optimal parameters are also approached and pole placement will be approximate. So it was expected that in the above circumstance with a sampling time of 20ms, the optimal pole placement parameters would not be achieved. However the result is that for a unit step input disturbance rejection, the closed loop matches that of the model when sampled at that frequency. On the other hand, because the pole placement is not exact, the closed loop projection for setpoint changes will not follow the equivalent setpoint-tracking model. The setpoint-tracking output will be different for all the processes because the resultant closed loop poles are all different.

There are two reasons why the model output and the closed loop matched. The first is that as shown in section 5.2 that the time of decay does not relate directly to the characteristic equation. The time constant can be obtained by using the time that it takes for the integral of the input disturbance to reach within a certain percentage of its final value. For example 7T will be the time for the integral to reach within 0.1% of its final value. Even though all closed loops in

Appendix B.3 followed the model and the rejection took the same amount of time, they all had different maximum heights. Therefore they all should have a different time when the integral of the output is equal to a certain value. Therefore they all have different time constants as can be seen by the final closed loop pole positions.

The other reason is that the some of the sampling times may have been too long to capture some of the high frequency responses. Thus the response matched the model only at that specific sampling frequency. The larger sampling time acts as a delay on the loop.

6.5.2 Second order pole placement used on a first order process

It is rare to have a process that is purely first order and most first order systems are in fact second order or higher but have damped poles making them appear first order. In that case second order pole placement with a non-dominant closed loop pole should be used.

Consider the extreme situation that the system is in fact purely first order. When second order pole placement IFT was attempted, the gain of the controller continued to rise at a constant rate as shown in fig 6.5.2 below.

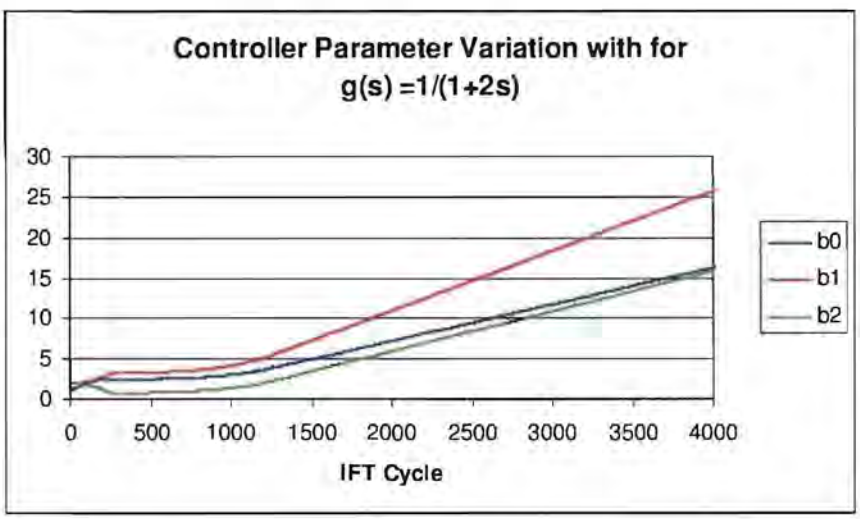


Fig 6.5.2- Controller parameter evolution for APPIFT on $g(s) = \frac{1}{1+2s}$

This was because the cost function was headed to the global minimum at controller parameters with an infinite gain. Eventually the cost function derivatives became big and led to the cost function magnitude rising until it was too large for the program to display the numerical value.

The Nyquist plot of the closed loop with the controller $k(s)=\frac{18.451s^2+27.528s+17.510}{s(0.01s+1)}$ (the controller before the cost function becomes too large) is used to analyse why this occurs. The sampling time is approximated as a dead time equal to $e^{-0.5Ts} = e^{-0.01s}$. This is shown in fig 6.5.3.

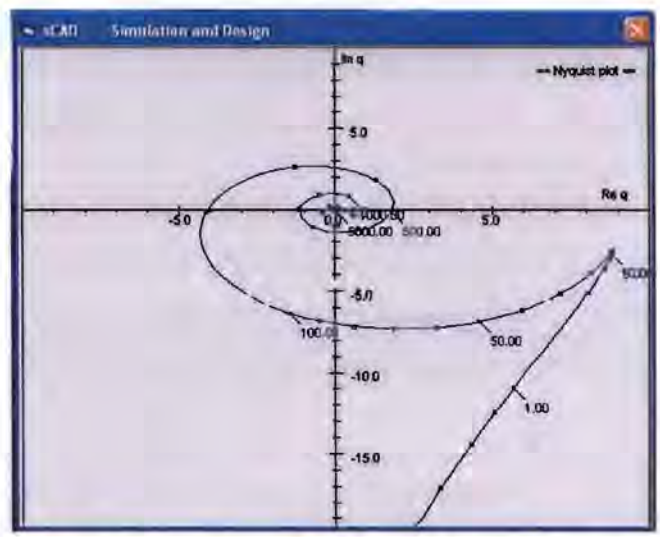


Fig 6.5.3- Nyquist plot with controller before cost function becomes too large

It can be seen that the Nyquist plot encircles the critical point at -1 yet there are no unstable open loop poles therefore according to the Nyquist criteria the above system is unstable. Therefore the reason for the rise in the cost function was that the closed loop became unstable for this controller at a sample time of 20ms.

Figure 6.5.4 shows a Nyquist plot of the process $g(s)=\frac{2}{s^2+4.5s+2}$ using the same controller as above.

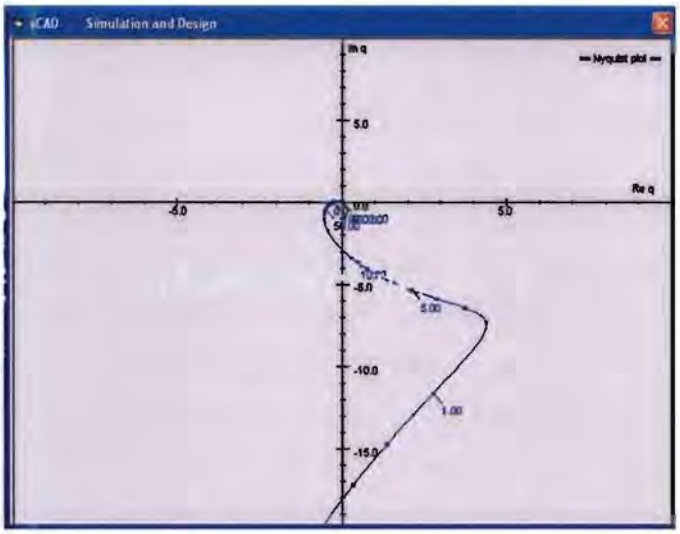


Fig 6.5.4- Same controller applied on a different process $g(s) = \dfrac{2}{s^2 + 4.5s + 2}$

The above closed loop is stable with the same controller. A Nyquist plot of the first order process $g(s) = \dfrac{1}{1+2s}$ with a sampling time of 0.001 is shown in fig 6.5.5 below.

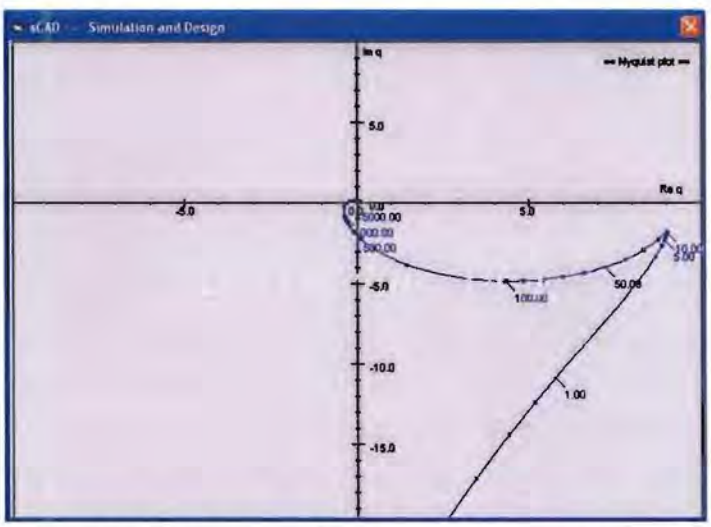


Fig 6.5.5- Nyquist plot with controller before cost function becomes too large with a sampling time of 0.001s

From fig 6.5.5 above it can be seen that a smaller sampling time increases the stability of the closed loop. Therefore the limited sampling time is the cause of the sudden increase in the cost function.

In conclusion, pole placement IFT for second order plants may not always result in the global optimum controller for achieving the desired closed loop poles. This is due to the limit in sampling time as shown earlier. It also has errors caused by the non-dominant pole added for causality. When deciding where to place the non-dominant pole of the controller, the two things to consider are the errors that it will bring to the characteristic equation and the sampling time constraints that may cause the desired pole placement not to be achieved.

6.6 APPIFT WITH IMPROVED SETPOINT TRACKING

If the sampling time is adequate for a second order process or if the closed loop is dominantly first order, approximate pole placement can be achieved. On the other hand a problem develops with the setpoint tracking response. The optimal controller gives a closed loop with a dominant zero. The result is that the step response over shoots the setpoint as mentioned in (Middleton and Graebe, 1999). This may be unacceptable for some processes and depending on the severity of the overshoot, it may result in high signals in the process input for the closed loop setpoint tracking.

On the other hand by placing the closed loop poles at a particular location, one can get the speed of disturbance rejection that they need. It is recommended that when doing IFT, it should not be done for setpoint tracking because the inclusion of a prefilter, p(s), can be used to give a desired setpoint tracking trajectory and speed as mentioned in the literature review. Should pole placement have occurred, this could be done by open loop design methods because the closed loop transfer function is known.

Assuming that there are no zeros in the process then the final model, m(s), will reflect the full transfer function closed loop. Therefore p(s) can be made a unity

gain transfer function with a pole located at the position of the controller zero. This cancels the zero that causes the overshoot in the closed loop. The result will be a closed loop that has a trajectory defined by the closed loop poles.

To be more persistent in obtaining the first order response that previously led to the pole-zero cancellation, the closed loop dynamics can be cancelled completely. This could be done because the numerator (of the controller) and the denominator (of the model) of the closed loop transfer function are known. An example of this is pole placement for a first order process $g(s)=\frac{1}{1+5s}$. The desired setpoint tracking is reflected in the model $m(s)=\frac{1}{1+s}$ and pole placement to s= - 1+/-j0.5. The result is simulated below. At time t = 0 there is a unit setpoint step change, t = 30 seconds a unit step input disturbance and t = 60 a unit step output disturbance. The final controller is $k(s)=\frac{9.371s+6.393}{s}$.

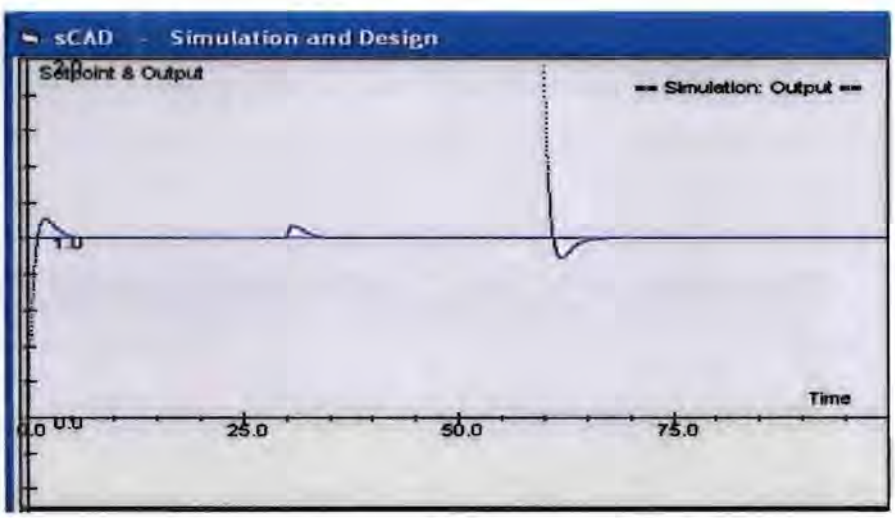


Fig 6.6.1- Pole placement results showing overshoot

It can be seen that there is overshoot in the setpoint tracking response. This is caused by the dominant zero located to the right of the poles. To give the desired setpoint tracking response a prefilter $p(s)=\frac{1}{s+1}\times\frac{5(s^2+2s+1.25)}{9.371s+6.393}$

$= \frac{5(s^2+2s+1.25)}{9.371s^2+15.764s+6.393}$ is added. Figure 6.6.2 is the response when the prefilter is added.

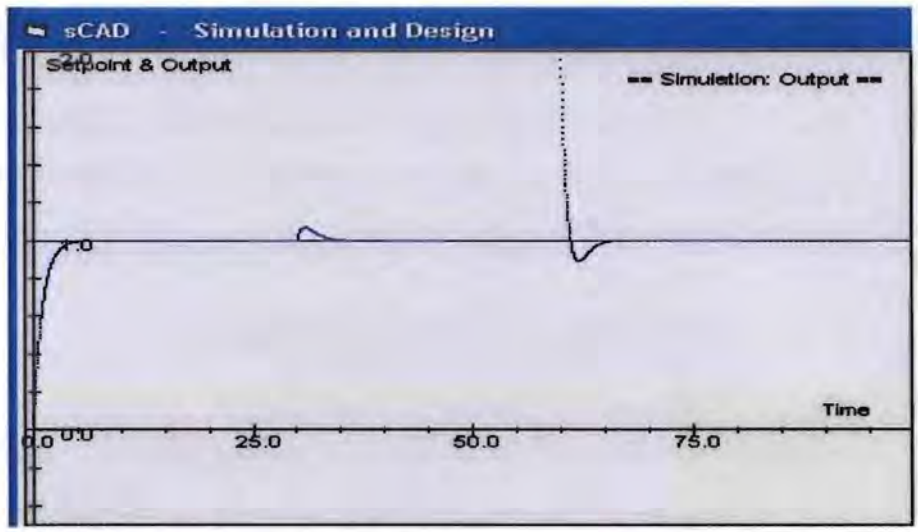


Fig 6.6.2- Pole placement results showing no overshoot when the prefilter is used

This is an improvement over a controller that standard IFT would produce i.e. one that gives pole-zero cancellation. The loss is that there is an overshoot in the output disturbance rejection; nonetheless the speed of transient decay is the same in all circumstances. The response when there is pole-zero cancellation is shown in fig 6.6.3.

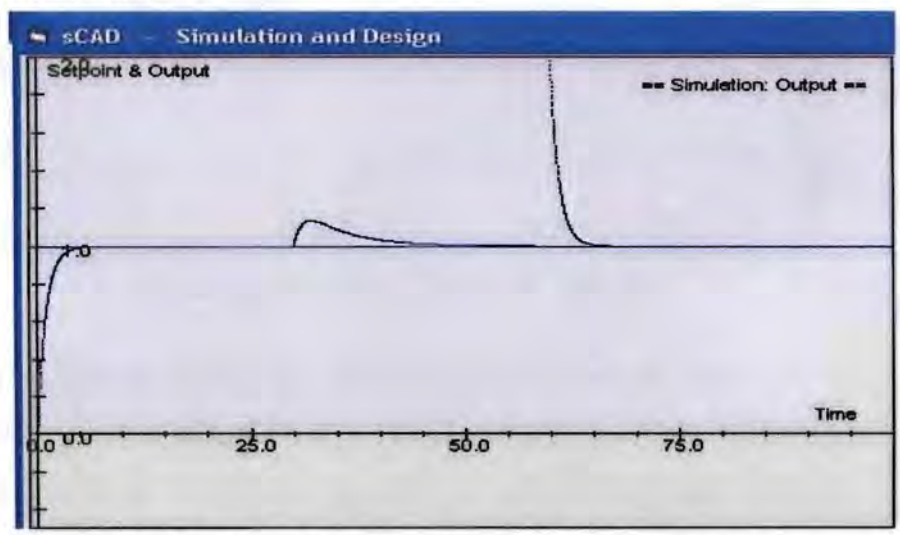


Fig 6.6.3- Normal IFT results when there is pole-zero cancellation

The controller used in the response above is $k(s)=\frac{1+5s}{s}$ which cancels the process pole. The above figure shows a closed loop that takes a long time to reject the input disturbance. Therefore the previous circumstances are overall an improved process. The end result is that a prefilter can be used to eliminate the undesired effects of pole placement while keeping the transient decay rate of all the other signals the same.

For a process with dominant or non-minimum phase zeros a possible algorithm would be to use the algorithm in (Lecchini and Gevers, 2002) to approximate the zero position. This way pole placement IFT can occur as normal and also avoid pole-zero cancellation. However this is not in the scope of this thesis.

This and the previous sections motivate the next chapter where a simpler version of approximate pole placement iterative feedback tuning is developed.

Chapter 7

Approximate Pole Placement IFT using a Prefilter

In the last chapter approximate pole placement IFT was analysed. In this chapter a simpler way of performing approximate pole placement IFT is developed. It is also analysed in a similar manner to that of the previous chapter.

7.1 DEVELOPMENT OF THE NEW POLE PLACEMENT ALGORITHM

If regular IFT for setpoint tracking is done by passing the setpoint signal through a prefilter, p(s), pole placement can occur. When p(s) has a pole that is located at the same place as the zero of the controller, the zero of the controller will be cancelled. The loop controller parameters are tuned in such a manner that the closed loop for setpoint tracking without the presence of the zero is tuned to resemble the model. Therefore it can be tuned such that the two denominators look the same, since the numerator will not have any zeros. The only difference will be the denominator. By doing this, the zeros have been cancelled in the open loop, since the zero is not present to cancel the pole to achieve the desired performance. Thus no pole-zero cancellation should be possible. These observations are now demonstrated in the mathematical equations below. In the loop considered the transfer function for setpoint-tracking is:

$$h_{yr} = \frac{n_p}{d_p} \times \frac{n_k n_g}{n_k n_g + d_k d_g} \tag{7.1.1}$$

Therefore if $d_p = n_k$ then this becomes:

$$h_{yr} = \frac{n_p n_g}{n_k n_g + d_k d_g} \tag{7.1.2}$$

The numerator n_p can then be varied to give the transfer function a unity gain at low frequency.

7.2 APPIFT USING A PREFILTER APPLIED ON FIRST ORDER PROCESSES

When the procedure is applied to a first order process the closed loop transfer function becomes:

$$h_{yr} = \frac{\frac{Ab_0}{T}}{\frac{Ab_1 s}{T} + \frac{Ab_0}{T}} \times \frac{\frac{Ab_1 s}{T} + \frac{Ab_0}{T}}{s^2 + \frac{(Ab_1 + 1)s}{T} + \frac{Ab_0}{T}}$$

$$= \frac{\frac{Ab_0}{T}}{s^2 + \frac{(Ab_1 + 1)s}{T} + \frac{Ab_0}{T}} \tag{7.1.3}$$

For the above to be equal to the second order process

$$m(s) = \frac{d}{s^2 + cs + d} \tag{7.1.4}$$

$$b_0 = \frac{dT}{A} \tag{7.1.5}$$

$$b_1 = \frac{cT - 1}{A} \tag{7.1.6}$$

Therefore pole placement to $\varphi_c = s^2 + cs + d$ can be achieved.

This was attempted for a number of first order processes, where the aim is to place the closed loop poles at s = -1+/-j0.5. The results are tabulated in table 7.2.1.

The results confirm that approximate pole placement had been achieved for all the first order processes. The pole positions were even more accurate than the

previous version of pole placement IFT. This can be seen by comparing table 7.2.1 with table 5.4.1.

TABLE 7.2.1- APPIFT using p(s) applied on first order processes

g(s)	g(s) poles	Final k(s) zeros	Closed loop h(s) poles	Damping Factor	Settling Time (7T)
$\frac{1}{1+2s}$	-0.500	-0.830	-1.01+/-j0.50	0.896	6.9
$\frac{1}{1+3s}$	-0.333	-0.713	-1.01+/-j0.49	0.900	6.9
$\frac{1}{1+4s}$	-0.250	-0.703	-1.01+/-j0.49	0.900	6.9
$\frac{1}{1+5s}$	-0.200	-0.692	-1.01+/-j0.49	0.900	6.9
$\frac{1}{1+6s}$	-0.167	-0.691	-1.00 +/- j0.50	0.894	7.0

Next to be investigated is the effect this will have on damped second order systems that appear first order.

7.3 APPIFT USING A PREFILTER APPLIED ON FIRST ORDER APPEARING SECOND ORDER SYSTEMS

The processes simulated are the same as those in section 6.2. This was to see how the closed loop performance would be affected when placement for first order processes is used on a damped second order process; hence they appear to be first order. The non-dominant poles are progressively shifted closer to the dominant pole at s = -0.5. As before a PI controller and a sampling time of 20ms are used so that the results are comparable. The aim was to place the closed loop poles at s = -1+/-j0.5 in all the cases considered.

Figure 7.3.1 is the final setpoint-tracking response for the process $g(s) = \frac{5}{s^2 + 10.5s + 5}$ once the controller parameters have settled. The prefilter is

included in the response. A unit step setpoint is applied for ***IFT experiment one*** at time t = 0s. At time t = 20s the setpoint is brought back to its initial position in preparation for ***IFT experiment two*** that occurs at time t = 40s.

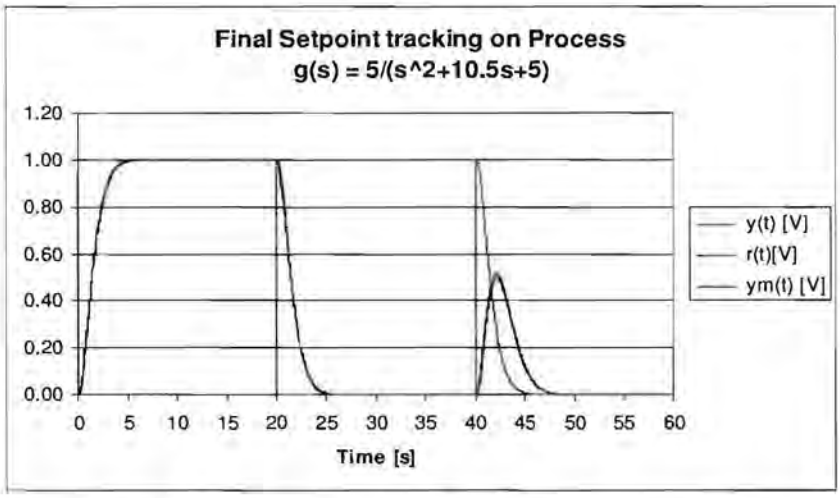


Fig 7.3.1- APPIFT using p(s) and a PI controller applied on damped second order process

The final controller was $k(s)=\frac{3.19s+3.58}{s}$ and hence the final prefilter was $p(s)=\frac{3.58}{3.19s+3.58}$. The closed loop poles are positioned at s = -8.114 and s = -1.193+/-j0.408. The closed loop with the prefilter appeared to track that of the model as shown in the fig 7.3.1. The imaginary poles are close to the desired pole positions at s = -1+/-j0.5. The differences are due to the effect of the pole at s = -10 therefore exact pole placement cannot occur.

The above experiment is repeated on a number of processes: $g(s)=\frac{4}{s^2+8.5s+4}$, $\frac{3}{s^2+6.5s+3}$, $\frac{2}{s^2+2.5s+2}$, $\frac{1}{s^2+2.5s+1}$ and $\frac{0.25}{s^2+s+0.25}$. These processes all have a pole at s = -0.5 and the non-dominant pole is progressively shifted closer to this pole. The simulation results are shown in Appendix B.4 and the results are summarised in the table 7.3.1.

TABLE 7.3.1- APPIFT using p(s) and a PI controller applied on second order processes

Process	Open loop Pole positions	Closed loop pole positions
$\frac{5}{s^2+10.5s+5}$	s = -10 and s = - 0.5	s = -8.114 and s = -1.193+/-j0.408
$\frac{4}{s^2+8.5s+4}$	s = -8 and s= - 0.5	s = -5.948 and s = -1.276+/-j0.337
$\frac{3}{s^2+6.5s+3}$	s = -6 and s = - 0.5	s = -3.372, s = -1.917 and s= -1.212
$\frac{2}{s^2+4.5s+2}$	s = -4 and s = - 0.5	s = -1.754+/j1.504 and s = -0.993
$\frac{1}{s^2+2.5s+1}$	s = -2 and s = - 0.5	s = -0.843+/-j1.630 and s = -0.814
$\frac{0.25}{s^2+s+0.25}$	s = -0.5 and s = - 0.5	s = -0.545 and s= -0.228+/-j1.175

Clearly the performance deteriorates and was worst when the two poles coincide. However, the final approximate pole placement IFT response was better than the pole-placement IFT used in section 6.2 in terms of speed and shape of response. There was always a resultant controller and the gain did not keep on rising as it did previously for pole placement when the process was $g(s)=\frac{1}{s^2+2.5s+1}$. Therefore this version of pole placement IFT is a better method than that of chapter 6 because it gives a reasonable controller (i.e. one with a limited gain) even when the process is second order with dominant first order dynamics. This version also does not need the model derivative unlike the method in chapter 6.

7.4 APPIFT USING A PREFILTER FOR SECOND ORDER PROCESSES

Since the procedure is still the same, this algorithm should work equivalently for second order systems. The relevant calculations are shown below. In this case the transfer function for setpoint tracking becomes:

$$h_{yr}(s) = \frac{b_0}{b_2 s^2 + b_1 s + b_0} \times \frac{ad(b_2 s^2 + b_1 s + b_0)}{s^3 + (adb_2 + c)s^2 + (adb_1 + d)s + adb_0}$$

$$= \frac{adb_0}{s^3 + (adb_2 + c)s^2 + (adb_1 + d)s + adb_0} \qquad (7.4.1)$$

Consider when the closed loop model is $m(s) = \frac{D}{s^3 + Bs^2 + Cs + D}$. The optimal controller parameters are the same as in equations (6.3.4), (6.3.5) and (6.3.6), namely, $b_0 = \frac{D}{ad}$, $b_1 = \frac{C-d}{ad}$ and $b_2 = \frac{B-c}{ad}$. Therefore pole placement should occur on the assumption that the pole added for causality on the controller has little effect on the process.

Figure 7.4.1 is the response obtained by doing APPIFT at a sample time of 0.001s i.e. 1 tenth of the time constant of the fastest open loop dynamic coming from the non-dominant pole added to the controller for causality. Again there is a unit step in the setpoint is applied for ***IFT experiment one*** at time t = 0s. At time t = 20s the setpoint is brought back to its initial position in preparation for ***IFT experiment two*** that occurs at time t =40s.

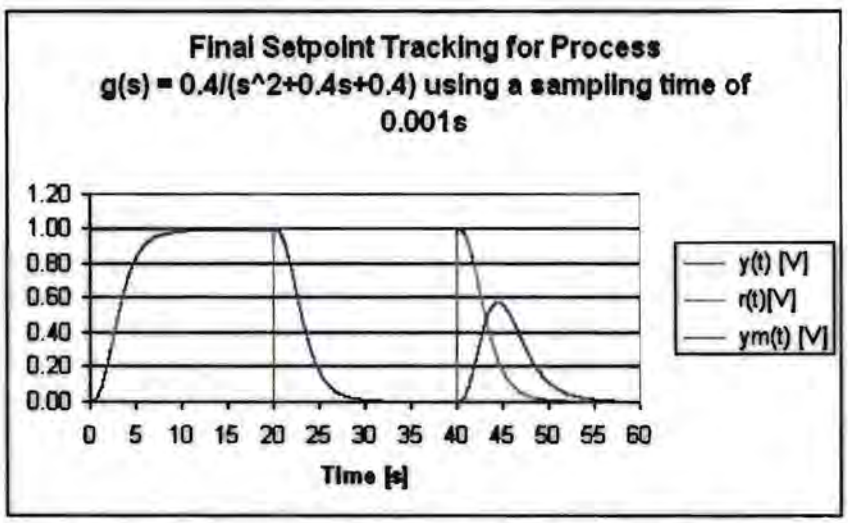


Fig 7.4.1 - Final responses for APPIFT using p(s) on second order process

The process output tracked that of the model. Pole placement was chosen to occur to places s = - 0.7+/-j0.7 and s = -0.5. The final controller obtained was $k(s)=\frac{3.65s^2+3.11s+1.19}{s(0.01s+1)}$ and thus the closed loop poles are located at s = -98.525, s = -0.692+/-j0.711 and s = -0.491. Therefore approximate pole placement had occurred. The controller parameters are close to those that were predicted.

7.5 EFFECT OF SAMPLING TIME CHANGES ON APPIFT USING A PREFILTER FOR SECOND ORDER PROCESSES

The effect of sampling time on the final closed loop poles was investigated. This was to check if this version of approximate pole placement IFT is affected in the same manner as the previous one considered in section 6.4. Pole placement was attempted for sampling times of 10, 5, 2 and 1 ms. The responses are shown in Appendix B.5. Figure 7.5.1 illustrates how the controller parameters change with sampling time

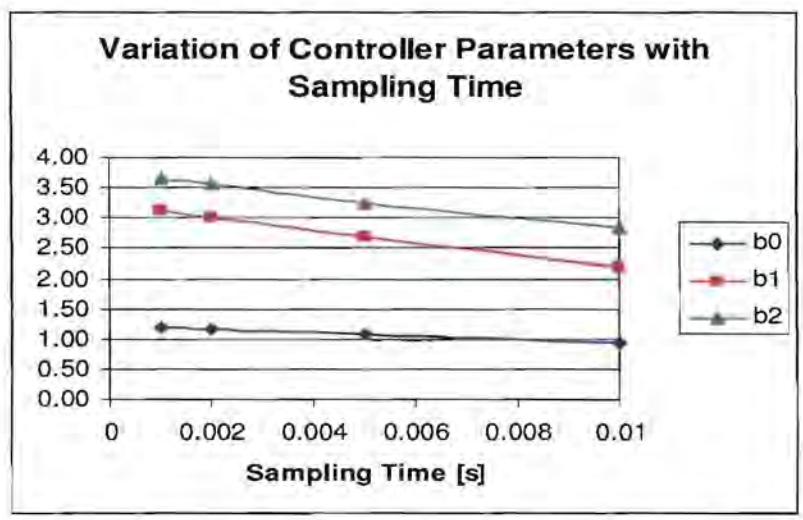


Fig 7.5.1- Sampling time changes on APPIFT using p(s) for second order processes

As before the parameters varied with sampling time, though they appear to vary more than they did when approximate placement was done by changing the

closed loop model, m(s) in section 6.4. The reason may be that the prefilter output is also affected by the sampling time. However, as shown in Appendix B.5, all the closed loop outputs tracked the model output, even though the optimum controller parameters differed in all the cases. The explanation may be that in each case the combination of the delay caused by the larger sampling time and the controller parameter values gave a closed loop output that was the same as the model. Figure 7.5.2 below is Fig 7.5.1 above with a trend line added to predict what the parameter values would be at an infinitely small sampling time.

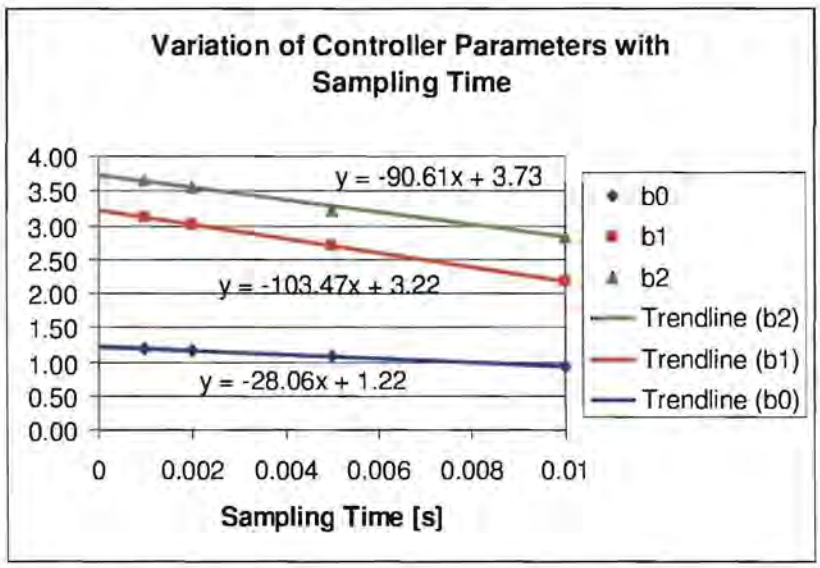


Fig 7.5.2– Trend of sampling time changes on APPIFT using p(s)

The predicted parameters are $b_2 = 3.73$ $b_1 = 3.22$ and $b_0 = 1.22$. This is close to the calculated values of $b_2 = 3.75$, $b_1 = 3.20$ and $b_0 = 1.23$. These parameters are closer to the predicted values than when using the pole placement IFT on a changing model as in section 6.4. This could be because there are fewer calculations to be made in this version of IFT therefore the resultant error is less.

When the sampling time was increased to 0.02s the controller parameter b_0 became negative. This caused a problem in the closed loop gradient calculation because this calculation uses the inverse of the controller and so this filter becomes unstable as can be seen from equation (2.1.4). However the sampling

time of 0.02s is impractical as it is twice the time constant of the fastest dynamic in the open loop (from the controller).

The controller zero is removed when tuning for the closed loop therefore is no pole-zero cancellation therefore all the closed loops have the same characteristic equation, as was the aim of this thesis. However what was exposed was that the presence of controller zero also changes the trajectory of the loop response and therefore is also a characteristic of the loop closed loop. However the speed of response is governed by the characteristic equation. The overshoot is governed by the characteristic equation in conjunction with the controller zero.

If it is undesirable to have a prefilter, p(s), in a control system, for whatever reason, it can be viewed as the filter that gives the setpoint signal required for the closed loop to tune the controller for pole placement. The prefilter does not have to be used during normal process operation. It can be used only for tuning purposes. However it is still desirable to use the prefilter in all operations so that the output does not overshoot when there is a setpoint change as expected in pole-shifting control strategies (Middleton and Graebe, 1999). The severity of the overshoot will depend on the dominance of the controller zero, which is neglected when doing pole placement.

In conclusion it is advisable to do pole placement using p(s) because it avoids most of the problems associated with the pole placement algorithm in chapter 6. It is also more accurate and allows IFT to be operated as normal. The only change to normal IFT is that the prefilter parameters must be updated whenever the controller parameters are updated and that the characteristic equation must be chosen wisely i.e. must be second order for a first order process and third order for a second order process.

7.6 APPROXIMATE POLE PLACEMENT SUMMARY

- As shown in chapter 6, when doing APPIFT with a changing model, the model derivative is not needed if the process characteristic equation can match that of

the model exactly. However where it cannot, the model derivative is needed to obtain the true minimum point of the cost function. For pole placement with a second order process the closed loop characteristic equation cannot be achieved exactly because of the pole added for causality, therefore the model derivative must be included.

- When doing APPIFT for a second order plant a non-dominant pole is needed. This non-dominant pole puts a constraint on sampling time for approximate pole placement to occur. The more non-dominant this pole becomes, the more the closed loop characteristic equation is capable of resembling that of the desired model. This will also mean that the sampling time will have to be faster.

- The sampling time affects the values of the final controller parameters and mostly the coefficient of the highest of s. Thus a small sampling time is needed to capture the high frequency responses for exact pole placement.

- Pole placement using a prefilter and the normal IFT for setpoint-tracking is advantageous. The reason is that the only change from the normal IFT is the inclusion of a prefilter that cancels the controller zero. It was also shown to be more accurate and stable than the method used in chapter 6.

In the next chapter the approximate pole placement method is applied to a DC motor to observe its performance on a physical process.

Chapter 8

Practical Application on a DC Motor

In the previous chapter approximate pole placement iterative feedback tuning was analysed for real world scenarios. In this chapter it is applied on the DC motor that gave pole-zero cancellation in chapter 4.

8.1 THE DC MOTOR SYSTEM

In this section the DC motor that was used in chapter 4 is discussed as approximate pole placement will be performed on this DC motor for speed control. The DC motor will serve as a demonstration of how APPIFT can be performed on a physical system as opposed to simulation. The DC motor set up is shown in fig 8.1.1.

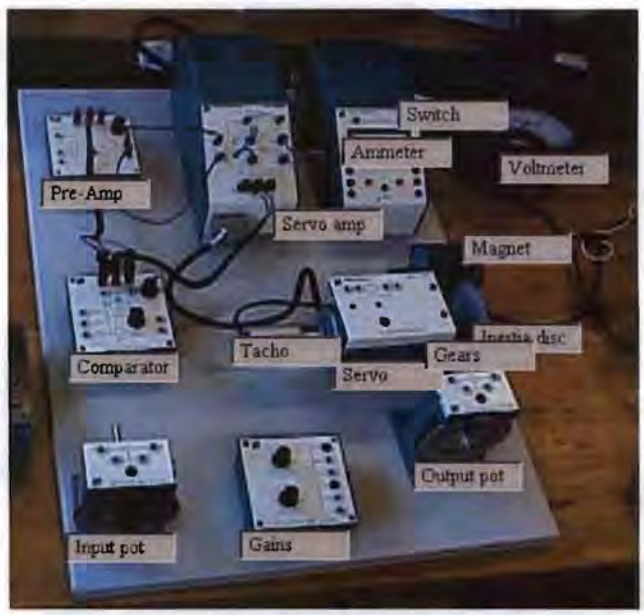


Fig 8.1.1- The field controlled DC motor System

The DC motor is a second order process with a transfer function of the form: $g(s) = \frac{A}{(1+sT_m)(1+sT_f)}$. The parameter T_m is the mechanical time constant and T_f is the electrical time constant. In many instances $T_f << T_m$ (Braae, 1994). In the investigations presented in this chapter the motor was used with a heavy inertial disc and the mechanical time constant was thus longer since it is proportional to the inertia of the disc. Therefore this makes the mechanical time constant even longer; hence the process has dominant first order dynamics and can be approximated as the process model $g(s) = \frac{A}{(1+sT_m)}$.

The motor speed is proportional to the voltage across the tachometer. To find the values of A and T, the output data was logged for nine step tests in each direction by using a square wave input signal. The step tests were done when the output is in the operation region of 5 - 8V for the speed signal. The output of the DAC was fed through the gain attenuator that was set so that its output was a third the size of its input. The first and second step tests are shown in fig 7.1.2

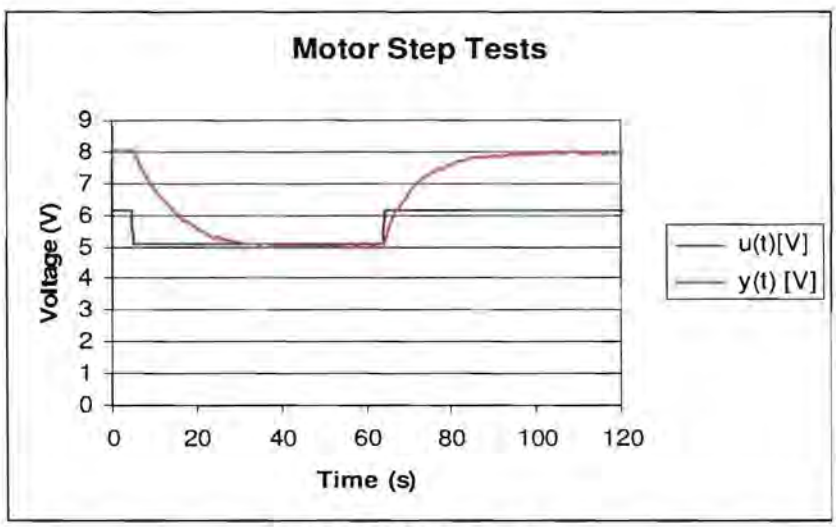


Fig 8.1.2- DC Motor Step Tests

The data from the step tests were then analysed to derive the parameter constant by nonlinear regression using an existing Microsoft Excel macro. This macro program minimizes a least-squares cost function that is based on the response data

to find the best first order representation of the data input. The step tests of this motor gave the following transfer functions:

$$g(s) = \frac{3.0 \pm 0.10}{1 + (7.3 \pm 0.34)s} \quad \text{(Step Up)}$$

$$g(s) = \frac{3.0 \pm 0.10}{1 + (10.3 \pm 0.40)s} \quad \text{(Step Down)}$$

When looking at the transfer functions it can be seen that there is a significant deviation in the time constants of transfer functions. The convergence criterion for the IFT algorithm is dependent on the process being time invariant, so this difference in motor dynamics may cause the algorithm not to settle to one point. The outcome is a DC motor that has a different transfer function when stepping up and stepping down. These linear transfer function models also change with the speed of operation because the motor is a non-linear system.

The next section discusses the program that was used on the motor.

8.2 VISUAL BASIC PROGRAM

8.2.1 Choosing a Sampling time

The program that was used for the experiments in this chapter is similar to the program that was used in the previous chapter. The difference was that the process model is not simulated. The computer DAC outputs a voltage to the motor and the motor tachometer voltage is the input into the computer ADC. For this reason the experiments had to be run in real time and required the use of timer interrupts in the Visual Basic program.

The timer in the Visual Basic program used on the computer in the University of Cape Town control laboratory had limited accuracy. And the accuracy was dependent on the timer interval desired. For this reason the sampling interval was chosen to be 250ms which was adequate in sampling the process that has a

nominal time constant of 7.3 seconds when stepping up. It can also be used for sampling a closed loop time constant of down to 2.5 seconds (i.e. ten times the sampling interval). This would allow the closed loop to be almost three times the speed of the motor in open loop; therefore it is a reasonable desirable goal.

8.2.2 Experiment Variables

This thesis involves ways of preventing pole-zero cancellation. Thus the speed of convergence is not the focus. The R matrix used in IFT is the unit matrix. The update constant γ again chosen to be $\gamma = \frac{0.01}{|Costfunctiongradient|}$. A reason for choosing the upgrade vector magnitude to be a constant size is to prevent the controller from becoming unstable from updates caused by large cost function gradients. Another problem with a large gradient update is that the controller parameters can become negative when the parameters are close to zero and the size of the upgrade is large. This can cause problems such as changing the sign of the feedback into positive feedback.

In the previous chapters this manner of defining the update constant, γ, worked well for the processes because the gains of the processes were all unity. This was partly the reason for reducing the size of the gain of the motor to one third of its actual size by including an attenuation unit in the loop. However the gain was not reduced to unity to show that APPIFT can work on a process that has a gain other than one. The reason for a process gain causing a problem with the way of defining the update constant is that a high gain reduces the size of the final controller parameters because the required gain of the controller is less. This means that the size of γ may be too big in the sense that it changes a controller parameter by a high percentage. This can prevent the controller parameters from settling to reasonable values.

8.3 APPIFT APPLIED ON THE DC MOTOR

APPIFT in this chapter is done using a prefilter. The reason is that it was decided in chapter 7 that this method was easier and more accurate than that using a changing model presented in chapter 6. Seeing that the aim is to track one transfer function, a setpoint is applied to the closed loop for ***IFT experiment one***. Then the setpoint is removed for one experiment time interval before ***IFT experiment two*** is performed. Since there are three experiment intervals and each experiment lasted 55 seconds (to capture the whole closed loop output before it settles) the time period for a cycle of iteration is 2.75 minutes. Therefore if the initial parameters are far from the optimum parameters the algorithm may take a long time to settle to the optimal.

Since the initial model that led to pole-zero cancellation was one with a time constant of 2.5 seconds; the model chosen for the desired closed loop response was one of second order with a high damping factor of 0.854 and a time constant of 2.5 seconds. Its transfer function model chosen is $m(s)=\frac{0.2}{s^2+0.8s+0.2}$ to place the poles at s = -0.4+/-j0.2. The VRFT method was not used to obtain an initial controller because it would require a smaller sampling time than the than the 250ms needed by the DC motor as shown in Appendix A.1. For this reason the initial controller was again chosen to be $k(s)=\frac{s+1}{s}$ and hence the initial prefilter was thus $p(s)=\frac{1}{1+s}$ to cancel the closed loop zero. The aim of this section is to show that APPIFT works in real time on a physical process and it avoids pole-zero cancellation. Since the upward and downward motor dynamics were different it was decided that the upward transfer function of the motor was to be tracked. The responses that were obtained during the initial cycle of the IFT optimization are shown in fig 8.3.1. Once again at time t = 3 seconds there is a 3 Volt step in setpoint into the closed loop for ***IFT experiment one***. At approximately t = 58 seconds the setpoint is removed. At t = 113 seconds the error that was recorded during the ***IFT experiment one*** is added to the setpoint for ***IFT experiment two***.

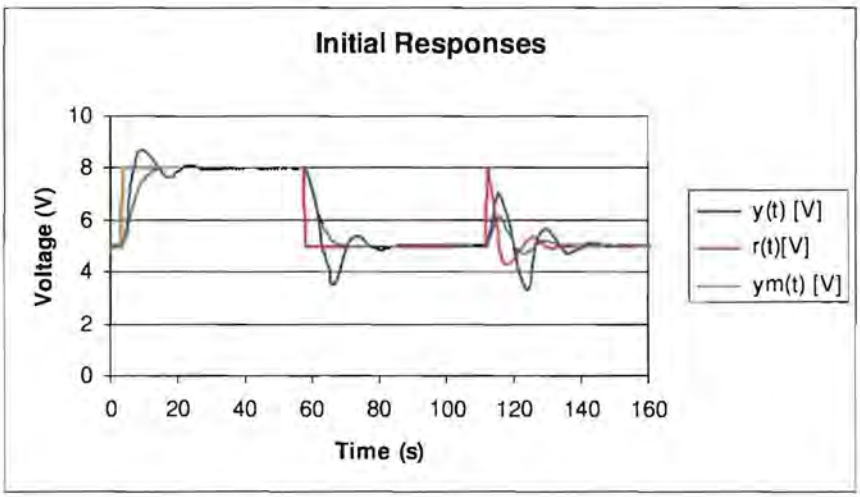


Fig 8.3.1 – Initial responses using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

As can be seen, initially the closed loop had a longer time constant and a smaller damping factor than the model.

Figure 8.3.2 shows the closed loop after APPIFT has been performed.

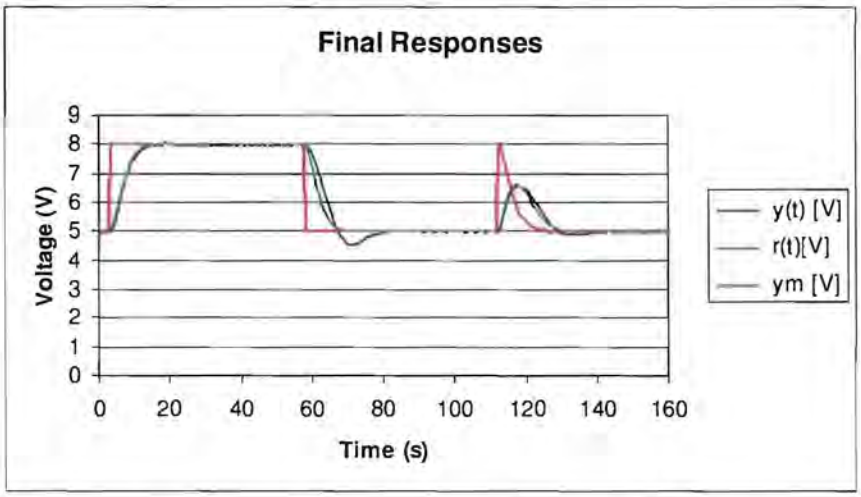


Fig 8.3.2- Final responses using model $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

The closed loop and the model track each other for the upward trajectory. Therefore approximate pole placement IFT has taken place.

Figure 8.3.2 shows that the controller zero did not cancel the process pole. The controller zero settled at approximately s = -0.3. This is not near the process pole.

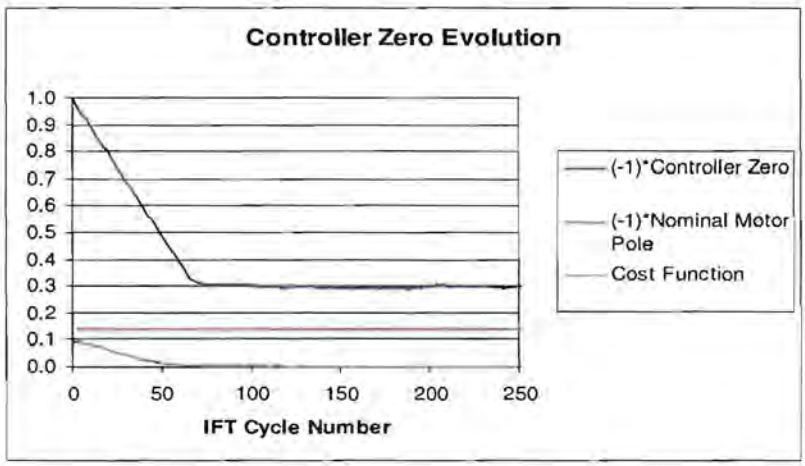


Fig 8.3.3- Controller zero evolution using model $m(s) = \frac{0.2}{s^2 + 0.8s + 0.2}$

This experiment was repeated again however this time with the initial controller $k(s) = \frac{1.2s + 0.1}{s}$ with a zero at s = -0.083, providing a zero position that is on the other side of the process pole. This is to show that the closed loop will not settle at the closed loop pole even if it reaches it at some point. The evolution of the controller zero is shown in fig 8.3.4.

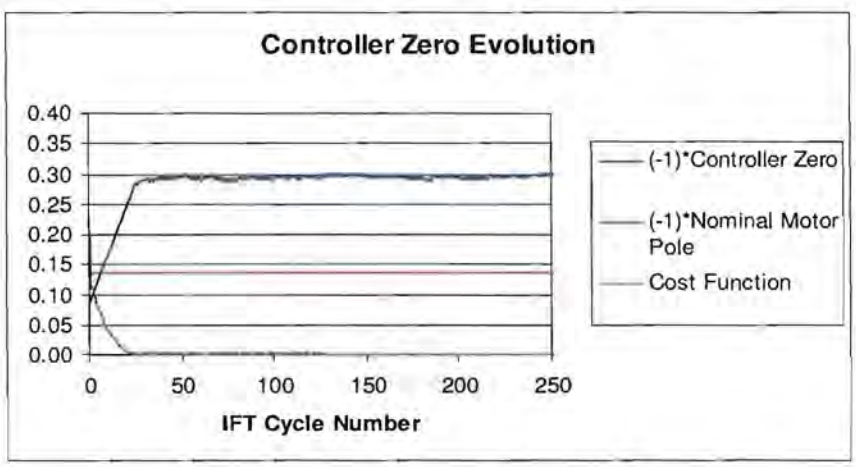


Fig 8.3.4- Controller zero evolution using $m(s) = \frac{0.2}{s^2 + 0.8s + 0.2}$ part two

Again the controller parameters settle at around the same values irrespective of where the initial controller zero is.

This experiment was repeated for pole placement to different pole positions. The new model chosen is $m(s) = \frac{0.18}{s^2 + 0.6s + 0.18}$ so as to place the poles at s = -0.3+/-j0.3. This ensures that the response can have maximum overshoot without oscillation (the damping factor is 0.707) and a larger time constant.

The initial responses obtained during the PPIFT are shown in fig 8.3.5.

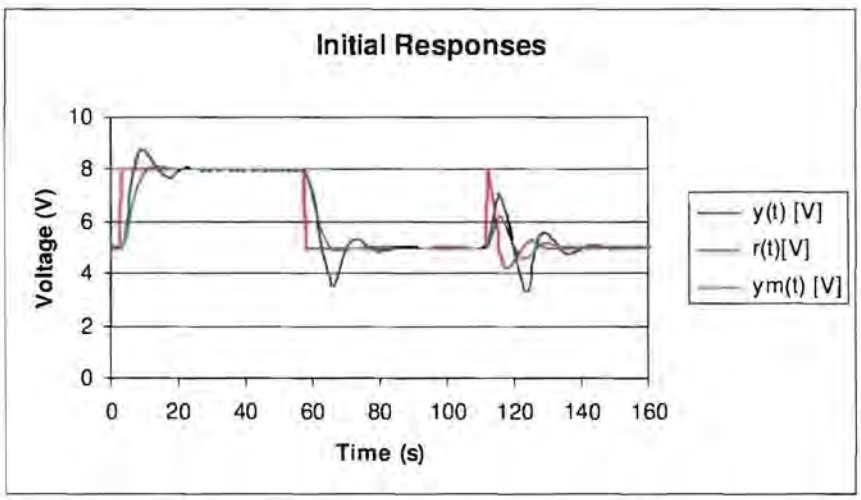


Fig 8.3.5- Initial responses using $m(s) = \frac{0.18}{s^2 + 0.6s + 0.18}$

Again the signals occur at the same times as the previous case and the initial controller and prefilter are $k(s) = \frac{s+1}{s}$ and $p(s) = \frac{1}{s+1}$ respectively.

Figure 8.3.6 shows the final responses after the controller parameters have settled. As can be seen the closed loop follows that of the model for the upward trajectory even if the model is one with overshoot.

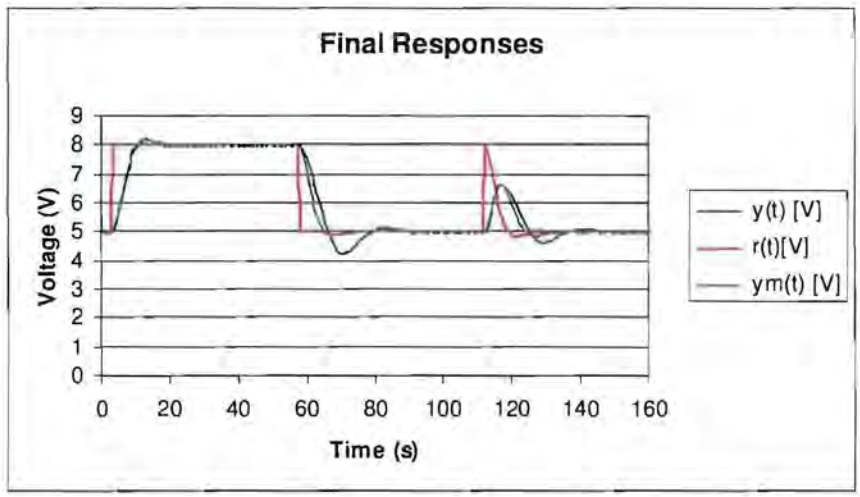


Fig 8.3.6- Final responses using $m(s) = \dfrac{0.18}{s^2 + 0.6s + 0.18}$

The evolution of the controller parameters is shown in fig 8.3.7.

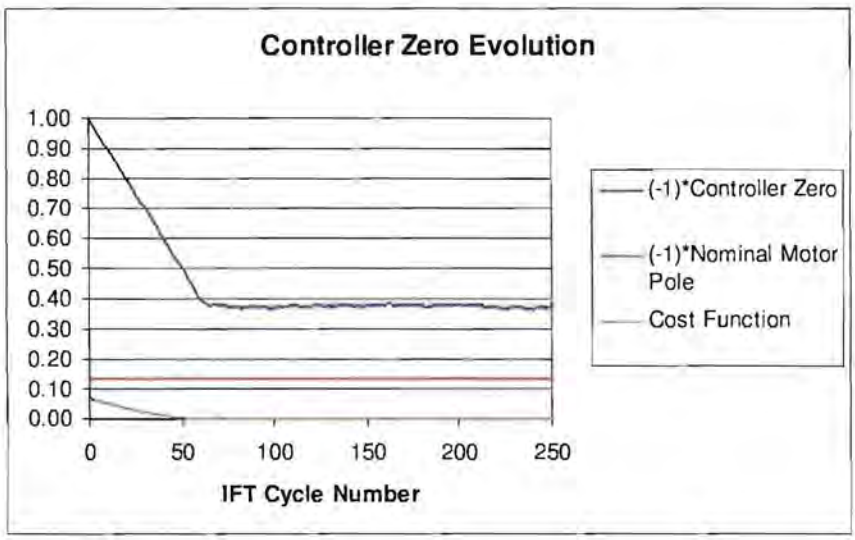


Fig 8.3.7- Controller zero evolution using model $m(s) = \dfrac{0.18}{s^2 + 0.6s + 0.18}$

The final average controller is $k(s) = \dfrac{1.33s + 0.5}{s}$ (data on the CD) which has a zero which settles at a different place as expected. This is unlike normal IFT that would have caused the controller zero to settle on the process pole irrespective of the desired model time constant.

The next attempt is to show that this pole placement can work on a process with a different transfer function. Therefore the aim now is to track downwards trajectory of the motor to a 3 volt step. The initial responses are shown in fig 8.3.8 below.

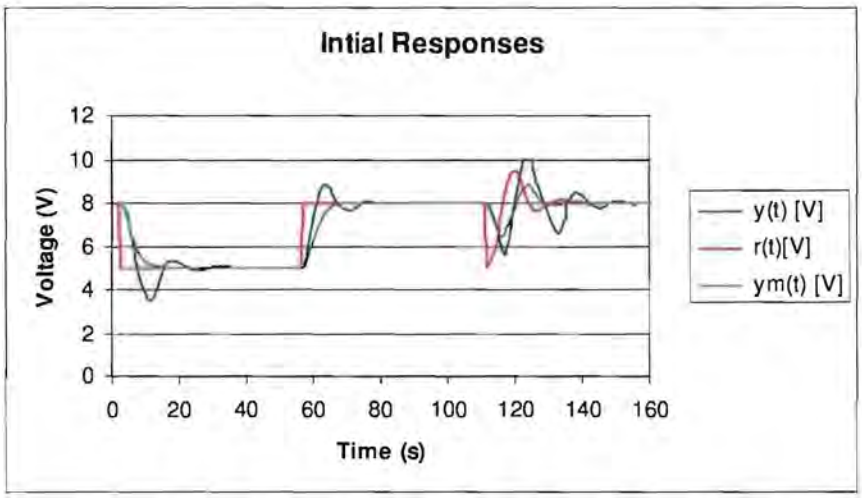


Fig 8.3.8- Initial responses for downward step using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

As before the initial controller and prefilter are $k(s) = \dfrac{s+1}{s}$ and $p(s) = \dfrac{1}{s+1}$ respectively. The closed loop response after 250 APPIFT cycles is shown in fig 8.3.9 below.

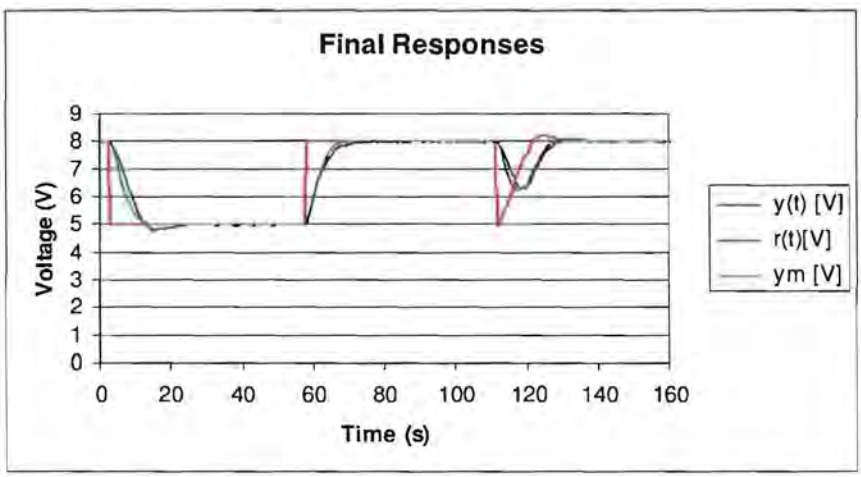


Fig 8.3.9- Day 1 responses for downward step using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

The above response did not track the model exactly after 250 cycles. The controller parameter evolution is shown in fig 8.3.10.

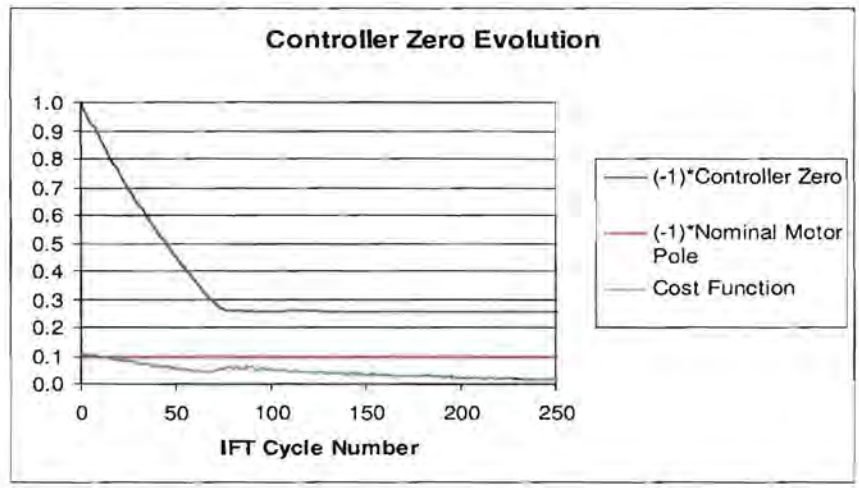


Fig 8.3.10- Day 1 Controller zero evolution when using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

It can be seen that the controller zero position has settled away from the process pole, as expected. However, after the 250 IFT cycles the closed loop did not look like the model because the controller parameters were still incrementing. This is illustrated in fig 8.3.11 below.

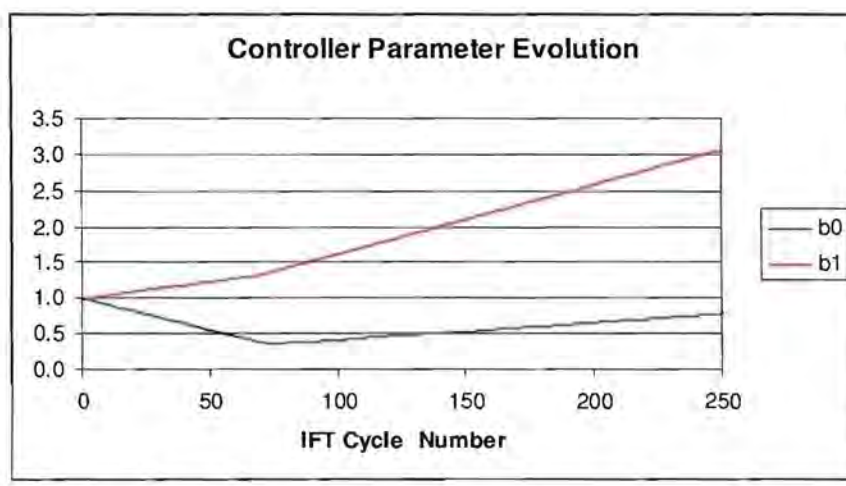


Fig 8.3.11- Day 1 Controller parameter evolution using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

However the controller zero position did not seem to be moving to any other value. Nevertheless to make sure this experiment was done on a second day. However because the controller values did not appear like they were about to settle soon, this was repeated but this time with larger initial controller parameters. The controller chosen was $k(s)=\frac{8s+2}{s}$. This kept the controller zero at s= -0.25 of which it already was and started the algorithm at higher controller values. Figure 8.3.12 shows the zero evolution after a further 200 cycles.

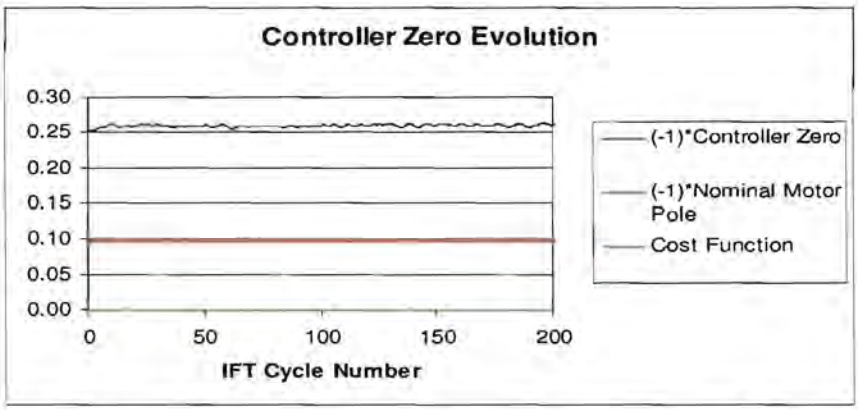


Fig 8.3.12- Day 2 controller zero evolution using model $m(s)=\frac{0.2}{s^2+0.8s+0.2}$

The controller zero has settled at a constant value confirming that pole-zero cancellation did not occur. Figure 8.3.13 below shows that the controller parameters are settling to a final value.

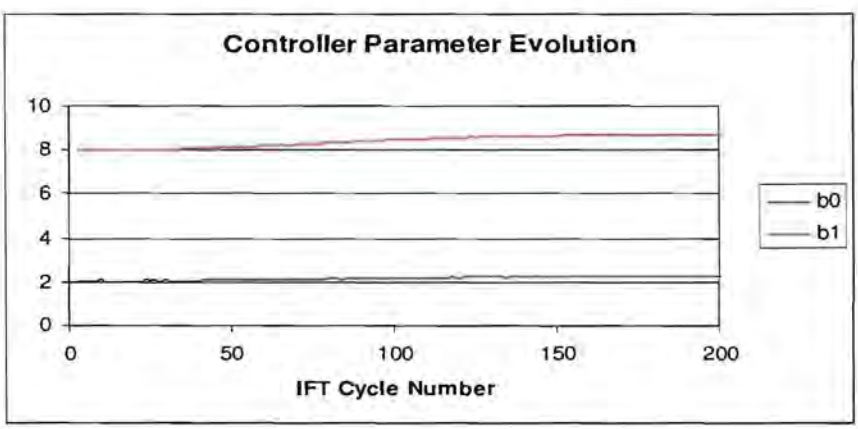


Fig 8.3.13- Day 2 controller parameter evolution using $m(s)=\frac{0.2}{s^2+0.8s+0.2}$

Figure 8.3.13 shows that the controller parameters had now settled fig 8.3.14 shows that the closed loop and the model were now the same.

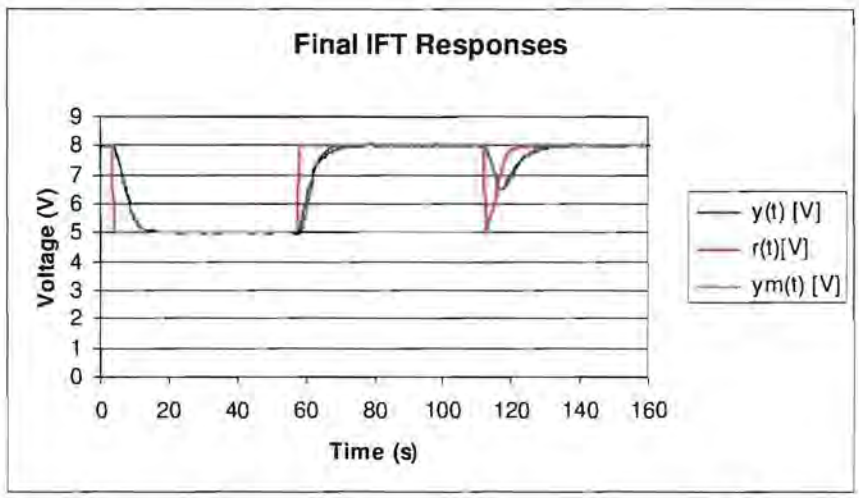


Fig 8.3.14- Final responses with downward step using $m(s) = \dfrac{0.2}{s^2 + 0.8s + 0.2}$

As can be seen the downward response now tracked the model. However in this case the upwards step was not too different from the model. This suggests that when having a system that has two different time constants when stepping up and down, the pole placement should occur on the one with a larger time constant. This may be because the larger time constant will require more gain on the controller for pole placement to occur. However a detailed study of such nonlinear systems are not part of the aim of this thesis which was to avoid pole-zero cancellation and show that the benefits of APPIFT noted in the simulations are achieved on a physical process, like the DC motor.

From the above experiments it can be said that the approximate pole placement has been successfully applied to a physical system. The next chapter concludes this thesis.

Chapter 9

Conclusions and Recommendations for Future Work

This chapter concludes the observations from this thesis and makes recommendation for future work.

9.1 CONCLUSIONS

In this section the conclusions are drawn from the results of the experiments and discussions in this thesis.

9.1.1 IFT can give a system with poor internal performance

This thesis demonstrated that under some conditions the controller given by IFT can be one that cancels the process poles. This was shown to occur through simulation and on a real system. Though this will only occur if the controller complexity is enough for pole-zero cancellation to be achieved. Nevertheless it is shown that through pole-zero cancellation, the desired model can be achieved. In this thesis this is demonstrated through the used of a PI controller and a first order process and model.

9.1.2 IFT for avoiding pole-zero cancellation

Four different methods for preventing pole-zero cancellation were proposed and investigated. All these methods avoided pole-zero cancellation under certain conditions. Still it was approximate pole placement IFT that gave a closed loop with dynamics that resembled that of the model specifically in terms of damping and speed of rejection. Therefore it does not give a closed loop with poor internal

performance. This method was then chosen as the best method for further investigation in this thesis and was applied to a DC motor where it achieved the desired results.

9.1.3 Approximate pole placement is best for first order and second order dominant systems

Approximate pole placement iterative feedback tuning was analysed in this thesis. It was found to work very well for first order and second order processes. The APPIFT method for second order processes is highly dependent on the sampling time of the program. The reason is that it requires the use of a non-dominant pole and has controller parameter that is a coefficient of a high power of s. However without the limit on sample time this method works very well.

9.1.4 Approximate pole placement iterative feedback tuning works in practice

Approximate pole placement was done on a DC motor system. It was shown that pole placement does in fact work on nonlinear physical system that can be approximated by linear models. If IFT is to be done using a process model, the better option would therefore be to use a prefilter that will cancel the controller zero so that the closed loop is tuned to yield the closed loop poles that resemble those of the model characteristic equation. This will give a closed loop that is the same as the desired model in terms of damping factor and speed of response.

9.2 FUTURE WORK RECOMMENDATION

In this section some future work is recommended. The recommendations are based on parts of the IFT methods that were explored during this thesis project but were not within of the scope of the present project.

9.2.1 Performing APPIFT on a microprocessor

APPIFT in this thesis was done on a motor for speed control. For position control the motor system can be approximated as a second order process thus APPIFT for

second order processes is needed. As it was shown in chapter 6, pole placement for second order processes needs a controller with a non-dominant pole added for causality. This controller places a requirement on the sampling time. A microprocessor with an internal crystal can accurately sample at high speeds. Therefore pole placement for a second order process can be performed for position control.

9.2.2 Performing approximate pole placement virtual reference feedback tuning

The APPIFT done in this thesis was all performed with the same initial controllers. VRFT is a method used to obtain initial controller parameters before using IFT for "fine tuning". This can not be used for the APPIFT because the controllers zeros are cancelled by the prefilter. A possible method for performing approximate pole placement virtual reference tuning is proposed in Appendix A.2.

9.2.3 Finding a faster means for achieving a settling point

When doing IFT with controller parameters that are far from the optimum, it can take a long time before the controller parameters settle. When performing IFT, data is collected for a specific period of time. This period of time must be longer than it takes for the process transients to settle for the IFT experiments. However usually the aim is to decrease the closed loop settling time, thus the settling time is usually faster on the last IFT cycle. There to reduce convergence time, there can be a means of detecting when the process output has settled before beginning the next IFT experiment. For example: when the output has reached within a certain standard deviation of the model output then the next IFT experiment begins.

9.2.4 Finding a means of having unrestricted controller when doing APPIFT

As shown in chapter 6, when doing pole placement for a first order process a controller with two variable parameters is needed and for a second order process a controller with three variable parameters is needed. IFT is designed for a controller with restricted complexity. However for the purpose of APPIFT, there is a need to be able to increase the degrees of freedom of the controller complexity

automatically if it is evident that pole placement will not be achieved with the given controller complexity.

9.2.5 Using IFT as a modelling technique

Normal IFT operates by tuning the controller parameters until the closed loop matches the model output. If the model and the process output look exactly the same, then closed loop has a transfer function that is like the model. Suppose that IFT was to be done to tune the process model to resemble the closed loop. When the process model and the closed loop look exactly the same, the transfer function of the closed loop will be that of the model. Using this model and controller parameters the model transfer function can be reverse engineered to find the transfer function of the process. An example of a possible manner of doing this is provided in Appendix A.4. This technique can also be compared to other normal system ID techniques (e.g. RLS).

References

Braae, M. 1994. *Control theory for electrical engineers.* Cape Town: UCT Press

Campi, M.C., Lecchini, A., & Savaresi, S.M. 2002. Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica.* 38:1337-1346.

Chen, D., & Seborg, E. 2002. PI/PID controller design based on direct synthesis and disturbance rejection. *Industrial and engineering chemical research.* 41:4807-4822. [Online] Available: http://pubs.acs.org/cgi-bin/article.cgi/iecred/2002/41/i19/pdf/ie010756m.pdf

Gevers, M. 2002. A decade of progress in the iterative process control design: from theory to practice. *Journal of Process Control.* 12:519-531.

Hildebrand, R., Lecchini, A., Solari, G., & Gevers, M. 2004. Prefiltering in iterative feedback tuning: optimization of the prefilter for accuracy. *IEEE Transactions on Automatic Control.* 49: 1801-1805.

Hildebrand, R., Lecchini, A., Solari, G., & Gevers, M. 2005a. Asymptotic accuracy of iterative feedback tuning. *IEEE Transactions on Automatic Control.* 50: 1182-1185.

Hildebrand, R., Lecchini, A., Solari, G., & Gevers, M. 2005b. Optimum prefiltering in iterative feedback tuning. *IEEE Transactions on Automatic Control.* 50: 1196-1200

Hjalmarsson, H., Gevers, M., Gunnarson, S., & Lequin, O. 1998. Iterative feedback tuning: theory and applications. *IEEE Control Systems.* 26-41.

Hjalmarsson, H., & Gevers, M. 2003. Special section on algorithms and applications of iterative feedback tuning. *Control Engineering Practice.* 11(9):1021

Hjarlmarsson, H. 2002. Iterative feedback tuning – an overview. *International journal of adaptive control and signal processing*. 16:373-395

Laplant, P.A. 1999. *Comprehensive dictionary of electrical engineering*. CRC IEEE Press

Lecchini, A., Gevers, M. & Maciejowski, J. 2006. An iterative feedback tuning procedure for loop transfer recovery. *Proc. of 14th IFAC Symposium on System Identification*. April 2006, Newcastle. 1097-1102.

Lecchini, A., & Gevers, M. 2002. On iterative feedback tuning for non-minimum phase plants. *Proceedings of the 41st IEEE Conference on Decision and Control.* December 10-13, Las Vegas. 4:5658-4663.

Maciejowski, J.M. 1989. *Multivariable feedback control design*. Reading: Addison-Wesley

Machaba, M. 2004. Investigation into iterative feedback control. MSc. Thesis. University of Cape Town. Cape Town

Middleton, R.H., & Graebe, S.F. 1999. Slow stable open-loop poles: to cancel or not to cancel. *Automatica*, 35:877-886.

Prochazka, H.M., Gevers, M., Anderson, B.D.O., & Christel, F. 2005. Iterative feedback tuning for robust controller design and optimization. [Online] Available: http://www.inma.ucl.ac.be/~gevers/PublisMig/S7.pdf [2007, August 16]

Rapson, M. 2007. Pareto analysis of controller design methodologies for integrator plus dead time processes. *Eurocon*. 2607-2606.

Sikaundi, J., & Braae, M. 2007. Preventing pole-zero cancellation for Improved input disturbance rejection. *Proceedings of the CISSE 2007 Conference*. [CD-ROM]. Available: CISSE 2007 international joint conferences.

Sobota, J., M. & Schlegal, M. Unknown. Iterative feedback tuning of a PID controller. [Online]. Available: http://www.rexcontrols.com/_new/downloads/clanky/ift_iftarticle.pdf [2007, June 22]

Solari, G.E. 2005. Iterative model-free controller tuning. Ph.D. Thesis. Universite Catholique de Louvain. [Online]. Available: http://edoc.bib.ucl.ac.be:81/ETD-db/collection/available/BelnUcetd-08042005-092900/unrestricted/IFT_Thesis.pdf [2007, April 22]

Wahlberg, B. 2005. On iterative feedback tuning and disturbance rejection using simple noise models. [Online]. Available: http://www.ee.kth.se/php/modules/publications/reports/2005/1773.pdf [2007, August 15]

Wei, H-Y,. Ginsberg, S.I,. & Braae, M. 2006. Engineering an iterative feedback controller. *Proc. SACAC Control Conf.* July, Durban. 64-69.

Appendix A

Complementary Techniques for Iterative Feedback Tuning

A.1 VIRTUAL REFERENCE FEEDBACK TUNING

The main content of this section is taken from (Campi et al, 2002).

Virtual Reference Feedback Tuning (VRFT) is similar to IFT in the sense that it does not need a model of the process to tune the controller. However it is a one shot technique that produces a controller with just one set of data based on a step test. It is sub-optimal for restricted controller classes therefore it is a complementary technique to IFT which is optimal for all controllers.

In this section the theory is developed in the s-plane and is a change from the z-plane used in (Campi et al, 2002). The idea of VRFT is that there is a process output $y = \frac{kg}{1+kg} r$ in closed loop for a one degree of freedom control structure and a desired model output, $y = mr$. When the closed loop and the desired model output are the same then the input into the two transfer functions will give the same output, y. There is a certain process output, y when a process has a particular input i.e. $y = gu$. Suppose that a known set of data of u is used as an input into the process g to obtain another set of data y. Using this particular output data set, y, the setpoint, r, required by the desired closed loop model, m, to obtain an output equivalent to y can be calculated. This is done by passing this output through the

inverse of the model (i.e. $r = m^{-1}y$). This setpoint, r is referred to as the virtual reference as it was not applied to the closed loop.

In closed loop feedback the input into a controller, k, is the error, e, between the setpoint and output i.e. $u = ke$. This error, e, can be obtained using the virtual reference and the output i.e. $e = r\text{-}y$. When both the closed loop and the desired model and have this virtual reference as a setpoint, the outputs will be the same for the optimal controller. Therefore the optimal controller is one that produces the vector of data u when fed with e.

As a result the criterion to be minimised is: $J = \frac{1}{N}\sum_{t=1}^{N}(u(t) - k(s;\rho)e(t))^2$.

Let $k(s;\rho) = \beta^T(s)\theta$

For example for a PI controller $k(s) = \frac{b_1 s + b_0}{s}$ will be $k(s;\rho) = \begin{pmatrix} \frac{1}{s} & 1 \end{pmatrix}\begin{pmatrix} b_0 \\ b_1 \end{pmatrix}$

Then,

$$J = \frac{1}{N}\sum_{t=1}^{N}(u(t) - \varphi^T(t)\theta)^2 \qquad \text{(A.1.1)}$$

Where

$$\varphi(t) = \beta^T(s)e(t) \qquad \text{(A.1.2)}$$

When the optimum parameters have been obtained, the cost function is zero and the parameter vector is defined by:

$$\theta_N = \left[\sum_{t=1}^{N}\varphi(t)\varphi(t)^T\right]^{-1}\left[\sum_{t=1}^{N}\varphi(t)u(t)\right] \qquad \text{(A.1.3)}$$

In the case that the controller does not belong to the class of controllers that can give the desired model, A.1.1 will not be zero at the minimum point. Therefore e

and u are passed through a filter L(s) to obtain an approximate minimum point. The procedure can be obtained from (Campi et al, 2002).

Consider the following case that was studied in this thesis: A PI controller, $k(s)=\frac{b_1 s+b_0}{s}$, a first order process $g(s)=\frac{A}{1+Ts}$ and a first order model $m(s)=\frac{1}{1+T_m s}$. This first order model can be achieved in closed loop with the complexity of this controller as shown in chapter 4. Using a step input, u(t), into the process g(s), the output data, y can be collected. However in obtaining the virtual reference r(t) the inverse of the model m(s) is needed. This will give the transfer function $\frac{1+T_m s}{1}$ but this is non-causal. This will thus require the use of a filter to add non-dominant poles so that the inverse transfer function can be achieved. This non-dominant pole will have to have a steady state gain of one so that it does not change this model by much such as:

$$m(s)=1+\frac{T_m s}{(1+T_c s)} \tag{A.1.4}$$

where T_c is chosen to be small., compared to T_m. For example when T_c = 0.01 the inverse model is $m(s)^{-1}=1+\frac{T_m s}{(1+0.01s)}$. However this will put a constraint on the sampling time of the simulation that was chosen to be less that 10 times smaller than T_c. Therefore a sampling time of dt = 0.001 can be used.

This was applied on the transfer function for the process $g(s)=\frac{1}{1+5s}$, using the model $m(s)=\frac{1}{1+2s}$ and its approximate inverse $m(s)^{-1}=1+\frac{2s}{(1+0.01s)}$. From equation A.1.3 the controller parameters were given by:

$$\begin{pmatrix} b0 \\ b1 \end{pmatrix} = \begin{bmatrix} \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)^2 & \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times e_t \\ \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times e_t & \frac{1}{N}\sum_{t=1}^{N} e_t^2 \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times u_t \\ \frac{1}{N}\sum_{t=1}^{N} e_t \times u_t \end{bmatrix} \quad \text{(A.1.5)}$$

and resulted in the controller $k(s) = \frac{0.5 + 2.5s}{s}$. This is the optimum controller that can be achieved with this configuration as shown in section 4.2.1. Note that the summations are divided by N to prevent them from growing too big as N becomes large.

A.2 POLE PLACEMENT VIRTUAL REFERENCE FEEDBACK TUNING

Since Virtual Reference feedback tuning can be used to obtain the initial parameters before performing IFT as a fine tuning technique, there needs to be a method to obtain the initial parameters for the approximate pole placement algorithm developed in this thesis. VRFT uses the inverse of the model to obtain the optimal parameters needed to achieve the closed loop model, however in pole placement the model zeros are unknown since these are the zeros of the optimal controller that is being sought. Therefore the traditional VRFT cannot be used for pole placement.

However the pole placement algorithm in chapter 7 performs pole placement using a prefilter that cancels the process pole. VRFT for pole placement is developed in this section. Once again consider the case of the use of a PI controller, $k(s) = \frac{b_1 s + b_0}{s}$ and first order process $g(s) = \frac{A}{1+Ts}$. In closed loop this will give

the closed loop $h(s) = \dfrac{\frac{A}{T}(b_1 s + b_0)}{s^2 + \frac{Ab_1 + 1}{T}s + \frac{Ab_0}{T}}$ that was the case study in this thesis.

Clearly the zero positions that would achieve pole placement to the required positions are not known because there is no process model. However if the zero of the controller is cancelled by a prefilter on the setpoint the closed loop transfer function becomes $h(s) = \dfrac{\frac{Ab_0}{T}}{s^2 + \frac{Ab_1 + 1}{T}s + \frac{Ab_0}{T}}$. Therefore there are no zeros in the closed loop and as a result no zeros are needed in the model. Thus the model can be of the form $m(s) = \dfrac{d}{s^2 + cs + d}$. The normal VRFT procedure can now be applied. The inverse of the above model will be needed; therefore non-dominant poles will have to be added to the inverse model $m(s)^{-1} = \dfrac{s^2 + cs + d}{(es^2 + fs + d)}$ where e and f will have to be small. This will allow for the reference model to be obtained. The error signal is now needed to obtain the controller that will give that output when the input is the virtual reference signal $e_p = pr\text{-}y$ however p is not known since it is the inverse of the final controller numerator with a gain of one. The process input $u = ke_p = k(pr\text{-}y)$ so that:

$$u = \frac{b_1 s + b_0}{s}\left(\frac{b_0}{b_1 s + b_0} r - y\right)$$

$$= \left(\frac{b_0}{s} r - \frac{b_1 s + b_0}{s} y\right)$$

$$= \left(\frac{(r - y)b_0 - b_1 sy}{s}\right)$$

$$= \left(\frac{eb_0 - b_1 sy}{s}\right)$$

$$= \begin{pmatrix} \frac{e}{s} & -y \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} \tag{A.2.1}$$

Therefore $\varphi^T = \begin{pmatrix} \frac{e}{s} & -y \end{pmatrix}$ can be used in equation (A.1.3) and VRFT can be performed as normal to obtain:

$$\begin{pmatrix} b_0 \\ b_1 \end{pmatrix} = \begin{bmatrix} \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)^2 & \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times -y_t \\ \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times -y_t & \frac{1}{N}\sum_{t=1}^{N} y_t^2 \end{bmatrix}^{-1} \begin{bmatrix} \frac{1}{N}\sum_{t=1}^{N}\left(\frac{e_t}{s}\right)\times u_t \\ \frac{1}{N}\sum_{t=1}^{N} - y_t \times u_t \end{bmatrix} \tag{A.2.2}$$

The challenge is to select the position of the non-dominant poles to obtain the virtual reference. This will also pose a restriction on the sampling time requirements in digital implementations.

A.3 IFT FOR NON-MINIMUM PHASE PROCESSES

This section is obtained mainly from (Lecchini and Gevers, 2002)

When doing IFT for non-minimum phase processes that includes those processes with a delay, both the zero and the delay must be present in the model. The reason is simply that in model following control, if the zero is not present in the model the controller may attempt to cancel the zero as it tries to reduce the phase lag. Since the integrator is fixed in IFT the controller zero will be tuned to cancel the integrating pole. However when doing IFT it is assumed that there is little or no knowledge of the process. Therefore the non-minimum phase zero location is not known.

Assuming that the transfer function from the setpoint to the output is stable, it is generally known that it can be expressed as a Laguerre expansion such as:

$$T(z,\rho)=\sum_{k=1}^{\infty} g_k(\rho)L_k(z,a) \tag{A.3.1}$$

$$L_k(z,a)=\frac{K}{z-a}\left(\frac{1-az}{z-a}\right)^{k-1} \quad \text{with} \quad K=\sqrt{1-a^2}$$

The adjustable reference model can be modelled by:

$$M(z,\eta)=\sum_{k=1}^{n}\eta_k L_k(z,a) \tag{A.3.2}$$

subject to the constraint: $M(1,\eta)=1$.

For pole placement, the criterion will be minimised with respect to both the controller parameters and the model parameters i.e. $\frac{\partial J}{\partial \rho}$ and $\frac{\partial J}{\partial \eta}$ are needed. The adjustable parameters of the model will be tuned to obtain the process zeros and delay. Therefore this algorithm models the process zeros while also performing IFT. The zeros of the model are completely free so the controller will spend its degrees of freedom into pole placement. However this is approximate pole placement because the controller may not have enough degrees of freedom to place the process poles. Since the model is a sum of Laguerre functions in terms of poles in a, the model will have n poles located at the same place. There will be a possibility of n-1 zeros. Therefore there are n extra derivatives to calculate.

In (Lecchini and Gevers, 2002) when the Cost function was optimised for pole placement, the resulting controller cancelled the process pole. The model numerator also duplicated the remaining controller zero. This can be avoided by using this method in conjunction with the pole placement algorithm in chapter 7. This is because the use of the prefilter cancels the controller zero and the desired closed loop model can contain zeros that are also optimised during the IFT. The result will

be that the model will neither duplicate the controller zero nor cancel the process pole.

A.4 OFFLINE POLE PLACEMENT USING IFT

Normal IFT is done by tuning the controller parameters so that the closed loop can resemble that of the desired model in a least squares sense according to a cost function such as the errors squared cost function shown below.

$$J = \frac{1}{2N}\left[\sum_{i=1}^{N}(y - ym)^2\right] \tag{A.4.1}$$

The derivative of the cost function with respect to the controller parameters is used to tune the controller parameters iteratively. The model derivative is zero so it stays constant.

For simplicity once again consider the following case: A PI controller, $k(s) = \frac{b_1 s + b_0}{s}$ and a first order process $g(s) = \frac{A}{1+Ts}$. It is shown in section 5.4 that if a second order model is used and the zero of this model has zeros that are at the same position of the controller zero, the IFT algorithms will perform pole placement. Therefore if the model is $m(s) = \frac{d}{b_0} \times \frac{(b_1 s + b_0)}{s^2 + cs + d}$ the controller parameters will be adjusted so that the closed loop characteristic equation becomes $s^2 + cs + d$.

Another manner of performing pole placement is to cancel the controller zero from the closed loop using a prefilter and using a model will no zero. Therefore the model will be of the form: $m(s) = \frac{d}{s^2 + cs + d}$. The closed loop transfer function for any controller parameters will be:

$$h(s) = \frac{\frac{Ab_0}{T}}{s^2 + \frac{Ab_1 + 1}{T}s + \frac{Ab_0}{T}} \tag{A.4.2}$$

since the prefilter has cancelled the controller zeros.

In this section a different manner of performing pole placement is suggested. IFT should be performed and optimised according to what model parameters would result in the smallest cost function in a least squares sense. Therefore ***IFT experiment one*** is performed by applying a setpoint into the closed loop and that of the chosen initial model. During this experiment the output, y and the model output ym are collected. The difference is that this ***IFT experiment one*** happens once on the closed loop. The data is stored and is continuously used to tune the model offline. Thus for the cost function stated earlier the derivatives:

$$\frac{\partial J}{\partial \sigma} = -\frac{1}{N}\sum_{i=1}^{N} em \frac{\partial ym}{\partial \sigma} \tag{A.4.3}$$

will be needed. Where σ represents the model parameters.

The model:

$$m(s) = \frac{d_0}{s^2 + d_1 s + d_0} \tag{A.4.4}$$

can then be tuned using the same IFT algorithm i.e. $\sigma_{i+1} = \sigma_i - \gamma_i R_i^{-1} \frac{\partial J}{\partial \sigma}(\sigma_i)$.

Once the optimal parameters have been obtained (i.e. the optimum values of d_0 and that of d_1) the optimal parameters that can give pole placement can be obtained. Therefore since d_0, d_1, b_0 and b_1 are known, the values of A and T can be calculated.

Using these values of A and T that are calculated one can calculate the required values of b_0 and b_1 that would give the desired pole positions. Using:

$$b_1 = \frac{cT - 1}{A} \tag{A.4.5}$$

$$b_0 = \frac{dT}{A} \tag{A.4.6}$$

Therefore this method uses iterative identification to perform pole placement. Normal pole placement IFT can then also be used tune the system so that the bias of noise in the original signal is eliminated.

Appendix B

Supporting Figures for the Analysis of Approximate Pole Placement IFT

B.1 APPIFT FOR FIRST ORDER PROCESSES APPLIED ON FIRST ORDER LOOKING SECOND ORDER PROCESSES

This section is a continuation of the experiments mentioned in section 6.2, where the aim was to place the closed loop poles at s = -1+/-j0.5. The second order processes investigated have damped poles and thus they appear to be first order. At time t= 0s there is a unit step input disturbance applied to the processes for ***IFT experiment one***. ***IFT Experiment two*** begins at time t = 20s where the output during experiment one is added to the closed loop as a new input disturbance.

It can be seen from Fig B.1.1, Fig B.1.2 and Fig B.1.3 that the initial input disturbance was similar in all cases. All the closed loops improved on the input disturbance rejection. However process $g(s) = \frac{2}{s^2 + 4.5s + 2}$ had a final input disturbance rejection time that was almost double that of the model. Clearly the input disturbance rejection became worse as the faster process pole was moved closer to the slower one.

As mentioned in Section 6.2 any further movement of the slower process pole past s = -2 resulted in the controller parameters increasing without bounds.

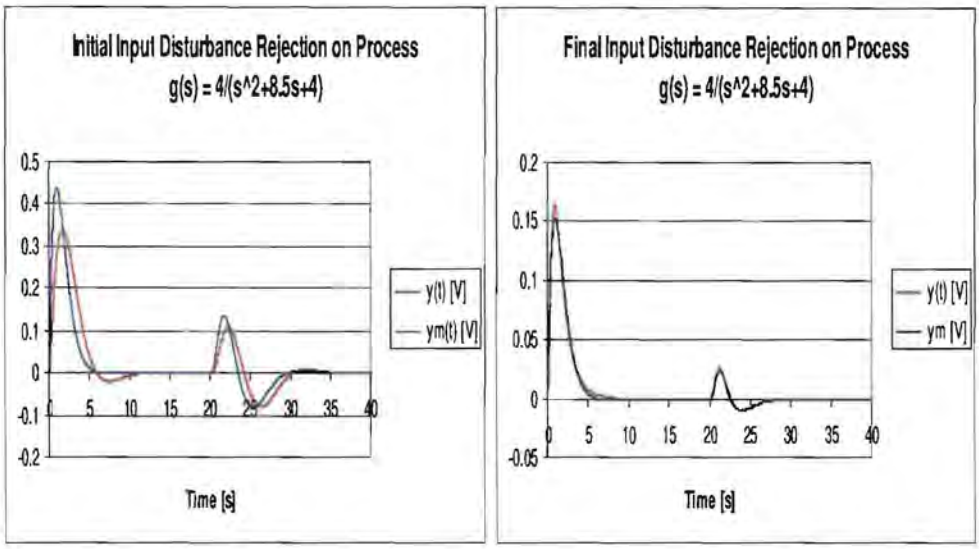


Fig B.1.1- Initial and final input disturbance rejection using process $g(s) = \dfrac{4}{s^2 + 8.5s + 4}$

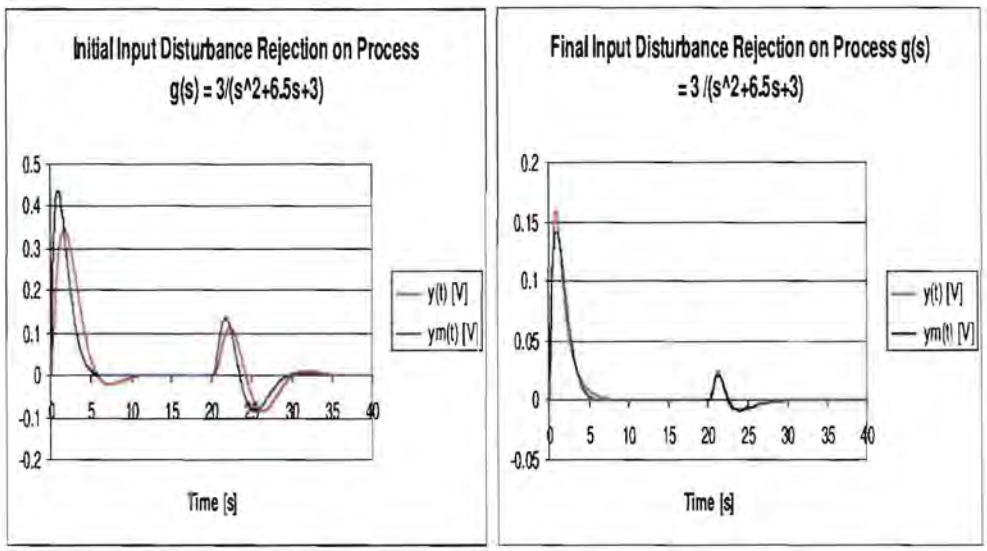


Fig B.1.2- Initial and final input disturbance rejection using process $g(s) = \dfrac{3}{s^2 + 6.5s + 3}$

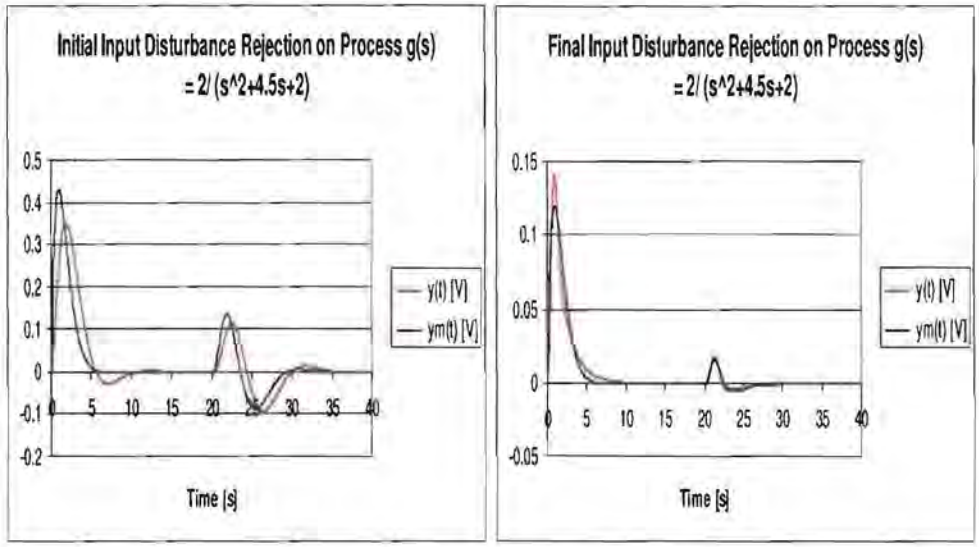


Fig B.1.3- Initial and final input disturbance rejection using process $g(s)=\dfrac{2}{s^2+4.5s+2}$

B.2 EFFECT OF SAMPLING TIME ON APPIFT FOR SECOND ORDER PROCESSES

This section is a continuation of section 6.4 where APPIFT for second order processes is implemented on the process $g(s)=\dfrac{0.4}{s^2+0.4s+0.4}$ for different sampling times. The aim was to see how the sampling time affects pole placement.

It can be seen in fig B.2.1 that the model output and the closed loop followed each other for all the sampling times. However the final values for parameter b_2 was different and hence the closed loop pole positions are different. The reason for this is that the high frequency coefficient b_2 needs a small sampling time to be tuned to the optimum. The square wave used for tuning the controller provides little high frequency excitation therefore a small sampling time is needed to tune this parameter. This is shown in section 6.4.

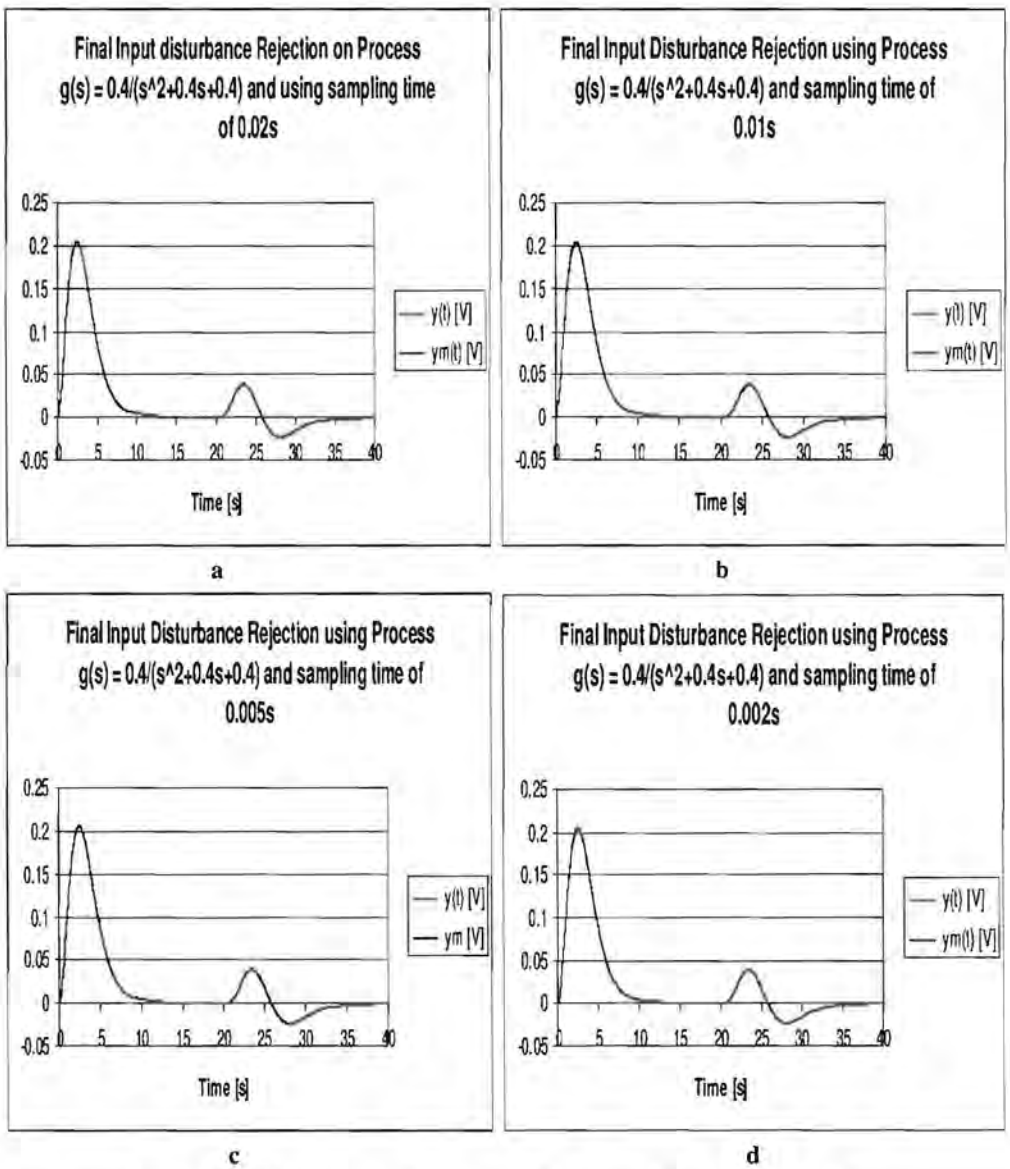


Fig B.2.1- Effect of sampling time on closed loop disturbance rejection

B.3 APPIFT FOR SECOND ORDER PROCESSES APPLIED ON FIRST ORDER APPEARING SECOND ORDER PROCESSES

This section is a continuation of section 6.5. Pole placement for second order processes was applied to damped second order processes. The algorithm was operated at a sampling time of 0.02 seconds so as to compare the results with when

first order pole placement was used. The chosen locations for pole placement are s = - 7 and s = 1+/-j0.5. The closed loop reactions are shown below.

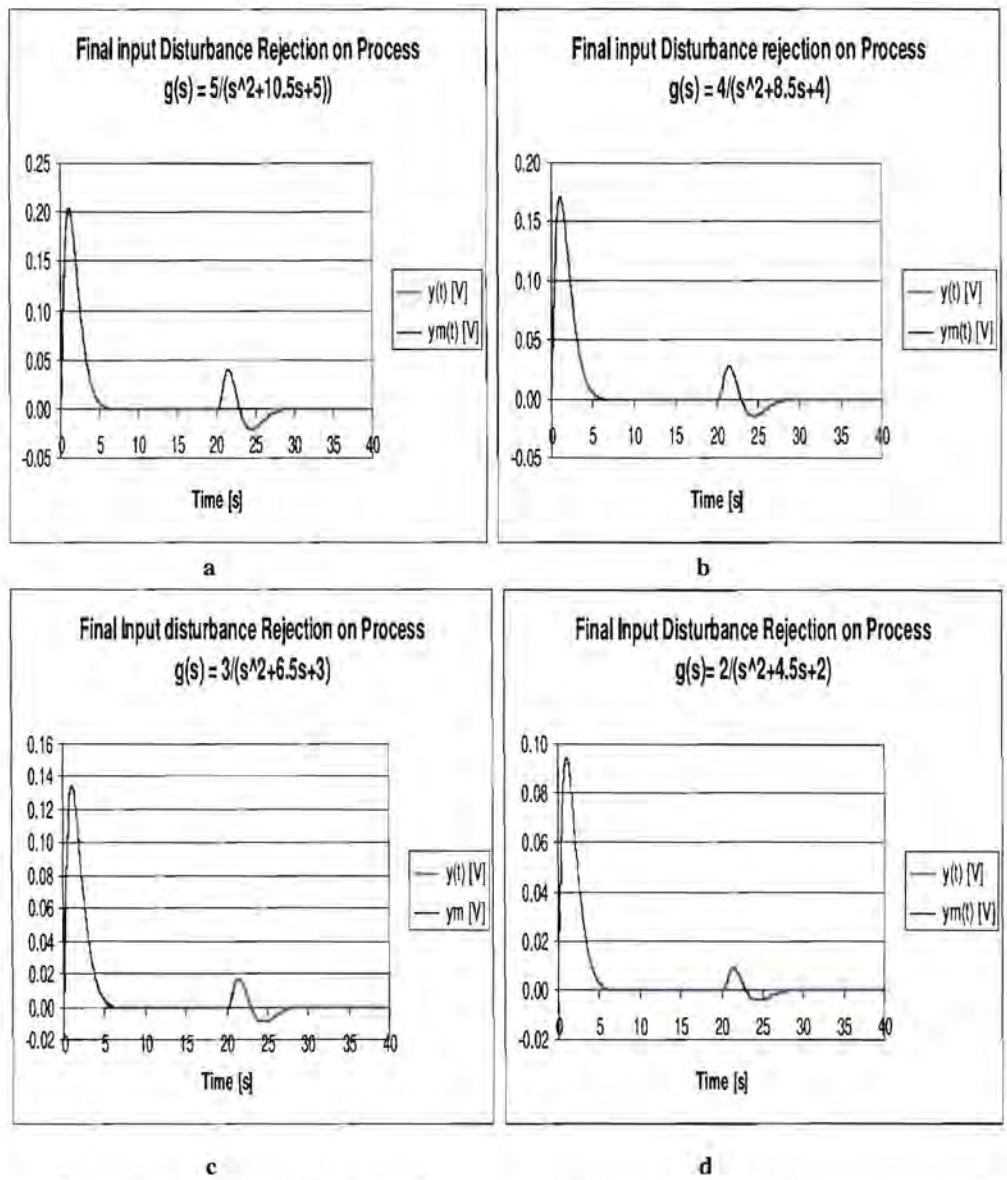


Fig B.3.1- APPIFT for second order processes applied on damped second order processes

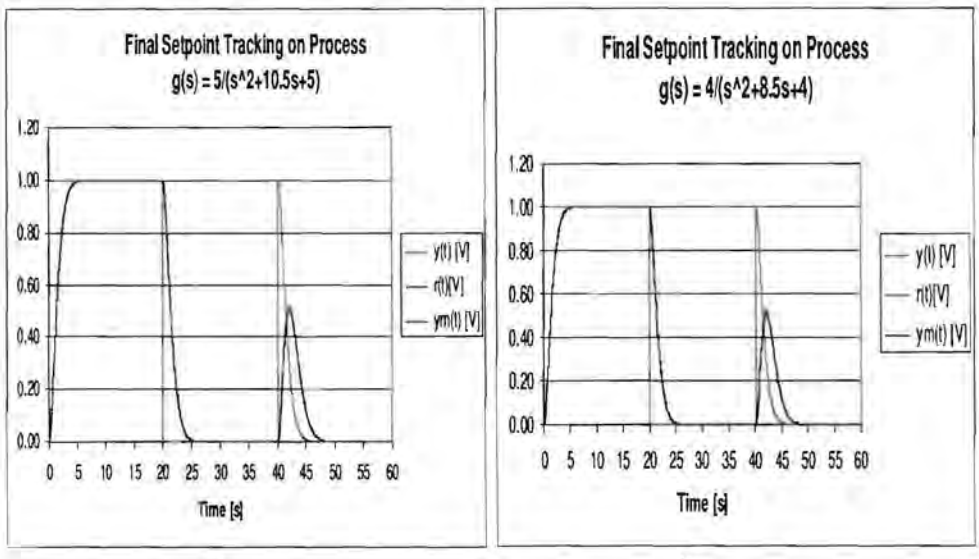
All the above processes followed the output of the model. The disturbance rejection time was the same in all cases. However it can be seen that the maximum height of the disturbance rejection was different in all the cases. Exact pole placement was

not expected in any of the above cases as this was done at a sampling time of 0.02s. It was shown that this sampling time does not provide an optimum controller.

B.4 APPIFT USING A PREFILTER FOR FIRST ORDER PROCESSES APPLIED ON FIRST ORDER APPEARING SECOND ORDER PROCESS

In chapter 7 pole placement using a prefilter is developed. This was also applied on second order processes that appear first order so that it can be compared to the pole placement in chapter 6. This section is a continuation of section 7.2. The aim was to place the closed loop poles at s = -1+/-j0.5. The results are shown fig B.4.1.

In this figure there is a unit step setpoint change at time t = 0 seconds for ***IFT experiment one***. The setpoint is then brought back to zero in preparation for ***IFT experiment two*** that begins at t = 40 seconds. The prefilter is included in the response.



a b

Continued

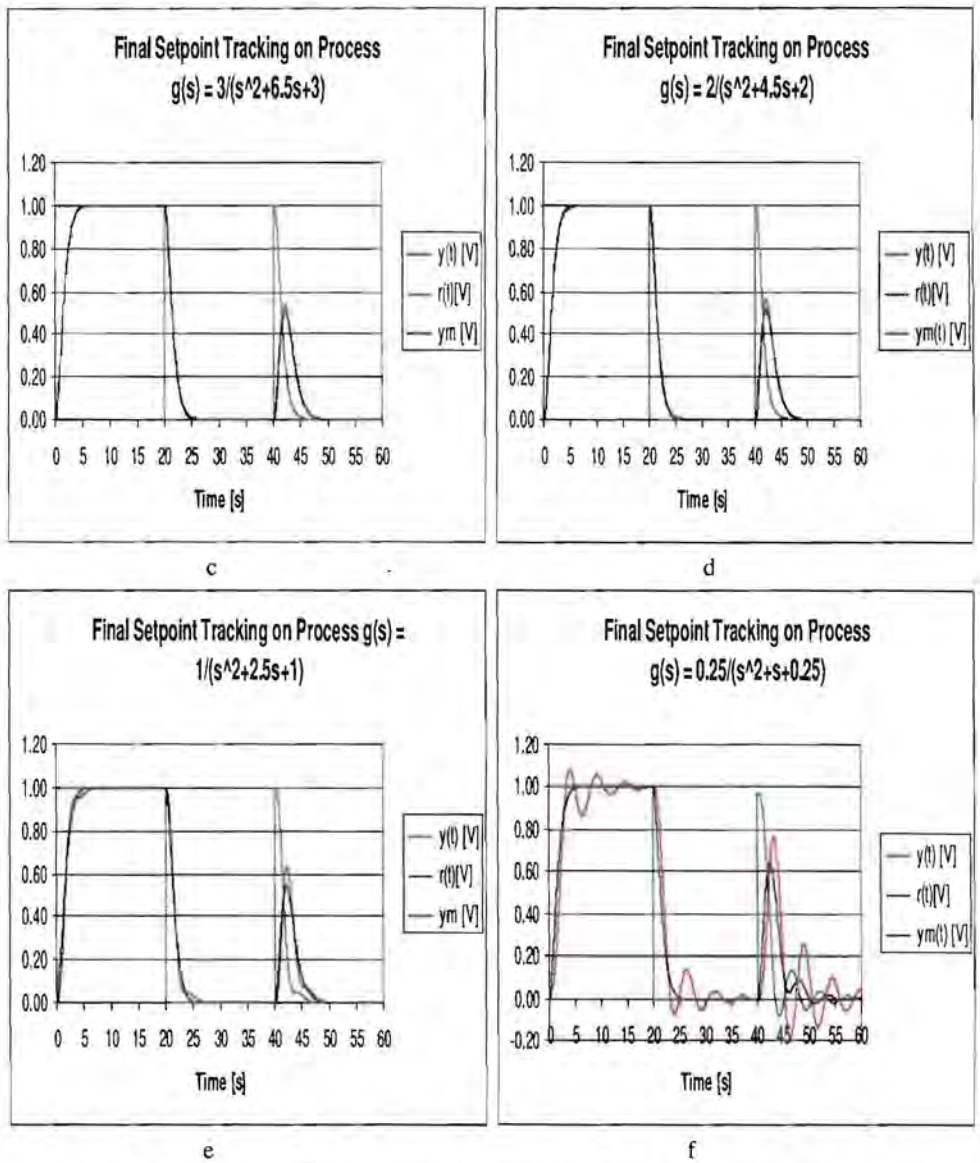


Fig B.4.1- APPIFT with a PI controller applied on damped second order processes

It can be seen that from fig B.4.1 this version of pole placement works better than that in chapter 6 in terms of model following. The worst controller occurs when the poles are on top of each other. The process becomes oscillatory. However final parameter values are reached as opposed to the controller parameters increasing without bounds as occurred with the other version of pole placement.

B.5 EFFECT OF SAMPLING TIME ON APPIFT USING A PREFILTER ON SECOND ORDER PROCESSES

This section is a continuation to section 7.5 where the effect sampling time of prefilter pole placement is investigated. This is shown in fig B.5.1. As before there is a unit step setpoint change at time t = 0 seconds for ***IFT experiment one***. The setpoint is then brought back to zero in preparation for ***IFT experiment two*** that begins at t = 40 seconds. The prefilter is included in the response.

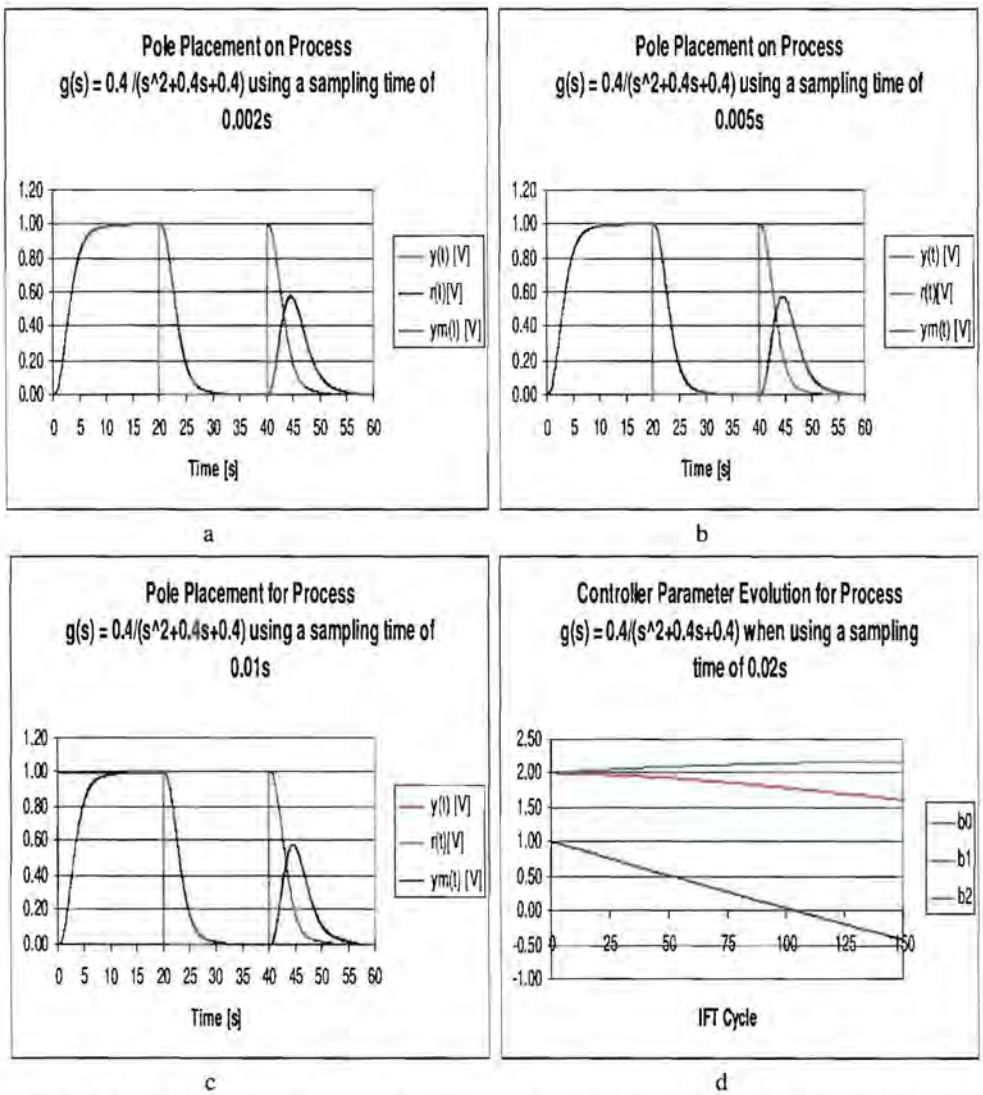


Fig B.5.1- Effect of sampling time changes on APPIFT using p(s) on second order processes

All the closed loop outputs matched the model output for all the sampling times. However when the sampling time was 0.02 seconds, the parameter b_2 became negative shown in fig B.5.1d. This means that there is a zero on the right half s-plane. Since IFT was performed using the normal IFT for setpoint tracking, the cost function gradient calculation involves the use of an inverted controller as a filter as shown in chapter 2. Therefore the output of this filter would be unstable. This will cause the cost function derivative to increase. A way of preventing this is to input the second experiment as an input disturbance in ***IFT experiment two***. The IFT equations for the cost function derivative will be the same however there will be no inverted controller. However the sampling time of 0.02 seconds will not allow exact pole placement because it is too long to tune the high frequency coefficient. It is also twice the length of the non dominant pole added for causality.

B.6 DC MOTOR TRANSFER FUNCTION

Nine step tests up and down were done to find the DC motor transfer function when speeding up and slowing down. The fig B.6.1 shows the step tests.

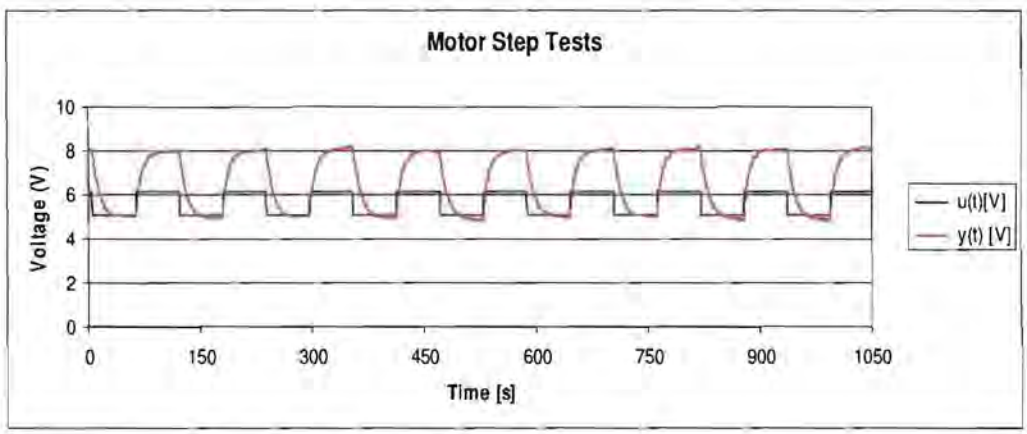


Fig B.6.1- Motor Step Tests

The step tests were done to keep the motor output speed between 5V and 8V. The data from the step tests can be found in the folder "/Data/StepTests" on the CD available with this thesis.