

University of Jordan

Faculty of Graduate Studies

٢ - ١٥
٤ - ٤
١٨

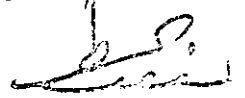
Self-Tuning Control of a Flame Cutting Machine

by

Fuad Fayez Nicola Bajjali

Supervisor

Dr. Yusef Al-Assaf

عميد كلية الدراسات العليا


Submitted in partial fulfillment of the requirements for the degree of Master
of Science in Industrial Engineering / Design and Manufacturing

Faculty of Graduate Studies

University of Jordan

July, 1996

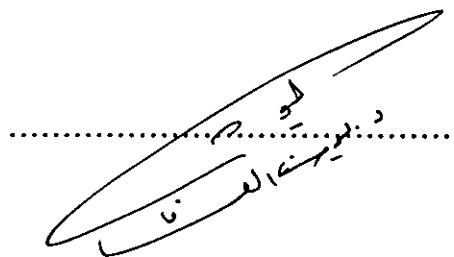
This thesis was defended successfully on 31.7.1996

Committee Members

Signature

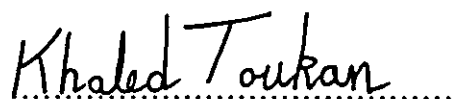
Dr. Yousef Al-Assaf

Chairman of Committee



Dr. Khaled Toukan

Member of Committee



Prof. Mohammad Zaki Khader

Member of Committee



Dedication

To my parents,

My beloved wife,

My brother and sisters,

With All My Love

Acknowledgment

This work would not have seen the light without the continuous support of my supervisor, Dr. Yousef Al-Assaf. It is my duty and honor to extend my deepest appreciation to him, as a supervisor, as well as a friend.

My thanks are also directed to Beta Rotary Compressors Manufacturing Company for their financial and technical support throughout this research.

Table of Contents

Dedication	iii
Acknowledgment	iv
Table of Contents	v
List of Tables	viii
List of Figures	ix
Nomenclature	xii
Appendices	xiv
Abstract	xv
CHAPTER ONE	
Introduction	1
CHAPTER TWO	
Self-Tuning Predictive Control	
2.1 Introduction	8
2.2 Parameter Estimation	10
2.2.1 The Recursive Least Square (RLS) Estimator	13
2.2.2 Design Parameters of the RLS	14
2.2.2.1 Initial Values of the Covariance Matrix	14
2.2.2.2 Initial Parameter Values	15
2.2.2.3 Model Order	16

2.2.2.4 Covariance Management and Parameter Tracking	17
2.3 The Generalized Predictive Controller GPC	24
2.3.1 Review of Self-Tuning Control Procedures	24
2.3.2 Derivation of the GPC Algorithm	26
2.3.3 The Predictive Controller	28
2.3.4 Design Parameters and Testing	30
2.3.4.1 The Effect of N_2	31
2.3.4.2 The Effect of NU	31
2.3.4.3 The Effect of λ	32

CHAPTER THREE

Experiments and Simulation

3.1 The Pneumatic Speed Engine	38
3.2 Simulation Experiment	46
3.3 Proto-type X-Y-Machine	64
3.4 Comparing the Performance of GPC and Artificial Neural Networks	79

CHAPTER FOUR

Flame Cutting Machine Design

4.1 Introduction	81
4.2 Mechanical Structure	82
4.3 The Control System	88

4.3.1 Hardware Components	88
4.3.1.1 The Personal Computer	88
4.3.1.2 The Interface Card	88
4.3.1.3 The Power Amplifier Unit	89
4.3.1.4 The Optical Encoders	89
4.3.1.5 Digital Interlocks	90
4.3.1.6 The Actuators	91
4.3.2 Software Components	92
4.3.2.1 The User Interface	93
4.3.2.2 The Layout Optimization Module	93
4.3.2.3 The Tool Path Generator	94
4.3.2.4 The Control Module	94
4.4 The Flame Torch Assembly and Gas Supply	95
CHAPTER FIVE	
Conclusions	96
REFERENCES	98
APPENDICES	
Appendix A : RLS Program Listing	100
Appendix B : GPC Program Listing	102
Appendix C : Data Sheets for Flame Cutting Machine Parts	107
Abstract in Arabic	113

List of Tables

Table 3.1 List of Set Point Profiles and Their Corresponding Responses	66
Table 4.1 Digital Interlocks of the Flame Cutting Machine	91

List of Figure

Figure 1.1 : Schematic Diagram of a Computer Monitoring System	2
Figure 1.2 Schematic Diagram of Computer Open-loop System.	3
Figure 1.3 Schematic Diagram of Computer Closed-loop System	5
Figure 2.1 Self-tuning Controller Structure.	10
Figure 2.2 The Prediction Error	13
Figure 2.3 The Effect of the Initial Size of the Covariance Matrix, $P(0)=1$	20
Figure 2.4 The Effect of the Initial Size of the Covariance Matrix, $P(0)=10$	20
Figure 2.5 The Effect of the Initial Size of the Covariance Matrix, $P(0)=100$	20
Figure 2.6 The Effect of the Initial Parameter Set on the Performance of the RLS $\theta(0)=[1 \ 1 \ 1]$	21
Figure 2.7 The Effect of the Initial Parameter Set on the Performance of the RLS $\theta(0)=[0 \ -1 \ 2]$	21
Figure 2.8 Over-Parameterization Effect on the Performance of the RLS	22
Figure 2.9 Under-Parameterization Effect on the Performance of the RLS	22
Figure 2.10 The Effect of the Forgetting Factor on the Performance of the RLS, $\beta=1$	23
Figure 2.11 The Effect of the Forgetting Factor on the Performance of the RLS, $\beta=0.9$	23
Figure 2.12 Effect of N_2 on the Performance of the GPC	34
Figure 2.13 Effect of N_U on the Performance of the GPC	35
Figure 2.14 Effect of $\lambda=0$ on the Performance of the GPC	36
Figure 2.15 Effect of $\lambda=0.5$ on the Performance of the GPC	37
Figure 3.1 Schematic of the Pneumatic Speed Engine Setup	42

Figure 3.2 Excitation Signal for the Identification	42
Figure 3.3 Controlling the Speed Engine with $\lambda=75$	43
Figure 3.4 Controlling the Speed Engine with $\lambda=85$	44
Figure 3.5 Controlling the Speed Engine with $\lambda=90$	45
Figure 3.6 Step Response of the PID System	49
Figure 3.7 Step Response of the GPC System	50
Figure 3.8 Profile1 to be followed by the modeled X-Y-Machine	51
Figure 3.9 Performance of the PID System in Tracking Profile1	52
Figure 3.10 Performance of the GPC System with $N2=3$ in Tracking Profile 1	53
Figure 3.11 Performance of the GPC System in Tracking Profile1 with $N2=7$	54
Figure 3.12 Response 1 of the PID system to a Ramp Input	55
Figure 3.13 Response 2 of the PID system to a Ramp Input	55
Figure 3.14 Profile 2 to be followed by the modeled X-Y-Machine	56
Figure 3.15 Profile 3 to be followed by the modeled X-Y-Machine.	56
Figure 3.16 Profile 4 to be followed by the modeled X-Y-Machine	57
Figure 3.17 Performance of the PID System in Tracking Profile 2	58
Figure 3.18 Performance of the GPC System in Tracking Profile 2	59
Figure 3.19 Performance of the PID System in Tracking Profile 3	60
Figure 3.20 Performance of the GPC System in Tracking Profile 3	61
Figure 3.21 Performance of the PID System in Tracking Profile 4	62
Figure 3.22 Performance of the GPC System in Tracking Profile 4	63
Figure 3.23 Path 1 to be followed by the Proto-type X-Y-Machine	68
Figure 3.24 Path 2 to be followed by the Proto-type X-Y-Machine	68
Figure 3.25 Path 3 to be followed by the Proto-type X-Y-Machine	69

Figure 3.26 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=500$	70
Figure 3.27 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=1000$	71
Figure 3.28 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=1500$	72
Figure 3.29 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=400$	73
Figure 3.30 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=500$	74
Figure 3.31 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=700$	75
Figure 3.32 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=400$	76
Figure 3.33 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=500$	77
Figure 3.34 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=750$	78
Figure 3.35 Response of the Speed Engine using Artificial Neural Networks	79
Figure 4.1 General Overview of the Flame Cutting Machine	86
Figure 4.2 The Ball Screw-Nut Assembly	87
Figure 4.3 The Guiding and Supporting Rods	87
Figure 4.4 The 90° Phase Shift between the Encoder Signals	90
Figure 4.5 Connection of the Encoder Signals to the Counter	90

Nomenclature

A,B,C	Polynomials of the CARIMA Model
ANN	Artificial Neural Networks
E_j, F_j	Polynomials in the Diophantune equation
$e(t)$	Error between predicted output and the set point
G	Matrix whose elements are g_i
G_j	Polynomial equals to $E_j B$
\bar{G}	Part of G_j associated with unknown terms
\tilde{G}	Part of G_j associated with known terms
GPC	Generalized Predictive Control
I	Identity matrix
J	Value of a quadratic cost function
K_d	Process delay time in samples
$K(t)$	Correction gain vector in RLS (Kalman Gain)
n	Number of estimated parameters
N1	Minimum prediction horizon
N2	Maximum prediction horizon
NU	Control horizon
RLS	Recursive Least Square Estimator
$P(t)$	Covariance matrix in RLS algorithm
$u(t)$	Control signal input to the process
\tilde{U}_{opt}	Vector of future control increments

$w(t)$	Set point for the process
W	Vector of set points in future
$x(t)$	Vector containing input/output data in RLS
$y(t)$	Measured output of the process
Y	Vector of $y(t)$
$\hat{y}(t)$	Predicted output of the process
z^{-1}	Backward shift operator
β	Constant forgetting factor
$\beta(t)$	Variable forgetting factor
Δ	The difference operator $(1-z^{-1})$
$\varepsilon(t)$	The error between estimated and true parameters
$\hat{\Theta}$	Vector of estimated parameters
λ	Control weighting factor
$\zeta(t)$	Zero mean white noise
σ	Tuning knob for $\beta(t)$
τ	The process time constant

Appendices

Appendix A : RLS Program Listing	100
Appendix B : GPC Program Listing	102
Appendix C : Data Sheets for Flame Cutting Machine Parts	107

Abstract

Self-Tuning Control of a Flame Cutting Machine

By

Fuad Fayez Nicola Bajjali

Supervisor

Dr. Yousef Al-Assaf

The cutting of sheet metals is a widely spread process in the industry. It is usually performed using oxy-acetylene flame cutting. In this flame cutting process the control of both the speed and position is essential in order to perform a defined clean cut. Specifically on sharp edges and curved lines, the constant speed control applied in most available flame cutting machines might not give the optimal relationship between these two process variables.

In this research a self-tuning controller has been introduced that can be implemented on a computerized flame cutting machine. This controller consists of a process parameter identification algorithm and a control procedure, the Generalized Predictive Control. This controller, which will be able to change the speed in response to predicted changes in the shape to be cut, can eliminate the above mentioned problems and improve the quality of the products and the productivity of the process. This algorithm has been studied and analyzed in this work. Experimental and simulation results proved its advantages over conventional controllers, such as the PID controller, in controlling the flame cutting process. Furthermore, a design for the computerized flame cutting machine has been introduced and its components and subsystems have been described.

Chapter One

Introduction

The flame cutting process is an important process for all the industries that use sheet metal as their raw material. The cutting of sheet metals by flame is done nowadays, either manually or on specialized machines. The two main characteristics that the final product of this cutting process should possess are accuracy of cut and a good surface finish of the cut line. To assure these two specifications, the position as well as the speed of the cutting torch should be well controlled. In manual cutting, all these parameters depend largely on the skill and good-workmanship of the operator. The current controllers of flame cutting machines, on the other hand, are not sophisticated enough to assure good quality cuts under different operating conditions. This will be clarified more later on in this section.

The impact of the rapidly developing electronics and computer technology is accelerating the trend towards automation and the use of computer control in the manufacturing industry [1]. The application of a digital computer for control can take several forms. The simplest being an efficient data collector and calculator, Its role is restricted here to be just a process monitoring system (see Figure 1.1).

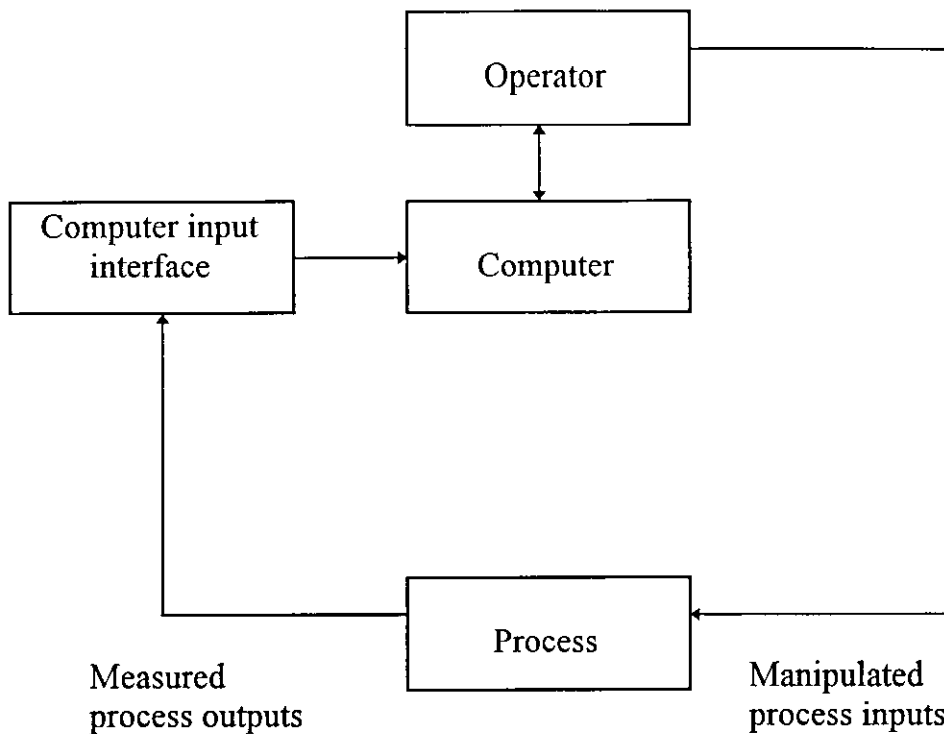


Figure 1.1 Schematic Diagram of a Computer Monitoring System

A second application is in computer open-loop control systems. In this scheme the computer is used as an on-line processor of programs and data to generate specific commands for the manipulation of machine and process actuators (see Figure 1.2). Open-loop principles are commonly employed in the use of computers to manipulate electrical or electro-hydraulic stepping motors for machine tool control.

The most sophisticated application of computers is in feedback-control systems or closed-loop systems, as shown in Figure 1.3, where the computer collects data from process sensors, compares it with given set-points and, through a control algorithm, generates commands that cause specific changes in the process output.

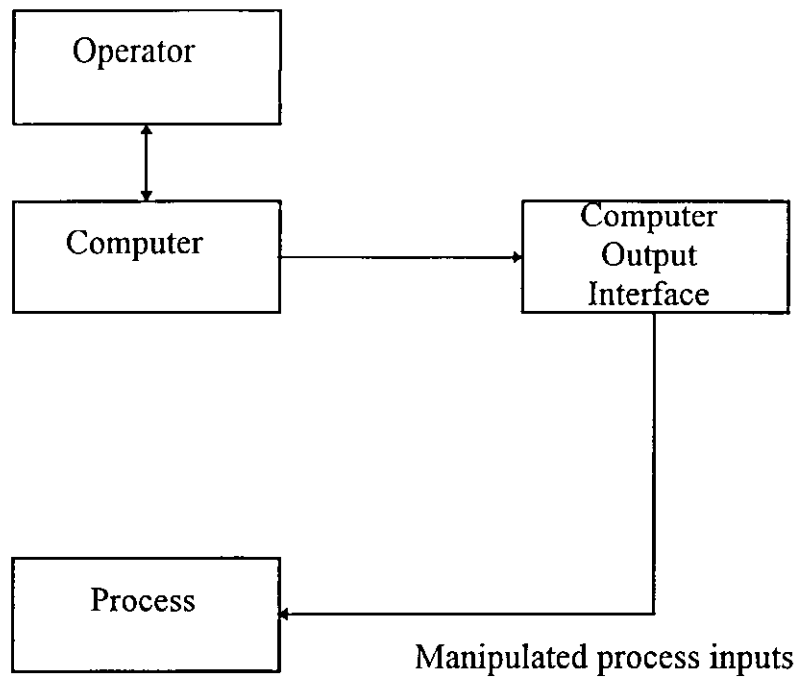


Figure 1.2 Schematic Diagram of Computer Open-loop System.

This form of control, the closed-loop form, has many advantages over the open-loop structure. In a closed-loop control system, i.e. a feedback system, the actuating error signal, which is the difference between the input signal and the feedback signal (which may be the output signal itself or its derivatives), is fed to the controller so as to reduce the error and bring the output of the system to a desired value. On the other hand, in open-loop control systems the output has no effect on the control action. Hence, the output of an open-loop control system is neither measured nor fed back for comparison with the input. Thus, to each reference input there corresponds a fixed operating condition; as a result, the accuracy of the system depends on calibration. In the presence of

disturbance, an open-loop control system will not perform the desired task. Open-loop control can be used in practice only if the relationship between input and output is known to a certain tolerated degree of accuracy (an adequate model exists) and if there are no disturbances. The use of feedback makes the system response relatively insensitive to disturbances. If using a closed-loop control system the issue of stability should be addressed, since it is a major problem of the closed-loop control system, which may tend to correct errors in a way that will lead to oscillations of constant or changing amplitude. It is also known that closed-loop control systems have a higher cost than open-loop control systems. Therefore, it is advised to use open-loop control systems where such a system is applicable [2].

Digital control techniques, for both open-loop and closed-loop forms, have been utilized extensively in machine tools, chemical process plants and many other processes. Hence, a mathematical basis for analyzing systems as viewed by a computer has to be developed in order to create mathematical models of real elements and systems that are compatible with the discrete nature of the computer. The next step is to choose or synthesize the suitable control algorithm to be applied on the process.

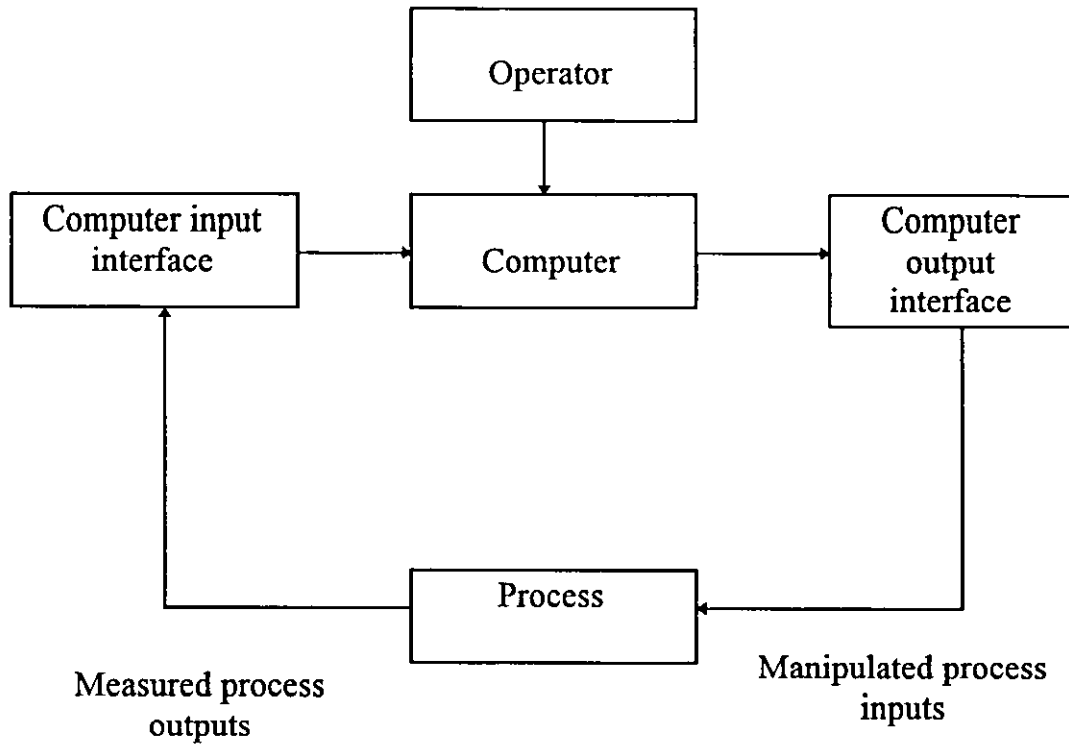


Figure 1.3 Schematic Diagram of Computer Closed-loop System

In the flame cutting process the control of both the speed and position is essential in order to perform a defined clean cut. Currently, most of the flame cutting machines available in the market are copying machines, i.e. a photo sensor will follow a pre-drawn shape, driving the head of the flame with it on a one-to-one scale on a preset constant speed chosen by the operator.

Especially on sharp edges and curved structures, the constant speed control applied will not give the optimal relationship between the two control signals mentioned above. The operator, merely by experience, sets the speed in accordance to the thickness of the sheet or plate to be cut. This speed can either be relatively low, to maintain an accurate position control, which can cause

excessive heat on the line of cut, and as a result, a low quality surface finish, or relatively high speeds, which in turn has one of the following disadvantages (or both): An overshoot at turning points, such as angles and curves, can occur or in worst cases, the speed will be so high that there will be no penetration of the flame, i.e. no cut.

A computerized flame cutting machine, controlled by a modern self-tuning predictive control algorithm, which will be able to change the speed in reflection to changes in the shape to be cut, can eliminate the above mentioned problems and improve the quality of the products and the productivity of the machine. For example, the controller can set relatively high speeds (within the constraints of the cutting process) when cutting straight lines then reducing this speed at corners and turning points. This would give "optimal" combination between total cut time and cut quality. The machine should also recognize the end of cut and be able to move independently between several objects to be cut. In self-tuning predictive control, the algorithm will "look ahead" to preview any turning points, and starts to adjust the speed to accommodate these points without overshooting, i.e. maintaining an accurate position control while keeping a clean surface finish by this adjustment.

Scope of the Thesis

The objective of this work is to build a flame cutting machine and to investigate the possibilities of applying modern control techniques to control its motion.

Chapter two describes an identification algorithm as well as a control algorithm that will together form a self-tuning controller to be used on the machine.

Chapter three investigates the suitability of these algorithms to the flame cutting application through simulation and experimental results.

Chapter four describes the flame cutting machine that should be built during this work.

And finally chapter five includes some conclusions, recommendations for further research on these topics as well as possibilities for future work.

Chapter Two

Self-Tuning Predictive Control

2.1. Introduction

In the real world, most processes and systems are non-linear and time variant. The dynamic characteristics of most control systems are not constant for a number of reasons, such as deterioration of components as time elapses or changes in the environment.

Nevertheless, the mostly applied conventional design methods for controllers produce constant coefficient algorithms based on a linear, time-invariant assumption of the system or process. In self-tuning control the main philosophical step from this is to introduce control algorithms with coefficients which can vary with time, i.e. to construct an algorithm that will automatically change its parameters to meet a particular requirement or situation. Figure 2.1 gives an overview about the general structure of a self-tuning controller. Having an adequate model is a cornerstone for successful control algorithm implementation. In self-tuning control, this model can be obtained using an identification mechanism based on the input/output data which converges to the unknown parameters of the process [3, 4, 5].

The principal tasks involved in control system engineering consist of the following :

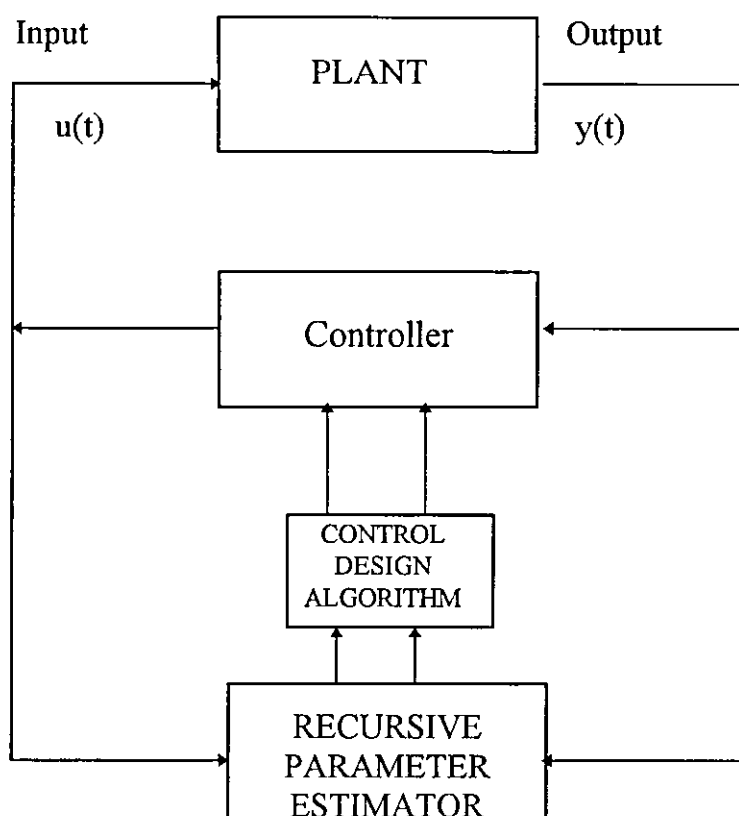


Figure 2.1 Self-tuning Controller Structure [5].

2.2. Parameter Estimation

There are several ways to construct a mathematical representation (model) of the system. One technique is to apply the laws of physics and chemistry to the system and hence construct a model based upon first principles. However, in most cases there are two main obstacles to this approach : the first one is that in many systems and processes, it is very difficult to determine the laws that govern the behavior of the system and put them in exact mathematical formulation; the second obstacle is that resulting differential equations are complex and demand a

large computational effort. An alternative to that approach is experimental modeling, based on the response of the system to different test input signals and then fitting the system to an assumed model. For example, a system can be assumed to be a second order system of the general form :

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

then by recording the unit step response of that system and calculating the maximum overshoot and the damped natural frequency, the two unknown parameters ζ and ω_n can be determined. A yet another approach to model a system is experimental modeling based on regression, i.e., identification or parameter estimation. This method can be described as constructing a model from observed and logged behavior of the system. This can be done off-line or on-line. The first is used for a complete set of data using a fitting criterion. The second fits a dynamic model of an assumed structure using running input/output data and a recursive criterion [3].

On-line identification is a basic step in self-tuning control. Its main function is to provide parameter estimates of the plant model for the controller at each time instant. A locally linearized model is identified, where changes in the dynamics are transformed into parameter changes.

The estimation is usually accomplished by assuming discrete time form for the system model and then using a recursive estimation algorithm to obtain estimates of the parameters of the model.

One well known discrete-time representation of systems is the CARIMA model (Controlled-Auto-Regressive-Integrated-Moving-Average) [4, 6]:

$$A(z^{-1})y(t) = z^{-k_d}B(z^{-1})u(t-1) + \frac{C(z^{-1})\zeta(t)}{\Delta} \quad (2.1)$$

where :

$y(t)$ is the controlled process output

$u(t)$ is the manipulated input

$\zeta(t)$ is uncorrelated random sequence noise signal

z^{-1} is the backward shift operator

Δ is the difference operator $(1-z^{-1})$

k_d is the dead-time of the process

A , B and C are polynomials in z^{-1} of the orders n_a , n_b and n_c

respectively, such that :

$$A(z^{-1}) = 1 + a_1z^{-1} + \dots + a_{n_a}z^{-n_a}$$

$$B(z^{-1}) = b_0 + b_1z^{-1} + \dots + b_{n_b}z^{-n_b}$$

$$C(z^{-1}) = 1 + c_1z^{-1} + \dots + c_{n_c}z^{-n_c}$$

The parameters of these polynomials are those to be estimated. The model can be written in a vector form as:

$$\Delta y(t) = x^T \hat{\Theta}$$

where :

$$\hat{\Theta} = [a_1, a_2, \dots, a_{n_a}, b_0, b_1, \dots, b_{n_b}, c_0, c_1, \dots, c_{n_c}]$$

and

$$\mathbf{x} = \begin{bmatrix} -\Delta y(t-1), -\Delta y(t-2), \dots, -\Delta y(t-na), \\ \Delta u(t-1), \Delta u(t-2), \dots, \Delta u(t-nb), e(t-1), e(t-2), \dots, e(t-nc) \end{bmatrix}$$

2.2.1 The Recursive Least Square (RLS) Estimator

The basic estimator used in many practically successful self-tuning identifiers is the Recursive Least-Square (RLS) algorithm. This iterative procedure allows the estimated model of the system to be updated at each sample interval as new data become available. This algorithm can be summarized in the following steps [3]:

1. At time step (t) form the new regression (data) vector $\mathbf{x}(t)$ using the new data.
2. Calculate the new prediction error

$$\varepsilon(t) = y(t) - \mathbf{x}^T(t) * \hat{\Theta}(t-1) \quad (2.2)$$

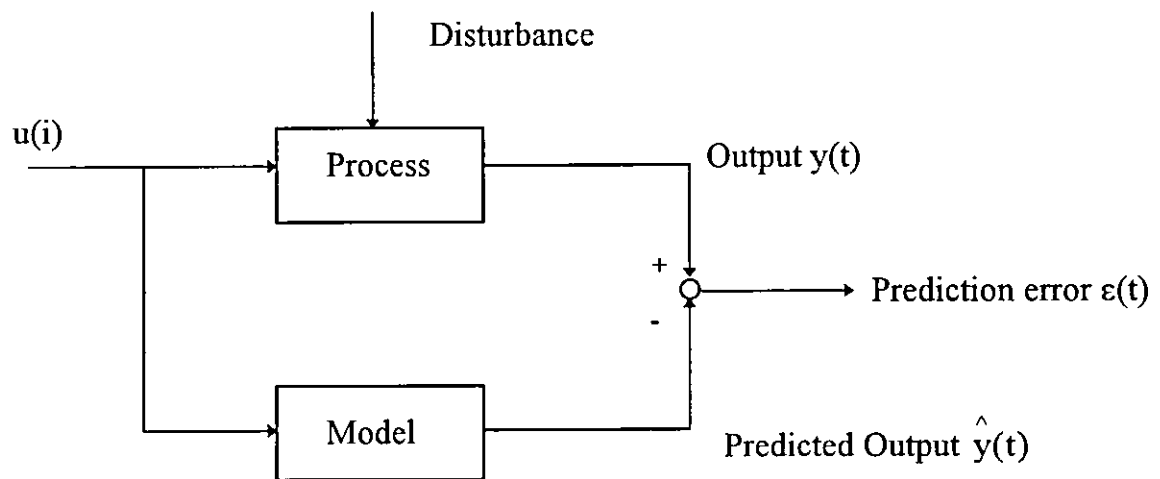


Figure 2.2 The prediction error [7]

3. Calculate the gain adjustment vector or the Kalman gain :

$$\mathbf{K}(t) = \frac{\mathbf{P}(t-1) * \mathbf{x}^T(t)}{1 + \mathbf{x}(t) * \mathbf{P}(t-1) * \mathbf{x}^T(t)} \quad (2.3)$$

4. Update the parameter vector

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) - K(t) * \varepsilon(t). \quad (2.4)$$

5. Update the covariance matrix

$$P(t) = P(t-1) - K(t) * [P(t-1) * x(t)]^T \quad (2.5)$$

6. Shift data and wait for next sample then go back to step 1.

A complete derivation of this algorithm, as well as a program listing in MATLAB, can be found in Appendix A.

2.2.2 Design Parameters of the RLS

In practical situations, there are some problems to be considered prior to operating the RLS algorithm. These problems come mainly from considerations related to initializing the estimator. The issues which arise here are the choices of the initial values for the data vector (regression vector) $x(t)$, the parameter vector $\theta(t)$ and the covariance matrix P . The following sections describe and illustrate readily the effects of all these parameters through a simulation example.

2.2.2.1 Initial Values of the Covariance Matrix

The size of the initial covariance matrix reflects the degree of uncertainty of the initial parameter values. A large P would reflect that no prior information of these unknown parameters is available, therefore, little or no confidence in the initial set of parameters is exerted, where a small P would be used if the initial values of the parameters $\hat{\Theta}(0)$ are known to be close to the true values [3].

Considering a model of the following form:

$$y(t) = -a_1y(t-1) + b_0u(t-1) + b_1u(t-2) \quad (2.7)$$

where $a_1 = -0.5$; $b_0 = 1.6$; $b_1 = 1$ are the actual parameters of the system. The input signal $u(t)$ was a zero mean white noise and $y(t)$ is the output. A program was written in MATLAB to test the effect of different parameters on the behavior of the RLS. The above statements about the initial value of the P matrix can be illustrated in Figures 2.3, 2.4 and 2.5. With increasing value of $P(0)$, initial value of P was 1, 10 and 100 respectively while keeping the other parameters constant, the convergence behavior of the RLS improves noticeably. In Figure 2.3 the RLS was not able to converge to the exact parameters at the end of the run, while in Figure 2.4 and in Figure 2.5 it did converge with faster convergence associated with the higher value of $P(0)$.

2.2.2.2 Initial Parameter Values

Prior knowledge of the system under analysis enables to obtain good initial values for the coefficients of the polynomials A, B and C. However, in more complicated systems, where the exact system order and accurate time delay are not exactly known, the choice of the initial parameter values can drive the system to local minima, that can be far from the global minimum of the system. This issue is also related to the initial size of the covariance matrix P, which governs the level of confidence in these initial parameters, as shown earlier [3]. Using the same simulation model as above with $P(0)=100$, the system was started with two different sets of initial parameter values. As can be seen in Figure 2.6, where $\theta(0)$

was $[1 \ 1 \ 1]$, and Figure 2.7, $\theta(0)=[0 \ -1 \ 2]$, that the initial values of the parameter vector are not crucial to the estimation process as mentioned above, since exact parametrization is available. In both cases the RLS converged to the right parameters, but some differences in the transient region are visible.

2.2.2.3 Model Order

One of the decisions that largely affect the performance of the RLS algorithm is the choice of the model order, i.e. the selection of the integers n_a , n_b and n_c in equation 2.1. Physical considerations of the system can prove helpful in determining these numbers. If the choice of the parameters is inaccurate, problems related to the terms over- and under-parametrization can arise. If the system was started with more parameters to be estimated than actually needed (over-parameterization), i.e. by adding more elements to the data vector $x(t)$, the system can still converge to the right set of parameters, with the extra added parameter of the new element converging to zero. In the simulation example, a new element ($y(t-2)$) was added to the data vector. Figure 2.8 illustrates that the algorithm converged to the actual parameters and the additional parameter converged to zero. In the opposite case of under-parameterization, i.e. removing elements from the data vector and thus reducing the order of the model, the algorithm will not converge to the actual parameters. A reason for this behavior is that the remaining parameters should also include the effects of the removed ones. Figure 2.9 clarifies this effect. The element $u(t-2)$ in the demonstration example was removed from the data vector. It can be easily seen here that the

convergence behavior was distorted from the beginning especially for the coefficient of $y(t-1)$.

2.2.2.4 Covariance Management and Parameter Tracking

The RLS algorithm is not aimed at only determining a constant parameter vector. In many cases it is required of the RLS to detect and track parameter changes which reflect changes in the dynamics of the process without prior knowledge of when or what changes will occur. These changes may be due to nonlinear phenomena which influence local linear models after an operation condition changes or to variations which occur over time due to external factors. In the current formulation of the RLS as described in equations 2.1 to 2.5, the covariance matrix will decrease to a minimum level after convergence to the first set of parameters, hence, the RLS will not have enough “energy” to track the changes. The basic mechanism, therefore, for parameter tracking is to control the size of the matrix $P(t)$. The most common way of getting the RLS to adapt to parameter changes is the Forgetting Factor technique [3]. This factor β is a number between 0 and 1 which is used to progressively reduce the emphasis placed on past information, i.e. reduce the importance of old data. This factor will change the algorithm slightly, but will give more weight for recent data than past data. The equation for the Kalman gain and the covariance matrix (2.3 and 2.5) will be changed as follows :

$$K(t) = \frac{P(t-1) * x^T(t)}{\beta + x(t) * P(t-1) * x^T(t)} \quad (2.3')$$

and

$$P(t) = (P(t-1) - K(t) * [P(t-1) * x(t)]^T) / \beta \quad (2.5')$$

Practically β will tend to increase the value of the matrix $P(t)$ by dividing it over a fraction, which will improve the estimator's efficiency. To demonstrate the effect of the forgetting factor β , a change in the dynamics of the above mentioned example was introduced at sample 150. The new parameters of the system in equation 2.7 are set to

$$a_1 = -1; b_0 = 0.75; b_1 = 1.5.$$

In Figure 2.10 the forgetting factor was set to 1, but in Figure 2.11 it was set to 0.9. In both cases $P(0)=100$. It is noticeable how the estimator was able to converge very quickly to the new parameters in the case of $\beta=0.9$, where it needed much more time to converge in the other case. Nevertheless, a problem is associated with a constant forgetting factor which is termed "Estimator Wind Up" [3, 6]. If the plant is accurately regulating around a steady state and the data vector is not rich, i.e. $x(t)$ brings little or no new information into the estimator, the equation for the covariance matrix is reduced to:

$$P(t) = P(t-1) / \beta$$

which means that the elements of $P(t)$ will become very large with increasing time, therefore the estimator will be very sensitive to system changes, leading to a rapid movement of the estimated parameters when new rich data becomes available. This may lead to failure or instability of the self-tuning system. It was necessary to introduce another modification to the RLS algorithm by making the

forgetting factor variable [3]. An equation that governs the forgetting factor could be in the form:

$$\beta(t) = 1 - \frac{\left[1 - K(t-1) * x(t)^T\right] * \varepsilon^2(t)}{\sigma} \quad (2.6)$$

where σ is a design parameter chosen according to operating conditions of the plant. This scheme must be used carefully in the presence of noise, since $\varepsilon(t)$ will be large and will not be distinguished from the true error. This will result in decreasing the forgetting factor and therefore increasing the $P(t)$ resulting in large parameter changes at every time instant. The problem can be handled through proper choice of σ , where a higher value of σ will reduce the sensitivity of $P(t)$ to changes in the input. It is also advisable to set low and high limits for the forgetting factor (Jacketing) and make all changes between these limits.

The RLS method includes a lot of design parameters. Choice of these parameters depends on the nature of the controlled process. When applying the Generalized Predictive Control (GPC) algorithm, proper choice of the RLS parameters is required to provide the GPC with a good model, otherwise the controller may fail in its function.

The following legend applies for Figures 2.3 to 2.11

-----	a_1
-----	\hat{a}_1
-----	b_0
-----	\hat{b}_0
-----	b_1
-----	\hat{b}_1

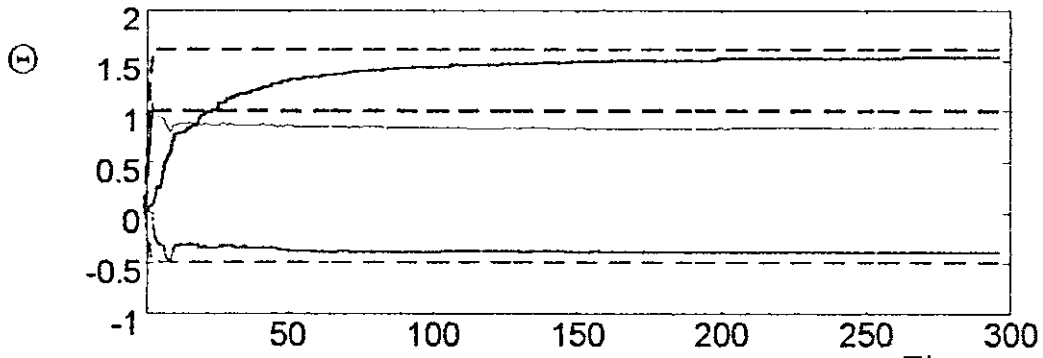


Figure 2.3 The Effect of the Initial Size of the Covariance Matrix

$$P(0)=1$$

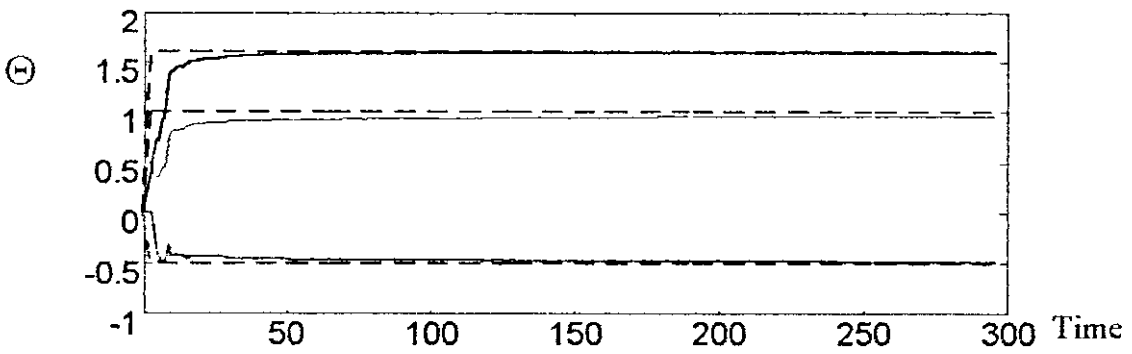


Figure 2.4 The Effect of the Initial Size of the Covariance Matrix

$$P(0)=10$$

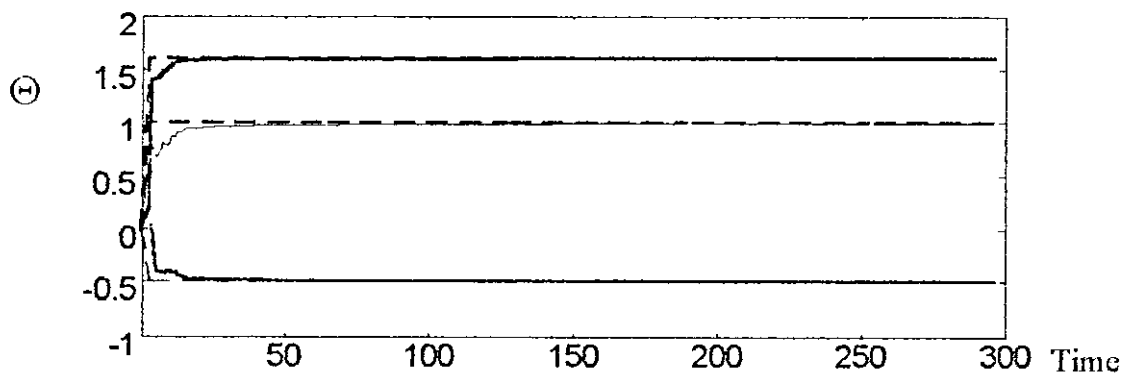


Figure 2.5 The Effect of the Initial Size of the Covariance Matrix

$$P(0)=100$$

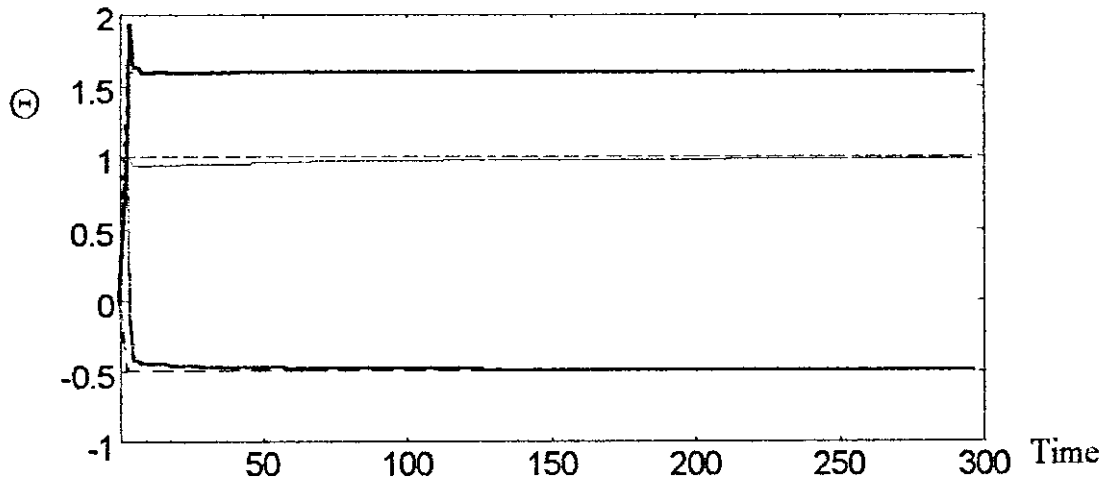


Figure 2.6 The Effect of the Initial Parameter Set on the Performance of the RLS

$$\theta(0)=[1 \ 1 \ 1]$$

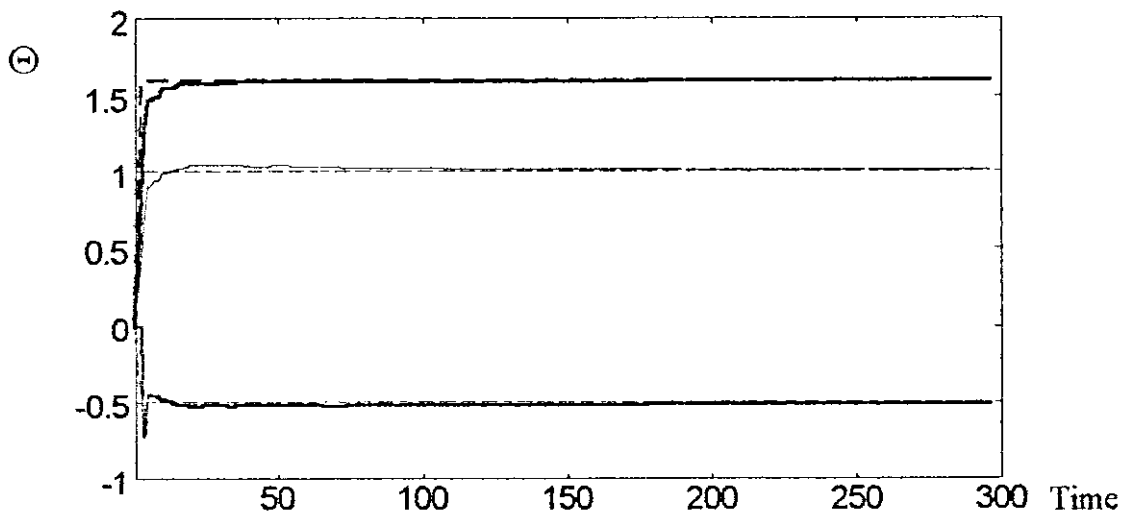


Figure 2.7 The Effect of the Initial Parameter Set on the Performance of the RLS

$$\theta(0)=[0 \ -1 \ 2]$$

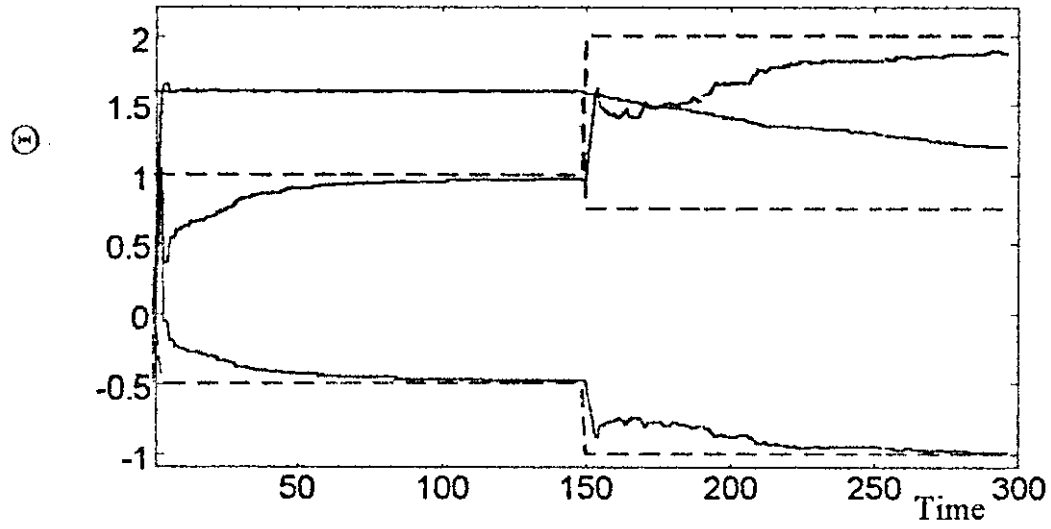


Figure 2.10 The Effect of the Forgetting Factor on the Performance of the RLS

$$\beta=1$$

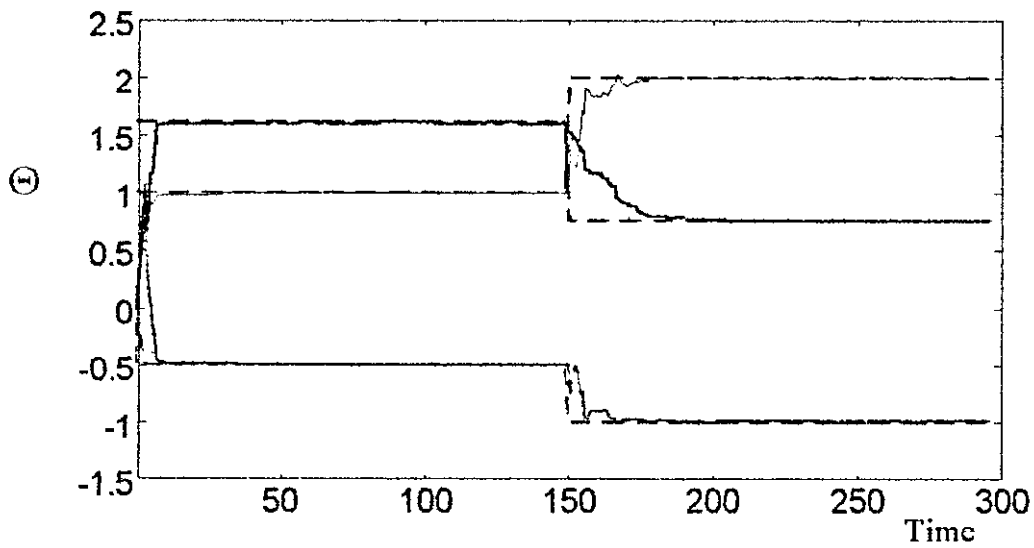


Figure 2.11 The Effect of the Forgetting Factor on the Performance of the RLS

$$\beta=0.9$$

2.3. The Generalized Predictive Control (GPC)

2.3.1 Review of Self-Tuning Control Procedures

A self-tuning controller consists of a recursive parameter estimator (plant identifier) coupled with an analytical control design procedure, such that the currently estimated parameters are used to provide feedback controller coefficients [4]. The plant identifier was discussed earlier in this chapter. In the next sections the analytical control procedure will be addressed and discussed.

The objective of a self-tuner could either be minimizing a quadratic cost function of the process inputs and outputs or placing the closed-loop poles at a pre-chosen location [5]. One of the first introduced self-tuning controllers was the Minimum Variance (MV) controller [5, 6]. The objective of this controller is to minimize the error between the one step ahead predicted output and the desired set-point profile at that point. The MV suffers from some problems such as instability with non-minimum-phase processes due to cancellation of unstable process zeros and other problems. As an outcome of several researches to solve the problems of MV, the Generalized Minimum Variance (GMV) controller was introduced. The cost function in this algorithm introduces penalty on the control action as well as minimization of the error between measured output and desired set point. It incorporates a variety of design polynomials to ensure stability and correct tracking of the set point. The use of control weighting reduces the control effort and allows for stabilizing certain non-minimum-phase processes. MV and GMV (to a certain extent) are very sensitive to the choice of the dead-time in the

process. Instability is expected if the dead-time is incorrectly chosen in these algorithms.

Parallel development of self-tuning controllers lead to the Pole-Placement (PP) controller, that was able to handle varying dead-time and non-minimum-phase systems. Nevertheless, this controller fails if there are model-process mismatches, i.e. it depends on a correct choice of the model order.

A further development in self-tuning control is the Generalized Predictive Controller (GPC) [8].

The GPC algorithm predicts a series of future outputs and decides a series of controls at each sample to set the actual output at these samples to the desired set points. A Controlled-Auto-Regressive-Integrated-Moving-Average (CARIMA) model is used in the controller design which introduces an integral effect, so the offsets in the response are eliminated. In addition this algorithm contains a number of design parameters 'knobs', upon their choice the performance of the controller can be altered to meet the application control specification. In contrast to the mentioned algorithms, the GPC can be robust even with varying process dead-time and model order. In addition it is capable of controlling non-minimum-phase processes.

As the GPC can consider future set-point changes it is suitable for robotics applications, flame cutting machines, as well as any application where a preprogrammed trajectory is available. Very close tracking of the desired trajectory is achievable using this algorithm.

2.3.2 Derivation of the GPC Algorithm

As mentioned above, the GPC algorithm assumes that the plant has a CARIMA model as in equation 2.1:

$$A(z^{-1})y(t) = z^{-k_d}B(z^{-1})u(t-1) + \frac{C(z^{-1})\zeta(t)}{\Delta}$$

where the variables of the equation are described in section 2.2. The role of Δ is to ensure integral action in the controller.

In a plant with a dead-time more than k_d samples the leading elements of the polynomial $B(z^{-1})$ are zero. For simplicity the polynomial $C(z^{-1})$ is chosen to be 1 (alternatively $C(z^{-1})$ is truncated and absorbed into the A and B polynomials).

This simplifies the model to the following form :

$$A(z^{-1})y(t) = z^{-k_d}B(z^{-1})u(t-1) + \frac{\zeta(t)}{\Delta} \quad (2.1')$$

To derive a j -step predictor based on the CARIMA model the following identity is considered :

$$1 = E_j(z^{-1})A(z^{-1})\Delta + z^{-j}F_j(z^{-1}) \quad (2.9)$$

Where E_j and F_j are polynomials of order $j-1$ and n_a respectively and are uniquely defined given $A(z^{-1})$ and the prediction interval j . Combining equations (2.1') and (2.8) gives the prediction equation :

$$y(t+j|_t) = G_j\Delta u(t+j-1) + F_j y(t) + E_j \zeta(t+j) \quad (2.10)$$

Where $y(t+j|_t)$ is the predicted output after j time samples in the future based on data up to time t and $G_j(z^{-1}) = E_j(z^{-1})B(z^{-1})$ is a polynomial of order n_b+j-1

As $E_j(z^{-1})$ is of degree $j-1$, the noise components are all in the future and unknown. The optimum prediction of y is obtained assuming the noise term is zero :

$$\hat{y}(t+j|t) = G_j \Delta u(t+j-1) + F_j y(t) \quad (2.10)$$

where $G_j(z^{-1}) = E_j B$.

In GPC algorithm, the prediction is extended to become a multi step prediction, where a set of future outputs is predicted in the range 1 to N , based on recursion of equation (2.9) (Diophantine Identity) over that range. This results in a set of equations similar to equation (2.11). The first term of the right-hand-side of these equations can be divided into two parts at each prediction step. One will be of known terms and the other will be future terms defined by the following equation:

$$G(z^{-1}) = \overline{G}_j(z^{-1}) + z^{-j} \tilde{G}(z^{-1}) \quad (2.11)$$

where $\overline{G}_j(z^{-1})$ is a polynomial corresponding to future terms and $\tilde{G}(z^{-1})$ is another polynomial that corresponds to known terms.

Let :

$$f(t+j) = \tilde{G} \Delta u(t-1) + F_j Y(t) \quad (2.12)$$

which represents the known terms [8].

The predicted outputs for all future steps in the prediction range can be combined in vector form as follows :

$$Y = [y(t+1), y(t+2), \dots, y(t+N)]^T \quad (2.13)$$

Similarly, the other terms are written as:

$$\tilde{U} = [\Delta u(t), \Delta u(t+1), \dots, \Delta u(t+N-1)]^T \quad (2.14)$$

$$f = [f(t+1), f(t+2), \dots, f(t+N)]^T \quad (2.15)$$

and considering a set point vector W , over the prediction range :

$$W = [w(t+1), w(t+2), \dots, w(t+N)]^T \quad (2.16)$$

so equation (2.11) can be given in the form :

$$Y = G\tilde{U} + f \quad (2.17)$$

where G is a matrix of dimension $N \times N$, it is a lower triangle matrix of the form :

$$G = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ g_2 & g_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_N & g_{N-1} & \dots & g_1 \end{bmatrix}$$

2.3.3 The Predictive Controller

The objective of the predictive controller is to minimize the difference between future outputs $y(t+j)$ and future set-points $w(t+j)$. To do that the GPC considered a quadratic cost function of the form :

$$J(N1, N2, \lambda) = E \left[\sum_{j=N1}^{N2} e^2(t+j) + \lambda \sum_{j=1}^{N2} \Delta u^2(t+j-1) \right] \quad (2.18)$$

where E means expectation over the prediction range, which depends on data available up to time t , e is the difference between set-point and predicted output defined by :

$$e(t+j) = w(t+j) - \hat{y}(t+j)$$

$N1$ is the minimum prediction horizon, $N2$ is the maximum prediction horizon, λ is the control weighting factor and $w(t+j)$ is a future set point which is known a priori or assumed to be equal $w(t)$.

The prediction is performed in the range $N1$ to $N2$ and a set of projected control increments is decided. The control signal is weighted by a positive scalar λ .

Combining equations (2.18) and (2.19) yields the following equation :

$$J = (W - G\tilde{U} - f)^T (w - G\tilde{U} - f) + \lambda \tilde{U}^T \tilde{U} \quad (2.19)$$

minimizing this equation with respect to the control increments vector gives the projected control increment vector \tilde{U} , where the first element in this vector is applied to the process and is given by:

$$\tilde{U}_{opt} = (G^T G + \lambda I)^{-1} G^T (W - f) \quad (2.20)$$

where I is a unit matrix, $G^T G$ is a matrix of order $N2 \times N2$, so at every sample a matrix of order $N2 \times N2$ will be inverted. This needs a large computational capacity especially for large values of $N2$. The use of what is called the Control Horizon NU will reduce this problem. The control horizon means, that after the interval NU , all control increments ΔU will be assumed to be zero. Which in turn results as if an infinite control weighting to future control increments is applied. In the cost function, this is achieved by replacing $N2$ by NU in the second summation on the right hand side. So the cost function becomes :

$$J(N1, N2, \lambda) = E \left[\sum_{j=N1}^{N2} e^2(t+j) + \lambda \sum_{j=1}^{NU} \Delta u^2(t+j-1) \right] \quad (2.18')$$

This assumption reduces the order of the G matrix to $(N2-N1+1 \times NU)$. If NU is chosen to be 1, the matrix $G^T G$ will be a scalar term. The inversion of a matrix will be reduced to calculating the reciprocal of a number, saving much computation time and effort.

A self-tuning GPC algorithm performs the following steps at each sample :

1. Identify process parameters using the RLS algorithm.
2. Calculating the predicted outputs in the prediction range.
3. Minimizing the cost function with the specified parameters with respect to future control action and hence, obtaining the optimal future control vector.
4. Applying the first element of the optimal control vector on the process and shifting the data vector.
5. Wait for next time sample, shift data and then repeat at step 1.

The following sections will discuss the effect of the ‘tuning knobs’ of the GPC algorithm, namely N2, NU and λ ., on the control system.

2.3.4 Design Parameters and Testing

To test the GPC algorithm a plant given by

$$\frac{Y}{U} = \frac{0.1 + 0.2z^{-1}}{1 - 0.9z^{-1}}$$

was considered. The set point profile was chosen to be a square wave of amplitude 10 to 30 with a period of 50 sampling times. In order to demonstrate the effect of the GPC parameters on the performance and not to confuse it with the effect of the identification technique, the estimator was not employed in the experiments.

2.3.4.1 The Effect of N_2

N_2 is the maximum prediction horizon. It is chosen to include all the transient response that is affected by the current control signal. At least $N_2 \geq 2n_b - 1$ [5], and it will be better if chosen up to the rise time of the plant [8]. However, a value of 10 is usually sufficient to lead to stable control [5]. On the above mentioned plant, at samples [1,75,175,275,375] N_2 was set to [1,2,5,10,20] respectively. The other parameters were set to [0.1,1,1] for $[\lambda, N_1, N_2]$. Figure 2.12 shows that with increasing N_2 a more slower response is obtained. When $N_2=1$, the role of λ was dominant which causes the oscillations in the output. Increasing the prediction horizon gives the controller more time to correct the present error, and this is the reason for the more sluggish response. Note here that the minimum output horizon N_1 has also to be chosen. Its value should be at least equal to the dead-time of the process. If this dead-time is unknown, N_1 can be set to 1 with no loss in generality and the degree of $B(z^{-1})$ is increased to encompass all possible values of the dead-time. Nevertheless, this will increase the computation time required.

2.3.4.2 The Effect of NU

A value of $NU=1$ gives generally acceptable control. Increasing NU makes the control and the corresponding output response more active, due to decreasing the constraints on the control signal, until a stage is reached where any further increase in NU makes little difference[8]. It can be demonstrated in Figure 2.13 that increasing NU causes a more active control, until the point where $NU \geq 3$, after which little or no differences can be observed. For this experiment NU was given the values [1,2,3,5,10] at the samples [1,75,175,275,375] respectively with $[N1, N2, \lambda]$ set to [1,10,0.5]. In the case of $NU=1$, the controller would “assume” at each stage that only one chance is available to make the output reach the set-point, and consequently it tries to apply the control signal that, if left constant, would lead the output to reach the set-point with its natural bandwidth (DC gain control). However, if NU increases, the controller tries to apply a control signal which will speed up the process (i.e. minimize the first term of the cost function to be minimized) and then settle to the steady-state control signal achieved by $NU=1$ (see Figure 2.13b).

2.3.4.3 The Effect of λ

Figure 2.14 shows the performance of the GPC with no control weighting, i.e. $\lambda=0$. The control output is active especially at points of changing the amplitude, but it is noticed that tight tracking is achieved. In Figure 2.15 λ was set to 0.5. The control output is more sluggish, which resulted in the oscillations at turning points, although the control is less active. This matches the theory presented

above. A higher λ means placing more emphasis on slow changes of the control signal sacrificing though the tight tracking of the set point profile.

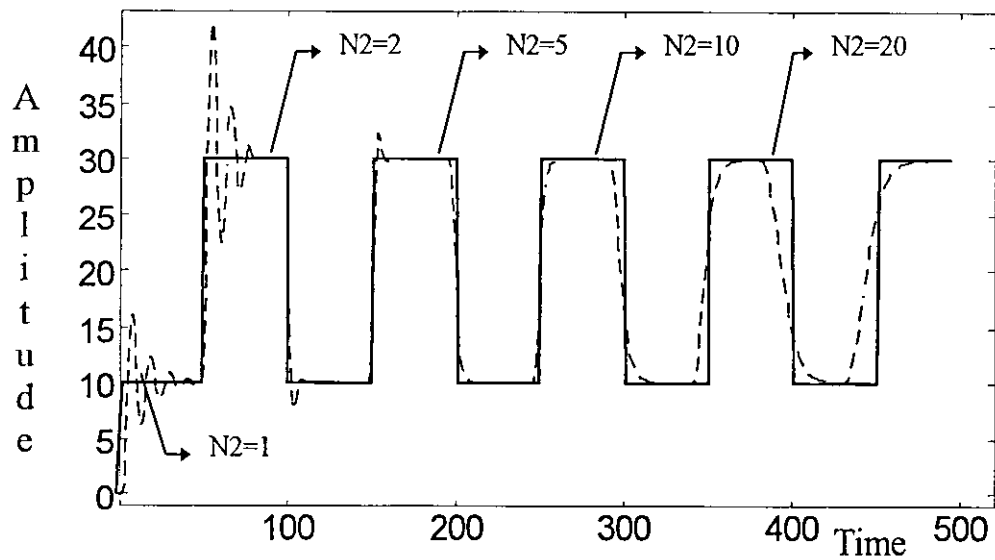


Figure 2.12a The Output Signal

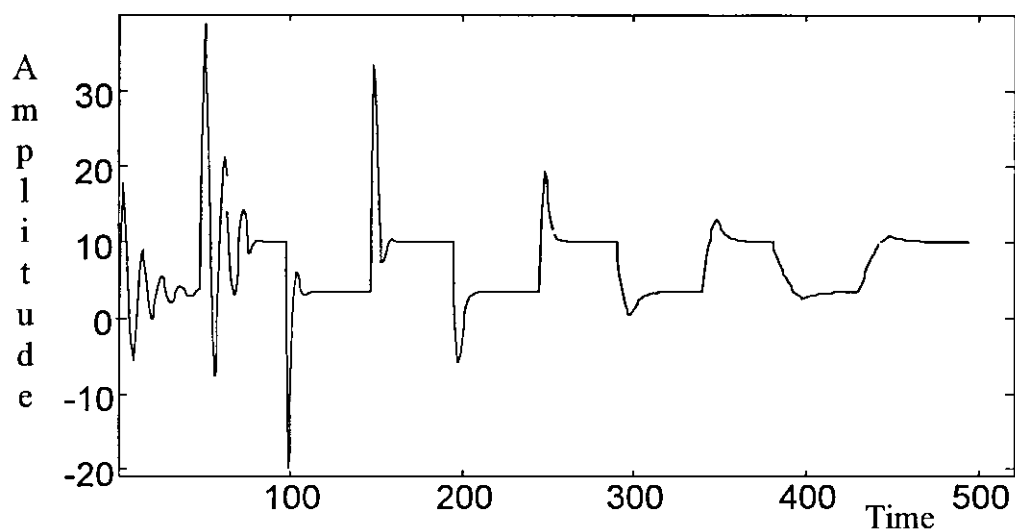


Figure 2.12b The Control Signal

Figure 2.12(a and b) Effect of N_2 on the Performance of the GPC

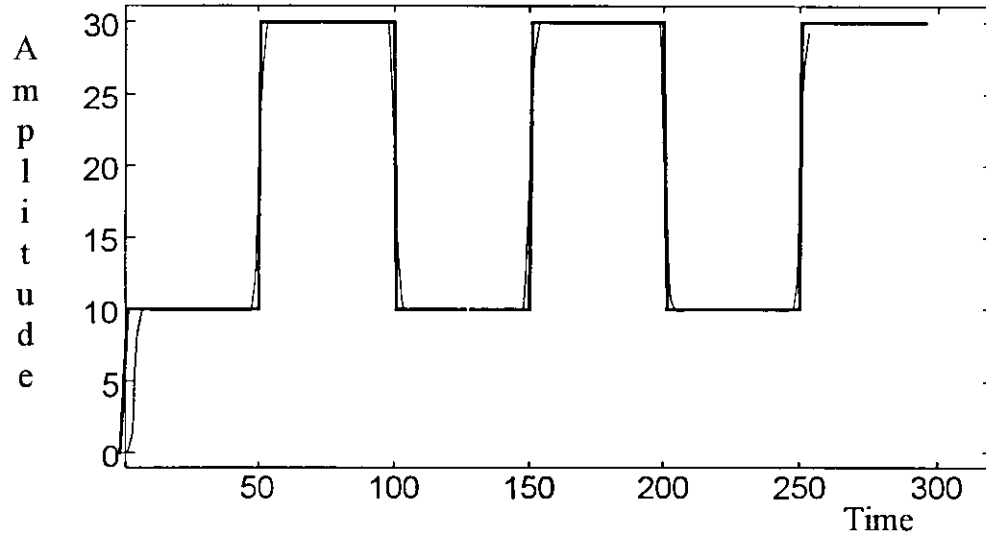


Figure 2.14a The Output Signal

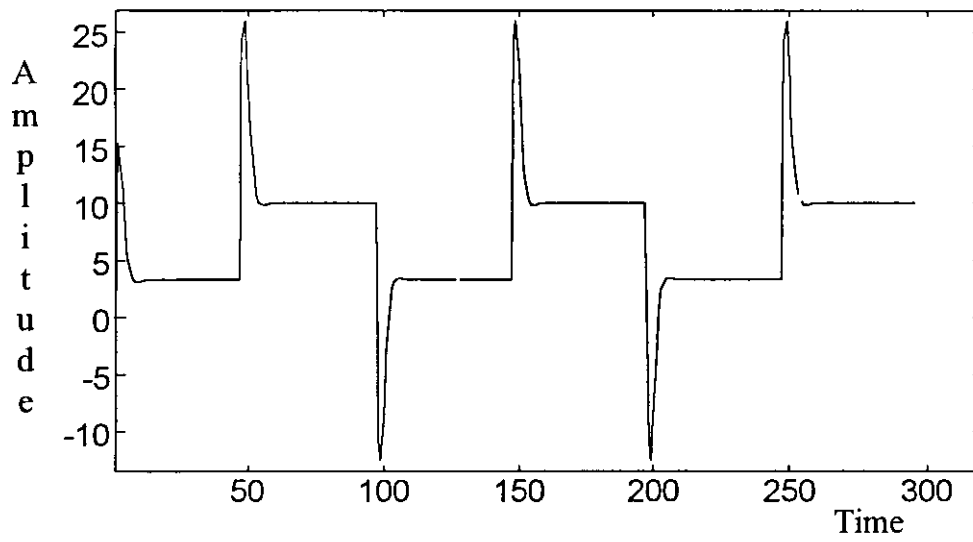


Figure 2.14b The Control Signal

Figure 2.14(a and b) Effect of $\lambda=0$ on the Performance of the GPC

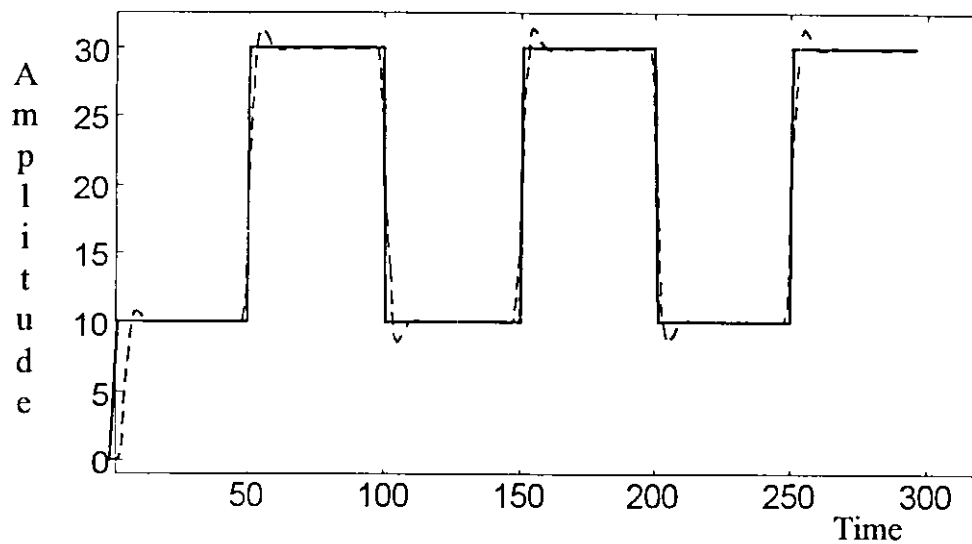


Figure 2.15a The Output Signal

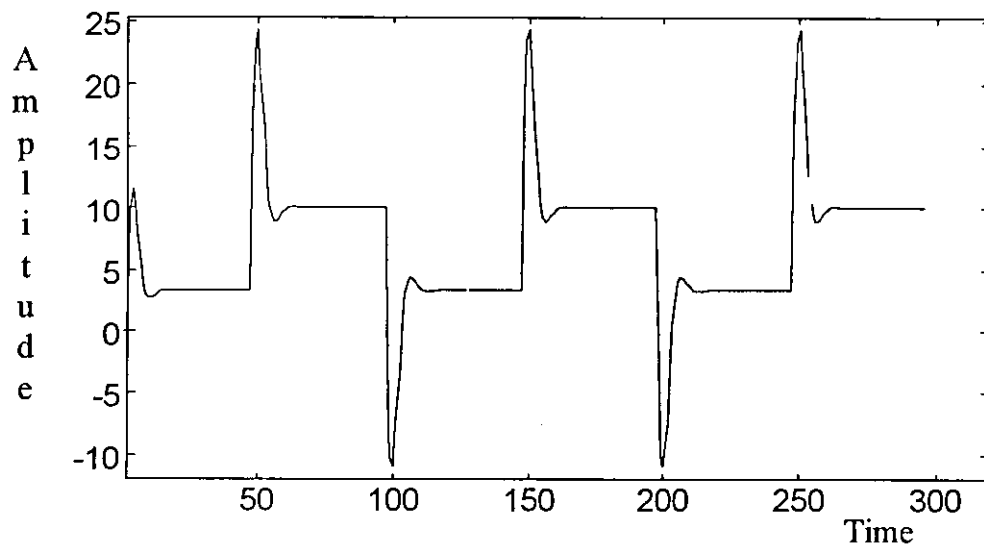


Figure 2.15b The Control Signal

Figure 2.15(a and b) Effect of $\lambda=0.5$ on the Performance of the GPC

Chapter Three

Experiments and Simulation

In order to test the two algorithms mentioned above in this thesis, namely the RLS and the GPC, they were applied on three experiments: two of these were on experimental setups in the laboratory, where the third one was done as a computer simulation using a program written in MATLAB and SIMULINK. These tests were necessary to check the suitability of the algorithms in dealing with problems of identification and control of systems, as well as to compare the performance of the GPC algorithm with that of a conventional, fixed term PID control, as will be shown in section 3.2.

3.1. The Pneumatic Speed Engine

To test the performance of the RLS and GPC algorithms, a program was written in Turbo Pascal V7.0 and was tested on an experimental setup. The setup is a pneumatic speed engine control system [9]. This engine apparatus consists of a scale model steam engine, which is driven through a motor controlled valve. The supply pressure of compressed air to the engine governs the speed of rotation of the engine crankshaft and the load. The engine supply pressure is controlled by means of an air valve, which in turn can be controlled by an electric motor. A constant motor drive voltage, produces a constant electric motor speed. Therefore, this setup can be seen as a simple built-in integrator in the pneumatic engine speed control loop.

The electrical control signal is applied through a power amplifier, and a feedback voltage signal representing the speed coming from an analog speed transducer can be measured on the amplifier unit. Figure 3.1 shows a schematic diagram of the system.

In order to control the system properly, a mathematical model has to be initiated, and hence, some assumptions about the order of the system and delay-time are to be made. By comparing the modeling of this system to a one of speed control of a DC-motor, it was assumed that the system is a second order system with integrator of the following general form:

$$G(s) = \frac{e^{-\tau s}}{s(as^2 + bs + c)} \quad (3.1)$$

where τ is the time delay of the system.

This transfer function represents the relationship between the speed of the engine as an output, and the applied voltage as an input.

Due to static friction in the gears and the valve seals, a finite input voltage is required before the valve begins to move, which gives rise to the dead-band characteristic of a motor. The delay-time of the system, i.e., the time between applying a control signal, and sensing the effect of it on the output, was assumed to be 5 samples (sampling time = 0.5 seconds). An experimental measurement of this delay time resulted in an average of 2.2 seconds.

In order to run the identification, an excitation signal has to be chosen. In theory the best signal for this purpose is a zero-mean white noise signal. Nevertheless, this signal could not be applied here because of the existence of a noticeable

time-delay in the system. The signal would have been too fast for the mechanical system and the output would not reflect truly the changes on the input. Since an integrator is included in the system, a step input on the other hand, would have driven the system into saturation due to the built-in integrator. Therefore, an excitation signal as shown in Figure 3.2 was chosen. This signal would drive the motor to a fixed speed and then hold it there.

After applying the chosen $u(t)$ and on-line identification of the system the following set of parameters was obtained :

$$\Theta = [-0.5308 \quad -0.3824 \quad -0.0867 \quad -0.3567 \quad -0.0723 \quad 0.1236]^T$$

Note that the data vector supplied to the RLS algorithm was as follows:

$$x = [-y(t-1) \quad -y(t-2) \quad -y(t-3) \quad u(t-5) \quad u(t-6) \quad u(t-7)]$$

where $u(t)$ is the control signal applied to the valve motor and $y(t)$ is the voltage signal representing actual speed of the engine. Note also the incorporation of the delay time in this data vector.

It is worthwhile mentioning here, that this setup was very sensitive to changes in the 'tuning knobs' of the RLS algorithm. Several experiments had to be carried out in order to tune these parameters. As discussed in the previous chapter, there was a problem of converging to a different local minimum in every run. Hence a different set of parameters was the result. After performing multiple experiments with different sets of parameters, the above mentioned parameter set proved to be most suitable to be a basis for the GPC controller. These experiments were performed by fixing all GPC parameters and changing only the estimated parameter set.

Figures 3.3, 3.4 and 3.5 represent three runs of controlling the engine speed using the self-tuning GPC controller. The set point was a step change of the speed from a zero rpm speed to a speed equivalent to -1.5 Volts on the speed transducer. The parameters for the estimator were set as follows:

$$\Theta(0) = [-0.5 \ -0.2 \ -0.1 \ -0.2 \ -0.1 \ 0.1].$$

$$P(0) = 2 * I, \text{ where } I \text{ is a } 6 \times 6 \text{ unity matrix.}$$

The difference between the three figures is actually in the value of λ , the control weighting factor. In these figures, λ was 75, 85 and 90 respectively. The effect on the response and especially on the control signal can be easily deduced. Where in Figure 3.3 the control signal was very active, especially in the first part of the run, it became more smooth and sluggish in the next two experiments. The other GPC parameters were :

$$N1 = 5, N2 = 15 \text{ and } NU = 1$$

Note that N1 was not set to 1 as usual, but to the value of the delay time of the system which was 5 as mentioned above.

The behavior of the control signal and that of the actual output signal, corresponds to the already mentioned effect of λ on the performance of the GPC controller (see section 2.3.4.3). In Figure 3.3 the control signal was given the possibility to change actively (lowest λ), this results in a very active control signal, and hence in an oscillatory behavior of the output. As λ increases, the emphasis was put on keeping the control signal from changing fast, and though, a more sluggish response on the output was achieved (Figures 3.4 and 3.5).

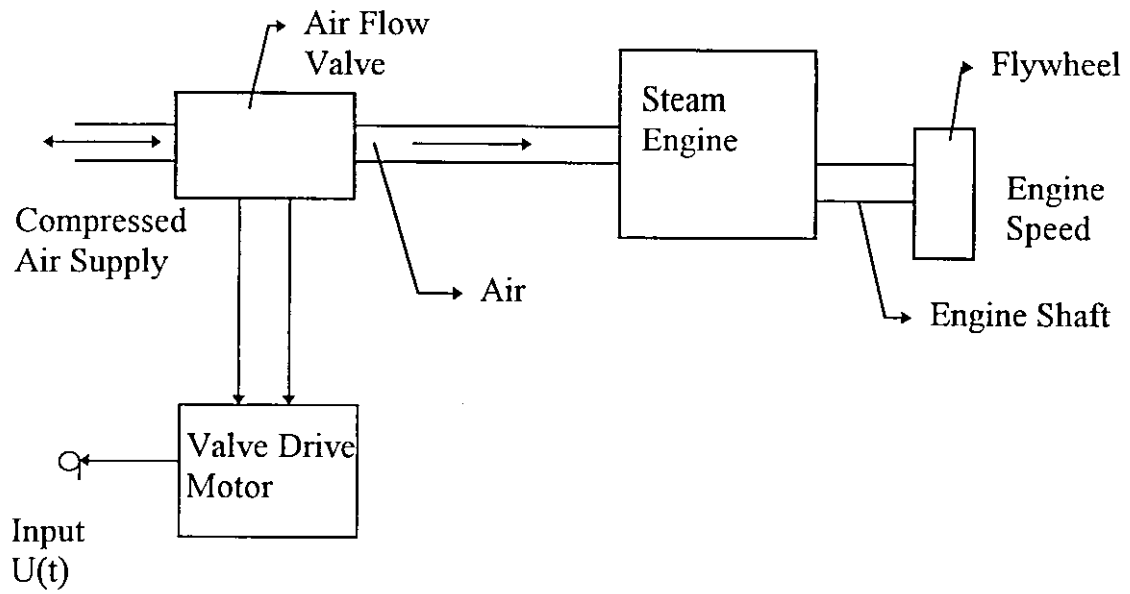


Figure 3.1 Schematic of the Pneumatic Speed Engine Setup [9]

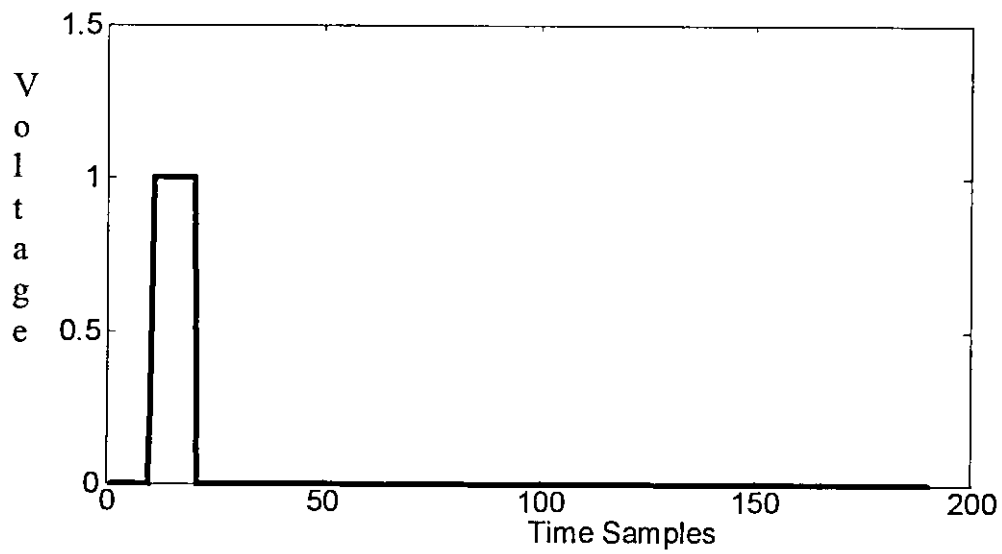


Figure 3.2 Excitation Signal for the Identification

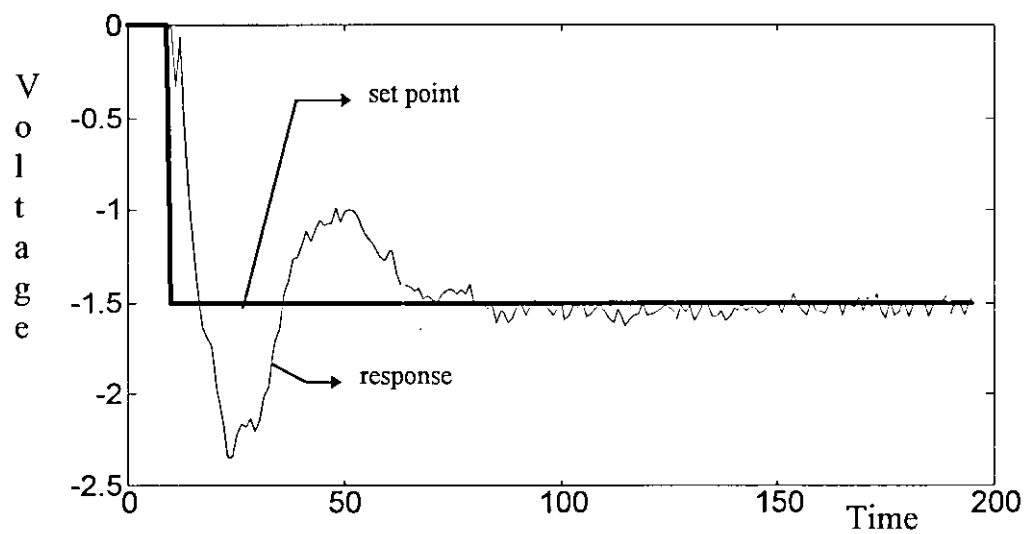


Figure 3.3a The Output Signal

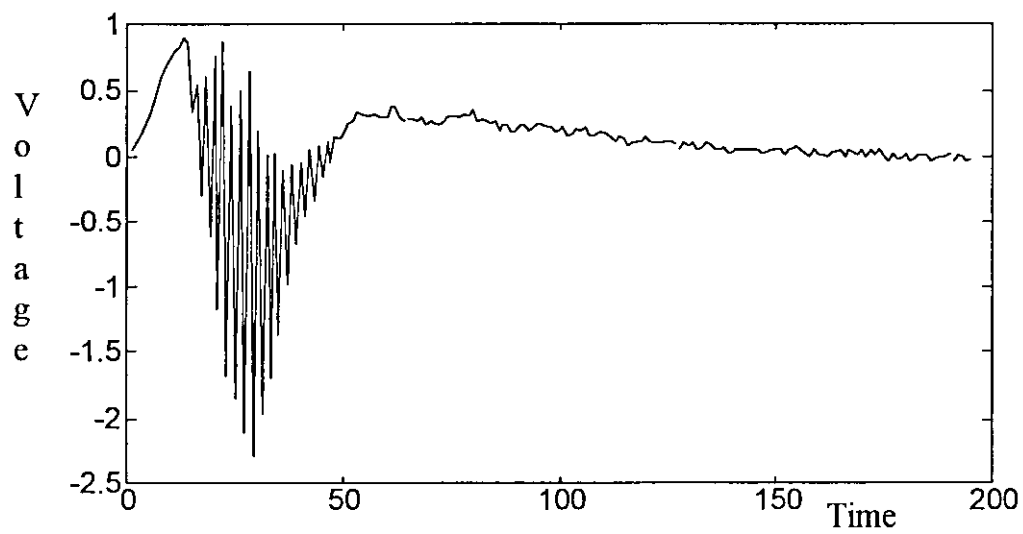


Figure 3.3b The Control Signal

Figure 3.3 Controlling the Speed Engine with $\lambda=75$

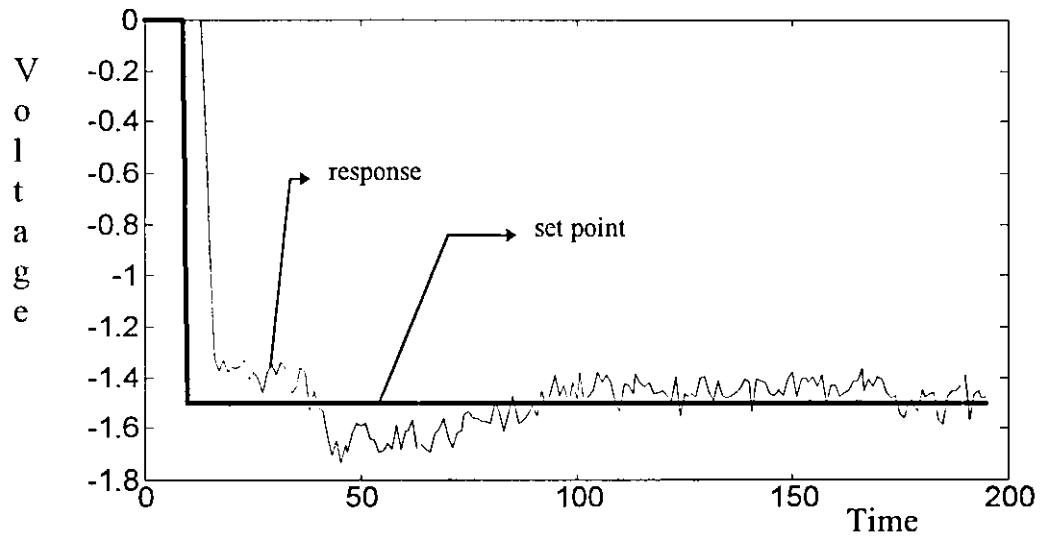


Figure 3.4a The Output Signal

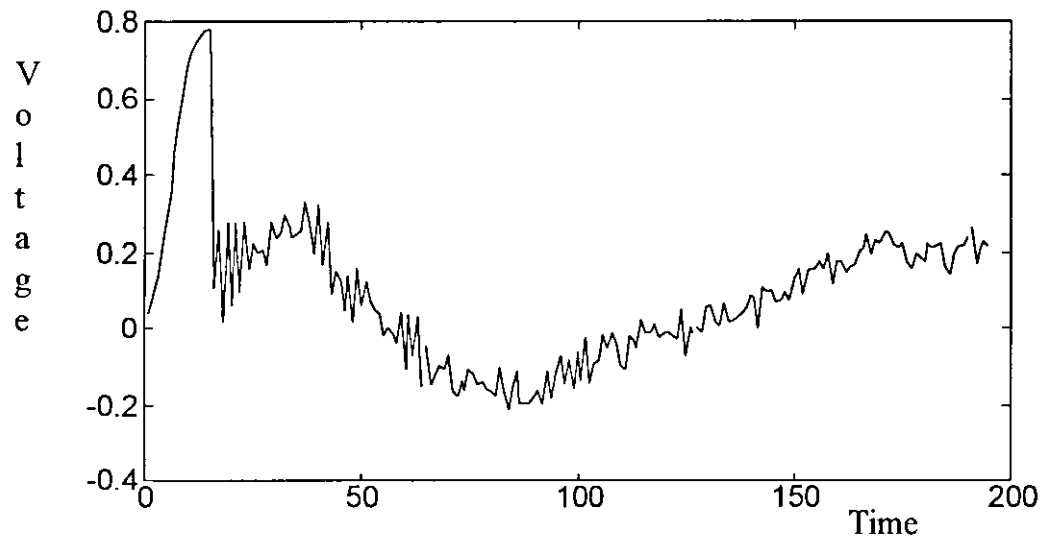


Figure 3.4b The Control Signal

Figure 3.4 Controlling the Speed Engine with $\lambda=85$

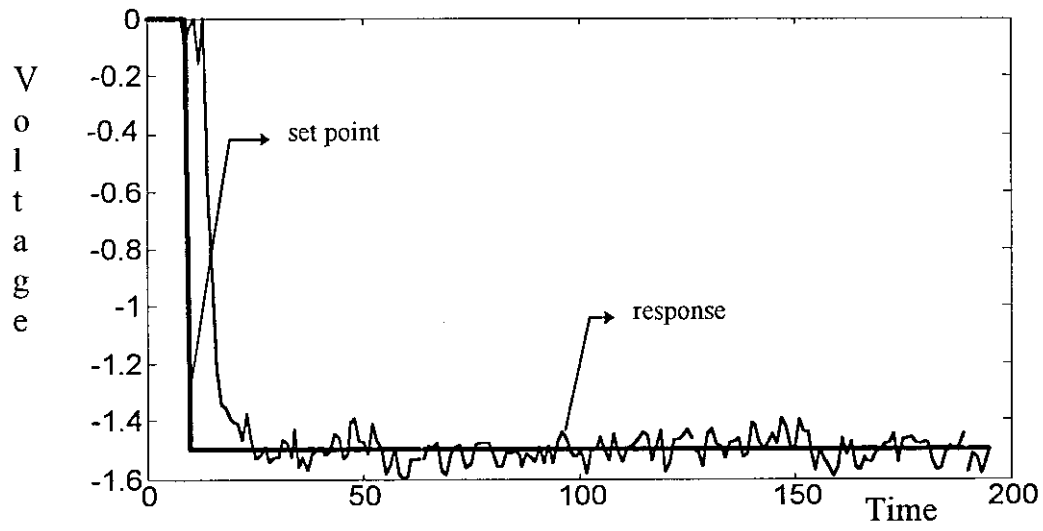


Figure 3.5a The Output Signal

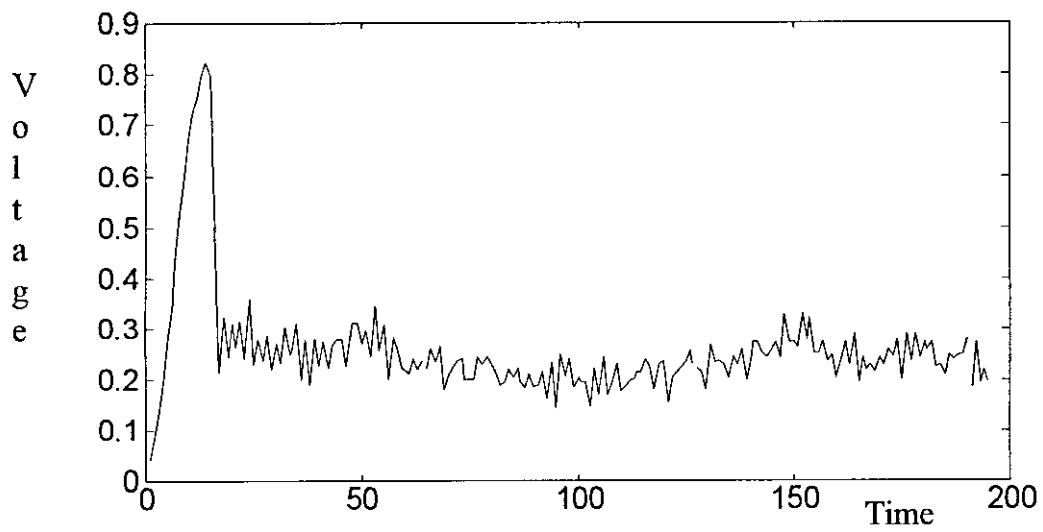


Figure 3.5b The Control Signal

Figure 3.5 Controlling the Speed Engine $\lambda=90$

3.2. Simulation Experiment :

To test the performance of the GPC algorithm on a 2-axes system similar to the actual purpose of this thesis, which is to apply the GPC on a flame cutting X-Y-machine, and to demonstrate the advantages of self-tuning predictive control over conventional fixed-term PID control for such problems, the following experiment was carried out.

The objective of the experiment is to control two identical motors simultaneously, such that a pre-drawn profile should be tracked. The model of each of the motors was considered to be:

$$G(s) = \frac{1}{s(s^2 + 1.2 * s + 1)} \quad (3.2)$$

The PID and the GPC were tuned initially to give as similar step input responses as possible, especially from the point of view of maximum overshoot and settling time, it is noticed that the PID system has a percentage overshoot of 10.37% while the GPC system has a percentage overshoot of 10.6%. The settling time of the PID system was 23 samples and that of the GPC system was 16 samples. Figures 3.6 and 3.7 show the step response of the GPC system and the PID system respectively. Note here the higher value of the control signal of the PID controller, where the control signal of the GPC was much lower in amplitude due to the predictive nature of the GPC, where the coming event (the step) was anticipated, and though, no severe control action had to be taken in order to track the change. Also note that the GPC system started raising the output before the

actual step was introduced, which is a clear demonstration of the predictive nature of the GPC.

The parameters for the PID were chosen as:

$$K_p=0.5; \quad K_i=0.01; \quad K_d=0.3.$$

while the parameters of the GPC were :

$$N_1=1; \quad N_2=3; \quad NU=1; \quad \lambda=1.$$

The path to be followed by the X-Y-system can be seen in Figure 3.8.

Comparing Figures 3.9 (a and b) and 3.10 (a and b), it can be seen the PID controller performed relatively good at the beginning, but as soon as the first turning point was reached, the controller was not able to track the sudden change in the preset profile at a relatively late stage. This scenario repeated itself at every turning point. Note also that the PID system did not reach the final point in the line. Actually, this behavior is due to an error of the PID system in response to a ramp input because the value of K_i is small. At steady-state, i.e. with increasing time, this error will become zero as a result of the integrating action. Nevertheless, this error can be made zero at earlier stages by increasing the value of K_i , but this will increase the relative instability of the system and can make the system more oscillatory or even unstable.

The tuning of the PID parameters was very difficult due to the existence of the integrator in the system itself, which reduced the stability margin of the system and increased the settling time of the system. Figure 3.12 shows the response of the PID system to a ramp input using the same parameters as in the case of a step input. In this figure the above mentioned error can be easily seen. Figure 3.13

shows another response of the PID system to a ramp input. Note here that the steady-state error will become zero as time increases.

The GPC on the opposite side, and due to its predictive nature, was able to foresee the turning points and adjust itself early to them. The result was a clean tracking of the profile, missing only the corner point of it. If the prediction horizon N_2 was set to a larger value, even more points would be missed. This can also be explained by the predictive nature of the GPC. The larger the prediction horizon, the earlier the GPC will start taking measures to accommodate forthcoming changes in the track. Therefore, for an application where it is expected to face many sharp turning points, it is advisable to keep the N_2 parameter as low as possible. Figure 3.11 shows the performance of the same GPC with only N_2 changed to 7. It is noticed that the radius of curvature at turning points was larger than the previous case because the GPC algorithm starts taking action towards changing the direction much earlier than in Figure 3.10.

Three other experiments were performed in order to underline the results obtained and illustrated in Figures 3.9 and 3.10. The new profiles were introduced to the two systems. These profiles are shown in Figures 3.14, 3.15 and 3.16. The performance of the PID system in tracking these profiles is shown in Figures 3.17, 3.19 and 3.21 respectively, and that of the GPC system is shown in Figures 3.18, 3.20 and 3.22. Comparing the performance of the two systems for the new profiles, the above mentioned statements about the superiority of the GPC controller over the PID controller can be underlined.

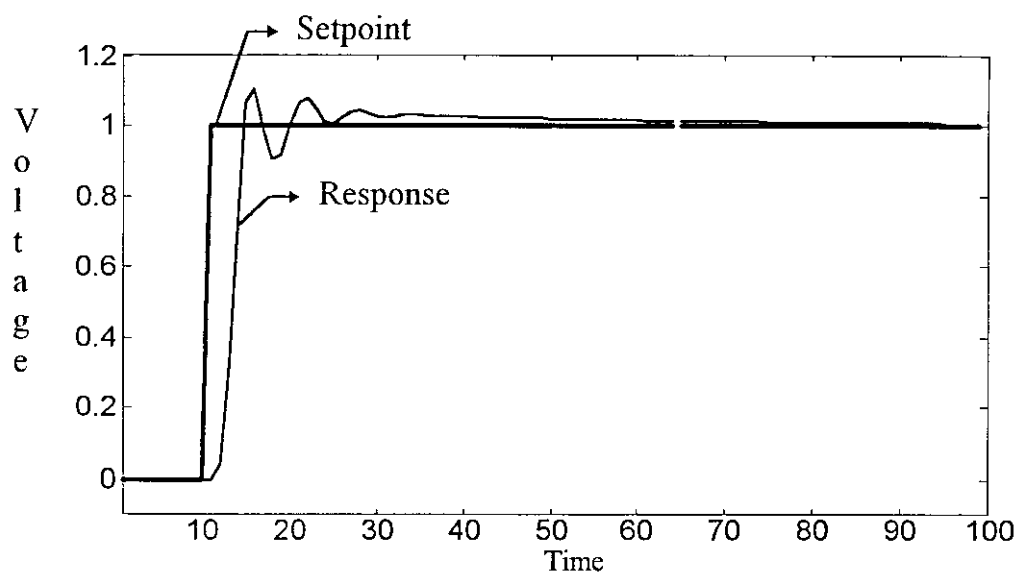


Figure 3.6a The Output Signal

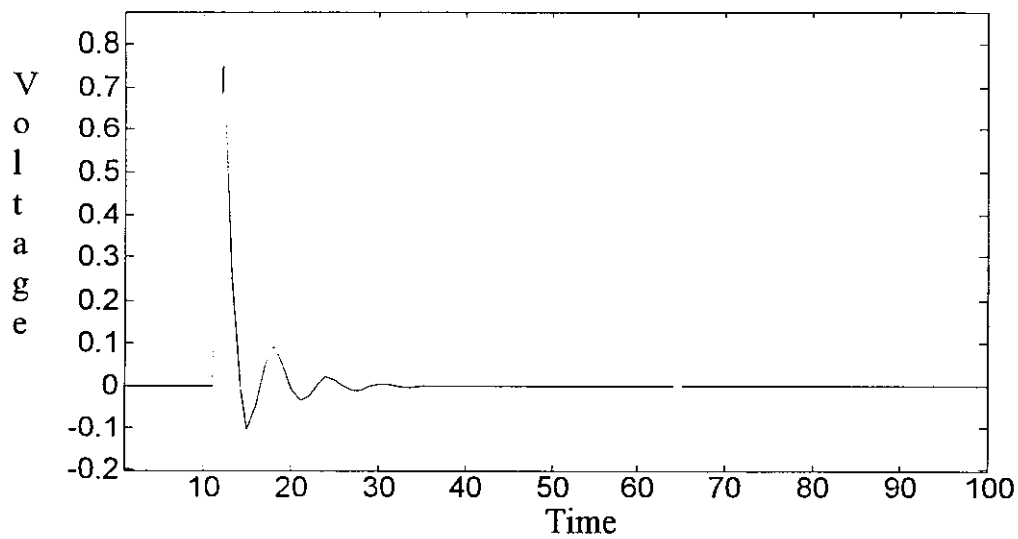


Figure 3.6b The Control Signal

Figure 3.6 Step Response of the PID System

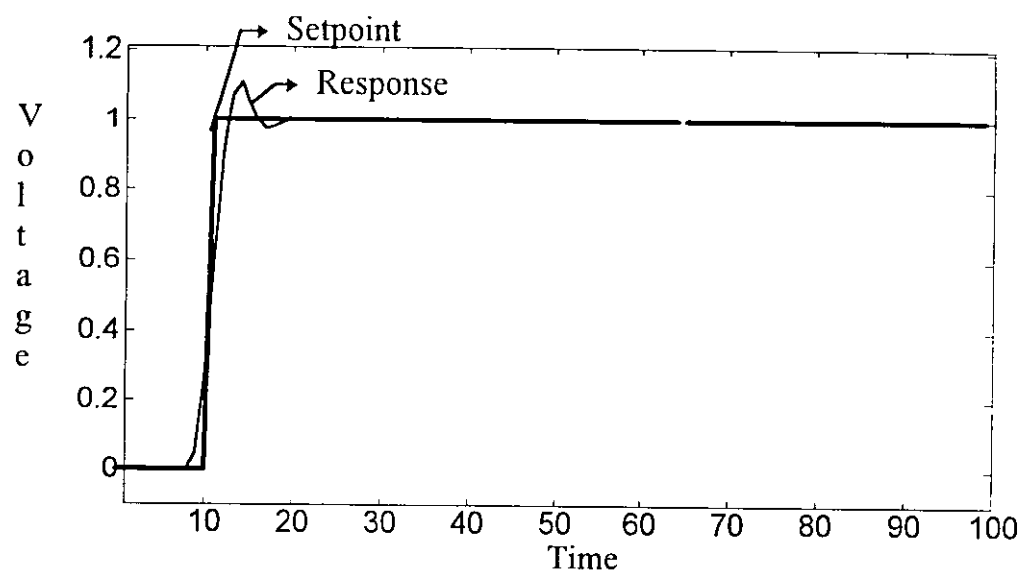


Figure 3.7a The Output Signal

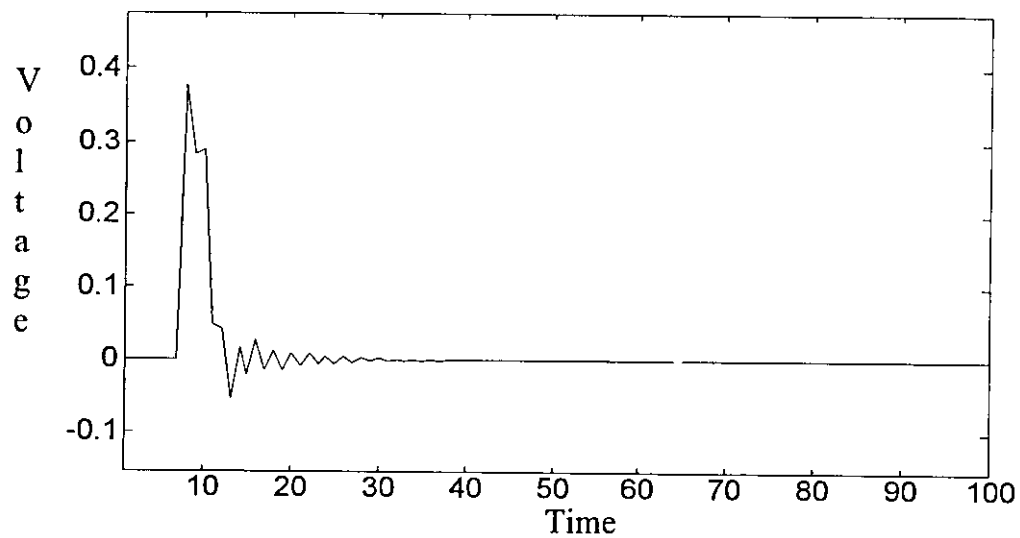


Figure 3.7b The Control Signal

Figure 3.7 Step Response of the GPC System

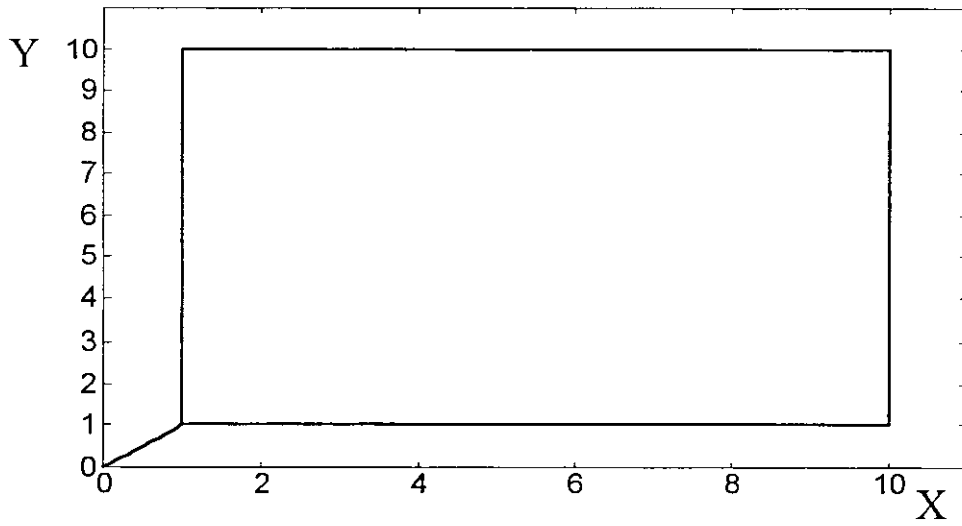


Figure 3.8 Profile 1 to be followed by the modeled X-Y-Machine.

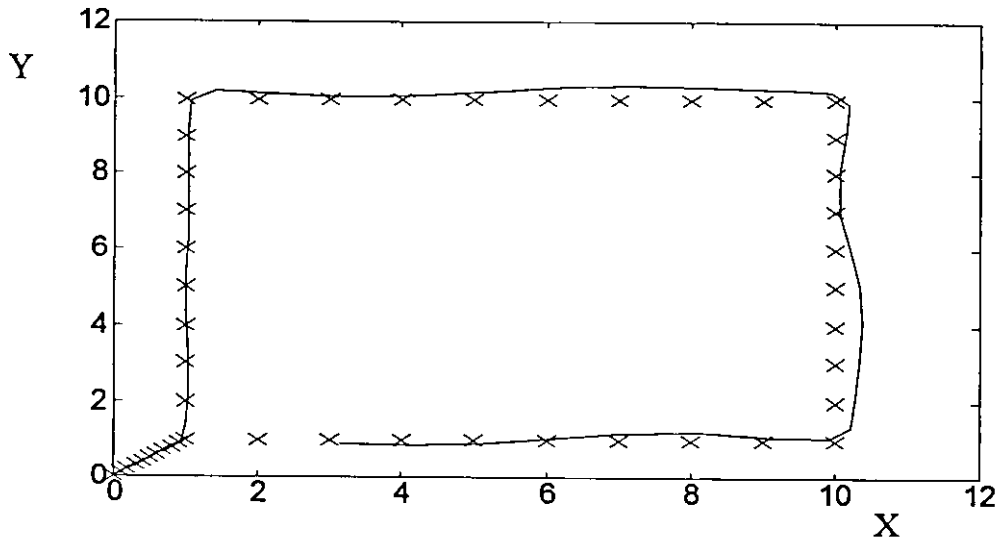


Figure 3.9a Actual Profile (line) vs. Set Profile (x) of the PID System

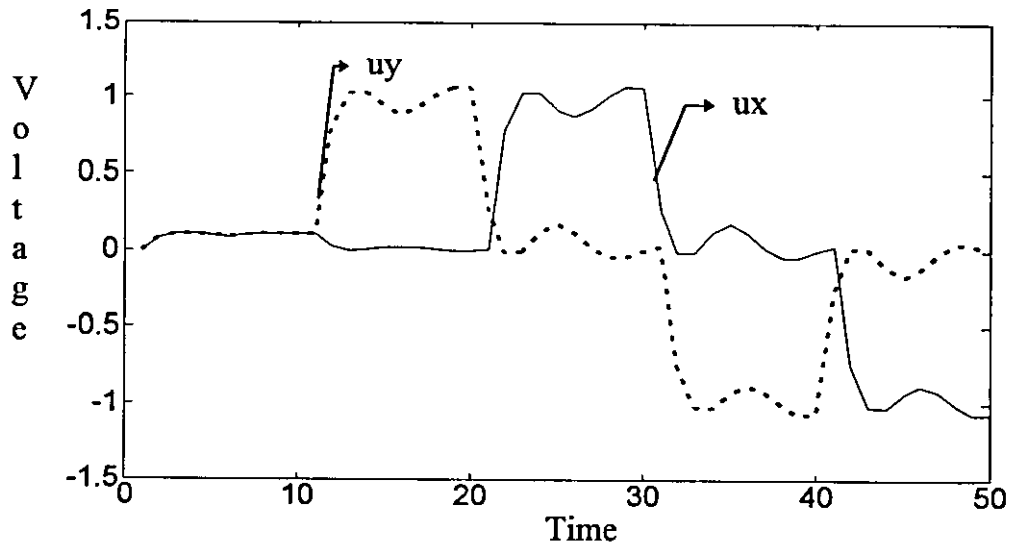


Figure 3.9b The 2 Control Signals for the X- and Y-Motors (PID System)

Figure 3.9 Performance of the PID System in Tracking Profile 1

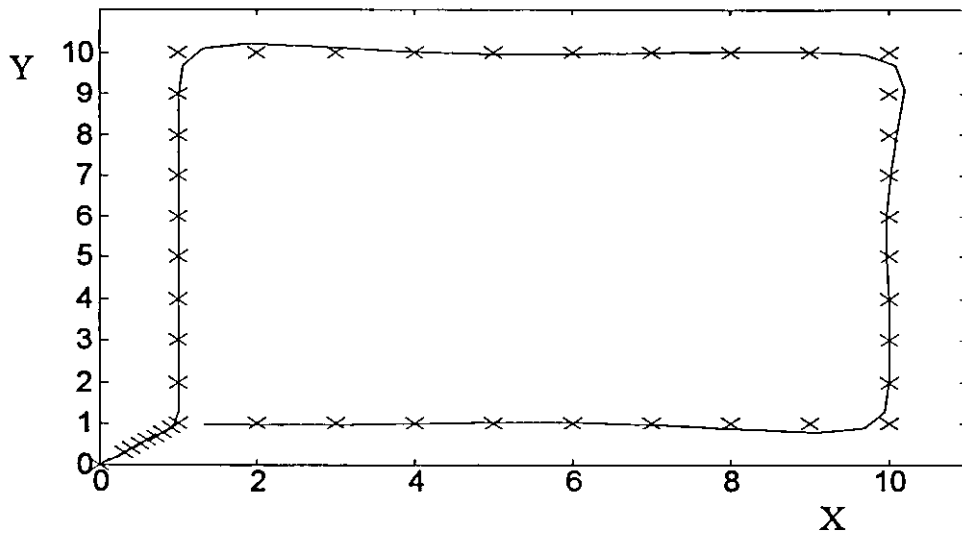


Figure 3.10a Actual Profile (line) vs. Set Profile (x) of the GPC System

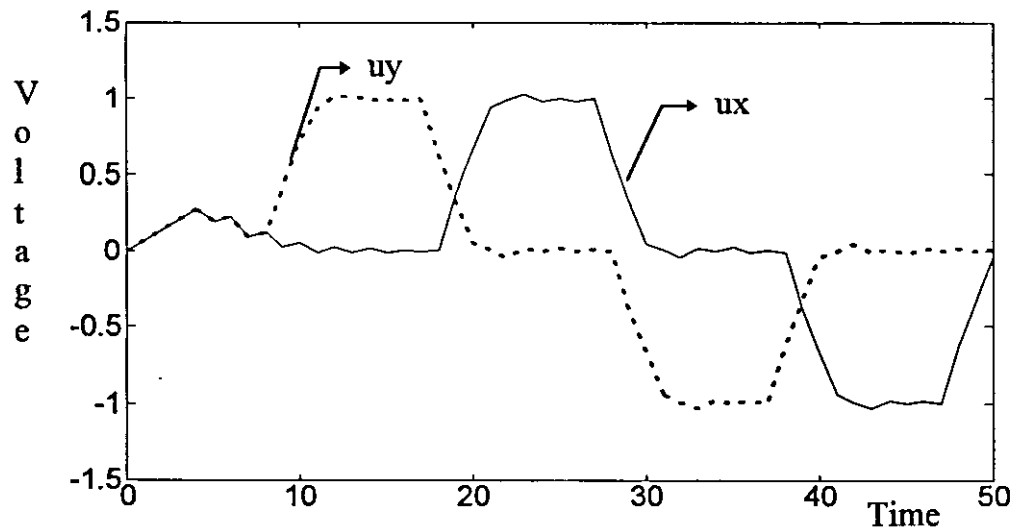


Figure 3.10b The two Control Signals for the X- and Y-Motors (GPC System)

Figure 3.10 Performance of the GPC System with $N_2=3$ in Tracking Profile 1

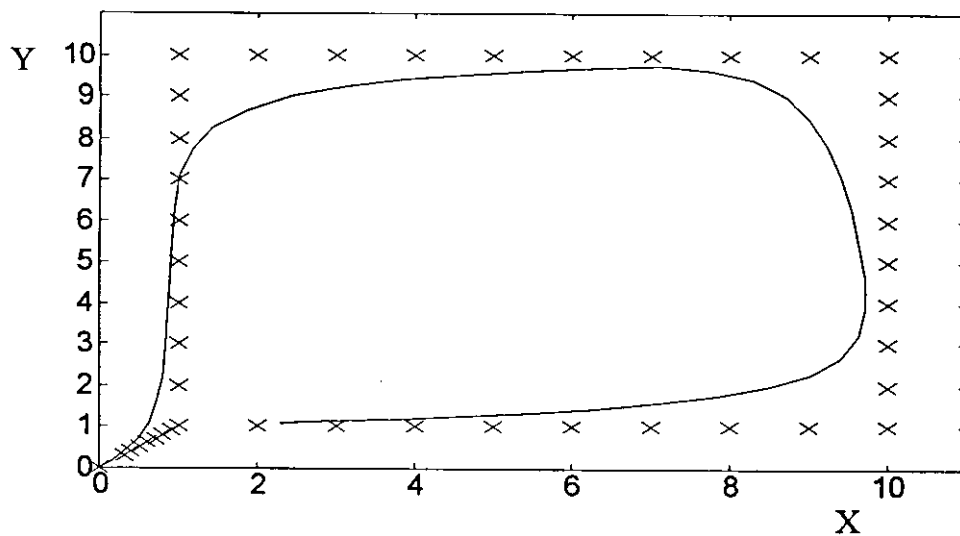


Figure 3.11a Actual Profile (line) vs. Set Profile (x) of the GPC System ($N_2=7$)

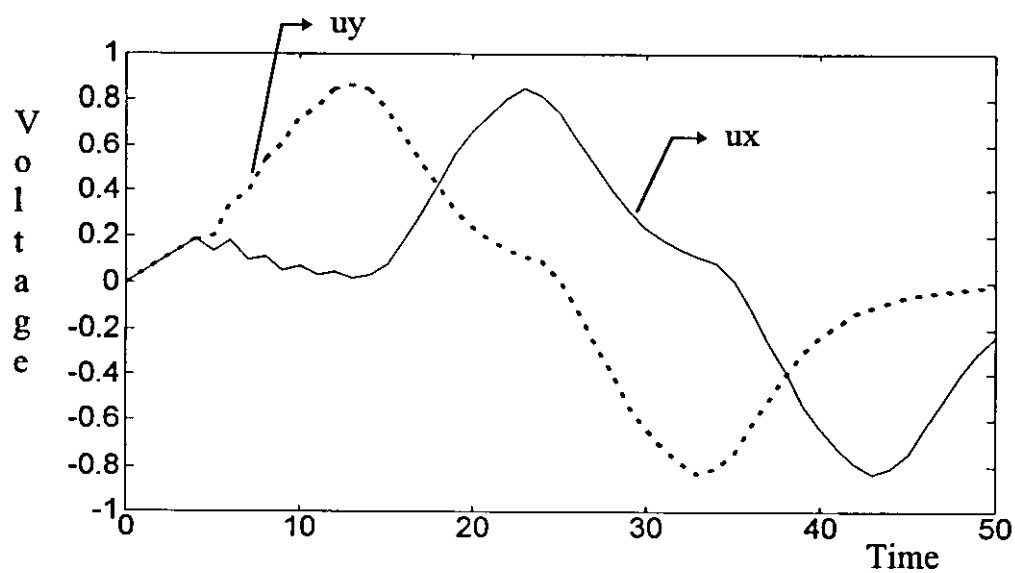


Figure 3.11b The two Control Signals for the X- and Y-Motors

Figure 3.11 Performance of the GPC System in Tracking Profile1 with $N_2=7$

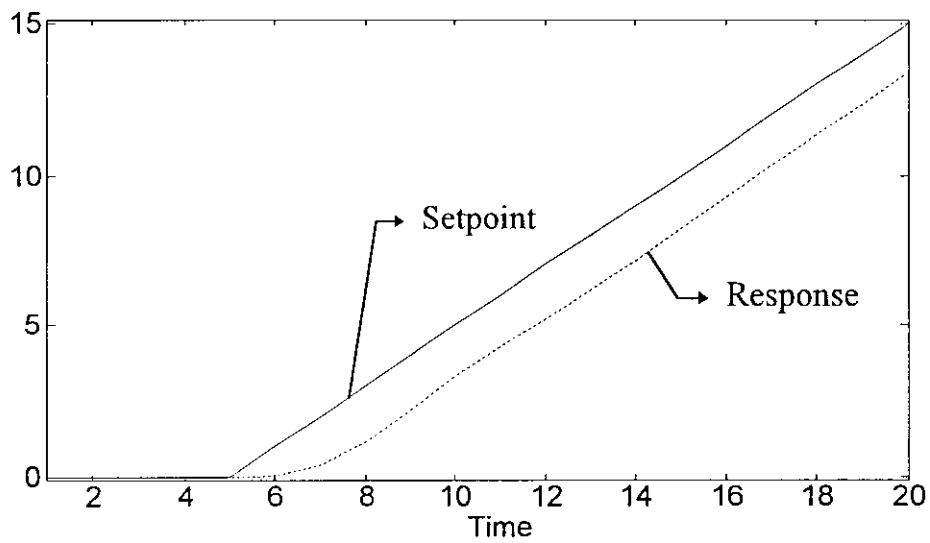


Figure 3.12 Response 1 of the PID system to a Ramp Input

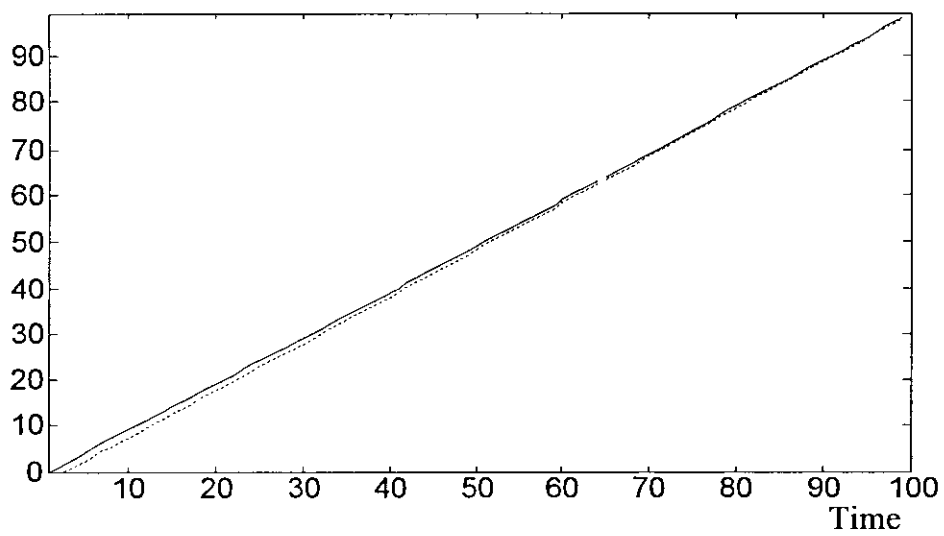


Figure 3.13 Response 2 of the PID system to a Ramp Input

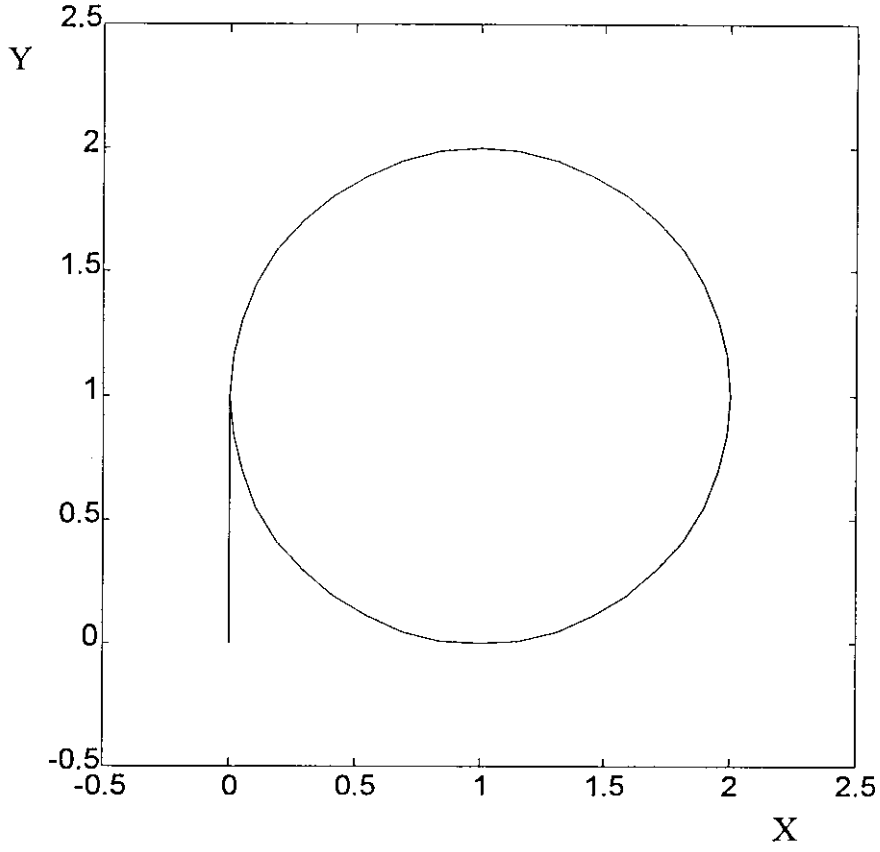


Figure 3.14 Profile 2 to be followed by the modeled X-Y-Machine.

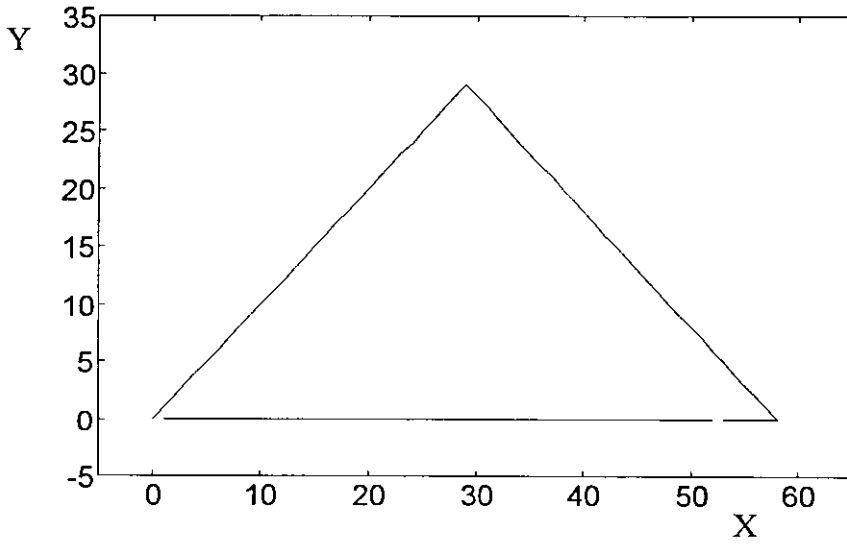


Figure 3.15 Profile 3 to be followed by the modeled X-Y-Machine.

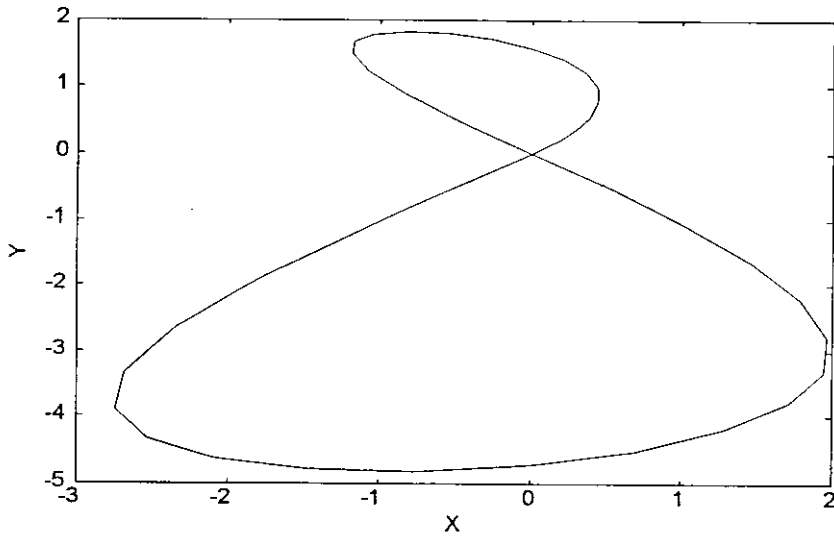


Figure 3.16 Profile 4 to be followed by the modeled X-Y-Machine.

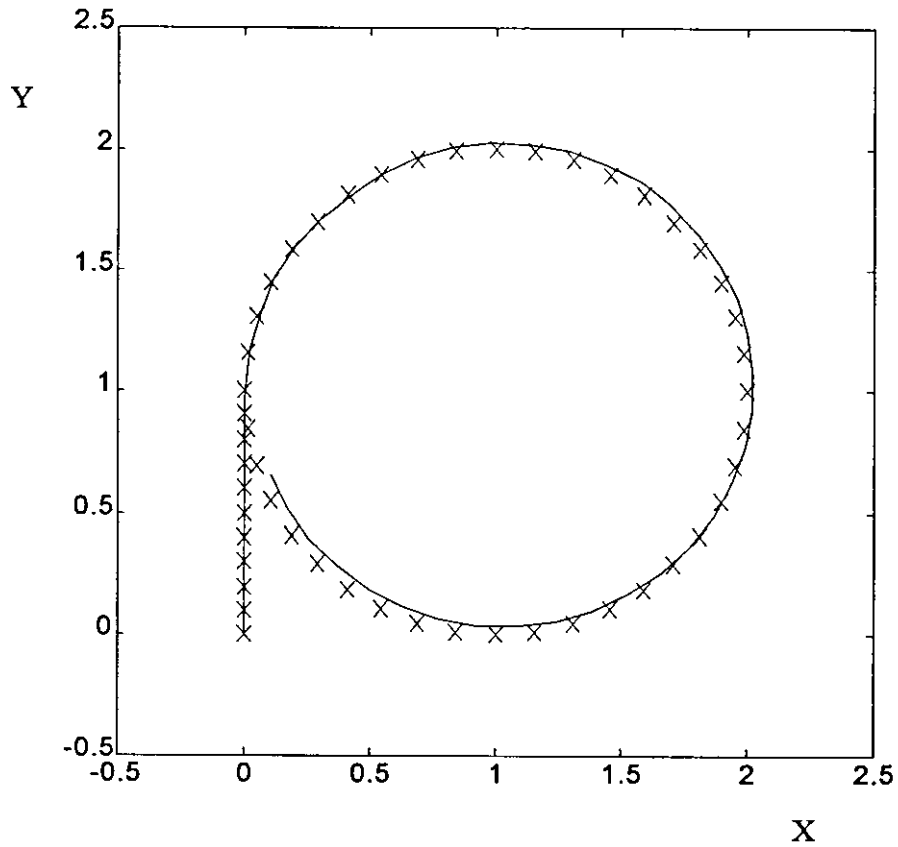


Figure 3.17a Actual Profile (line) vs. Set Profile (x) of the PID System

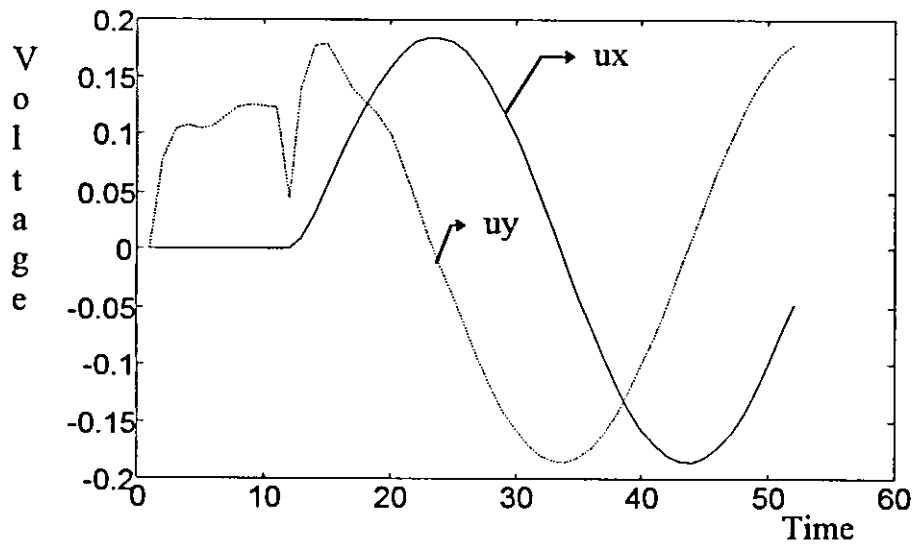


Figure 3.17b The two Control Signals for the X- and Y-Motors

Figure 3.17 Performance of the PID System in Tracking Profile 2

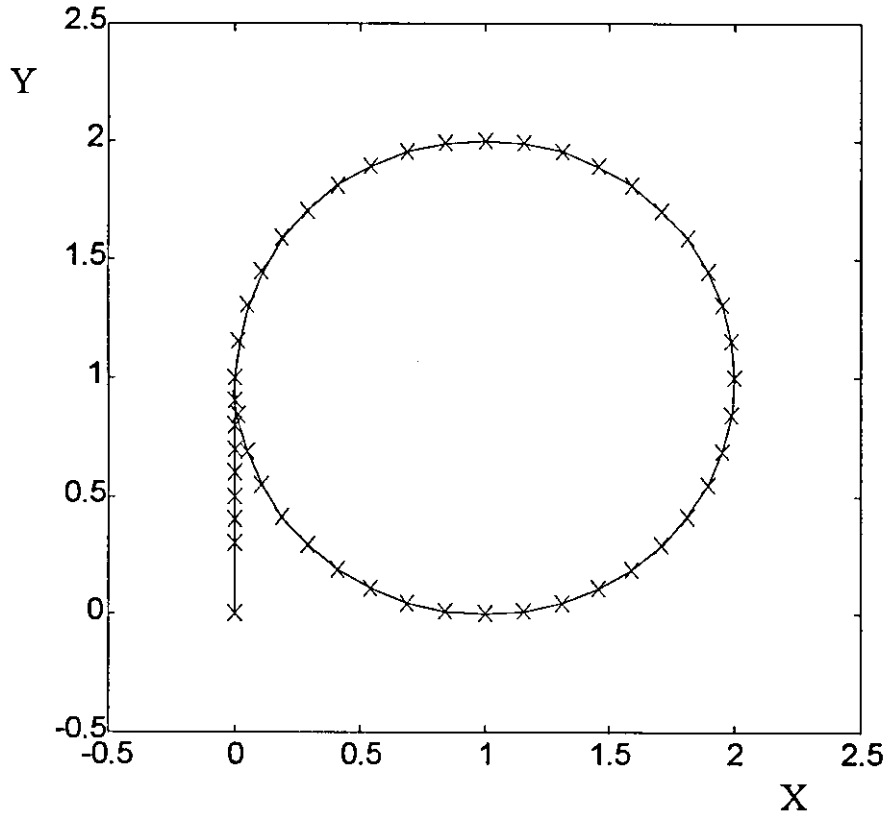


Figure 3.18a Actual Profile (line) vs. Set Profile (x) of the GPC System

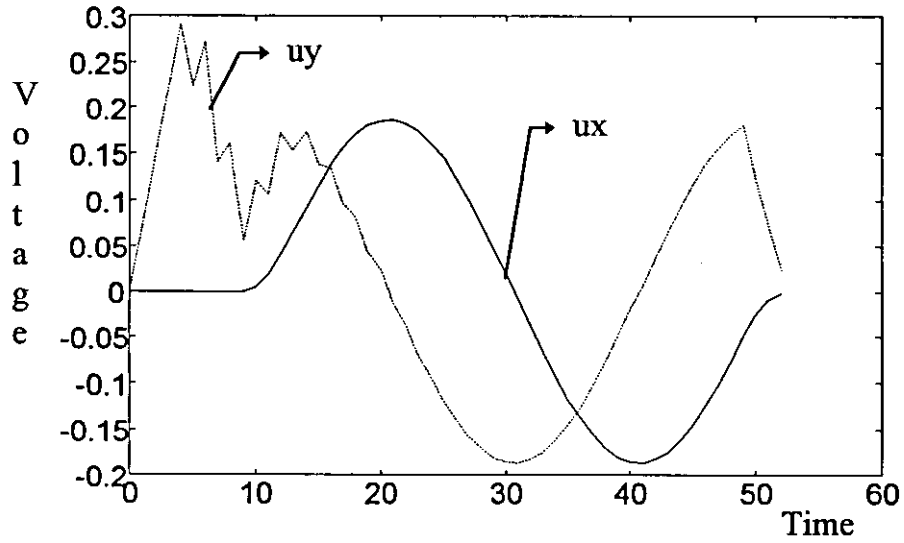


Figure 3.18b The two Control Signals for the X- and Y-Motors

Figure 3.18 Performance of the GPC System in Tracking Profile 2

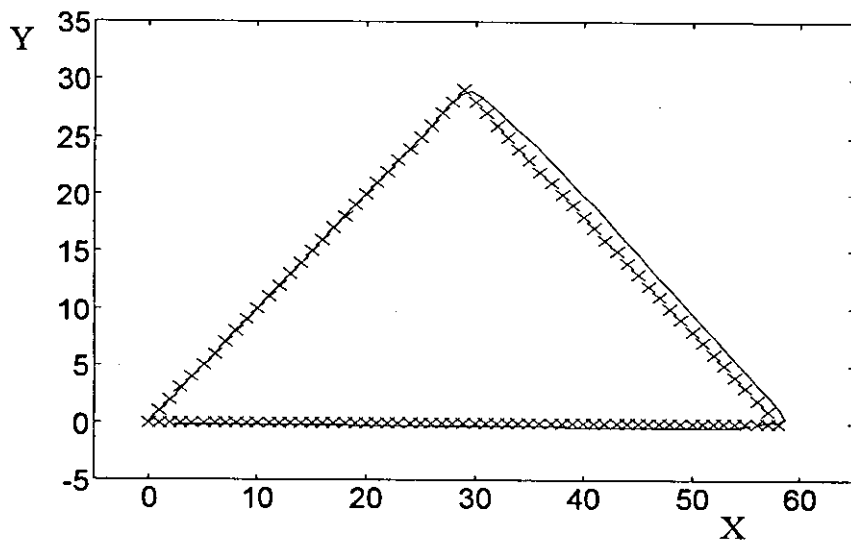


Figure 3.19a Actual Profile (line) vs. Set Profile (x) of the PID System

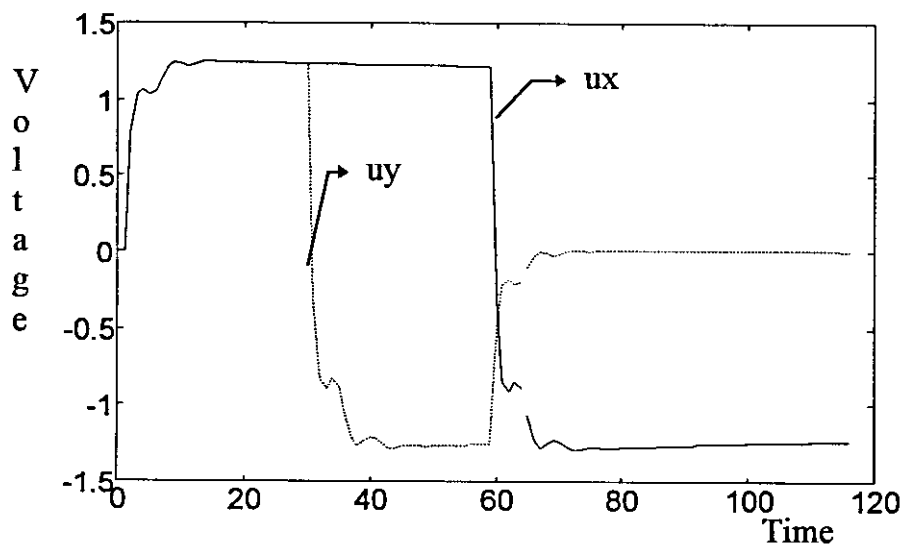


Figure 3.19b The two Control Signals for the X- and Y-Motors

Figure 3.19 Performance of the PID System in Tracking Profile 3

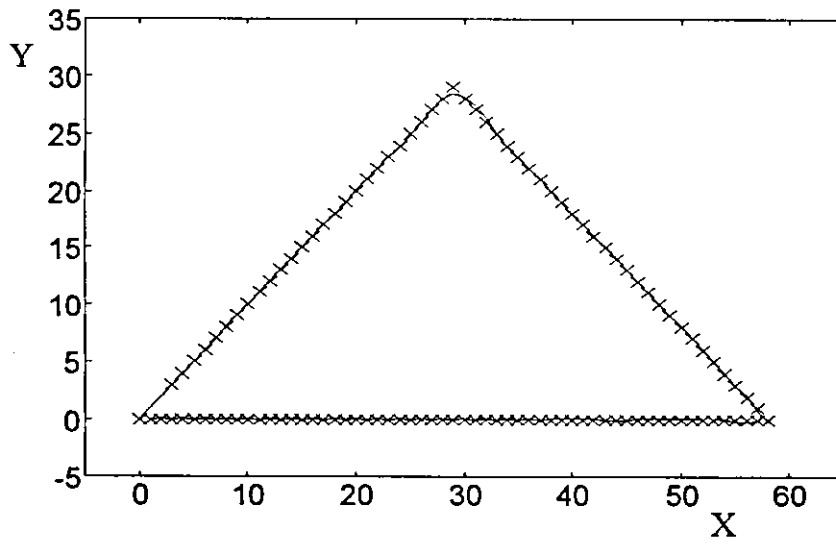


Figure 3.20a Actual Profile (line) vs. Set Profile (x) of the GPC System

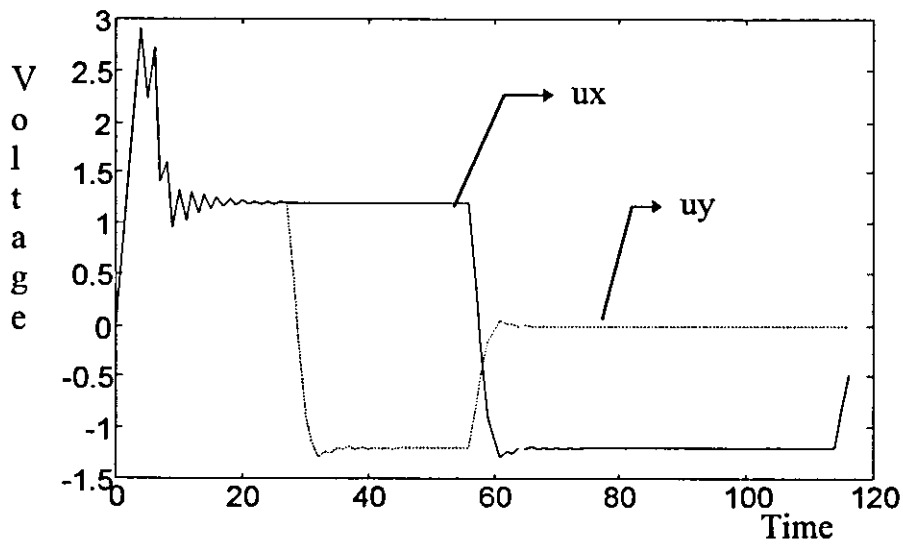


Figure 3.20b The two Control Signals for the X- and Y-Motors

Figure 3.20 Performance of the GPC System in Tracking Profile 3

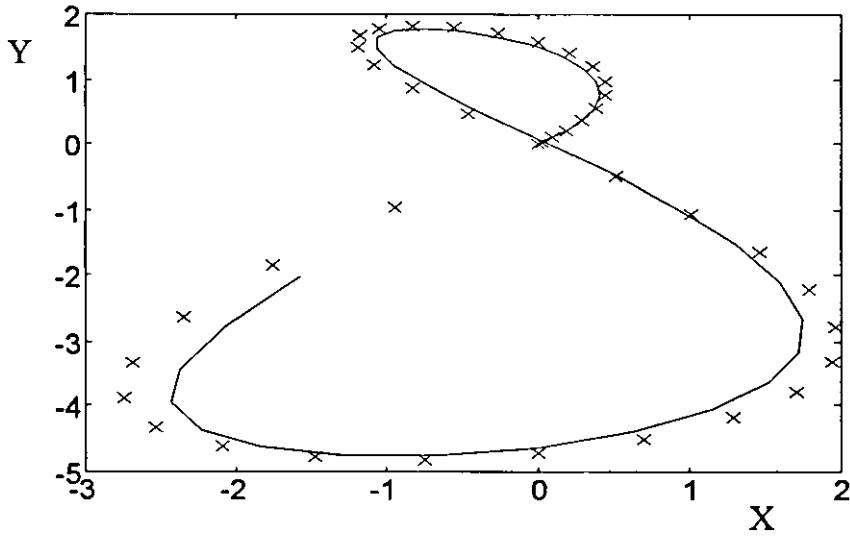


Figure 3.21a Actual Profile (line) vs. Set Profile (x) of the PID System

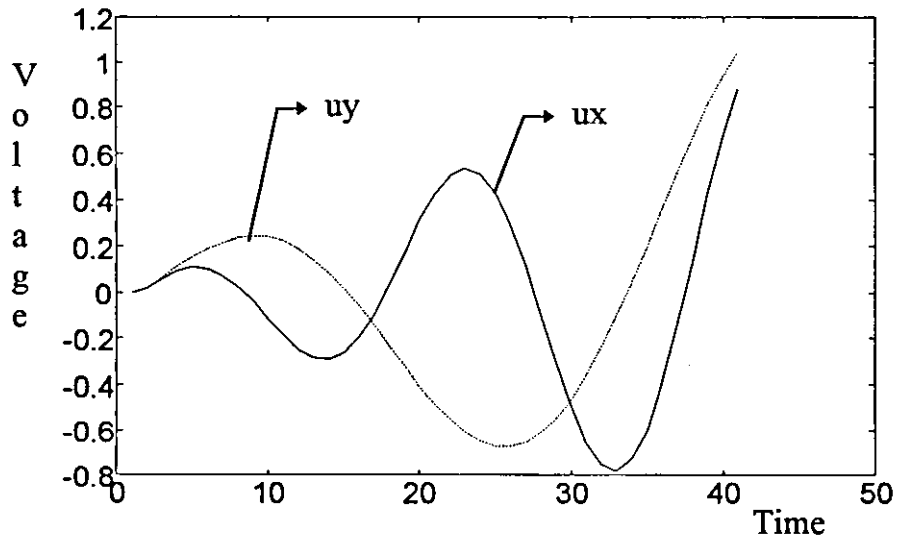


Figure 3.21b The two Control Signals for the X- and Y-Motors

Figure 3.21 Performance of the PID System in Tracking Profile 4

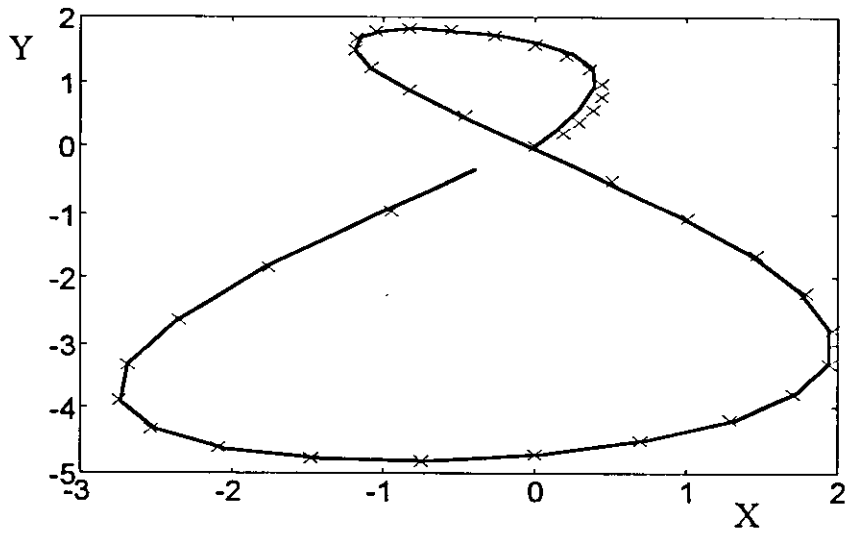


Figure 3.22a Actual Profile (line) vs. Set Profile (x) of the GPC System

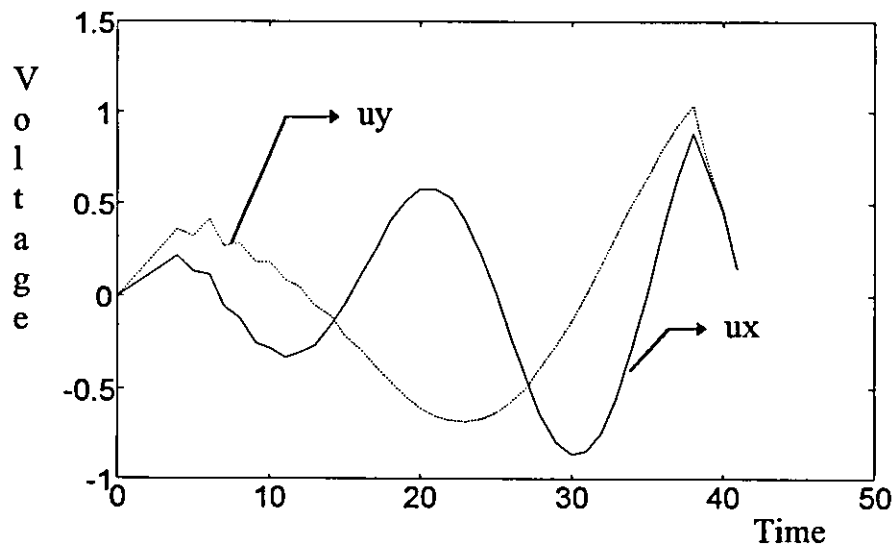


Figure 3.22b The two Control Signals for the X- and Y-Motors

Figure 3.22 Performance of the GPC System in Tracking Profile 4

3.3 Proto-Type X-Y-Machine

To determine the suitability of the algorithms described above for the motion control of a flame cutting machine, a test was performed on a proto-type X-Y-machine with two permanent magnet DC motors as actuators and two optical encoders as feedback devices. The power rating of the motors is 12 V, 4 A each and the encoders can give 100 pulses per revolution. The maximum stroke of this machine in the X-direction is 600 mm and 250 mm in the Y-direction. The program for this test was written in Turbo Pascal V7.0.

To initiate the mathematical model of each motor, the same assumption was made regarding the transfer function of the motors as in equation 3.1.

$$G(s) = \frac{e^{-\tau s}}{s(as^2 + bs + c)}$$

This transfer function represents the relationship between the position of the carriage in either direction as an output and the input voltage on the specific motor. The delay time of the system was determined to be one sample, i.e. 0.5 seconds.

Different runs of the RLS algorithm were performed using different excitation signals on both motors. These signals were a square wave, a white noise and pseudo-random signals. The suitability of the produced models to control each motor separately using the GPC algorithm was tested. The best parameter sets suitable for this task were found to be:

$$\Theta_x = [-1.5879 \ 1.1139 \ -0.4155 \ 3.0820 \ 1.0473 \ 1.6088]$$

$$\Theta_y = [-1.5795 \ 0.7235 \ -0.1565 \ 6.2459 \ -0.2310 \ -0.9802]$$

where

Θ_x is the parameter set for the X-axis motor

Θ_y is the parameter set for the Y-axis motor

Note that the data vector supplied to the RLS algorithm was as follows:

$$x = [-y(t-1) \ -y(t-2) \ -y(t-3) \ u(t-2) \ u(t-3) \ u(t-4)]$$

where $u(t)$ is the control signal applied to the motor and $y(t)$ is the number of encoder counts representing actual position of the carriage.

The test was performed using an interface card with two 16-bit Digital-to-Analog Converters to give the analog output signal to the motors and two 24-bit up-down counters to receive and count the pulses produced by the optical encoders.

The power amplification between the interface card used in this setup and the motors was done using push-pull amplifier circuits.

Several tests has been performed on the machine to test its ability to follow a set point profile. The introduced profiles are shown in Figures 3.23, 3.24 and 3.25.

Each test has been repeated for different values of λ . The set point profiles, λ and results of these tests are listed in Table 3.1.

Set Point Profile	λ	Actual Response	Control Signal
Figure 3.23	500	Figure 3.26a	Figure 3.26b
Figure 3.23	1000	Figure 3.27a	Figure 3.27b
Figure 3.23	1500	Figure 3.28a	Figure 3.28b
Figure 3.24	400	Figure 3.29a	Figure 3.29b
Figure 3.24	500	Figure 3.30a	Figure 3.30b
Figure 3.24	700	Figure 3.31a	Figure 3.31b
Figure 3.25	400	Figure 3.32a	Figure 3.32b
Figure 3.25	500	Figure 3.33a	Figure 3.33b
Figure 3.25	750	Figure 3.34a	Figure 3.34b

Table 3.1 List of Set Point Profiles and their corresponding Responses

The findings in these experiments confirm the above mentioned theoretical effect of λ on the response of the control system and the activity level of the control signal. Nevertheless, the following remarks have to be taken into consideration in order to better understand the results and to explain them:

1. There are mechanical problems in the test machine. Some of these problems can be noticed in the gearing of both motors and in the existence of high static friction. In particular, the Y-axis is affected by these problems.

2. The problem with the static friction introduces a relatively large dead-band for both motors. This dead-band is also dependent on the actual position of the carriage, i.e. the dead-band performance follows a hysteresis pattern.
3. The oscillations that can be noticed on the responses are due to two factors: the first factor is the control signal applied from the GPC algorithm, specifically at relatively low values of λ where the restrictions on changing the control signal are low, and the second one is due to the problems mentioned in points 1 and 2 above. In this case the GPC applies a control signal, but due to the gearing problems and the changing static friction, the response is not smooth and some oscillations can result from this behavior.
4. The operating range of the power amplifiers is limited. Merely 1.5 volts, after the dead-band region, are needed to accelerate the motors from still stand up to the highest speed.
5. The units of X and Y in the actual response figures are encoder counts, where in the plots of the control signals the time is given in samples (1 sample = 0.5 seconds) and u_x and u_y are given in volts.

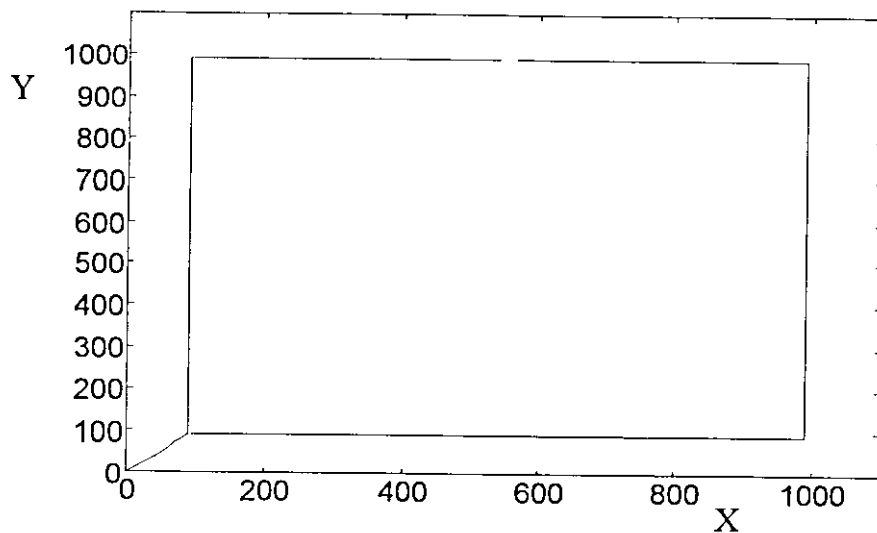


Figure 3.23 Path 1 to be followed by the Proto-type X-Y-Machine

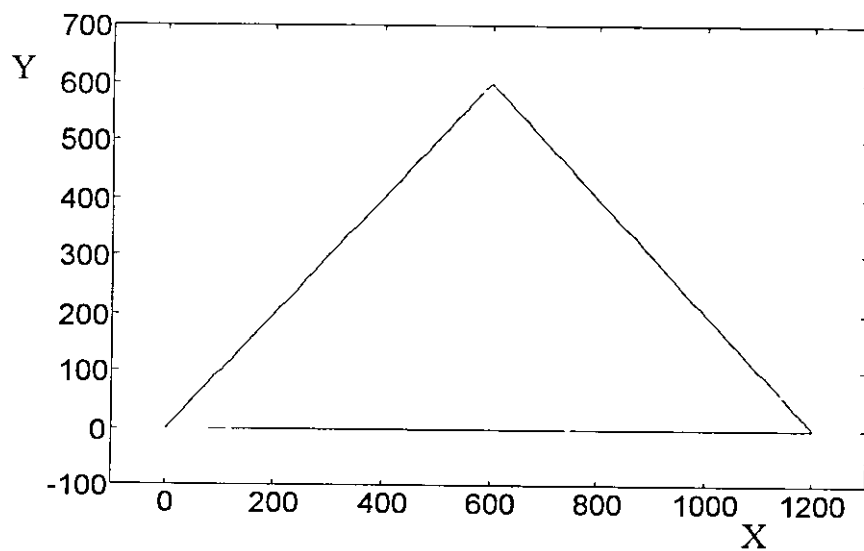


Figure 3.24 Path 2 to be followed by the Proto-type X-Y-Machine

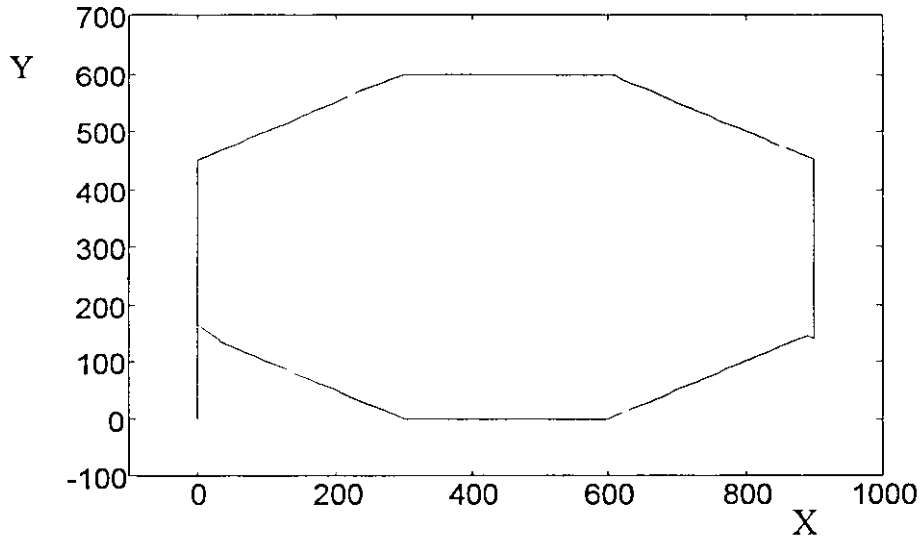


Figure 3.25 Path 3 to be followed by the Proto-type X-Y-Machine

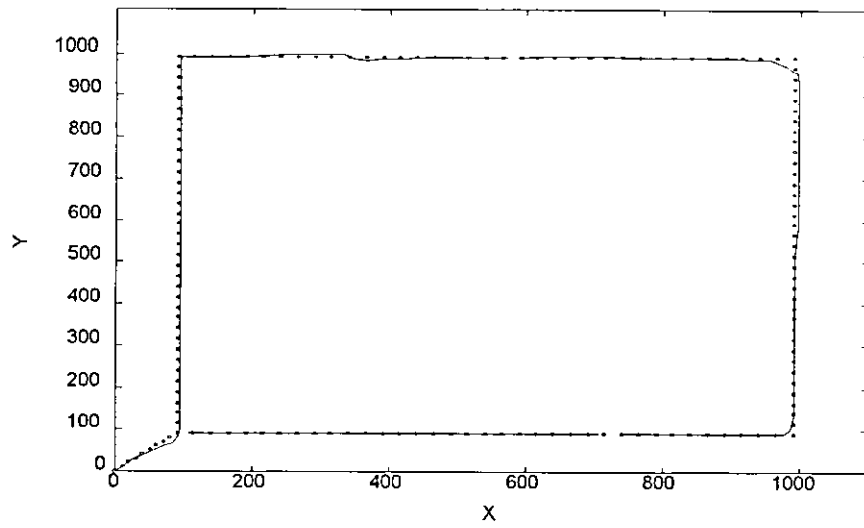


Figure 3.26a Response of the X-Y-Machine

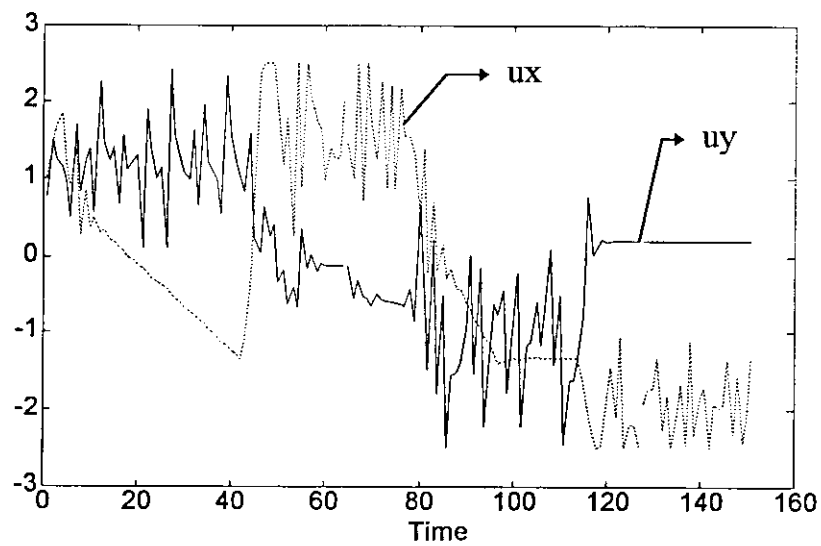


Figure 3.26b Control Signals for the two Motors

Figure 3.26 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=500$

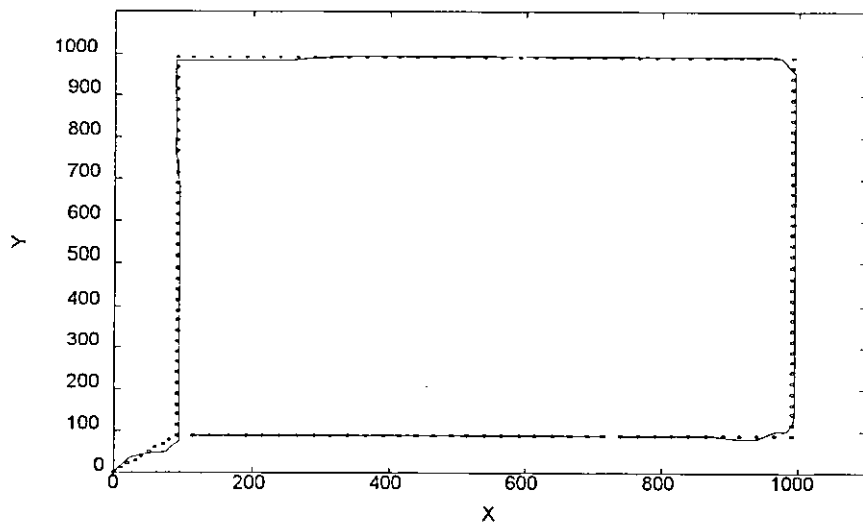


Figure 3.27a Response of the X-Y-Machine

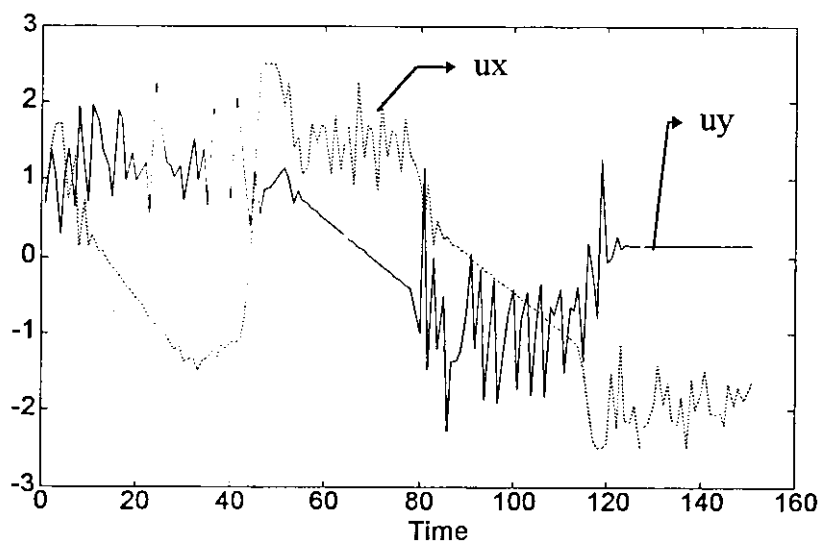


Figure 3.27b Control Signals for the two Motors

Figure 3.27 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=1000$

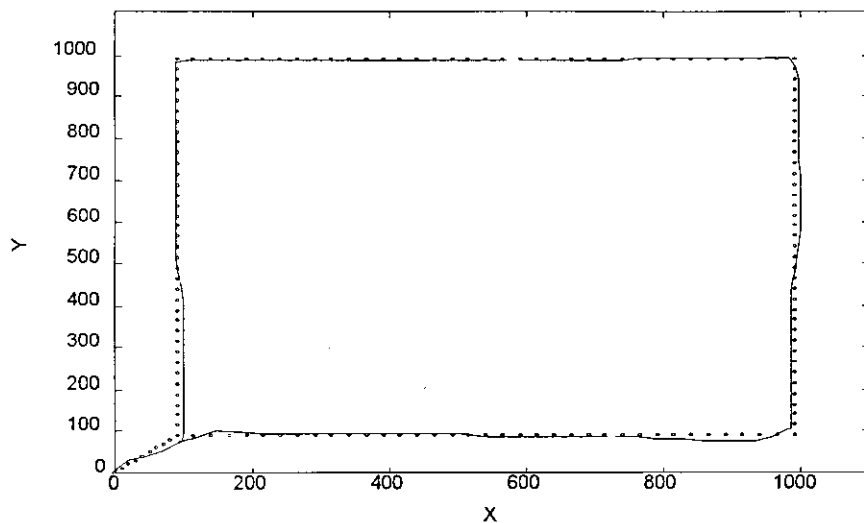


Figure 3.28a Response of the X-Y-Machine

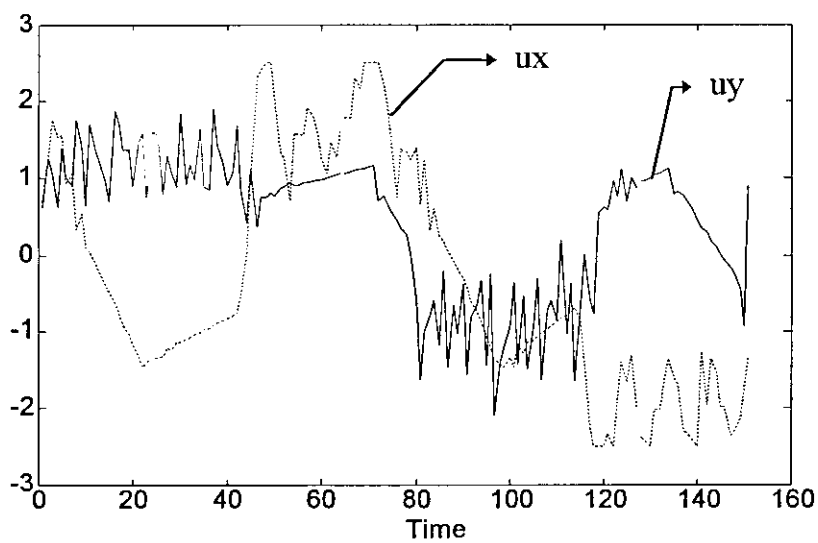


Figure 3.28b Control Signals for the two Motors

Figure 3.28 Performance of the X-Y-Machine in Tracking Path 1 with $\lambda=1500$

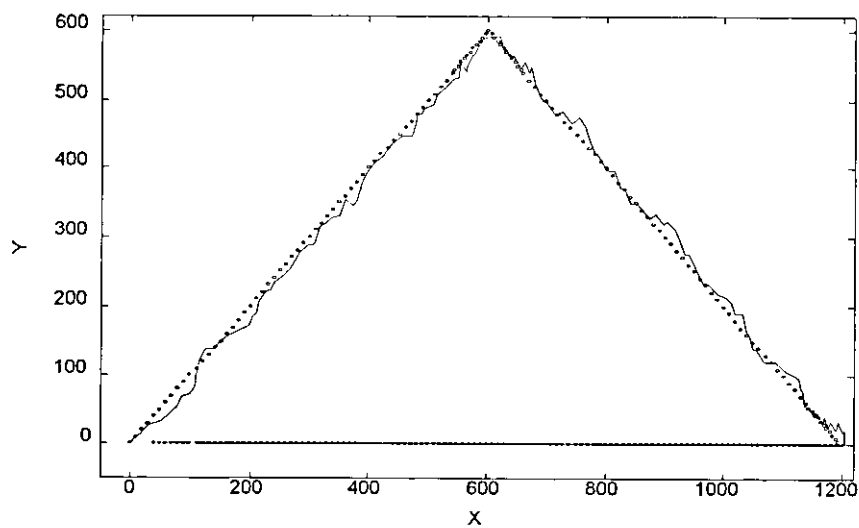


Figure 3.29a Response of the X-Y-Machine

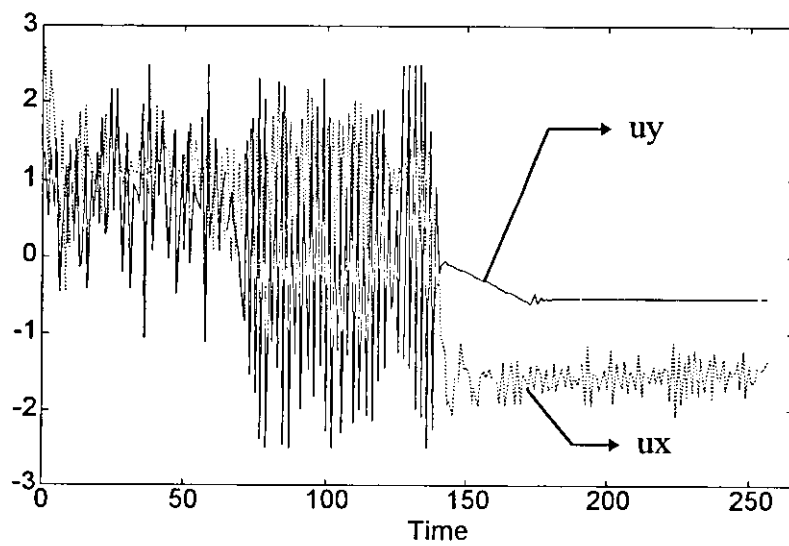


Figure 3.29b Control Signals for the two Motors

Figure 3.29 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=400$

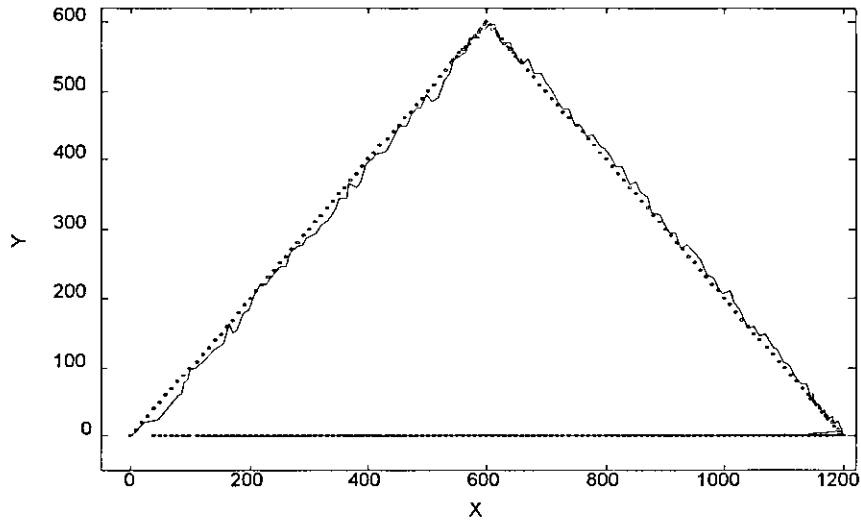


Figure 3.30a Response of the X-Y-Machine

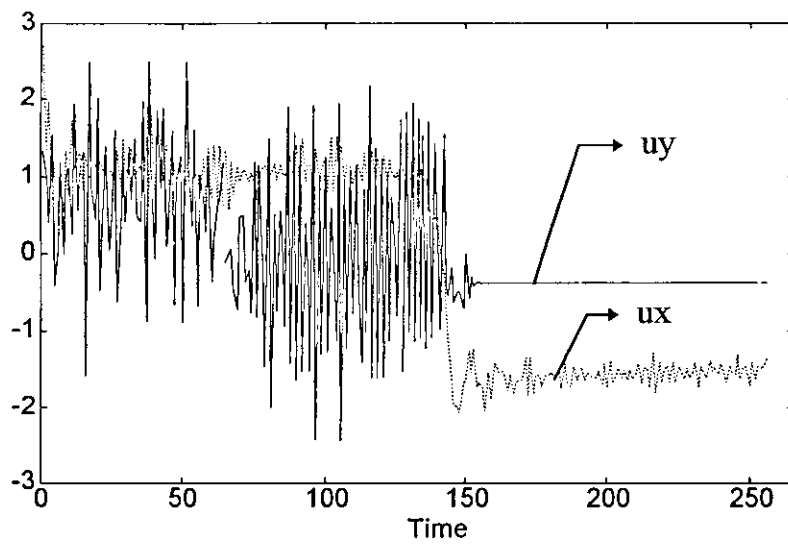


Figure 3.30b Control Signals for the two Motors

Figure 3.30 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=500$

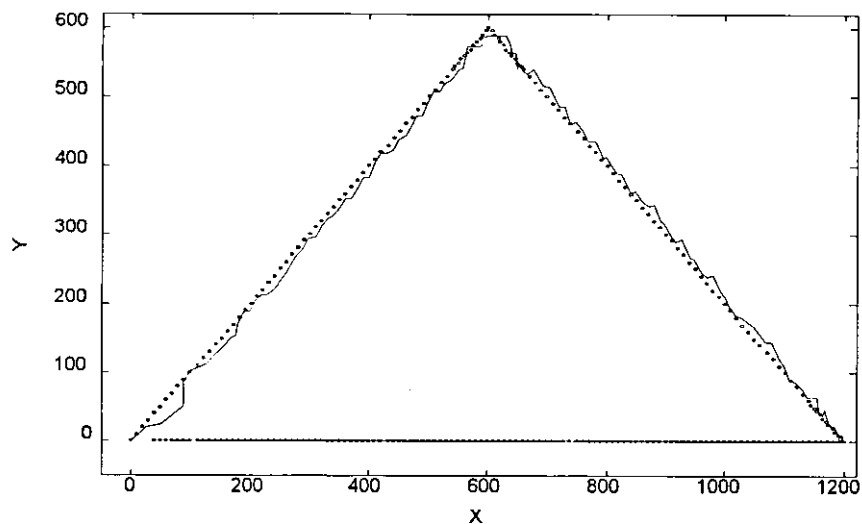


Figure 3.31a Response of the X-Y-Machine

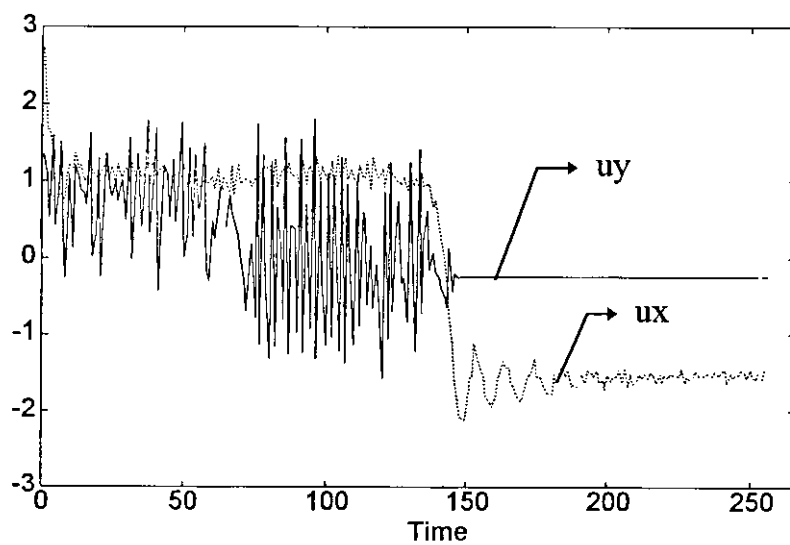


Figure 3.31b Control Signals for the two Motors

Figure 3.31 Performance of the X-Y-Machine in Tracking Path 2 with $\lambda=700$

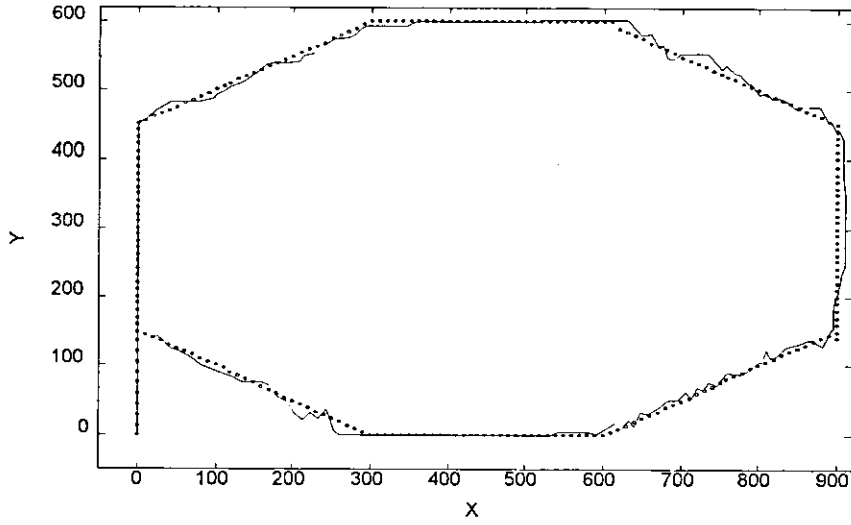


Figure 3.32a Response of the X-Y-Machine

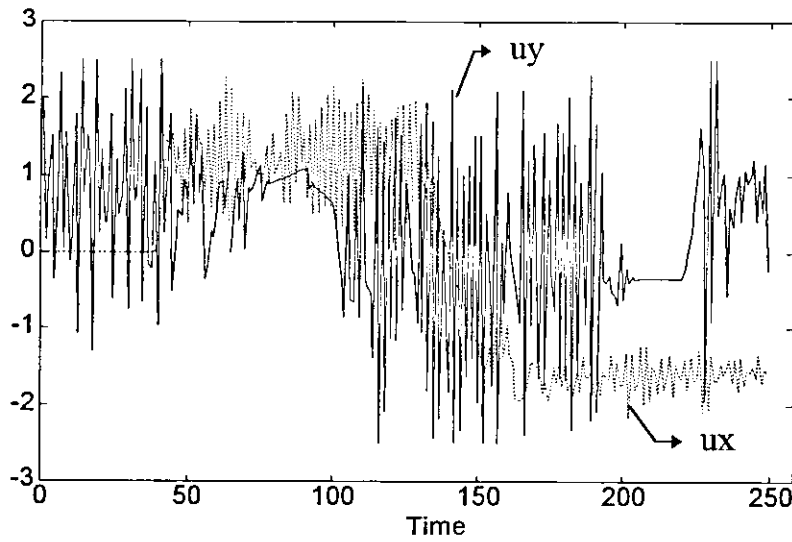


Figure 3.32b Control Signals for the two Motors

Figure 3.32 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=400$

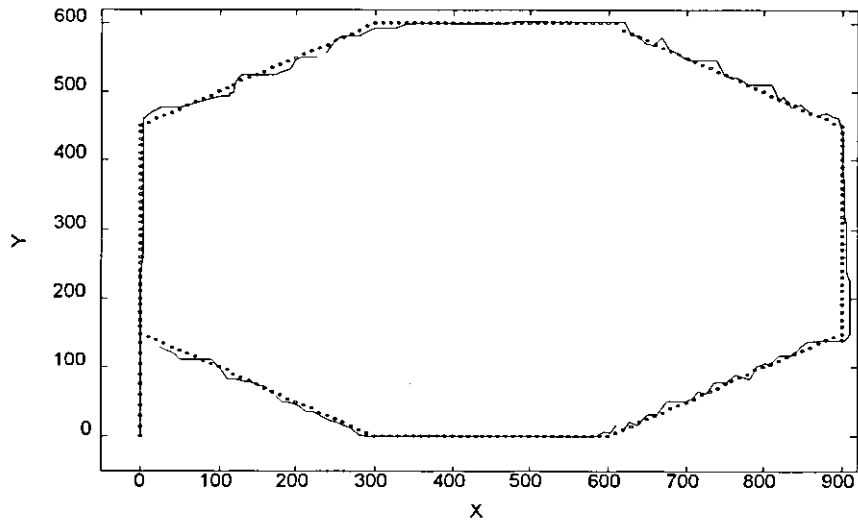


Figure 3.33a Response of the X-Y-Machine

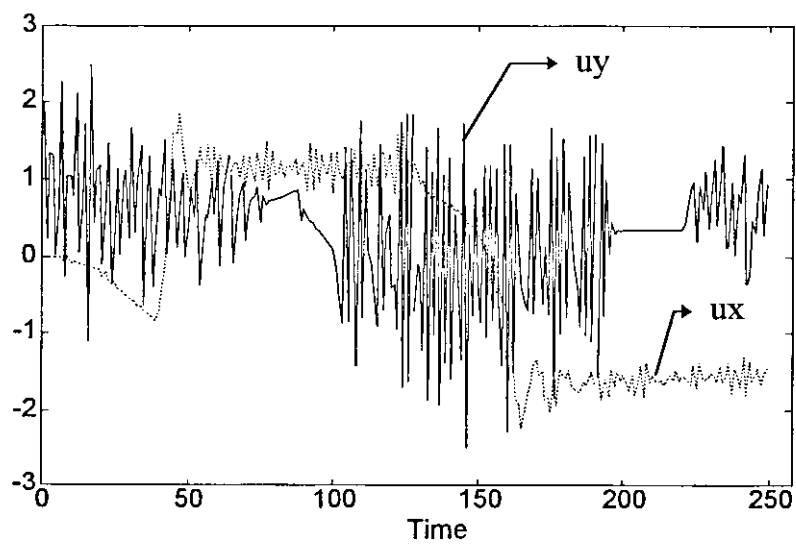


Figure 3.33b Control Signals for the two Motors

Figure 3.33 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=500$

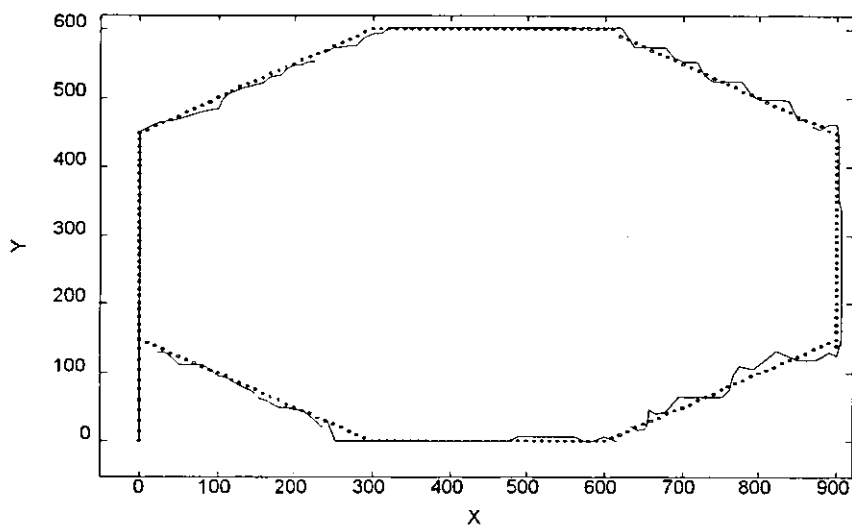


Figure 3.34a Response of the X-Y-Machine

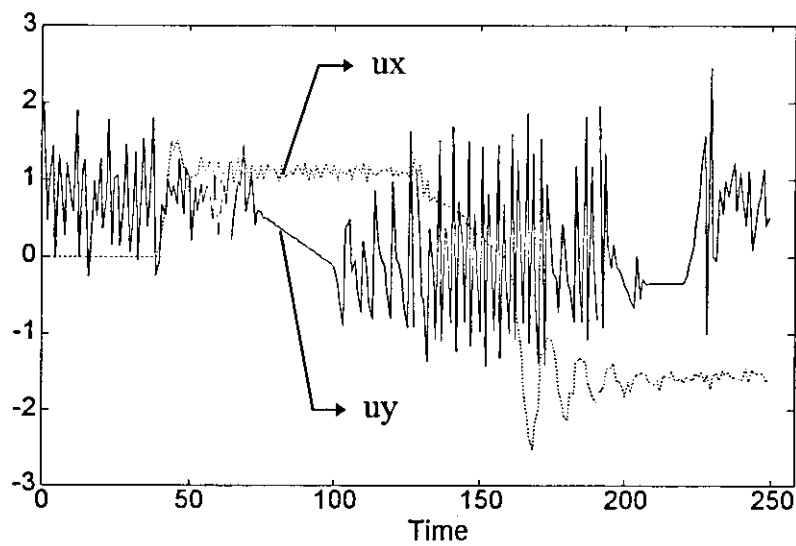


Figure 3.34b Control Signals for the two Motors

Figure 3.34 Performance of the X-Y-Machine in Tracking Path 3 with $\lambda=750$

3.4 Comparing the Performance of GPC and Artificial Neural Networks

The speed engine setup described in section 3.1, has been used in another research that applied Artificial Neural Networks (ANN) to identify the parameters of the system and control the engine speed [10]. A response of the motor speed to a step input is taken in order to compare the results of ANN and GPC. This comparison will not be thorough and is limited to this application. The comparison will be based on two criteria; first the response of the system in both cases and second the effort needed in both systems to achieve that response.

The response of the ANN can be seen in Figure 3.35, where the response of the GPC is shown in Figure 3.5a.

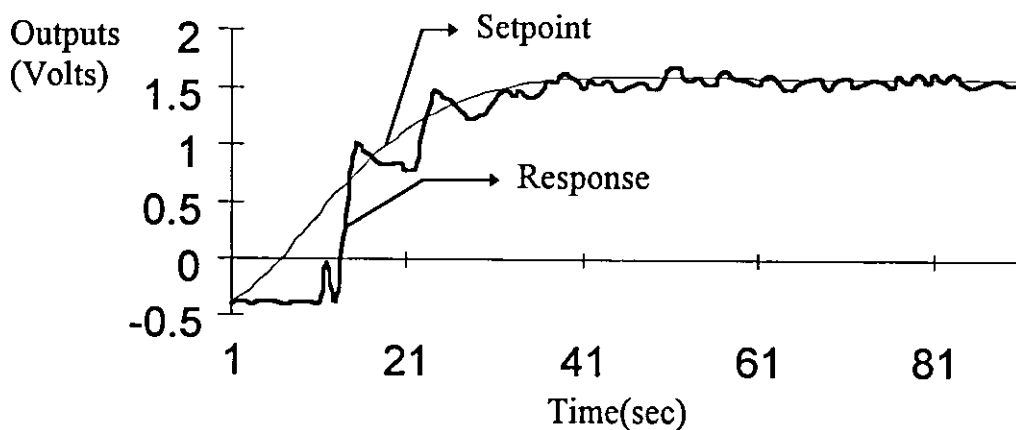


Figure 3.35 Response of the Speed Engine using Artificial Neural Networks

Note here that the step change in the set point was introduced gradually in ANN (Model Reference Control), where in the GPC system it was introduced as a step.

It is noticed that the two responses are comparable regarding the tracking of the set point and output regulation. On the other hand, the effort needed to design, initialize and train the ANN system is greater than that needed in the GPC system. In order to establish a successful ANN structure the following steps have to be performed:

1. Applying a pseudo-random input signal and measuring the output signal, these two signals will be logged and stored in a file or a matrix.
2. The two above mentioned signals are introduced to the ANN to accomplish the off-line learning phase. In this phase the ANN tries to identify the parameters of the process. The performance of the ANN in this phase depends greatly on the number of hidden layers.
3. The calculated parameters are then introduced to the ANN that performs the control of the engine speed. This ANN is not the same as in step 2 and its performance also depends on the number of hidden layers, which has to be chosen by the user.
4. If the error between the set point and the actual response exceeds a certain predetermined threshold value, on-line learning is introduced to the system where the results of the off-line learning will be taken as initial values for the parameters.

The effort needed to design and implement the GPC system is less as can be seen in the previous discussion in Chapter Two.

Chapter Four

Flame Cutting Machine Design

4.1 Introduction

Part of this research is to build a micro computer based flame cutting machine and apply modern techniques and algorithms for the control of the machine and for minimizing the waste produced by the cutting process.

An important point of view to be mentioned here is that the machine will be completely designed and built locally. That means gaining important experience in machine design and building, the availability of spare parts when needed, as well as the possibility of making changes to the machine in the future to suit any coming needs of the end users or any other potential users that might show interest in this machine. The design and build-up of the machine will be conducted in cooperation with two other colleagues

The general specifications and requirements of this machine are:

- Material to be cut: Plain carbon steel.
- Stock type: Standard 2D-sheets and plates (1x2m², 1.25x2.5m² and 1.5x3m²).
- Shapes of products: Mainly 2D basic regular shapes and some irregular shapes.
- Sizes of shapes to be produced: For circles and arcs $\Phi=20-300\text{mm}$, for rectangular and other shapes 50x60mm² up to 2m².

- Accuracy of cut : $\pm 0.5\text{mm}$.

In the phase of the design of the flame cutting machine the modularity concept was the guiding line for constructing the different subsystems of the machine.

The machine is divided into the following systems:

1. The mechanical frame and motion axes.
2. The control system.
3. The flame torch assembly and gas supply.

The following sections will discuss these three systems in more details, presenting their components and subsystems, as well as their functionality.

4.2 The Mechanical Structure

The machine configuration adopted is the *gantry* type, where two orthogonal actuators move the two machine axes relative to each other to position the cutting tool in a plane. This configuration resembles a bridge crane that spans a confined work area, and is typical in industry. A general overview of the designed machine is provided in Figure 4.1.

The frame of the machine has been constructed from standard structural steel sections and parts, which provide the required rigidity and support for other machine members and parts. In addition to their local availability and their relatively low costs, the manufacturing costs are reduced by using these standard parts. U-channel beams are used to carry the long axis guiding rods and other machine members, since they are ideal to support loads and provide sufficient rigidity.

I-section beams are used to carry the short axes assembly, including the carriage on which the cutting head is to be mounted. Twin I-sections are used to provide the required deflection resistance, and carry a maximum tool weight of 50 Kg. This structure would also support thermal stresses that are typical to be induced in the working environment of this machine.

The linear movement of both machine axes is provided through the use of high precision, low backlash, and low friction ball screw-nut assembly. These ball screws raise the efficiency of power transmission to 90% rather than 40% of conventional power screws through their ball-nut sub assembly as shown in Figure 4.2. These assemblies convert the rotational motion provided by the actuators into smooth linear motion. Technical specifications of the selected power screws are available in Appendix C. Rotary bearings are used to mount the ball screws at each end. Self-aligning type is used on the drive side, while a standard ball type is used on the other one.

Both moving axes are carried over and guided by high precision supporting and guiding rods (see Figure 4.3), through linear bearings. They intend to relieve the actuators from directly carrying loads, in addition to the guarantee of smooth running axes with optimum guiding accuracy. The linear bearings used to carry the moving parts of the axes over the guiding rod provide frictionless relative motion. More details about these guiding rods are provided in Appendix C.

The torch carriage is equipped with a “rack and pinion” mechanism to provide the manual height adjustment of the torch. An improvement of this assembly

could be easily made in the future by providing a motorized height adjustment, and swiveling mechanism to provide the capability of rotating the torch so that beveled cuts could also be made by the machine.

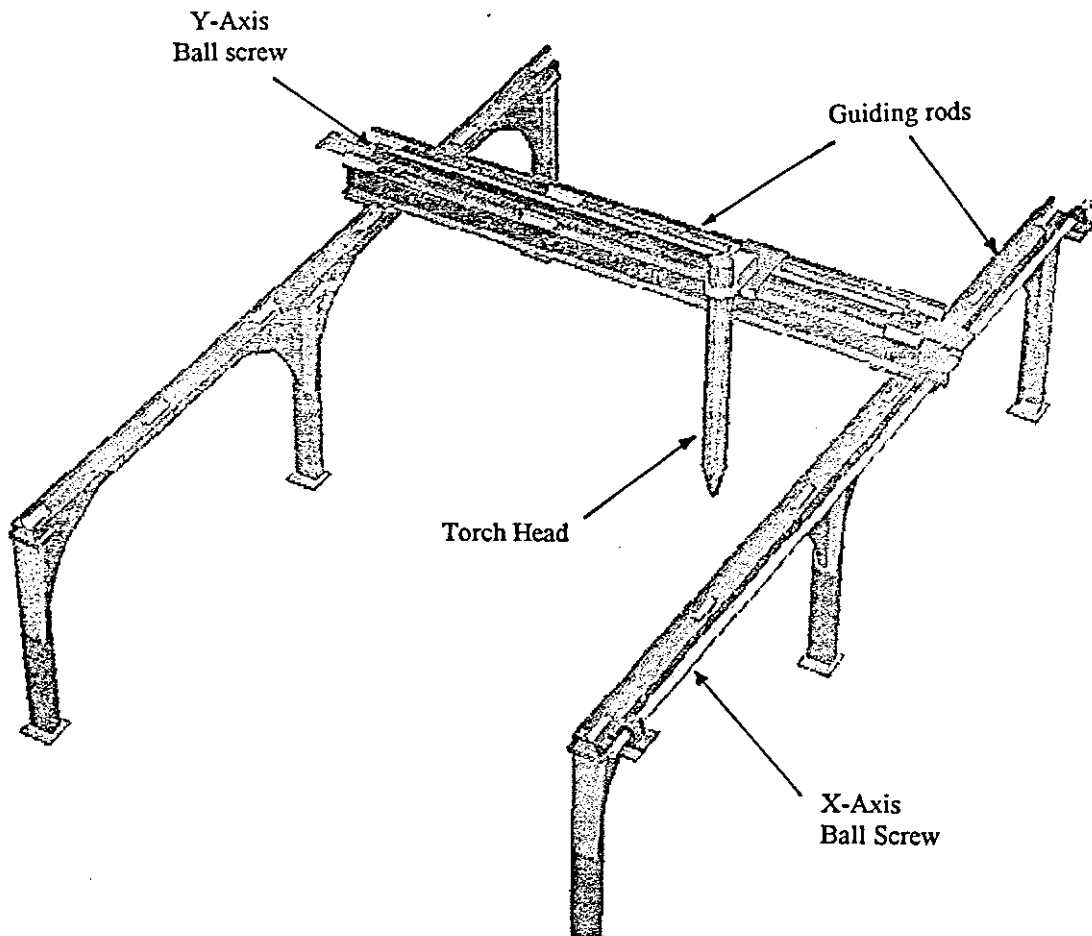


Figure 4.1a General View of the Flame Cutting Machine

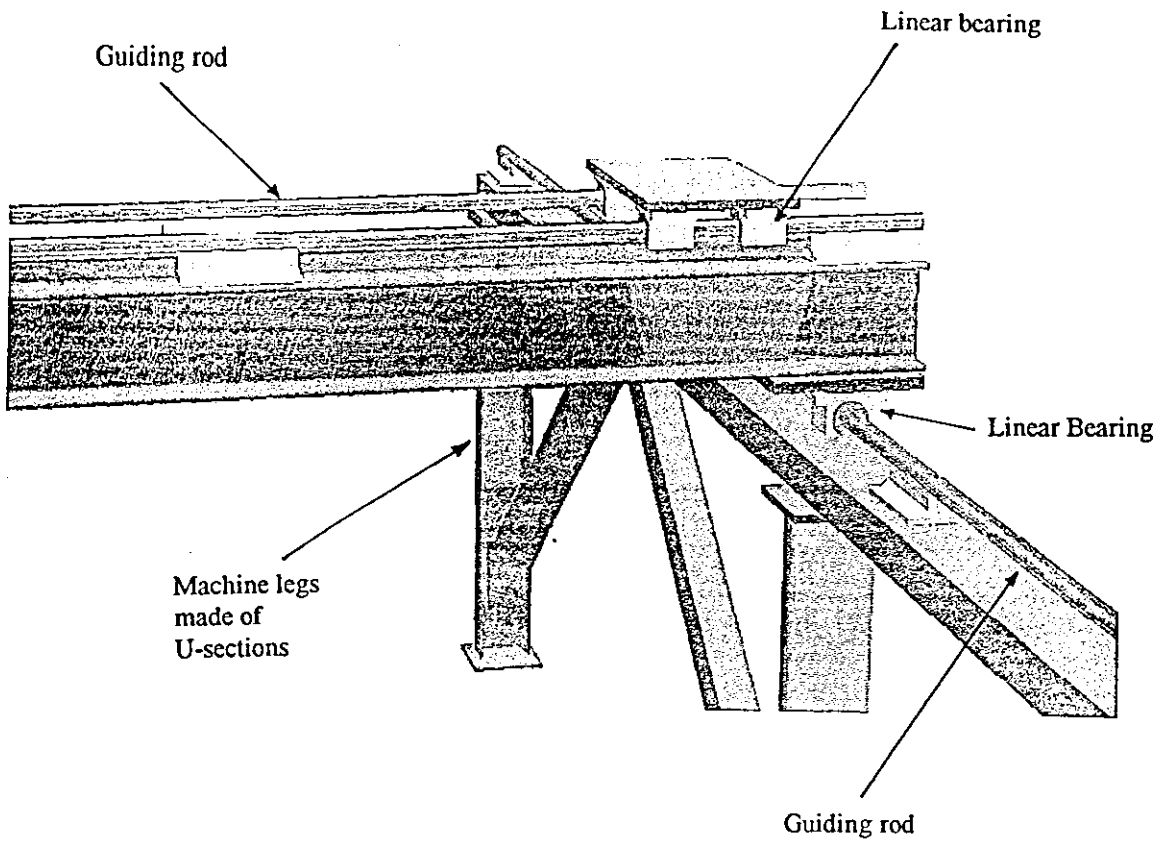


Figure 4.1b Guiding rods and linear bearings of the machine.

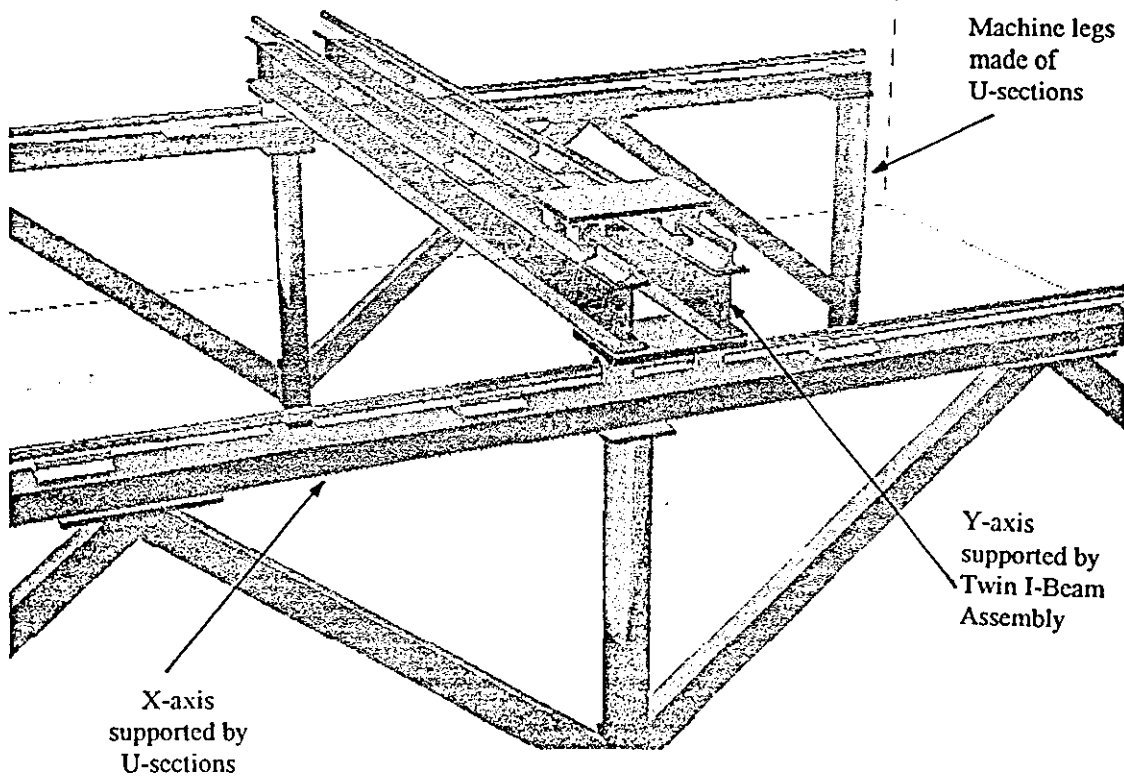


Figure 4.1c Structural Members of the Machine

Figure 4.1 Mechanical System of the Flame Cutting Machine

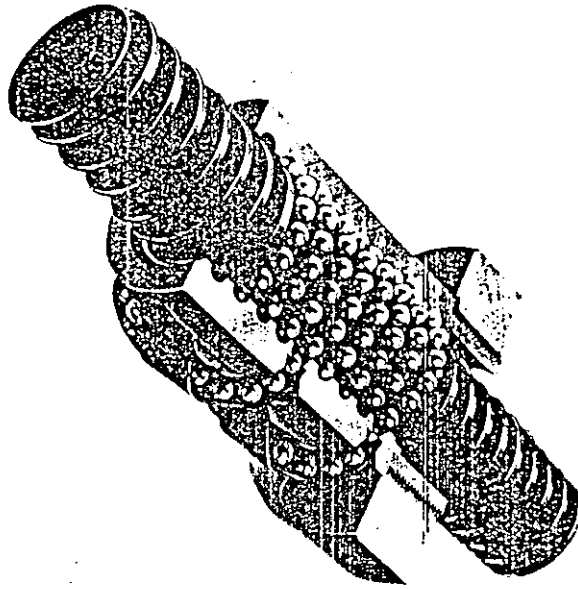


Figure 4.2 Ball Screw- Nut Assembly

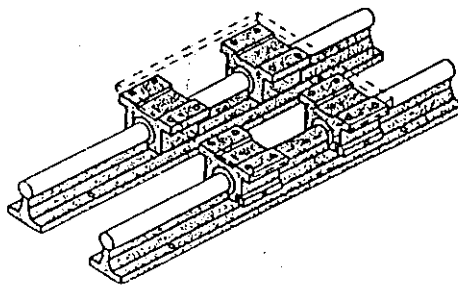


Figure 4.3 The guiding and supporting rods.

4.3 The Control System

The control system for the above described flame cutting machine has two main subsystems. The first one is the hardware and the second is the software. Following is a brief description of these two subsystems, their components and their functions.

4.3.1 Hardware Components :

4.3.1.1 The Personal Computer

The PC that is going to be used is a 80486 based machine with 33 MHz clock speed and 4 mega byte of Random Access Memory (RAM). It is equipped with a Fixed Disk Drive (Hard Disk) and a Floppy Disk Drive. The PC will run the software that will be described later on in this chapter.

4.3.1.2 The Interface Card

This is the card that will handle all the data I/O between the PC and the machine. Full specifications of the card are available in Appendix C. The main features of interest in this card are :

1. Two 12-Bit Digital-To-Analog Converters (DAC) which will convert the digital control signal issued by the software to an analog signal in the range ± 5 VDC. Simple calculations can show that the resolution of these DACs is sufficient for this application. The full speed range of the machine is 75 to 3000 mm/min or 1.95 to 50 mm/sec. If the

variation of the speed can be assumed to have 100 steps to cover the whole speed range, then a resolution of 8 bit would be sufficient.

2. Two 24-Bit Up-Down Counters that will receive the signals from the feedback devices of the flame cutting machine (Optical Encoders). The x-axis span is 3000 mm, which results in 6000 Basic Length Units (BLU), taking the required accuracy of ± 0.5 mm into account, i.e. a 16-bit counter should satisfy the needs of the feedback system.
3. Thirty two Digital I/O Lines. Some of these lines will be used as inputs, where some others will be used as outputs. In Table 4.1, Section 4.3.1.5, more details about these interlocks are provided.

4.3.1.3 Power Amplifier Unit

The input of this unit will be a PC signal (± 5 VDC, 4 to 20 mA). This signal will be transmitted to the output using the above described interface card. The output will be the drive signal for the motors (± 50 VDC, 4A). This unit is designed and manufactured by a local firm.

4.3.1.4 Optical Encoders

The encoders will provide feedback signals for the PC. Each encoder supplies two pulse train signals, between which there is a 90° phase shift. This will enable the controller to detect both position as well as direction of movement of the carriage. The following figure illustrates this connection and the phase shift between the two signal. The counter counts the 0-to-1 transitions on the clock

input, where the state of the gate input selects between up and down counting. If the shift is $+90^\circ$ then signal B is always High when the 0-to-1 transition takes place on the clock input. If the phase is -90° ,i.e., when the direction of movement is reversed then signal B is Low when the transition takes place.

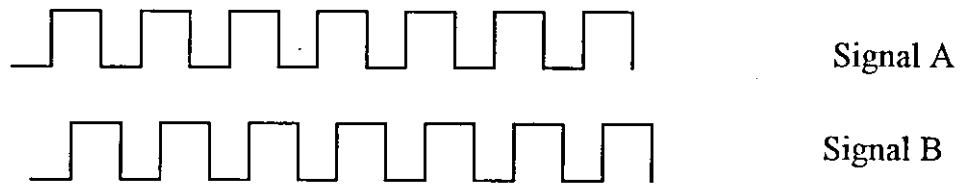


Figure 4.4 The 90° Phase Shift between the Encoder Signals

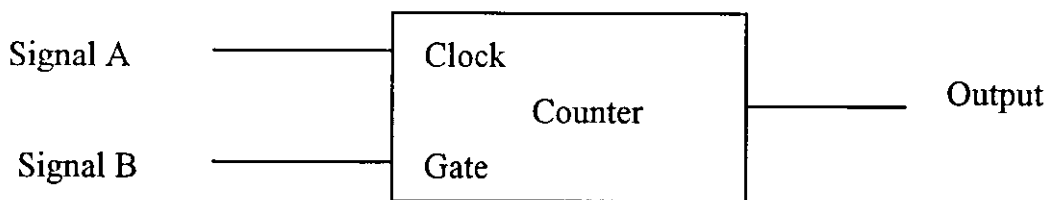


Figure 4.5 Connection of the Encoder Signals to the Counter

4.3.1.5 Digital Interlocks

The following table shows the number of digital interlocks that are built in the machine, their type and their functions. As noticed all of them serve to protect the machine against damage resulting from improper operation, or driving the machine beyond its operational limits.

Description of the Signal	Type (Input/Output)	Number of Signals	Source/Destination
Oxygen supply valve	Output	1	Solenoid valve on the oxygen supply hose.
Acetylene supply valve	Output	1	Solenoid valve on the Acetylene supply hose.
Cutting gas supply valve	Output	1	Solenoid valve on the cutting gas supply hose.
Ignition signal	Output	1	Spark plug
Ready/Operating signal	Output	1	Green light on operator desk.
Alarm signal	Output	1	Visual and audio alarm equipment
Emergency stop	Input	1	Emergency stop buttons
X-Axis end-of-travel limits	Input	2	Limit switches at the two ends of the X-Axis
Y-Axis end-of-travel limits	Input	2	Limit switches at the two ends of the Y-Axis

Table 4.1 Digital Interlocks of the Flame Cutting Machine

4.3.1.6 Actuators

DC-Servo motors were selected as actuators for this machine type. They are being widely used in similar applications in the industry. In light of the power requirements for this machine, that are detailed later on in this section, permanent magnet DC-motors were selected for this application. Their control using a PC is

relatively easy and they can meet the requirements of the control system. The following specifications were determined in the design phase to meet the requirements and specifications of the cutting process:

Speed : The maximum linear speed that the machine should travel at is 50 mm/s. Using a power screw with a 6 mm pitch as described in section 4.2 the maximum output speed of the motors should be 500 rpm.

Torque : According to a maximum carriage weight of 50 kgs, and taking into account other structural elements that should be moved by the motors and their inertia, the required torque to be generated by the motors was found to be 1.3 Nm for the X-Axis and 1.1 Nm for the Y-Axis.

Taking the above requirements into consideration, the nearest standard motors that could be found were of the following specifications:

For the X-Axis motor : Speed 480 rpm, torque 2.0 Nm.

For the Y-Axis motor : Speed 480 rpm, torque 1.2 Nm.

4.3.2 Software Components

The main functions that the software should perform for the operation of the machine can be seen in the following list:

1. Provide a user-friendly, graphical user interface for the operator.
2. Optimize the layout of the shapes on the sheet.
3. Generate the tool path for the machine.
4. Control the movement of the machine.

4.3.2.1 The User Interface

The main function of this module is to enable the operator to draw or select the shapes that are going to be cut, and to input the thickness of the stock sheet. To simplify the process of drawing a shape or selecting a pre-drawn shape from a library, the general purpose and commonly used drafting package AutoCAD® release 12 is going to be used as the user interface. The operator will draw the shapes (or select them) and generate a file of the Data Exchange File (DXF) format. This file will be used as the input for the layout optimization module.

4.3.2.2 The Layout Optimization Module

This module is used to minimize the waste produced by the cutting process by optimizing the layout of the shapes on the stock sheet. This module applies some Artificial Intelligence (AI) techniques to perform this task [11]. The input of this module is, as mentioned above, is the DXF file of the user drawn shapes generated by AutoCAD®. The output of this module will be also a DXF file containing the same shapes after relocating them to minimize the scrap percentage. The user has an option to accept this new layout as is or to make modifications on it using AutoCAD®, which implies a certain amount of recursion between user interface module and the optimization module. The layout optimization module can be called within the AutoCAD® environment.

4.3.2.3 The Tool Path Generator

This module will have the entities of the DXF file generated after points 1 and 2 above, and the thickness of the plate to be cut as inputs. The thickness will serve to decide on the minimum and maximum speed of cut to be applied to cut sheets of this thickness using a look-up table. Using the speed constraints and the entities of the DXF file this module will generate a set of point coordinates suitable for the controller module. It will also have an indication for the control module whether the flame should be on or off during a travel distance, i.e., will distinguish between a cutting line or a rapid traverse line between two shapes.

4.3.2.4 The Control Module

This module is responsible for the accurate tracking of the shapes to be cut by an accurate control of the position and speed of the cutting torch. The control algorithm applied here is a Self-Tuning Generalized Predictive Controller (Self-Tuning GPC). This module will also monitor the state of the digital interlocks in order to take certain actions when necessary, such as generating alarms or stopping the machine in case of emergency. The algorithms used in this module were presented, explained and tested in Chapters 2 and 3 of this work.

4.4 The Flame Torch Assembly and Gas Supply

The subsystems described in sections 4.2 and 4.3 constitute an X-Y-motion system. From modularity point of view, different cutting heads representing different cutting technologies can be installed on that motion system. Examples for that are water jet cutting machines or laser cutting machines. The Oxy-Acetylene cutting technology, suitable for cutting plane carbon steel sheets and plates, is used in this project. Therefore, a suitable head assembly has been selected. This head assembly consists of a straight-headed torch with the tip in line with the torch axis. A kit of different cutting tips (nozzles) is provided to suit a variety of sheet metal thicknesses. More details on this head assembly can be found in Appendix C.

The gases needed for this process, namely cutting oxygen, heating oxygen and acetylene, are supplied from three gas cylinders through a set of regulators and their flow is controlled using solenoid valves, which are controlled digitally from the controller module. Safety valves are also provided on the gas supply lines to prevent back fire.

The flame cutting machine, whose design was presented in this chapter, is currently in the phase of assembly. Some of the standard parts described above had to be imported from abroad, and therefore, considerable delays had to be taken into account.

Chapter Five

Conclusions

The work presented in this thesis tackled some of the problems faced in the position control of the flame cutting process. It was necessary to have an adjustable speed while controlling the position of the flame in order to achieve an optimal combination of these two process variables to ensure clean and defined cuts.

The study introduced the application of an advanced control algorithm, the Self-Tuning Generalized Predictive Controller (GPC), to the control of X-Y-Machines, represented here by a flame cutting machine. The advantages of self-tuning control algorithms over fixed term controllers such as the PID were investigated and demonstrated on hand of simulation examples. In order for the GPC to perform in a good manner, its parameters have to be understood and properly tuned. The effect of these parameters was demonstrated and analyzed.

It was also shown that the identification algorithm applied to identify the parameters of a process plays a vital role in the controller behavior. The control algorithm will depend on the parameters supplied by the identification algorithm. Therefore, the tuning knobs of one of the mostly applied algorithms, the Recursive Least Square (RLS) algorithm, were studied and analyzed. It was shown that these knobs of RLS have to be set properly and according to the

process properties, in order to produce adequate identification of process parameters.

The study presented a design for a flame cutting machine. The first action to be taken in future work is to actually build this machine and apply the above mentioned algorithms and investigate their suitability in a practical environment where the criteria of performance are set high, since this machine will work in a real industrial environment and will be used to cut shapes that will be part of the final product of an industrial firm.

References

- [1] John G. Bollinger and Neil A. Duffie, "Computer Control of Machines and Processes", Addison Wesley Publishing Company, Reading 1988.
- [2] K. Ogata, "Modern Control Engineering", 2nd Edition, Prentice Hall, New Jersey, 1990.
- [3] P. E. Wellstead and M. B. Zarrop, "Self-Tuning Systems: Control and Signal Processing", John Wiley and Sons, Chichester, 1991
- [4] D. W. Clarke, "Self-Tuning Controller Design and Implementation", OUEL Report No. 1535/84, University of Oxford, 1984.
- [5] Y. Al-Assaf, "Self-Tuning Control : Theory and Applications", D.Phil. Thesis, University of Oxford, 1988.
- [6] N. Abdel-Jalil, "Computer Simulation of Adaptive Control for Thermoplastics Injection Molding Process", M.Sc. Thesis, University of Jordan, 1992.
- [7] K. Ogata, "Discrete-Time Control Systems", Prentice Hall, New Jersey 1987.
- [8] D. W. Clarke, C. Mohtadi and P. S. Tuffs, "Generalized Predictive Control, 2 Parts: The Basic Algorithm", OUEL Report 1555/84, University of Oxford, 1984.
- [9] P. E. Wellstead, "Engine Speed Control Apparatus", TecQuipment, Nottingham, 1981.

- [10]R. Hirzallah, "Process Identification and Control using Neural Networks", Internal Report, University of Jordan, 1996.
- [11]R. Al-Naqa, "Application of Artificial Intelligence Techniques for Metal Cutting ", Internal Report, University of Jordan, 1996.

Appendix A

Program Listing for RLS Algorithm


```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% APPENDIX A
% Program for the test of the RLS algorithm
% Program name RLS_FF.M
% Written by : Fuad Bajjali
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;
sigma=2;
cov=input('Initial Value for the P matrix : ');
beta=input ('Enter forgetting factor : ');
num=1000;
th1=zeros(1,num);
th2=zeros(1,num);
th3=zeros(1,num);
th4=zeros(1,num);
ch=200;
p=zeros(3);
p=eye(3)*cov;
theta=[1 1 1]';
y=zeros(1,num);
yy=zeros(1,num);
e=zeros(1,num);
for j=1:400
    u(j)=rand-0.5;
end;
for j=401:800
    u(j)=0.1;
end;
for j=801:num
    u(j)=rand-0.5;
end;
eps=0;
for t=3:num
    e(t)=eps;
    if (t<ch)
        thy=[-1 0.75 0.5];
        y(t)=thy(1)*y(t-1)+thy(2)*u(t-1)+thy(3)*u(t-2)+e(t);
        % y(t)=-0.5*y(t-1)+1.6*u(t-1)+u(t-2);
    else
        thy=[-1 1.3 0.5];
        y(t)=thy(1)*y(t-1)+thy(2)*u(t-1)+thy(3)*u(t-2)+e(t);
        % y(t)=-1*y(t-1)+0.75*u(t-1)+2*u(t-2);
    end
    x=[y(t-1) u(t-1) u(t-2)];
    eps=y(t)-x*theta;
    k=p*x'/(beta+x*p*x');

```

```

theta=theta+k*eps;
p=(p-k*(p*x'))/beta;
yy(t)=theta(1)*y(t-1)+theta(2)*u(t-1)+theta(3)*u(t-2);
beta=1-((1-x*k)/sigma)*(eps^2);
if beta>1
    beta=1;
end;
if beta<0.7
    beta=0.7;
end;
tt(t)=t;
th1(t)=theta(1);
th_1(t)=thy(1);

th2(t)=theta(2);
th_2(t)=thy(2);

th3(t)=theta(3);
th_3(t)=thy(3);

% th4(t)=theta(4);
end;

figure(1);
plot(tt,th1,'k:',tt,th_1,'k-');
title('Parameter a=-0.5');
axis('manual');
axis([1 num min(th1)-0.5 max(th1)+0.5]);

figure(2);
plot(tt,th2,'k:',tt,th_2,'k-');
title('Parameter b0=1.6');
axis('manual');
axis([1 num min(th2)-0.5 max(th2)+0.5]);

figure(3);
plot(tt,th3,'k:',tt,th_3,'k-');
title('Parameter b1=1.0');
axis('manual');
axis([1 num min(th3)-0.5 max(th3)+0.5]);

figure(4)
plot(tt,th1,'k',tt,th_1,'k:',tt,th2,'r',tt,th_2,'r:',tt,th3,'b',tt,th_3,'b:');
axis('manual');
axis([1 num -1.5 2.5]);

```

Appendix B

Program Listing for GPC Algorithm

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% APPENDIX B
% Program for the test of the GPC algorithm
% Program name XCONT.M
% Written by : Fuad Bajjali
% This program calls the function
% GPC_FUN.M
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear
load('test.mat');
num=size(yin,1)-200;

LAM=0.1;
NU=1;           % Control Horizon
N1=1;N2=1;     % Minimum and maximum horizons
y=zeros(1,num); % the y output signal
yy=zeros(1,num); % the y output signal (predicted)
uy=zeros(1,num); % the y control signal
wy=zeros(1,num+N2); % the y preset points

N2=input ('Enter maximum horizon (N2)  : ');
NU=input ('Enter control horizon (NU)  : ');
LAM=input ('Enter Weighting factor LAMBDA : ');

% generating the set point sequence (w)
for i=1:size(yin)
    wy(i)=yin(i);
end;

for i=1:N2+25
    wy(num+i)=wy(num); % N2 samples ahead
end

thy=[-0.9 0.1 0.2];

% main loop
for t=3:num
    % if (t> 75) LAM=0.1; end;
    % if (t>175) LAM=0.25; end;
    % if (t>275) LAM=0.5; end;
    % if (t>375) LAM=1;end;

    y(t)=-thy(1)*y(t-1) +thy(2)*uy(t-1) +thy(3)*uy(t-2);

% GPC---> u --->call process y(t)=with new x

```

```

na=1; nb=2;
for i=1:N2
    WWy(i)=wy(t+i); % N2 samples ahead
end
% Calling the GPC controller function %
duy=gpc_fun(na,nb,thy,y,uy,WWy,t,NU,N1,N2,LAM);
uy(t)=uy(t-1)+duy;
tt(t)=t;
wiy(t)=wy(t);
end % end of the t loop

figure(1);
plot(tt,y,'k:',tt,wiy,'k');
title('set point solid / actual in dotted');
axis('manual');
axis([2 520 min(y)-1 max(y)+1]);

figure(2);
plot(uy,'k');
title('Control signal uy');
axis('manual');
axis([2 520 min(uy)-1 max(uy)+1]);

```

```

%%%%%%%%%%
% GPC Controller Function
% Written by : Fuad Bajjali
% called by XCONT.M
%%%%%%%%%%

```

```
function [du]=gpc_fun(NA,NB,THETA,YY,UU,W,t,NU,N1,N2,LAM);
```

```

F=zeros(NA+1,N2);F1=zeros(1,NA+1);
PF=zeros(1,N2);
I1=zeros(NU,NU);I2=zeros(NU,NU);
E=zeros(1,N2);
G=zeros(1,N2);G1=zeros(1,N2);
GM1=zeros(N2,N2); GM=zeros(N2,NU);
G2=zeros(1,NB);GB=zeros(NB,N2);
GG=zeros(NU,NU);GT=zeros(NU,N2);GTG=zeros(NU,NU);
GG1=zeros(NU,NU);GG2=zeros(NU,N2);GUD=zeros(1,N2);FYD=zeros(1,N2);

```

```
D=[1 -1]; % Delta =1-z^-1
```

```
A(1)=1;
```

```
A(2)=THETA(1);
```

```
B(1)=THETA(2);
```

```
B(2)=THETA(3);
```

```
AD=conv(A,D);
```

```
%
```

```
% Recursion of the Diophantite equations%
```

```
%%
```

```
G(1)=B(1);
```

```
E(1)=1;
```

```
for i=1:NA+1
```

```
  F(i,1)=-AD(i+1)*E(1);
```

```
end
```

```
for j=1:NB
```

```
  GB(j,1)=B(j);
```

```
end
```

```
for j=2:N2
```

```
  E(j)=F(1,j-1);
```

```
  G(j)=E(j)*B(1)+GB(1,j-1);
```

```

for i=1:NA
    F(i,j)=F(i+1,j-1)-AD(i+1)*E(j);
end
F(NA+1,j)=-AD(NA+2)*E(j);

for k=1:NB-1
    GB(k,j)=E(j)*B(k+1)+GB(k+1,j-1);
end
end
%%%%%%%%%%
% End of recursion%
%%%%%%%%%%

for j=1:N2
    for i=1:j
        G1(i)=G(j-i+1);
        GM1(j,i)=G1(i);
    end

    for k=1:NB-1
        G2(k)=GB(k,j);
    end

    for n=1:NA+1
        F1(n)=F(n,j);
    end

    %%%%%%%%%%%
    % Parameters of the F Vector%
    %%%%%%%%%%%
    GU=0;
    for m=1:NB-1
        GU=GU+G2(m)*(UU(t-1)-UU(t-2));
    end
    GUD(j)=GU;

%
FY=0;
for m=1:NA+1
    l=m-1;
    FY=FY+F1(m)*(YY(t-l));
end
FYD(j)=FY;

%%%%%%%%%%
end % Of the main loop

```

```

%%%%%%%%%
for I=N1:N2,
  for J=1:NU,
    GM(I-N1+1,J)=GM1(I,J);
  end
end

GT=GM';
GTG=GT*GM;
for I=1:NU,
  for J=1:NU,
    I1(I,J)=0;
    I1(I,I)=1;
  end
end
I2=LAM*I1;
GG=I2+GTG;
GG1=inv(GG);
GG2=GG1*GT;
PF=W-(GUD+FYD);
DU1=GG2*PF';
du=DU1(1);
end

```


Appendix C

Data Sheets of Machine Components

Appendix C:

Data Sheets of Major Machine Components

STANDARD BALL NUT ASSEMBLIES

SEE PAGES 5 & 6 FOR MATCHING SCREWS

(SEE PAGE 8 FOR LARGER SIZES)

For use with the Standard and Precision Rolled-Hardened Ball Screws shown on pages 5-6. Nuts are hardened to Rc 56-62. Stainless Steel Nuts are hardened to Rc 40-45.



SCREW SIZE		STANDARD BALL NUTS											FLANGE	
Nom. Dia.	Lead	Ball Nut Part No.	Type	CAPACITY		DIMENSIONS					BALL SIZE & QUAN.		TORQUE	Fits Flange Part No. Page 28
				Dynamic Load for 10 ⁶ Inches Life (Lbs.)	Max Static Load (Lbs.)	Outside Dia. D	Radius Over Tube R	Overall Length L	Mounting Thread Size T	Thread Length TL	Nominal Diameter	Quantity (±5%)	To Raise 1 Pound Lbs. in.	
3/8	.125	R5-10-2	1	130	1,300	.750	.460	1.00	664-32	.250	1/16	62	.021 Lbs. in.	R10-3
		R-11-2		260	2,600			1.875				124		R10-3
		*R5-15-2		25	230			1.00				62		*R15-3
		*R-16-2		50	460			1.875				124		*R15-3
1/2	.500	R5-20-2	3	700	3,900	1.062	.670	1.75	937-16	.375	1/8	70	.057 Lbs. in.	R30-3
		*R5-21-2		135	725			.670				1.75		70
5/8	.200	R-30-2	1	725	5,600	1.0 sq.	.780	1.71	937-16	.500	1/8	67	.035 Lbs. in.	R30-3
		R5-30-2		140	1,150	1.375	.80							*SR30-3
		*SR5-30-2		140	1,150	1.375	.80							*SR30-3
	.200 LH	R-31-2	2	725	5,600	1.0 sq.	.780	1.71	937-16	.500	1/8	67	.035 Lbs. in.	R30-3
		*R-31-2		140	1,150	1.375	.80							*SR30-3
		*SR-31-2		140	1,150	1.375	.80							*SR30-3
.200	R530A-2	1	1,300	10,000	1.375	.80	2.75	1.125-16	.500	1/8	165	.035 Lbs. in.	R30-3	
	*SR530A-2	1	250	2,000	1.375	.80	2.75						*SR30-3	
3/4	.230	R-34-2	1	1,900	18,800	1.250	.875	2.875	1.125-16	.500	1/8	64	.035 Lbs. in.	R35-3
		R-35-2		950	9,400			1.875						R35-3
		*R5-35-2		950	9,400			1.375						*R36-3
		*R5-36-2		160	1,350			1.875						*R36-3
	.500	R5-37-2	3	3,150	18,500	1.312	1.00	2.937	1.250-16	.500	5/32	150	.053 Lbs. in.	R37-3
		*R5-38-2		570	3,950			2.937						150
1 (More on Page 10)	.250	R-40-2	2	1,500	13,000	1.5 sq.	1.150	2.347	1.563-16	.600	5/32	85	.043 Lbs. in.	R40-3
		*R-40-2		290	2,500			3.00						*SR40-3
		R-40A-2		3,000	26,000			1.625						1.107
	.250 LH	R-41-2	2	1,500	13,000	1.5 sq.	1.150	2.347	1.563-16	.600	5/32	85	.043 Lbs. in.	R40-3
		*R-41-2		290	2,500			3.00						*SR40-3
		R-41C2-2		3,000	26,000			1.625						1.107

* Denotes Stainless Steel ** Note: Ball sizes may vary from nut to nut, but not in any one nut.

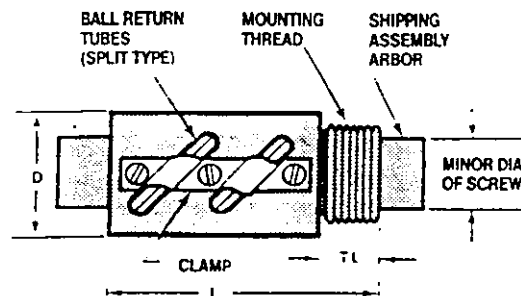
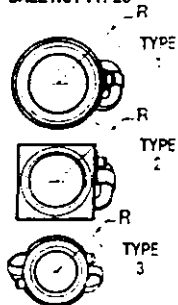
Never mix balls from one nut to another. Use only factory-approved balls.

† Denotes Hard Chrome Plate with Stainless Steel Balls.

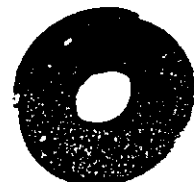
Dynamic load ratings are based on a lubricated assembly with a suitable lubricant. Normal ballnut backlash is .004 to .015" may be reduced to .002" on special order. Ball nuts are shipped on a shipping-assembly arbor unless assembly to the ball screw is requested. Specify direction of ball nut if screw ends are machined.

NSK NO. 1 GREASE IS RECOMMENDED FOR THESE ASSEMBLIES (STOCK ITEM).

BALL NUT TYPES



OPTIONAL NYLON BRUSH WIPER KIT PAGE 29



OPTIONAL MOUNTING FLANGES PAGE 28

STANDARD BALL NUT ASSEMBLIES

SEE PAGE 6 FOR MATCHING SCREWS

FOR BALL SCREWS 1" DIA. TO 4" DIA.

For use with the Standard and Precision Rolled-Hardened Ball Screws shown on page 6.

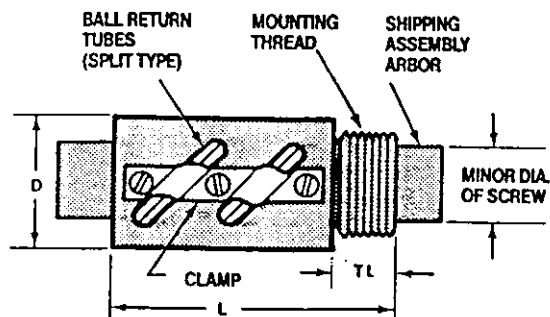
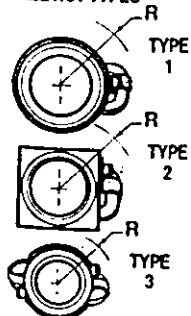
Nuts are hardened to Rc: 56-62. Stainless Steel Nuts are hardened to Rc: 40-45.

SCREW SIZE		STANDARD BALL NUTS											FLANGE	
See Page 6 For Matching Screws		Ball Nut Part No.	Type	CAPACITY		DIMENSIONS					BALL SIZE & QUAN.		TORQUE	Fits Flange Part No. Page 28
Nom. Dia.	Lead			Dynamic Load for 10 ⁴ in. Life	Max Static Load (Lbs.)	D Nut Dia.	R Radius Over Tubes	L Overall Length	T Mounting Thread Size	TL Thread Length	Nominal Diameter ††	Quantity (±5%)	To Raise 1 Pound	
1	.250	R-42-2	1	3,400	30,000	1.687	1.150	3.10	1.563-18	.600	5/32	178	.043 Lbs. in.	R40-3
	.500	R-43-2	3	4,300								186	.087 Lbs. in.	
	1.00	R-44-2	2	2,050	11,150	1.5sq.	3.00	98	.175 Lbs. in.					
		RC44-2		300	2,800			SR40-3						
1 1/8	.200	R-45-2	1	2,450	24,800	1.687	1.150	2.50	1.625-20	.485	1/8	244	.035 Lbs. in.	R45-3
		R-46-2		440	4,450							SR46-3		
	.200 LH	R-47-2	2,450	24,800	R45-3									
1 1/2	.500	R-50-2	1	9,050	54,100	2.625	1.812	4.687	2.548-18	.750	5/16	102	.087 Lbs. in.	R50-3
	.250 LH	R-53-2		4,525	44,800							2.093	1.340	3.00
	.250	R-54-2	3	6,900	32,400	2.25 sq.	1.70	3.625	2.250-20	1.00	1 1/32	64	.175 Lbs. in.	R55-3
	1.00	R-55-2										7,240	29,800	
	1.00 LH	R-56-2		23,000	130,000	3.250	2.30	6.375	3.00-12	1.50	3/8	160	.175 Lbs. in.	R61-3
	1.875	R-58-2										18,000	150	
2	1.00	R-61-2	1	19,800	132,000	3.375	2.275	6.75	3.137-12	1.56	3/8	154	.087 Lbs. in.	R60-3
	.500	R-62-2		22,500	138,500							4.00	2.75	6.75
2 1/2	.500	R-70-2	1	26,500	138,500	3.375	2.01	3.75	3.34-12	.750	5/32	194	.175 Lbs. in.	R74-3
	1.00	R-71-2		6,300	81,000							468	.035 Lbs. in.	
	.250	R-74-2		42,400	254,000							4.750	3.125	9.312
3	.660	R-80-2	1	85,700	476,950	5.875	3.756	12.593	5.497-12	2.00	5/8	190	.175 Lbs. in.	R90-3

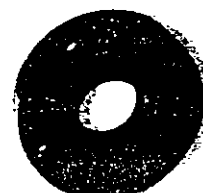
•Denotes Stainless Steel. Part No. RC44-2 is hard chrome plated. Dynamic load ratings are based on a lubricated assembly with a suitable lubricant. Normal Ball Nut Backlash is .004 to .015", may be reduced to .002 on special order. Ball Nuts are shipped on a shipping-assembly arbor unless assembly to the Ball Screw is requested. Specify direction of Ball Nut if Screw Ends are machined. ††Note: Ball sizes may vary from nut to nut but not in any one nut. Never mix balls from one nut to another. Use only factory-approved balls.

NSK NO. 1 GREASE IS RECOMMENDED FOR THESE ASSEMBLIES (STOCK ITEM).

BALL NUT TYPES



OPTIONAL NYLON BRUSH WIPER KITS PAGE 29

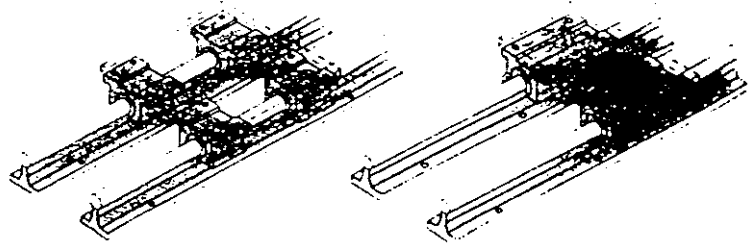


OPTIONAL MOUNTING FLANGES PAGE 28

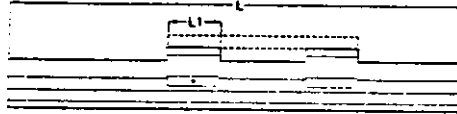
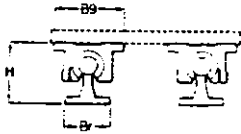
Group I System:

1CB

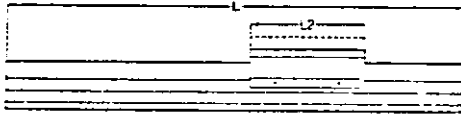
Double Shaft Fully Supported System



Double Shaft Fully Supported System with 4 Pillow Blocks



Double Shaft Fully Supported System with 2 Twin Pillow Blocks



Maximum Stroke Length is determined by subtracting pillow block length (L2) from total system length (L).

Part No	Nom. Shaft Dia	Load (lbf)*		Dimension (in.)				Pillow Block	Rail Assembly
		Max. On System	Max. On Any Bearing	L	H ±0.001	B	B9		
1CB-08-FAO	1/2	720	180	1.50	1.812	1.50	2.00	SPB-8-OPN-XS	SRA-8-XS
1CB-12-FAO	3/4	1880	470	1.88	2.437	1.75	2.75	SPB-12-OPN-XS	SRA-12-XS
1CB-16-FAO	1	3120	780	2.63	2.937	2.13	3.25	SPB-16-OPN-XS	SRA-16-XS
1CB-24-FAO	1 1/2	6240	1560	3.75	4.250	3.00	4.75	SPB-24-OPN-XS	SRA-24-XS

*Based on a travel life of 2 million inches.

Part No	Nom. Shaft Dia	Load (lbf)*		Dimension (in.)				Maximum Stroke Length (in.)	Pillow Block	Rail Assembly
		Max. On System	Max. On Any Bearing	L2	H ±0.001	B	B9			
1CB-08-HAO	1/2	720	180	3.50	1.812	1.50	2.00	L - (3.50)	TWN-8-OPN-XS	SRA-8-XS
1CB-12-HAO	3/4	1880	470	4.50	2.437	1.75	2.75	L - (4.50)	TWN-12-OPN-XS	SRA-12-XS
1CB-16-HAO	1	3120	780	6.00	2.937	2.13	3.25	L - (6.00)	TWN-16-OPN-XS	SRA-16-XS
1CB-24-HAO	1 1/2	6240	1560	9.00	4.250	3.00	4.75	L - (9.00)	TWN-24-OPN-XS	SRA-24-XS

*Based on a travel life of 2 million inches.

System	8"	12"	16"	18"	20"	24"	28"	30"	32"	36"	40"	42"	44"	48"
1CB-08	■	■	■	■	■	■	■	■	■	■	■	■	■	■
1CB-12	■	■	■	■	■	■	■	■	■	■	■	■	■	■
1CB-16	■	■	■	■	■	■	■	■	■	■	■	■	■	■
1CB-24	■	■	■	■	■	■	■	■	■	■	■	■	■	■

Systems ordered in standard lengths are typically shipped in one week. Custom length systems are available and require two to three weeks for delivery. Lengths exceeding 156.00 ins. require built joints and will need four to six weeks for delivery. For special requirements, please contact the Thomson Systems application engineering department.

Enhanced Multifunction I/O Boards ISA

Digital I/O

Number of channels
 AT-MIO-16E-10 8 input/output
 AT-MIO-16DE-10 32 input/output
 Compatibility TTL/CMOS
 Digital logic levels:
 DIO<0..7>

Level	Minimum	Maximum
Input low voltage	0 V	0.8 V
Input high voltage	2 V	5 V
Input low current (V _{in} = 0 V)	-	-320 μ A
Input high current (V _{in} = 5 V)	-	10 μ A
Output low voltage (I _{OL} = 24 mA)	-	0.4 V
Output high voltage (I _{OH} = 13 mA)	4.35 V	-

PA<0..7>, PB<0..7>, PC<0..7> (Remaining 24 lines of AT-MIO-16DE-10)

Level	Minimum	Maximum
Input low voltage	0 V	0.8 V
Input high voltage	2 V	5 V
Input low current (V _{in} = 0 V)	-	-60 μ A
Input high current (V _{in} = 5 V)	-	10 μ A
Output low voltage (I _{OL} = 2.5 mA)	-	0.4 V
Output high voltage (I _{OH} = 2.5 mA)	3.9 V	-

Handshaking
 (AT-MIO-16DE-10 only) 3-wire
 Power-on state Tri-state
 Data transfer:
 AT-MIO-16E-10 Programmed I/O
 AT-MIO-16DE-10 Interrupts, programmed I/O

Timing I/O

Number of channels 2 up/down counter/timers, 1 frequency scaler
 Resolution
 Counter/timers 24 bits
 Frequency scalars 4 bits
 Compatibility TTL/CMOS
 Base clocks available
 Counter/timers 20 MHz, 100 kHz
 Frequency scaler 10 MHz, 100 kHz
 Base clock accuracy $\pm 0.01\%$
 Maximum source frequency 20 MHz
 Minimum source pulse duration 10 ns in edge-detect mode
 Minimum gate pulse duration 10 ns in edge-detect mode
 Data transfers DMA, interrupts, programmed I/O
 DMA modes Single transfer

Triggers

Digital Trigger
 Compatibility TTL
 Response Rising or falling edge
 Pulse width 10 ns minimum

RTSI

Trigger lines -
 Bus interface Slave

Power Requirement

+5 VDC ($\pm 5\%$) 0.7 A
 Power available at I/O connector +4.65 VDC to +5.25 VDC at 1 A

Physical

Dimensions (not including connectors) 33.8 by 9.9 cm (13.3 by 3.9 in.)
 I/O connector
 AT-MIO-16E-10 68-pin male SCSI-II type
 AT-MIO-16DE-10 100-pin female 0.050 D-type

Environment

Operating temperature 0° to 55°C
 Storage temperature -55° to 150°C
 Relative humidity 5% to 90% noncondensing

AT-MIO-16E-10, AT-MIO-16DE-10

ONE: (512) 794-0100 • FAX: (512) 794-8411 • E-MAIL: info@niinst.com • WWW: http://www.niinst.com

NATIONAL INSTRUMENTS 3-31

133-2V/28	3 HOSE	28 x 65	133-2FV/28	3 HOSE	28 x 65
-----------	--------	---------	------------	--------	---------

acks and cutting tips are not supplied with torches. 198 torch can also be made with 30 mm or 35 mm barrel (Please specify your require-
 when ordering). All torches have inlet threads 9/16" x 18 U.N.F. Left and Right.
 98-4B and 198-4BF have 1/4" BSP inlet threads.

**MACHINE CUTTING TORCHES
 SERIES-198-2T-THREE HOSE**

MODEL	LONG IN.	GAS
198-2T	250	ACETYLENE
198-2TA	460	ACETYLENE
198-2TF	250	PROPANE
198-2TFF	460	MAT. GAS

ITEM NO	STOCK NO	DESCRIPTION
1	6259-BPS	Tip Nut
2	19844-2A	Head
3	19810-2	Body
4	C-62	Injector (ACETYLENE)
	C-62-F	Injector (PROPANE)
5	6219-3	Cap (ACETYLENE)
	6219-3F	Cap (PROPANE)
6	19892	Tube for 250 ^{mm} Torch (2)
	19892-A	Tube for 460 ^{mm} Torch (2)
7	19892-2	Tube for 250 ^{mm} Torch
	19892-2A	Tube for 460 ^{mm} Torch
8	19888-2T	Tail
9	9710-C	Body
10	L-97-2EPS	Needle Valve Assy
11	E-97-C	Control Valve (Cutting Ox) Complete
12	19843	Handle 250 ^{mm}
	19843-A	Handle 460 ^{mm}
13	2877	Screw (2)
14	A-62-G8PS	Needle Valve Assy (2)

**MACHINE CUTTING TORCHES
 SERIES-133-2**

MODEL	GAS
133-2	ACETYLENE
133-2/28	ACETYLENE
133-2F	PROPANE
133-2F/28	MAT. GAS

ITEM NO	STOCK NO	DESCRIPTION
1		BODY (SEE NOTE)
2	C-62	Injector (ACETYLENE)
	C-62-F	Injector (PROPANE)
3	13319-A	Inlet connections (ACETYLENE)
	13319-F	Inlet connections (PROPANE)
4	13357-2L	Inlet connections (GAS)
5	13357-2R	Inlet connections (OX)
6	4390-11	Handle

ملخص

التحكم الذاتي المتكيف بآلة القطع باللهب

إعداد

فؤاد فايز نقولا البجالي

المشرف

الدكتور يوسف العسّاف

تنتشر عمليات قطع صفائح وأواح المعدن بشكل كبير في الصناعة وتتفد عادةً باستخدام عملية القطع باللهب. في هذه العملية، يعتبر التحكم بالموقع أساساً للحصول على قطع واضح و "نظيف". إلا أن أسلوب تثبيت السرعة المستخدم في أغلب آلات القطع باللهب المتوفرة قد لا يعطي العلاقة المثلى بين إشارتي الموقع والسرعة وخاصةً عند الزوايا الحادة والمنحنيات.

لقد تم في هذا البحث تقديم متحكم ذاتي متكيف من الممكن تطبيقه في آلة محوسبة للقطع باللهب. يتكون هذا المتحكم من جزئين: الأول خوارزمية للتعرف على معالم العملية (Process Identification Algorithm) والثاني خوارزمية التحكم التنبؤي العام (Generalized Predictive Control Algorithm). هذا المتحكم يمكنه التغلب على المشاكل المذكورة أعلاه عن طريق تغيير السرعة نتيجةً للتغيرات المتوقعة في شكل القطع وبالتالي يرفع من جودة المنتج النهائي وإنتاجية العملية. لقد تم دراسة وتحليل هذه الخوارزمية في هذا البحث وإثبات أفضليتها مقارنةً بالمتحكمات التقليدية مثل (PID) في التحكم في عملية القطع باللهب.

بالإضافة إلى ذلك تم تقديم تصميم لآلة محوسبة للقطع باللهب وتم شرح أجزائها والأنظمة المكونة لها.