THE MULTIVARIABLE CONTROL OF A HOT BAR ROLLING MILL

This dissertation has been submitted to the Department of Electrical and Electronic Engineering, University of Cape Town in partial fulfillment of the requirements for the MSc degree in Electrical Engineering.

by

Grant Crosley

September 1991

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to the following:

Scaw Metals Ltd. for providing the funding for this research and my financial support over the last five years.

Prof. M.Braae of the Department of Electrical and Electronic Engineering, University of Cape Town, for his supervision and advice during this dissertation.

Mr B.C.Stevens, Chief Electrical Engineer at Scaw Metals Ltd., Germiston, for his ideas and help for the duration of this thesis.

Mr K.Welgemoed, Chief Electrical Engineer at the Hille Mill plant, Scaw Metals, for all his advice and discussions on the plant operation.

Mr T.Vermeulen, Electrical Engineer in charge of the Test Department at Scaw Metals, for organising and conducting the tests on the Hille Mill.

Mr T.Van Breugel, Electrical Engineer at Scaw Metals Ltd., for the present plant configuration block diagrams.

Mr C.Greensmith, Chief Electrical Engineer at the Morgan Mill plant, Scaw Metals, for his advice during the initial study of the plant setup.

SYNOPSIS

This thesis dissertation is an investigation into the multivariable control of a modern continuous hot bar rolling mill.

The objectives of this study were to analyse the Hille Mill Plant and to determine the manner in which this rolling mill operates. From this initial study the objective was to design two types of control systems for the process. The two control systems used are single-variable proportional-integral controller system and a multivariable control system designed using the Nyquist Array. The final objective was to ascertain what the best control strategy for this industrial process is.

The plant structure and setup was investigated first. The control problem was then formulated in terms of classical control theory.

The open loop structure derived from the initial problem formulation was then explained and used to derive the proposed closed loop control structure. Which input and output variables are used to control the plant was investigated.

Due to economic constraints, the study could not be conducted using the actual plant and since a mechanical model of the process was not available, a computer simulation of the rolling production line was written. The main aspects that had to be simulated were the D.C. drives, the rolling forces on the steel bar and the movement of the bar through the production line and the effects this caused.

The ramp test data obtained from the Hille Mill was found to be inadequate for the design of the simulator and therefore it was designed using mathematical models of the rolling aspects previously mentioned. Thus, these transfer functions were also not adequate for the design of the control systems. The transfer functions used to design the control systems were derived from step tests conducted on the computer simulation of this process.

The control structure required for a multivariable system was investigated. The structure was found to depend on the position of the bar in the production line. As the bar entered the stands sequentially the plant interaction changed each time. Similarly, the effect of the bar leaving the stands was to change the plant interaction matrix. The rolling process cycle describes the passage of a bar completely through the production line. The control structure required by the plant was derived for each step in the process cycle.

The transfer functions for the plant were derived for all the different stages in the production cycle. These transfer functions were then used to design the required control systems. The multivariable system, consisting of the different control structures required at different times in the rolling cycle, was designed using the Nyquist Array design methods. Single-variable proportional-integral controllers were also designed to control the plant outputs. This was done using the Characteristic Loci design method.

Both of these control systems were implemented on the rolling mill simulator in closed loop. The performance of these control systems was recorded under different plant conditions. These conditions were normal plant production, tension and production speed setpoint changes and the effects of a change in the metal model on the controller performance.

These results were then graphically and numerically compared in terms of the percentage interaction in each output after a setpoint change, the time the output took to settle to within 5% of it's setpoint value, it's overshoot and whether the output response was acceptable for this process. The disturbance rejection of the two control systems was also computed and compared.

The comparison of these results indicated that the multivariable control system was superior in all aspects of closed loop plant performance. The single-variable system did not control the plant within acceptable limits when any of the production tension or speed setpoints were stepped.

From the results obtained in this dissertation it was concluded that the best control strategy to adopt for this process, is a multivariable control system which switches between different decoupling controllers during the rolling process.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Chapter 4:

Appendix A:

Appendix F:

Appendix G:

Appendix I:

Appendix K:

Appendix O:

LIST OF TABLES

Appendix D:

Appendix E:

Appendix H:

Appendix N:

NOMENCLATURE

Symbol	Meaning
A	: Ammetter or Amperes.
a	: Triangle side length variable.
Ag	: Silver.
Al	: Aluminium.
A_o or A_i	: Cross-sectional area.
b	: Triangle side length variable.
b_i	: Width of metal cross-section.
B.I.M.	: Binary Interaction Matrix.
C	: Carbon.
$C_e\Phi$	: Back electro-magnetic force constant.
$C_m\Phi$	: Motor load constant.
Co	: Cobalt.
Cr	: Chromium.
Cu	: Copper.
C/L	: Closed Loop.
D or D_1	: Disturbance.
D.C.	: Direct Current.
da	: Change in triangle side length.
dl	: Change in length.
dt	: Change in time.
$\frac{di_a}{dt}$	: Rate of change of armature current w.r.t time.
$\frac{d^2i_a}{dt^2}$	: Second derivative of the current w.r.t. time.

$\frac{dn}{dt}$	:	Rate of change of velocity w.r.t time.
$\frac{d^2n}{dt^2}$	:	Second derivative of velocity w.r.t time.
$\frac{dv_a}{dt}$	:	Rate of change of supply voltage w.r.t time.
D.N.A.	:	Direct Nyquist Array.
dV or dV_i	:	Change in volume.
E or e	:	Error variable.
E_a	:	Armature generated voltage.
E.M.F.	:	Electro-magnetic Force.
E_y	:	Young's Modulus.
e^{-sT}	:	Time delay element.
F	:	Force.
Fe	:	Iron.
F_{li}	:	Metal tension force.
G_i or G_{ii}	:	Open loop transfer function.
G(s)	:	Open loop transfer function matrix.
H	:	Unit of inductance, Henry.
h	:	Loop height.
h_i	:	Height of metal cross-section.
H.M.D.	:	Hot Metal Detector.
H(s)	:	Closed loop transfer function matrix.
$H(s)^{-1}$	:	Inverse closed loop transfer function matrix.
Ht	:	Height variable.
H11	:	Aircraft steel.
i_a	:	Armature current.
iff	:	If and only if.
i_r	:	Armature current reference.

I.N.A.	:	Inverse Nyquist Array.
I_{pi}	:	P.I. controller integral constant.
J	:	Machine mechanical moment of inertia.
K	:	Controller.
k	:	Direct current machine constant.
K_a	:	Armature transfer function gain constant
k_{ii}	:	Control matrix element ii.
K_{pi}	:	P.I. controller gain constant.
K(s)	:	Controller matrix.
kW	:	Unit of power, kiloWatt.
k'	:	Direct current machine constant.
L	:	Loop height/length variable.
l	:	Length.
L_a	:	Armature inductance.
L.H.	:	Loop Height detector.
mm	:	Unit of length.
mm^2	:	Unit of area.
mm^3	:	Unit of volume.
mm/s	:	Unit of velocity.
M_a	:	Acceleration torque.
M_c	:	Rolling torque constant.
M_d	:	Drag torque.
Mg	:	Magnesium.
MIMO	:	Multi-Input Multi-Output.
M_l	:	Load torque.
Mn	:	Manganese.
Mo	:	Molybdenum.
MPa	:	Mega Pascal.
MV	:	Multivariable.
n or n_i	:	Rotational velocity.
N	:	Unit of force, Newton.
Ni	:	Nickel.

Nmm	:	Newton millimeter.
N/mm^2	:	Newton per square millimeter.
n_r	:	Rotational velocity reference.
Ohm	:	Unit of resistance.
O/L	:	Open Loop.
P	:	Phosphorous.
P.I.	:	Proportional Integral.
P.T.	:	Pulse Transmitter.
Q(s)	:	Open loop compensated plant matrix.
$Q(s)^{-1}$	:	Inverse open loop transfer function matrix.
r	:	Roller radius.
R_a	:	Armature resistance.
rpm	:	Revolutions per minute.
(s)	:	Laplace operator.
S	:	Sulphur.
s	:	Unit of time, second.
S.C.R.	:	Silicon Controlled Rectifier
Si	:	Silicon.
Sn	:	Tin.
Sp or S	:	Speed variable.
S(s)	:	Sensitivity Function.
S & H	:	Sample and Hold.
T	:	Tension variable.
T_o	:	Initial metal tension torque.
T_a	:	Armature circuit time constant.
T.G.	:	Tacho-generator.
T_{li}	:	Accelerating torque due to metal tension.
T_m	:	Motor mechanical time constant.
T(s)	:	Complementary Sensitivity Function.

u	:	Input variable.
V	:	Vanadium.
V_a	:	Armature supply voltage.
v_i	:	Metal speed through stand i.
w.r.t.	:	With respect to.
+/-	:	Comparative controller.
Φ	:	Total motor field flux.

CHAPTER 1: INTRODUCTION

Rolling mills of various kinds are located all over the industrial world. There are sheet metal processing plants, paper rolling mills and steel bar rolling mills. This thesis is a study of the latter, the modern continuous steel bar rolling mill.

The mill under consideration is the Hille Mill at the Scaw Metals Ltd. plant in Germiston, Transvaal. A schematic of the operation of the plant is shown in figure 1.1.

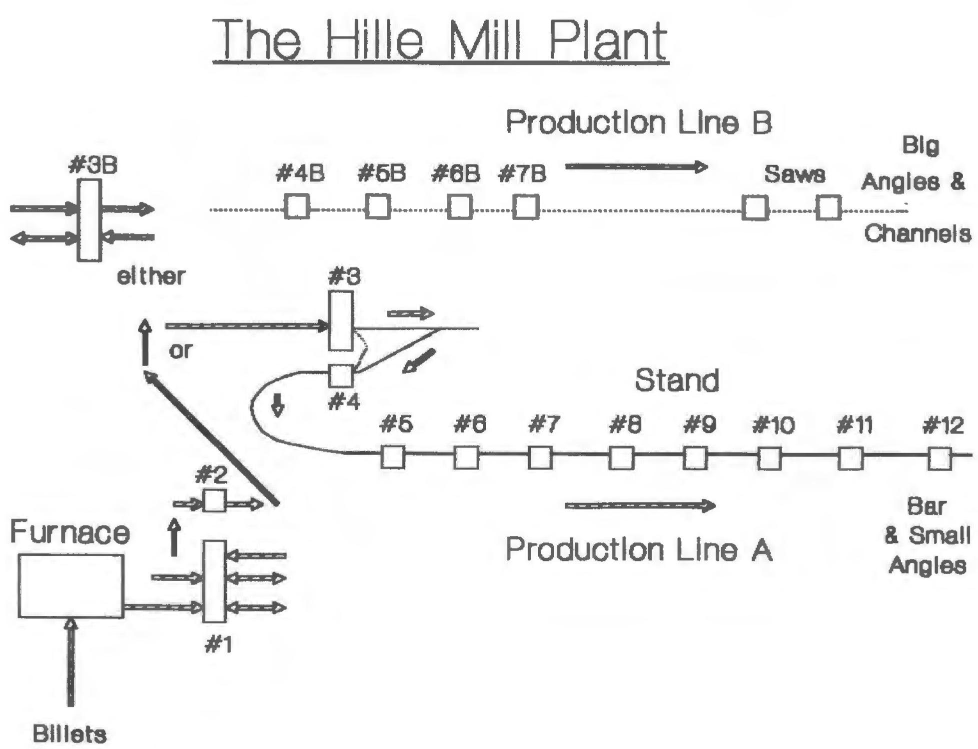


Figure 1.1: The Hille Mill Plant.

The steel used in the Hille Mill is melted down and formed into billets in the steel foundry section. These billets are heated up in the furnace at the beginning of the production process. They are then sent through the rolling process.

The initial rolling process involves coarse rolling down of the billet to smaller dimension. This is done by means of two stands of rollers, labelled #1 and #2 in figure 1.1. The billet passes through these stand six times in total. Each pass through the stands reduce the cross-sectional area of the steel bar. The bar is then transferred by means of conveyor belts to either stands #3 and #4, (the beginning of line A), in figure 1.1, or stand #3B at the beginning of production line B. These stands complete the initial rolling before the billet passes into production lines.

The steel billet is then rolled into the final required product by either of the two different production lines, line A and line B. Flat, round, square and small angled steel bars of varying dimensions are rolled in line A, while big angle and channel steel products are produced by line B.

Line A is the production line that is studied in this thesis. There are eight stands in production line A, labelled #5 - #12 in figure 1.1. The product shears that cut off the end of the bar or that cut the final products into the right length are not shown in this schematic. The set of rollers in each stand is controlled by a separate D.C. drive. The steel billet is rolled to the correct dimension while passing through the eight stands.

In order that a high quality of steel bar is produced and that production output is maximised, it is necessary to optimise the performance of these mills. This is done by controlling two rolling variables, namely the tension and the speed.

To ensure high quality steel the tension in the rolling process must stay constant. If the tension varies the amount that the steel is stretched also varies. This causes the steel product to be weak and of low quality. While the steel bar passes through the first four rolling stands the tension is kept constant by maintaining the first four drives at the correct speeds. In this manner the steel is not stretched or compressed between any two stands.

Tension-free rolling between the last five stands is achieved by forming the metal into a loop between the stands. The loop height will change if a drive's speed changes and the loop prevents the steel product from being stretched or compressed.

For maximum production the speed of the rolling mill must be kept as high as possible. The output is kept at a specific rate by controlling the speed of the last drive in the production line (#12, figure 1.1). All operating problems that arise are solved using the other drives.

The objectives of this study are:

1. To analyse the rolling mill plant at the Hille Mill and to identify it's open loop structure.
2. To model and simulate production line A.
3. To use proportional-integral (PI) controllers to control the simulator.
4. To use the Inverse Nyquist Array (INA) design method to design a control system.
5. To determine the best control solution to this problem.

In this thesis the plant structure and setup is investigated first and the control problem is formulated in terms of classical control theory. The plant inputs and outputs are identified and explained.

Then the open loop system is discussed and the present control system is briefly explained. The problems of this approach are mentioned and the differences between the methods used to design the present control system and the classical control theory used in this thesis are presented.

The configuration of the closed loop control system that is used in this thesis is given in the form of block diagrams and then analysed.

A computer simulation of the rolling mill is needed in order to evaluate this theoretical approach to the design of the best control system. Simulation is used because it is not possible to work on the actual plant due to economic considerations and the fact that there is no working model of the rolling mill plant at the University of Cape Town.

The design, calculation and programming of the complete rolling mill simulator is then discussed. The mathematical laws and equations used in the simulator are explained and justified.

The theory of control structures is then presented and it is related to the rolling mill system. How the required control structure changes depending on the position of the bar is then illustrated. The implications of this changing control structure are analysed using binary interaction matrices (BIM).

Equations describing the plant output responses to step changes in the input variables are calculated from the rolling mill simulator. These are compared to the equations derived from actual plant data.

The control of the rolling mill simulator is then looked at. The automatic feedback control systems used in this study are :

1. PI controllers. These controllers are designed using single variable control theory. They are a common industrial control device.
2. A multivariable control system. This system is designed using the Inverse Nyquist Array design method. This is a multi-variable design method which is used to minimise plant interaction.

Firstly, the optimum performance achievable using simple PI controllers is analysed. Then the equations calculated from the simulator are used in conjunction with the binary interaction matrices already derived, in order to design the required control structures for control of the complete rolling process. This is done using the Inverse Nyquist Array design method. The theory of this design method is presented and it's advantages and disadvantages as compared to other design methods are discussed.

The results of applying the control system designed using the Inverse Nyquist Array are then compared to the results of using simple PI controllers.

Finally conclusions are drawn as to the best control system (of those used in this study) to use in the Hille Mill.

CHAPTER 2: THE HILLE MILL ROLLING SYSTEM

In order that a control system can be designed for this process it is necessary to study the plant and the rolling process and to formulate the rolling mill problem in terms of classical control theory.

2.1 The Plant Schematic Diagram

A diagram illustrating the plant setup is shown in Figure 2.1.

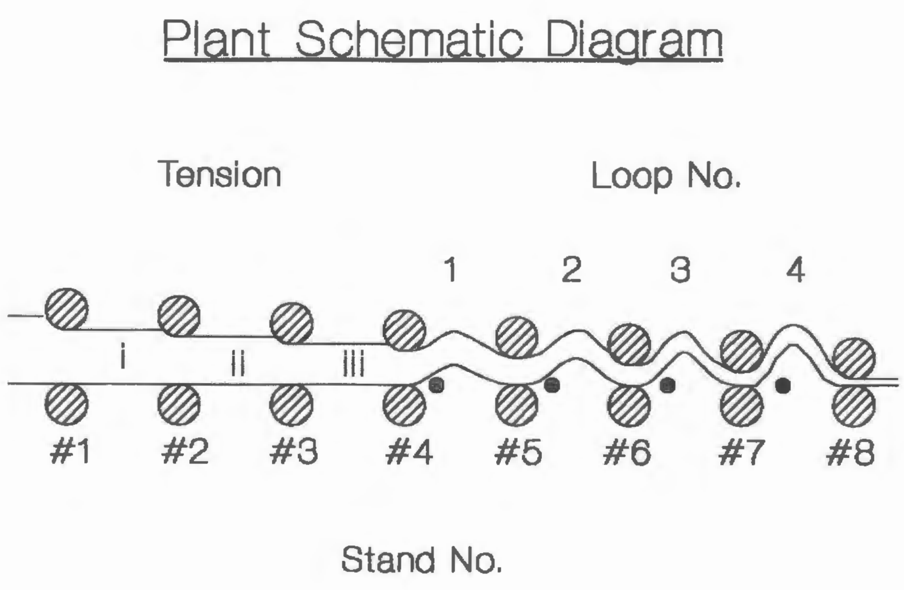


Figure 2.1: A simplified schematic of production line A

Figure 2.1 shows that there are eight stands in this production line. The steel billet travels from left to right on this schematic diagram.

The billet is rolled into the required dimensions by exerting pressure on the hot metal. This is done by squeezing the steel through pairs of rollers, represented in figure 2.1 by the shaded circles. Each pair of rollers is driven by a D.C. drive. A stand consists of a D.C. drive and a pair of rollers.

As is shown in figure 2.1, the steel is looped between the last five stands. This ensures that the steel bar is not stretched or compressed in the final stage of production. A change in the speed of one of these drives will affect the loop height and not the tension in the metal.

While the steel bar is between the first four stands it is too thick to loop in this manner. The bar passes directly from one stand to the next. The speeds of these four drives will affect the tension in the metal and the aim of any control system must be to keep this tension constant during rolling.

The production rate of line A must be kept constant. Stand eight is used to control the speed of the metal bar as it leaves the production line. Stand seven must then be used to control the height of loop no 4, as labelled in figure 2.1. In this study the preceding stand will always be used to control a plant feature. This is how the Hille Mill plant operates at the present time[1].

The following section deals with the identification of all the plant input and output variables and with which input is used to control a particular output variable.

2.2 The Plant Inputs and Outputs.

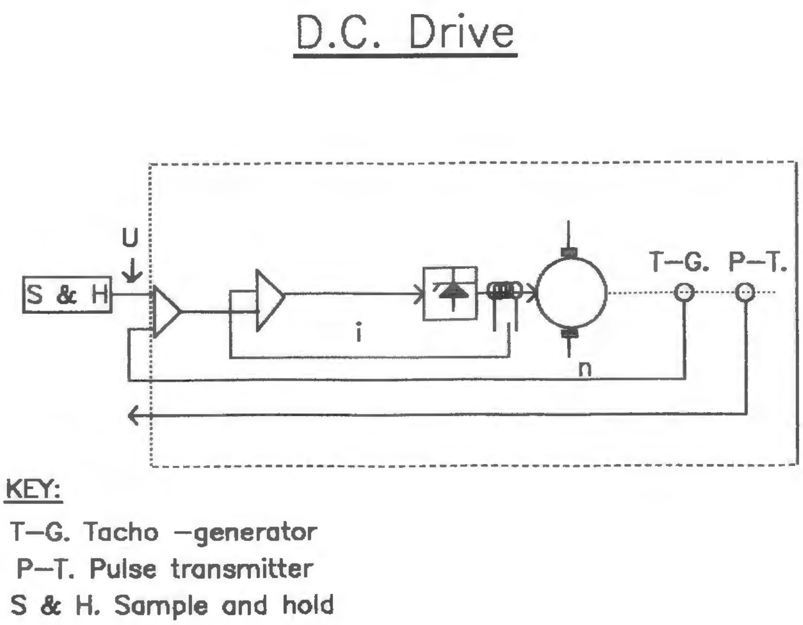


Figure 2.2: Block diagram of the D.C. drives.[2]

The D.C. drives are used to roll the steel into the desired shape. They are configured as shown in figure 2.2. From this diagram it is possible to identify the input and output variables of the stand. The input "u" is a setpoint for the D.C. drive speed control system.

This system controls the speed of the drive. The speed is measured by two devices, as shown in figure 2.2. These are a tacho-generator and a pulse transmitter. The tacho-generator signal is used as the feedback signal for the D.C. drive control loop.

Hence there are two highly correlated outputs from each drive in the production line. These are the actual speed and the pulse transmitter output.

In addition to these outputs associated with the D.C. drive there are the plant outputs which indicate the status of the metal. As figure 2.1 shows, the two types of outputs which reflect the status of the steel in the plant are the tensions, labelled i,ii and iii, and the loop heights 1,2,3 and 4.

The tension in the metal is measured or represented by the current consumption in the drive immediately preceding that section of bar. If a drive's speed increases it will pull more metal through that stand's rollers during a unit of time. This will cause the metal section preceding that drive to be stretched. Since the metal is being pulled by the drive, the preceding drive will have to do less work in order to maintain a constant speed. Therefore the current in the preceding drive will decrease.

If, on the other hand, a drive's speed decreases this will lead to the preceding drive having to do more work to maintain a constant speed. This will cause the current in the preceding drive to increase. Therefore the current in a drive is an indication of the work it is doing and hence of the relative tension of the steel billet.

The height of the loop formed by the metal is measured using infra-red detectors. This is possible because the temperature of the metal at this stage is approx. 1000°C [See Appendix A].

Thus there are eight plant inputs and twenty three plant outputs.

Inputs:
1. Eight D.C. drive speed setpoints.

Outputs:
1. Eight actual D.C. drive speeds.
2. Eight pulse transmitter outputs.
3. Three current outputs (from the first three drives).
4. Four loop heights.

As explained in section 2.1, it is important to control the tension of the rolling process and to keep the production rate constant.

Therefore the following eight outputs must be controlled to achieve these objectives.

Tension control: 1. Three current outputs.
2. Four loop heights.

Speed control: 1. One pulse transmitter output. (Drive eight)

2.3 The Open Loop Control Configuration.

In open loop the eight speed setpoints would be used to control the eight outputs mentioned in the previous section (three current outputs, four loop heights and one speed output). The D.C. drive represented by figure 2.2 is shown as a single block in figure 2.3.

Single variable controllers are needed to control the plant in open loop. These controllers would be positioned as is illustrated in figure 2.3. Each controller alters the input (speed setpoint) to a drive in order to control a plant output.

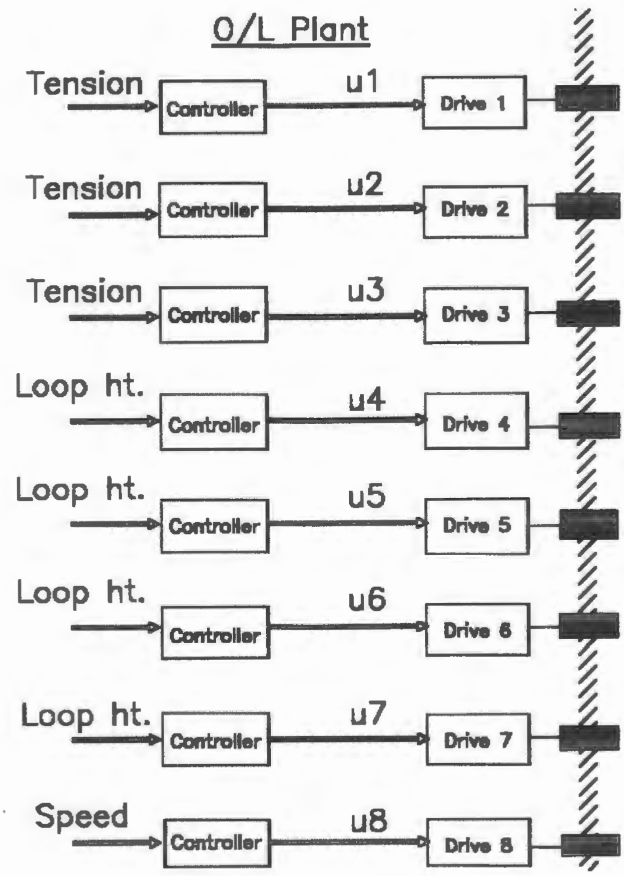


Figure 2.3: The open loop plant configuration.

In open loop the inputs to the drives are totally independent of the outputs. If a disturbance causes an output to change the open loop controller will not compensate for this. The performance of the rolling mill and the quality of the products will not be optimum.

Similarly, if one speed setpoint is changed it not only affects the desired output but it will affect other outputs as well. The open loop controllers trying to control these other affected outputs will again not compensate for this. Therefore an operator trying to solve one problem might create a series of problems down the production line.

For these reasons open loop control of this rolling mill plant is not acceptable. Feedback from the output is required in order to control this process effectively. Therefore only closed loop controllers are considered and designed in this thesis study.

2.4 The Present Closed Loop Plant Configuration.

Figure 2.4 is an illustration of the present control system at the Hille Mill plant, the Contritrol-P controller[3]. This was installed by the Brown Boveri Corporation. Although an in-depth study of the Contritrol-P system has not been conducted as part of this project, the basic method of operation has been investigated.

This system controls the rolling process in the manner determined in section 2.2. It controls the tension of the rolling process by keeping the current consumption in the first three drives constant.

The first drive in the production line is used to control the tension of the metal section immediately after the drive. This tension is labelled "i" in figure 2.2. Similarly, the second and third drives are used to control the tension in sections "ii" and "iii", respectively.

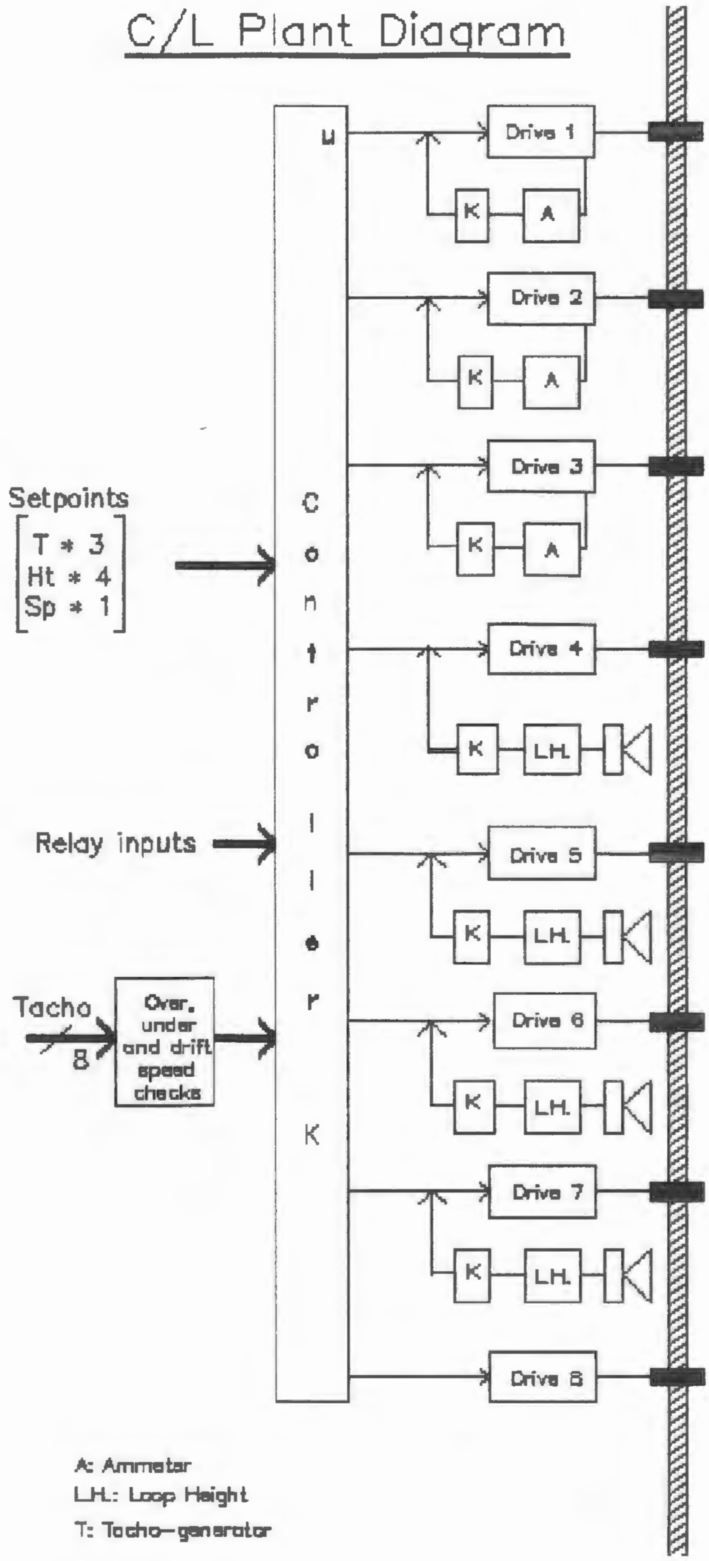


Figure 2.4: The C/L plant control diagram.

The loop heights, labelled 1,2,3 and 4 in figure 2.2, are controlled by using drives 4,5,6 and 7 respectively.

The rate of production is maintained by keeping the speed of the last drive, drive number eight, constant.

Figure 2.4 illustrates that the currents in drives 1,2 and 3 are fed back to (part of) the controller. These, along with the outputs of the loop height detectors ("LH"-in figure 2.4) are used in the first "control layer" of the Contritrol-P controller. The "control layers" are illustrated in figure 2.5.

The first layer consists of three tension control loops and four height control loops. Figure 2.5 shows how this "control layer" is implemented on the plant. Note that the speed of drive eight is already controlled as illustrated in figure 2.2. All eight drives have this inner speed control loop. This inner loop and the first control layer are also known as cascaded inner control loops. Figure 2.5 represents the complete D.C. drive configuration of figure 2.2 with a single block.

A comparative controller is used to control each tension output. This position of this controller is shown in figure 2.5 by the "+/-" block. This operates by comparing the actual current with a desired value. If the value is not equal to the desired value the speed reference to the drive is altered accordingly. This decision process is conducted by a micro-processor.

The loop height is controlled by means of a proportional integral (PI) controller. The height is measured using an infra-red scanner. The output of this scanner is the input to the PI controller. This controller alters the speed setpoint to the drive if the height is not correct.

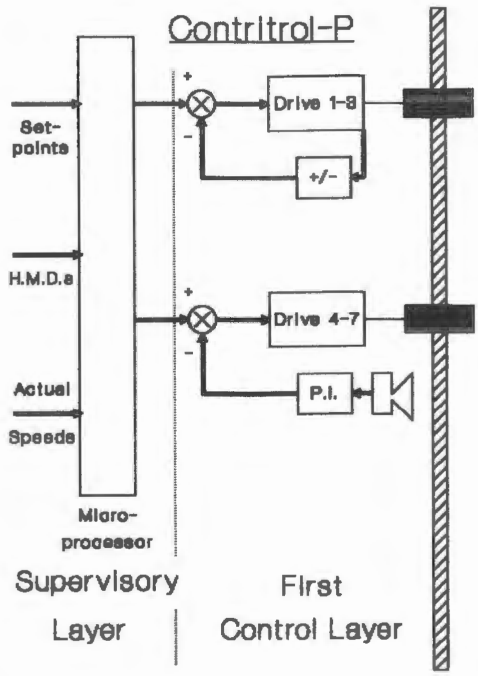


Figure 2.5: The Contritrol-P control structure[2].

Figure 2.5 also illustrates the "supervisory control layer" of the Contritrol-P controller. This layer is a micro-processor based control system[3]. Using setpoints chosen by the operator of the production line it calculates the speed references to the first "control layer" of the controller.

The position of the bar in the rolling mill is monitored by hot metal detectors (H.M.Ds). The outputs of these detectors are sent to the controller. Using these outputs and the relative speeds of all eight drives, the "supervisory control layer" alters the speed references to the drives to achieve the desired plant performance.

This controller was designed in an ad hoc engineering manner without using classical multi-variable control theory. The differences between the approach adopted in the Contritrol-P system and that of classical control will be illustrated in the next section.

2.5 The Proposed Closed Loop Control Configuration.

The block diagram of the closed loop control system designed in this thesis and it's setup in the plant is illustrated in figure 2.6.

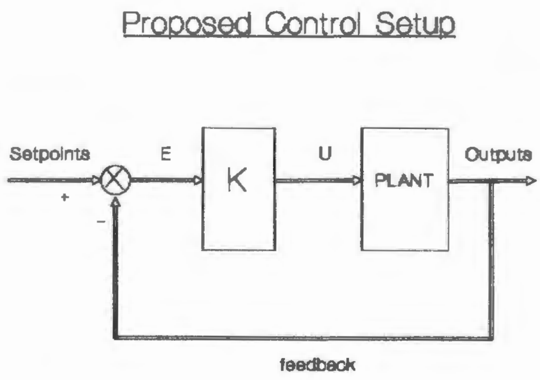


Figure 2.6: The closed loop block diagram.

Although exactly the same outputs are controlled and the same inputs are used by this proposed control setup, the manner in which it is done is different.

There are no supervisory control layers. The outputs that are being controlled are fed back and compared to the setpoints. The difference between the two, the error,"E", is then the input to the control unit K. The controller then alters the inputs,"U", to the eight drives accordingly.

The controller, marked "K" in figure 2.6, is designed in this thesis study. The aim of the controller is to regulate the plant tension, loop height and production rate at optimum level.

As was mentioned before, two types of control system will be designed for this rolling mill. The first system will consist of eight single variable PI controllers. The second will be designed using the multi-variable INA design method. Both will be implemented on the rolling mill simulator, designed and explained in Chapter 3.

References.

1.Welgemoed, K. Private communication. Electr. engineer in charge of the Hille Mill at Scaw Metals Ltd., Germiston, Transvaal.

2.Van Breugel, T. Private communication. Electrical engineer (until June 1990) in charge of the present Contritrol-P micro-processor, Scaw Metals, Germiston, Transvaal.

3.Brown Boveri Corporation. Operating and Maintenance Manual for Bar and Section Mill - Scaw Metals, contract DV 11000, 1980.

CHAPTER 3: THE ROLLING MILL SIMULATOR

Due to economic constraints it is not possible to work on the actual plant and it is, therefore necessary to test the different controllers designed in this thesis on a model or a simulator of the plant. Since a physical model of the plant is not available, a computer simulation was written. The computer simulation was based on mathematical equations describing various aspects of the rolling process. The main components of the rolling process to be simulated are the D.C. drives, the tension in the metal, the formation of the loops and the time scale of the rolling process.

3.1. The Simulation of the D.C. Drives.

A literature review on simulating D.C. drives was conducted. A number of applicable articles and books are reviewed in the next section. These articles and books are the main source of the information and equations used to design the simulator in section 3.1.2.

3.1.1. Literature Review.

In order to derive an accurate model of the D.C. drive and it's operation, the basic theory is investigated.

One of the first books on D.C. machines *The Performance and Design of Direct Current Machines* was written by Dr A.E.Clayton in 1927. It became a standard and widely read text. Since this book was published radical changes in the technology of D.C. machines have taken place. The book *Direct Current Machines* written by M.G.Say and E.O.Taylor[1] in 1977 endeavors to carry on with Dr.Clayton's belief that machine behaviour is best grasped through an understanding of flux-current interaction.

The principles and conventions, with regard to symbols and models, used in this thesis are the same as described in this book.

As explained in this book and in *Electrical Machines and Power Electronics*[2] by R.E.Steven, D.C. machines can be classified differently according to how they are excited. It is necessary to know what type of excitation a motor has so that it can be simulated using the correct equations.

The equations governing the motor behaviour change according to how they are excited. The D.C. drive in the Hille Mill is classified as a seperately excited machine phase drive with current and speed feedback. This type of excitation is also well described in *Electrical Machines and their Applications*[3] by J.Hindmarsh.

The equations governing the speed/current, torque/current, torque/speed relationships are covered in many of the books on this subject[1-4]. These relations are used in section 3.1.2 to derive the equations describing the drive performance.

The book *Electric Machinery*[4] by A.E.Fitzgerald, C.Kingsley Jr. and S.D.Umans describes the steady-state and dynamic behaviour of these machines. The book goes onto derive the transfer functions for parts of the D.C. drive system. The block diagrams for different systems, including the D.C. drive system simulated in this thesis, are explained and discussed in this book as well.

The block diagram and transfer functions of this seperately excited drive system are also dealt with in *Electrical Machines and Power Electronics* by R.E.Steven. However, this drive system is given an excellent coverage in two articles that are to be found in the University of Cape Town's Immelman Library.

The first article, entitled *Microprocessor-Based Control of Steel Rolling Mill Digital DC Drives*[5] by R.J.Hill and F.L.Lou, presents novel algorithms for the digital control of double-loop D.C. machine drives used for application where conventional torque or power control is inadequate. Such an application is the steel rolling mill drive.

Where this article is particularly useful is the good discussion it offers on the equations governing the D.C. drive, shown in figure 2.2. After considering the equations and laws describing the dynamical and electrical machine operation, the time constants, gains and transfer functions of this system are derived and explained.

The block diagram, used in section 3.1.2 to write the simulation, is derived in this paper. Various assumptions used in order to simplify the block diagram are explained and justified by R.J.Hill. One such simplification is the optimization of the current inner loop. The PI constants are calculated to achieve the so called *fast response*[5] used in this simulator package.

The second article is *Mathematical Modelling in Undergraduate D.C. Machine Drive Projects*[6] by R.J.Hill. The first section of this paper deals with the mathematical modelling and simulation of the D.C. electrical machine. It presents the equations governing the drive response in a slighlty different manner to the previous article, of which R.J.Hill is also a co-author.

The information contained in these books and articles is used to derive an accurate model of the D.C. drive system at the Hille Mill. This derivation is strongly based on the methods presented in the two articles mentioned[5,6].

3.1.2. THE D.C. Motor Equations.

Figure 3.1 is a representation of the motor armature circuit. This is for a separately excited D.C. machine phase drive with current and speed feedback.

The following equations govern the D.C. machine's behaviour. There are many different notations for these equations[1-6]. The notation adopted is that used by R.J.Hill and F.L.Lou[5]. They describe the effects of the armature current, the back EMF, the drag torque and the acceleration torque.

Armature Circuit Representation

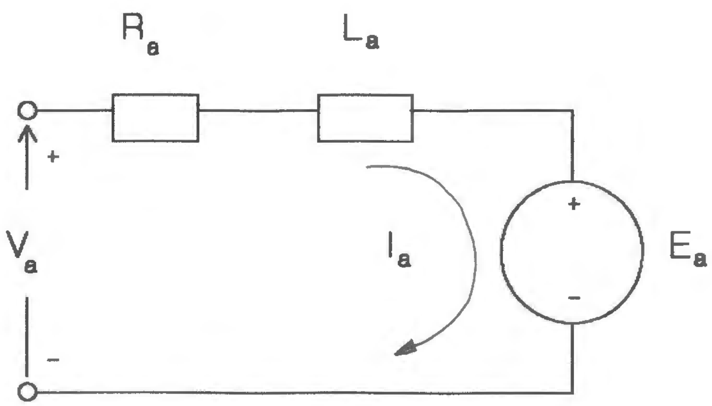


Figure 3.1: Motor armature circuit representation[5].

$$E_a + L_a * \frac{di_a}{dt} + R_a * i_a = v_a \qquad ...(3.1.1)$$

$$E_a = C_e * \Phi * n \qquad ...(3.1.2)$$

In equation 3.1.2, C_e is the back EMF constant and Φ is the total field flux.

$$M_d = M_l + M_a = C_m * \Phi * i_a \qquad ...(3.1.3)$$

In the above equation M_d is the drag torque, M_l the load torque, M_a the acceleration torque and C_m is the load constant. Friction is included in the load torque.

$$M_a = J * \frac{dn}{dt} \qquad \text{...(3.1.4)}$$

J is the machine mechanical moment of inertia.

Using these four equations it is possible to derive the dynamic equation for the D.C. motor (Eq. 3.1.5). This is done in Appendix B.

$$T_mT_a* \frac{d^2n}{dt^2} + T_m* \frac{dn}{dt} + n = k*v_a - k'*\left[M_l + T_a* \frac{dM_l}{dt} \right] \qquad \text{...(3.1.5)}$$

The equation describing the electrical current response is also derived from equations 3.1.1-3.1.4. This is done in Appendix C. Equation 3.1.6 is the final form of the torque-current response of the motor.

$$T_mT_a * \frac{d^2i_a}{dt^2} + T_m * \frac{di_a}{dt} + i_a = \frac{T_m}{R_a} * \frac{dv_a}{dt} + \frac{M_l}{C_m\Phi} \qquad \text{...(3.1.6)}$$

3.1.3: The D.C. Motor Speed Control System.

The motor mechanical dynamic and current torque equations qouted in the previous sections enable the block diagram of the D.C. speed control system to be derived. This is shown in figure 3.2.

This block diagram can be simplified. In a D.C. motor the armature current varies much faster than the armature voltage and speed[5]. This enables the back EMF and speed variations to be neglected for the optimization of the inner current control loop. The block diagram then becomes that shown in figure 3.3.

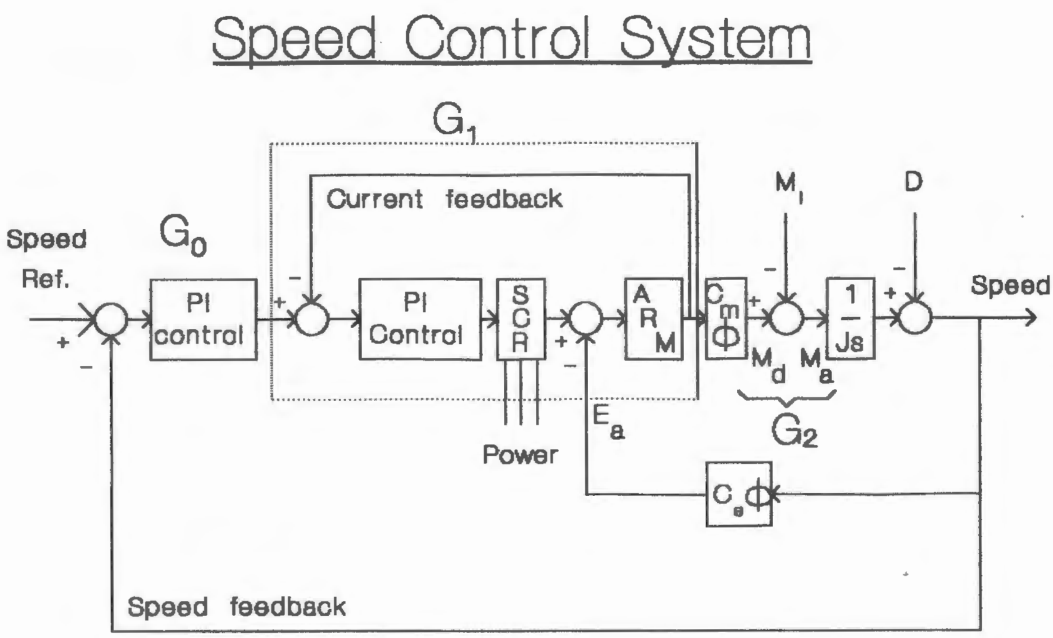


Figure 3.2: The D.C. motor speed control system[2,4,5].

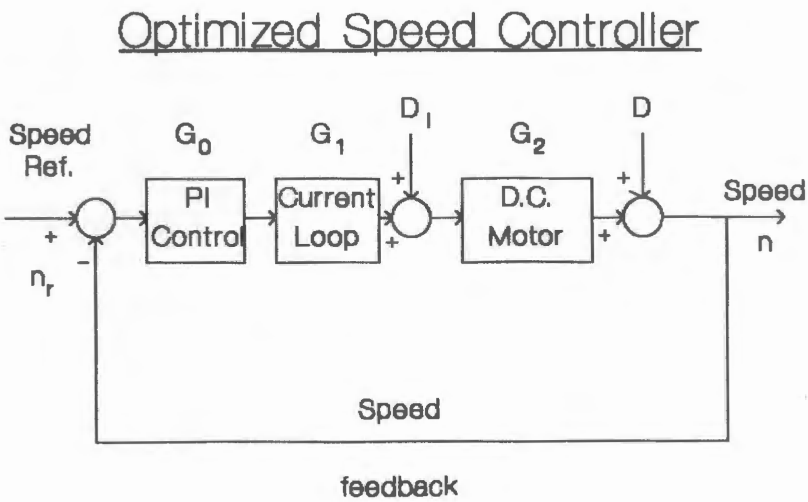


Figure 3.3: The speed controller with an optimized inner current loop

The computer simulation of the D.C. drives is written to simulate the above block diagram. M_l, the load torque is calculated in the simulation of the tension in the steel bar which is described in section 3.2.

The electrical and mechanical characteristics of the D.C. motors used at Scaw Metals are used in this simulation package. These are given in Appendix D.

The armature current inner loop shown in figure 3.4 is a stand alone thyristor current source under PI control[5]. The following equation, Eq 3.1.7, describes the transfer function of the motor armature.

$$G_{12}(s) = \frac{K_a}{1 + s*T_a} \qquad ...(3.1.7)$$

The transfer function of the PI controller, G_{10}, is

$$G_{10}(s) = \frac{K_{pi} * (1 + s*I_{pi})}{s * I_{pi}} \qquad ...(3.1.8)$$

The thyristor rectifier is a zero-order hold with the transfer function $G_{11}(s)$.

$$G_{11}(s) = \frac{1 - e^{-sT}}{s} \qquad ...(3.1.9)$$

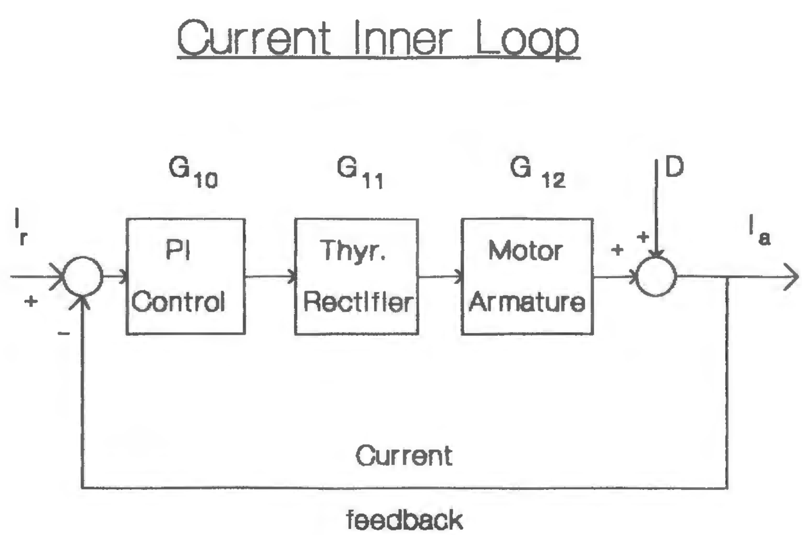


Figure 3.4: The armature current inner loop.

The optimization criteria for the current inner loop represented in figure 3.4 is that the armature current must reach it's new steady state value within one calculation interval. If this is the case then the current inner loop can be simplified and can be represented by equation 3.1.10.

$$G_1(s) = e^{-sT} \qquad \ldots(3.1.10)$$

This is the transfer function of a pure delay element. This is termed *fast response*[5]. The drives in the simulation package are assumed to be optimized and therefore the current inner loop is simulated as a pure delay element. The effect of varying this delay time is just to change the transfer function of the closed loop D.C. drive.

3.2. Simulating the Forces on the Steel Bar.

In order to simulate the forces involved in rolling steel, a good understanding of the mechanical and material properties of metals is required. There are numerous books on this subject and in the next section those used to obtain this information are reviewed. All the equations and information gathered is then brought together and used to derive the equations used to simulate the forces. in section 3.2.2.

3.2.1 Literature Review.

All materials respond in a certain way to an applied stress. The concepts of stress and the often resulting strain are introduced and discussed in the book *Mechanical Working of Metals: Theory and Practice*[7] by J.N.Harris.

The different types of behaviour under stress, mentioned in section 3.2.2, are rigid, elastic, plastic and fracture behaviour. These types of behaviour define the amount of strain a metal undergoes when a stress is applied. All metals react in a combination of these behavioural types. J.N.Harris explains that most working of metals is done in the plastic-elastic region of the stress-strain curve.

An in-depth study of the deformation of the metal crystal is given by R.W.K.Honeycombe in his book, *The Plastic Deformation of Metals*[8]. The properties of metals and the reasons for it's behaviour are explained. The stress-strain curves for different crystals in a number of different situations are given.

J.N.Harris shows that the stress-strain relationship is linear over the elastic-plastic region of the deformation curves of many metals. This slope of this linear relationship is described by a constant known as Young's Modulus. A list of Young's Moduli for different metals, including steel, is included in J.N.Harris' book.

However, this value cannot be used in this simulation for the following reasons.

Firstly, as is shown in *Principles of Industrial Metalworking*[11] by G.W.Rowe, the Young's Modulus of a metal depends on the temperature at which the metal is being processed. Therefore, the results taken at room temperature would have to be adjusted for the rolling temperature of approximately 1000^{o}C. (Appendix A presents the rolling temparatures measured after each stand in the Hille Mill).

The value of Young's Modulus also depends on the speed at which the metal is being processed. This is shown in *Developments in High Speed Metal Forming*[10] by R.Davies and E.R.Austin. The effects of high speed processing on the stress-strain relationship and, therefore, on Young's Modulus are explained. This is done for various metals including steel. The effects of friction, temperature, stress wave and inertia forces are also described in this text.

Hence, the value of Young's Modulus depend's on a number of factors. Since, the value of this constant for the steel that Scaw Metals rolls (Appendix E) is not reported in the literature[7-11,13,14], the value used is that of another steel rolled at this temperature.

Having determined a value of Young's modulus, the effect of the rolling process on the metal must be determined. The main aspects are the forces in the roll gap, the friction force, the rolling loads, the roll torque and the determination of roll pressure.

The equations and laws governing these subjects are presented and explained in four of the books used[7,9-11]. The most complete study of the rolling process is given in G.W.Rowe's book, *Principles of Industrial Metalworking Processes*[11].

The minimum and maximum reduction rates possible in a pass through a set of rollers are calculated in this text. These rates limit the choice of the cross-sectional areas of the steel after each stand in the simulation package. (Appendix F gives the cross-sectional areas used in the simulation)

Using the knowledge and equations obtained from these books, the laws governing the response of the metal to rolling speeds and conditions are derived in the following section.

3.2.2 Designing the Tension Simulator.

One of the aims of a controller for this process must be to ensure tension-free rolling. Figure 3.5 shows three stands, the position of the steel and the tensions therein. The tension is calculated in terms of the force of compression or expansion being exerted on that section of steel. These forces are labelled F_{L1} and F_{L2}.

In order to determine the effects of rolling forces on the load torque, M_l, it must be calculated in terms of these forces. They in turn must be expressed as a function of the speeds of the drives.

Therefore, it will be possible to simulate the load torque on a drive at any time during the rolling process, using the mechanical and material properties of steel and the speed of the drives.

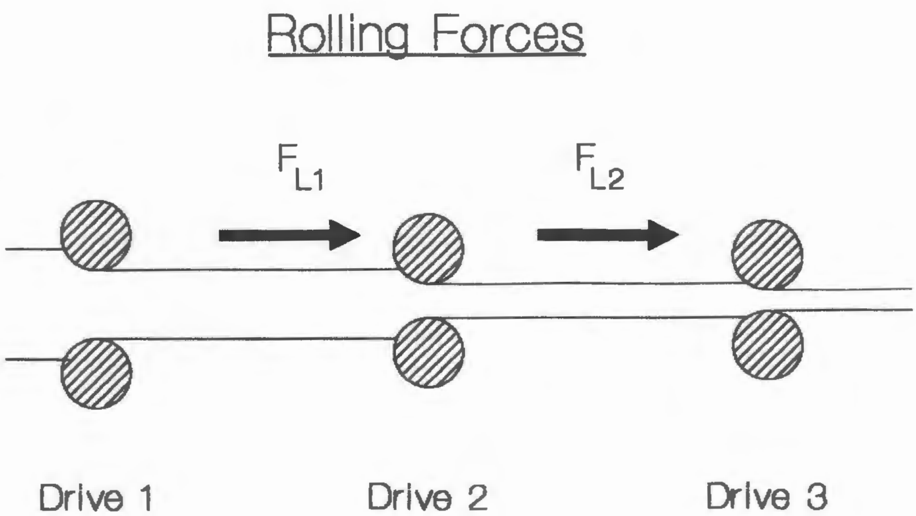


Figure 3.5: A schematic showing tensions, F_{L1} & F_{L2}.

These forces are due to tension in the metal caused by the relative speeds of the drives being mismatched. There are two possible situations.

Firstly, if drive 2, figure 3.5, is running too fast relative to drive 3, it will cause the metal to be compressed. The added force required to compress the metal will cause the drive to decelerate. However, the speed control system discussed in section 2.1 will try to maintain the set speed when the load increases. This requires a larger torque output to counteract this decelerating torque.

This force will, on the other hand, accelerate drive speed 3. The speed control system will keep this speed constant by decreasing the torque output of the drive. the load torque on the motor will be less.

Secondly, if drive 2's speed is too high compared to the previous stand, the force will be one of extension. Drive 2 will pull the metal and the added load torque required to extend the metal will cause the motor to decelerate. As before, the speed control system will counteract this by providing the extra torque required to maintain the speed. The load torque on the motor will increase.

This force will, on the other hand, accelerate drive speed 1, since the steel will be pulled through it. The speed control system will keep this speed constant by decreasing the torque output of the drive. The load torque on the motor will be less.

Therefore, the load torque will increase if a drive's speed is too fast relative to the drives on either side of it, and similarly it will decrease when the drive is running too slowly.

The work required to roll the steel billet down in dimension is assumed to remain constant in this study[9,11]. This is assigned the label M_c.

Therefore, M_1 [Nmm] will be equal to the constant load torque minus the accelerating torque, T_{11} [Nmm], plus the decelerating torque, T_{12} [Nmm]. (T_{11} is an accelerating torque due to the definition of the direction of force F_{11}, see figure 3.5)

$$M_l = M_c - T_{l1} + T_{l2} \qquad ...(3.2.1)$$

The torque required to exert a particular force at the edge of the roller depends on the radius of the roller[9,11,12]. The equation describing this is:

$$T = r \times F \qquad ...(3.2.2)$$

This is illustrated in Figure 3.6.

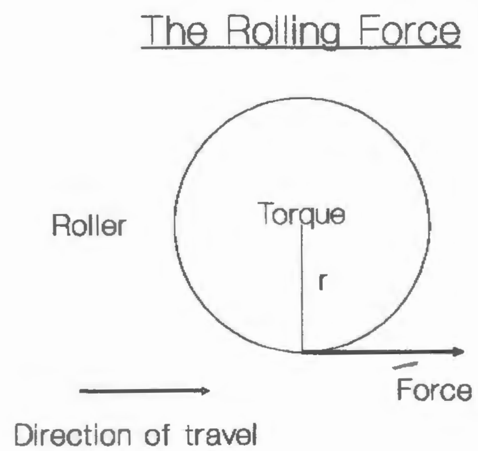


Figure 3.6: The torque-force relationship.

The force required to compress a length of metal depends on it's mechanical and material properties. Of fundamental importance is the stress/strain diagram for the material.

The stress [N/mm^2] in a metal is related to the force exerted on it and it's cross-sectional area[7,9,11]. (Eq 3.2.3) stress = force / area ...(3.2.3)

The strain in a metal is determined from the amount that the metal has been extended/compressed as a factor of it's total length[7,9,11]. (Eq. 3.2.4) Strain is a ratio and has no units.

strain = extension / length ...(3.2.4)

Depending upon the properties of the material various types of behaviour are observed.

(1) *Rigid behaviour:* In this behaviour the metal undergoes no strain under applied stress.
(2) *Elastic behaviour:* This is the situation when a body undergoes deformation under applied stress, but returns to it's original shape once the stress is removed.
(3) *Plastic behaviour:* This implies that a body undergoes deformation when subjected to a stress and that the deformation (strain) remains after the stress is removed.
(4) *Failure:* When the material ruptures into two parts.

Actual materials will deform in a combination of these individual types of behaviour. Metals tend to behave in an elastic-plastic manner.

Figure 3.7 shows such a stress/strain diagram[7]. This diagram has certain features labelled A, B, etc.

A. *Yield point* - The stress value at which behaviour of the metal changes from elastic to elastic-plastic.

B. *Ultimate tensile stress* - The maximum stress value obtainable.

C. *Young's Modulus* - This gives the relationship between stress and strain over the elastic portion of the diagram. According to Hooke's Law, it describes a linear relationship.

D. *Fracture stress* - The stress at which the metal fractures.

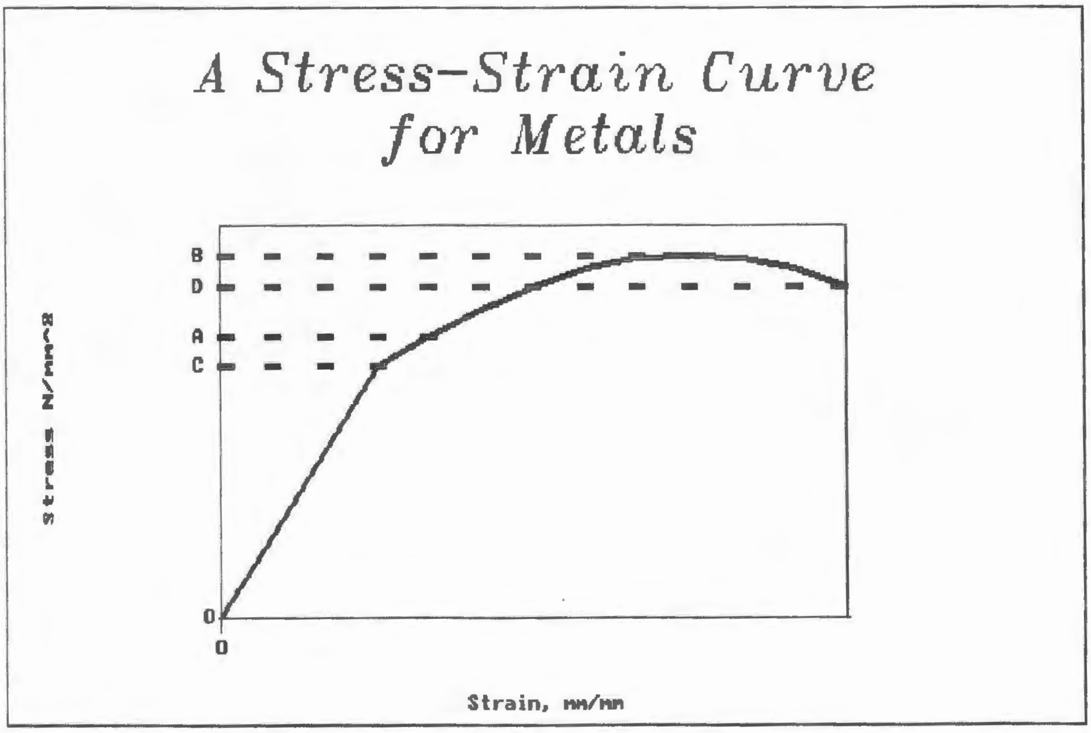


Figure 3.7: A common stress-strain curve.

For the majority of metals, including steel, the relationship between stress and strain is linear up

For the majority of metals, including steel, the relationship between stress and strain is linear up to point C on figure 3.7. Young's Modulus of a metal describes the slope of the curve up to this point. It's unit of measurement is the same as stress, i.e. Nmm^{-2}, and is given by

$$\text{Young's Modulus} = \frac{\text{stress}}{\text{strain}} \qquad \ldots(3.2.5)$$

Young's modulus, E_y, is different for various metals and varies with temperature. Table 3.1[7] shows E_y [Nmm^{-2}] for different metals.

Young's Moduli

Metal	E	Metal	E
Al	670	Mg	435
Cr	2410	Mn	1540
Co	2000	Ni	2000
Cu	1070	Ag	735
Steel	2000	Sn	400
Pb	174	H11 steel	2050

Table 3.1: Young's Modulus for different metals.

These Young's moduli are measured at room temperature and therefore have to be altered for steel at the rolling temperature[10]. The temperature of the steel varies depending on it's position in the rolling process. The temperature was measured after each stand at Scaw Metals and was found to be approximately 1000^{o}C. The results are presented in Appendix A.

The stress strain curves for the exact type of steels that Scaw Metals uses in the Hille Mill (See Appendix E) are not reported in the literature on this subject[7-11,13,14]. However, the stress-strain curves at different temperatures for Nb-V steel (BS 4360 GR 55EE) are shown in figure 3.8[15].

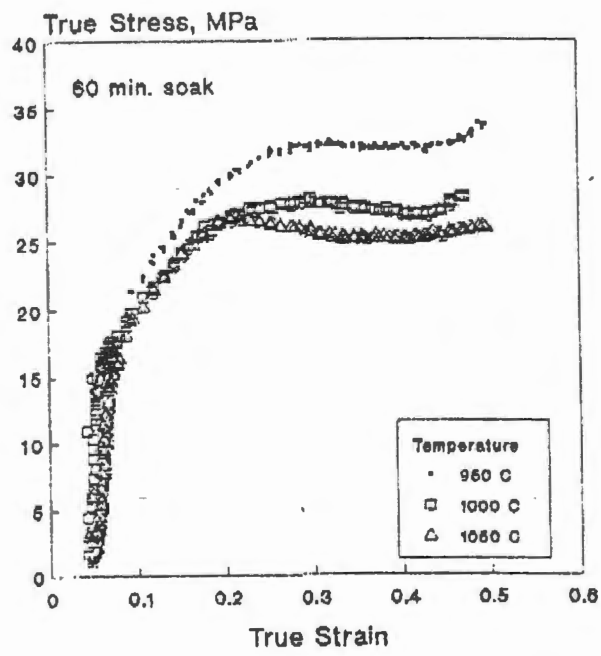


Figure 3.8: Stress-strain curves for Nb-V steel[15].

Figure 3.9 shows the variation in the Young's moduli between different types of steel versus temperature[13,14]. The variation in these results is approximately 40 kN/mm^2. This is a variation of approximately 25% in Young's Modulus.

The Young's Modulus used in this study is therefore, the value obtained from the stress-strain curve for Nb-V steel (at 1000^{o}C). This value (400 MPa or N/mm^2) is obtained from the slope of the linear deformation region discussed earlier.

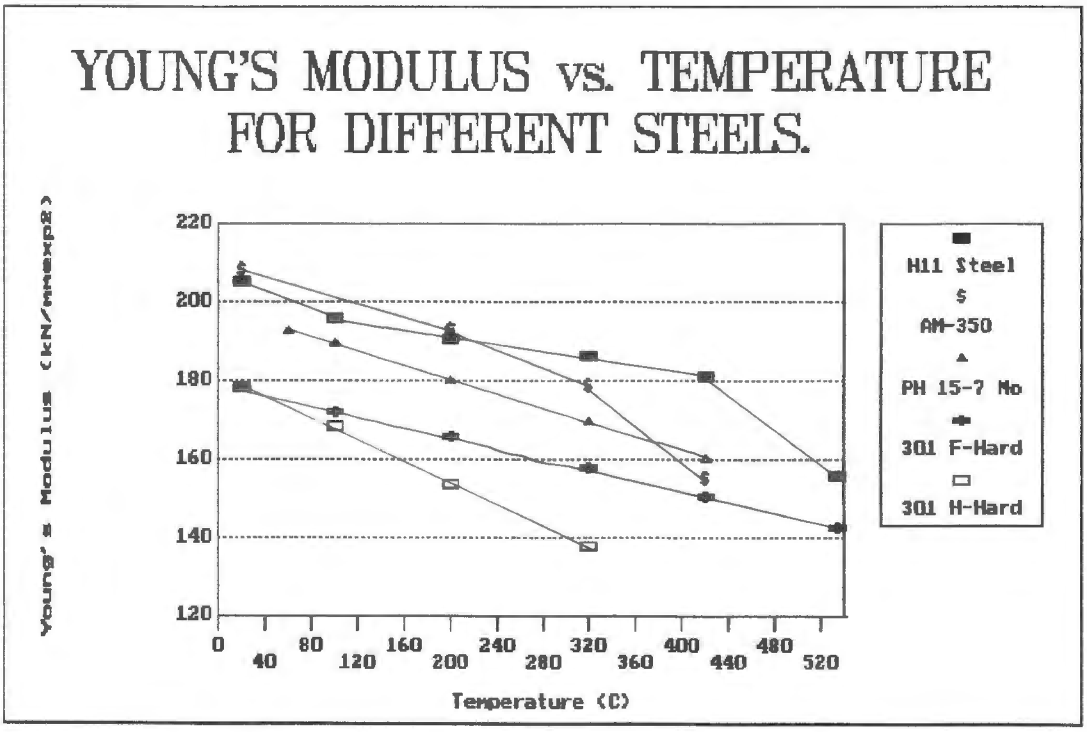


Figure 3.9: Young's Modulus vs. temperature for different steels

Young's Modulus varies with the rate of strain and is therefore slightly different at high speeds[10]. Since this is not the exact type of steel rolled and the fact that it has not been altered for high speeds[10], the effect of varying the value of Young's Modulus used in the simulator package on the controller performance is investigated and reported on in sections 6.1.2 and 6.2.2.

Compression and extension of a metal give identical stress/strain curves over small changes in length[7]. Eq. 3.2.5 can be rewritten using equations 3.2.3 and 3.2.4.

$$E = \frac{F * l}{A_o * dl} \qquad \text{...(3.2.6)}$$

where: F is the force exerted [N], A_o the cross-sectional area [mm^2], l the original length [mm] and dl the change in length of the metal piece [mm].

Rewriting this, the force [N] required to extend a metal is governed by equation 3.2.7.

$$F = E * A_o * \frac{dl}{l} \qquad \text{...(3.2.7)}$$

Figure 3.10 is a schematic showing the cross-sectional areas at different places in the rolling process.

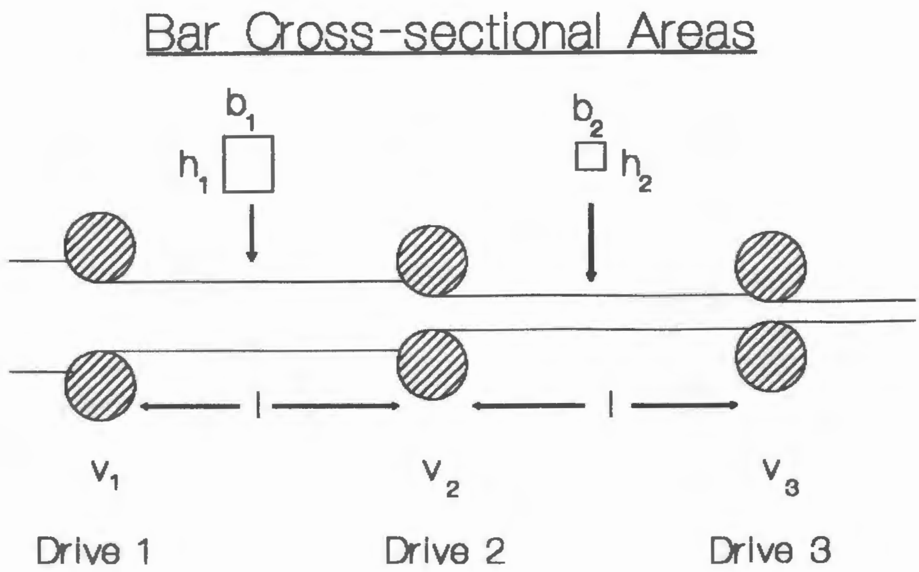


Figure 3.10: Cross-sectional areas in the rolling process

The change in the length of a section of metal is related to the speeds of the drives at either end of it. For the volume of metal between the two stands to stay constant the volume entering the gap must be equal to the volume leaving. The volume of metal entering, V_e [mm^3], in a unit of time dt [s], is given by the following equation:

$$V_e = (b_1 * h_1) * v_1 * dt \qquad ...(3.2.8)$$

"v_1" is the speed [mm/s] of the metal travelling through drive 1 in figure 3.10. Therefore the change in volume in the same time period, dV [mm^3], is calculated using equation 3.2.9.

$$dV = ((b_1 * h_1) * v_1 - (b_2 * h_2) * v_2) * dt \qquad ...(3.2.9)$$

The change in length, dl , compared to the actual length, l, is assumed to be small in this simulation. This is because the controller will attempt to keep the tension small and ideally zero. Therefore, there will be a negligible change in the cross-sectional area of the metal under strain, since this only occurs under increasing deformation conditions. Volume is equal to length multiplied by area. Therefore, since area has been assumed to remain constant, Eq. 3.2.9 becomes.

$$dl = \frac{((b_1 * h_1) * n_1 - ((b_2 * h_2) * n_2)}{b_1 * h_1} * dt \qquad ...(3.2.10)$$

Substituting into equation 3.2.7 and then into 3.2.2 the expression relating the torque to the speeds of the drive and the properties of the metal is derived.

$$T = T_o + F * r$$

$$T = T_o + E * \frac{((b_1 * h_1) * n_1 - (b_2 * h_2) * n_2)}{l} * dt * r \qquad ...(3.2.11)$$

The load torque, M_1, is calculated using the values for T_{11} and T_{12} determined using this equation. The effects of the tension on the rolling process are modelled in this simulator using the equations and constants described in this section.

3.3. Modelling the Loop Height.

The metal bar is looped between the last five stands of production line A. This forms four loops, the heights of which must be controlled. This is shown in the plant schematic in figure 2.1.

The reason that the loop is formed is so that there is no tension in the metal while it is between two stands. As has been stated before this is an efficient way of ensuring products of high quality.

To simulate the loop height and it's time response, it is assumed to be roughly the shape of an isosceles triangle. This is shown in figure 3.11.

The height of the loop is therefore assumed to be approximated by the height of the apex of the triangle, h [mm] in figure 3.11. The height is related to the two other variables of the triangle, a and b [mm], according to the following equation defined by Pythagoras' theorem.

$$h^2 = a^2 - \frac{b^2}{4} \qquad \text{...(3.3.1)}$$

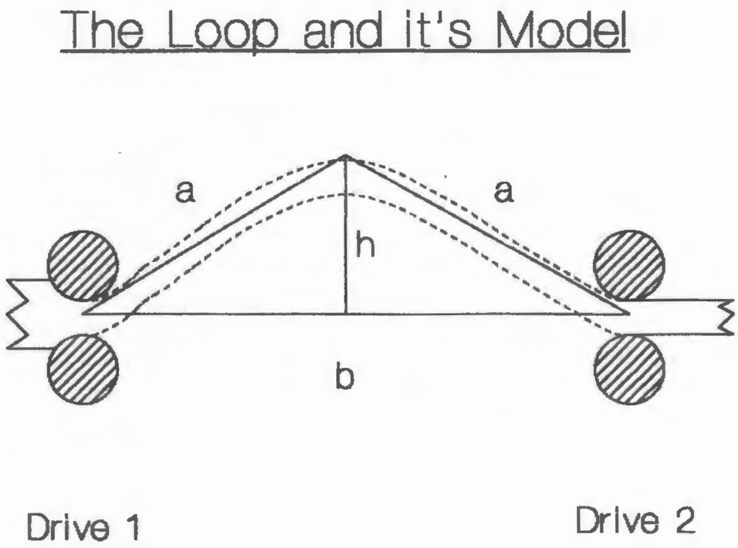


Figure 3.11: The loop and it's model.

The height of the loop after a unit of time is calculated by using the following equation. The variable da [mm] describes the change in length, a, in that unit of time. (Note: b is assumed to stay constant)

$$h = \left[(a + da)^2 - \frac{b^2}{4} \right]^{0.5} \qquad \ldots(3.3.2)$$

The height of the loop can be related to the speeds of the two drives between which it is formed. The triangle variable, a, is half the length of the metal between the two drives.

$$l = 2 * a \qquad \ldots(3.3.3)$$

The length [mm] of metal between the two drives depends on their relative speeds. If the volume of steel entering the loop, dV_1 [mm^3], is equal to the amount leaving, dV_2, then the volume of metal will be constant and dV will be zero.

$$dV = dV_1 - dV_2 \qquad ...(3.3.4)$$

Since volume is equal to length times area, Eq. 3.3.4 can be rewritten as

$$A_1 * \frac{dl}{dt} = A_1 * \frac{dl_1}{dt} - A_2 * \frac{dl_2}{dt} \qquad ...(3.3.5)$$

In Eq. 3.3.5, A_1 and A_2 are the cross-sectional areas [mm^2] of the steel billet as it leaves the respective drive. This is shown in figure 3.12.

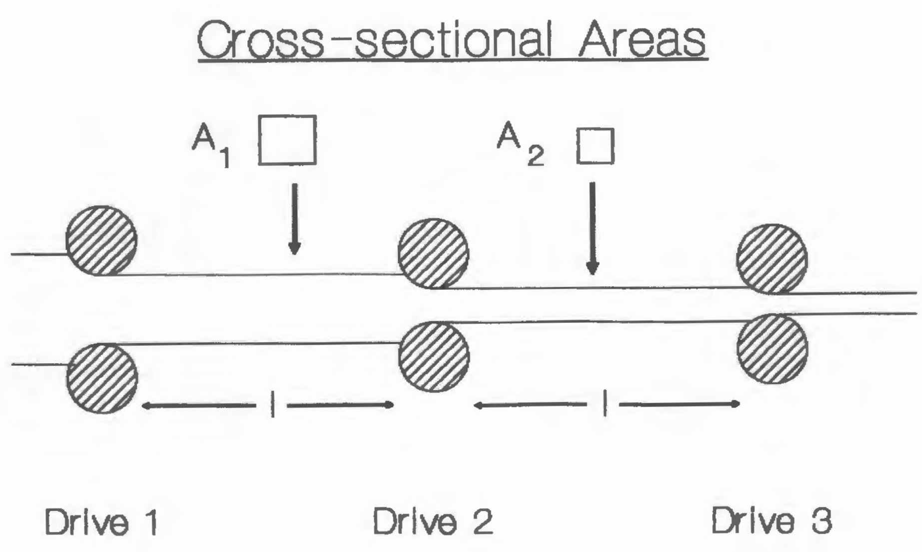


Figure 3.12: The different cross-sectional areas

In equation 3.3.5, dl_1/dt and dl_2/dt are the velocities of the metal between the rollers of stands 1 and 2, v_1 and v_2 [mm/s], respectively. Substituting for these and rewriting this in terms of the change of length, dl [mm], in a unit of time, dt [s], an expression relating the change in the length of metal between the drives to their speeds is derived (Eq. 3.3.6).

$$dl = \left[v_1 - \frac{A_2}{A_1} * v_2 \right] * dt \qquad \ldots(3.3.6)$$

As stated in Eq.3.3.3, a is half the length, l. Therefore, da, the change in length a per unit time, is half the value of dl. By substituting for da in Eq.3.3.2 an expression for calculating the height [mm] after time dt, is given by Eq.3.3.7.

$$h = \sqrt{\left[a + \left[v_1 - \frac{A_2}{A_1} * v_2 \right] * \frac{dt}{2} \right]^2 - \frac{b^2}{4}} \qquad \ldots(3.3.7)$$

Since the speeds of the drives are calculated in the D.C. drive simulator described in section 2.1, the four loop heights can be calculated using these speeds. The cross-sectional areas of the billet at various points in line A are chosen to be those shown in Appendix F.

As can be seen from equation 3.3.7, the loop height is a non-linear function of the speed , v_1. The relation between v_1 and the input (the speed setpoint to drive 1, figure 3.11) that controls it, is linear. Thus the height is a non-linear function of the input that is used to control it. In order that linear control theory can be used to design a controller, the height output is linearised in the manner described in section 5.2.1.

3.4: The Simulation Time Scale.

The amount of time that the bar spends in each of the rollers and the time period over which the production line is rolling are simulated using real plant data.

3.4.1: The Stand Rolling Period.

The time that the bar is between the rollers of a stand is illustrated by the figures I1 - I3, Appendix I. This is real plant data taken from the third stand in production line A.

As is shown in these graphs the stand is actively rolling the bar for approximately 7-9 seconds. The entrance of the bar into the stand is marked by the sudden increase in the drive current, due to the increase in the load torque. Similarly the drive current drops immediately after the bar leaves the stand.

The simulator has been written so that the bar spends about 8.4 seconds in each stand. This is illustrated by figure 5.9 which shows all the input and output variable responses of the plant during the rolling of one billet. Again the bar's entrance into and exit from the first three drives are marked by the sudden changes in the drive current.

3.4.2: The Complete Rolling Cycle.

The speed of each drive is determined by the thickness of the bar between the stands. This is in order to maintain the correct bar tension throughout the process, as explained in section 3.2.2., and to achieve a metal output speed[16] of 30 m/s^{-1}.

The final product chosen in this simulator is 70 x 30 mm flat bar[16] and the intermediate bar cross-sectional areas are given in Appendix F. The required drive speeds are calculated according to equation 3.2.11 and these are given in Appendix G.

The complete rolling of a single flat bar is illustrated by the process graphs in figure 5.9. This figure illustrates that the complete process takes about 10.8 seconds.

The time that the bar enters the first stand is the beginning of the cycle. The bar enters the eighth stand 2.4 seconds later. The billet is then between all eight stands for 6 seconds and it, similarly, takes 2.4 seconds to leave the production line. The process cycle begins again with the entrance of another bar into the first stand.

For the purpose of this study it is assumed that there is only ever one bar in the production line. There are sometimes two different billets in the production line at once in the Hille Mill. This increases the number of binary interaction matrices for the process cycle, as calculated in Chapter 4. The control strategy of switching controllers explained in Chapter 6 would then just switch between more control structures during the cycle.

The simulation package on which the control systems (designed in chapter 6) are tested is written according to the mathematical models explained in this chapter. (A full listing of the simulation package is given in Appendix L) These mathematical models are used rather than the transfer functions derived from actual plant data for the reasons discussed in chapter 5 of this thesis.

In chapter 5 the simulator is used to derive all the transfer functions for the plant and these are used later in the controller design. However, before deriving transfer functions it is very useful to analyse the plant interaction and how it changes depending on the position of the bar in the production line. The next chapter investigates the changing plant interaction matrix and the resulting control structures that are required for this process.

References.

1. Say, M.G. and Taylor, E.O. (1980), *Direct Current Machines.*, Pitman Publishing, London.
2. Steven, R.E. (1983), *Electrical Machines and Power Electronics.*, Van Nostrand Reinhold, England.
3. Hindmarsh, J. (1970), *Electrical Machines and their Applications.*, Pergamon Press, Oxford.
4. Fitzgerald, A.E., Kingsley, C.Jr. and Umans, S.D. (1985) , *Electric Machinery.*, 4th ed., McGraw-Hill, Singapore.
5. Hill, R.J. and Lou, F.L. (1989), *Microprocessor-Based Control of Steel Rolling Mill Digital DC Drives.*, IEEE Trans. Power Electronics, Vol. 4, No. 2, pp 289 - 297.

6.Hill, R.J. (1990), *Mathematical Modelling in Undergraduate D.C. Machine Drive Projects.*, Int. J. Elect. Enging. Educ., Vol. 27, pp 197 - 217.

7.Harris, J.N. (1983), *Mechanical Working of Metals: Theory and Practice.*, Pergamon Press, Oxford.

8.Honeycombe, R.W.K. (1984), *The Plastic Deformation of Metals.*, 2nd ed., Edward Arnold, London.

9.Avitzur, B. (1968), *Metal Forming: Processes and Analysis.*, McGraw-Hill, New York.

10.Davies, R. and Austin, E.R. (1970), *Developments in High Speed Metal Forming.*, The Machinery Publishing Co., Sussex.

11.Rowe, G.W. (1977), *Principles of Industrial Metalworking Processes.*, Edward Arnold, London.

12.Halliday, D. and Resnick, R. (1988), *Fundamentals of Physics.*, 3rd ed., Wiley, New York.

13.American Society of Metals (1978), *Metals Handbook: Vol. 1: Properties and Selections: Irons and Steels.*, Ninth Edition, ASM printers.

14.American Society of Metals (1978), *Metals Handbook: Vol. 3: Properties and Selections: Stainless Steel, Tool Materials and Special-Purpose Metals* , ASM printers.

15.Banks, K.M. (1991) *Private Communication*, Materials Engineer, Iscor Research and Development, Pretoria.

16.Welgemoed, K. (1991) *Private Communication,* Electrical Engineer, Hille Mill, Scaw Metals Ltd., Germiston, Transvaal.

CHAPTER 4: CONTROL STRUCTURES

All the controllers that are used to control a plant at a particular time, make up a control structure. Control structures can consist of a mix of single-variable, feedforward and multivariable controllers.[1] Appendix H gives a full coverage of the theory of control structures.

A control structure for a plant is derived from it's binary interaction matrix (B.I.M.). The method of derivation is described in section 4.2.

The theory of control structures is very applicable to the rolling mill situation. When using binary interaction matrices it becomes apparent that the control structure of production line A changes. The structure is found to depend on the position of the bar in the mill.

The binary interaction matrix is explained in the next section and all the possible B.I.Ms are derived. Section 4.2 concentrates on applying control structure theory to all these B.I.Ms.

4.1: The Binary Interaction Matrix.

A binary interaction matrix illustrates which plant inputs have an effect on (or interact with) which outputs. The B.I.M(s) for a plant can be drawn up solely from plant knowledge without any attempt at transfer function derivation.

The plant inputs are labelled along the columns of the matrix, while the plant outputs are represented by the rows. An example is shown in figure 4.1.

B.I.M. (Metal in Drives 1-8)

Inputs:

S : Speed

Outputs:

A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1	x	x	x	x	x			
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.1: A binary interaction matrix.

All the plant input and output variables are discussed in section 2.2. As is established there, the input variables that are used to control the plant simulator in this study, are:

Inputs:

1. Eight D.C. drive speed setpoints, S1 - S8.

The outputs that are controlled have been chosen (for the reasons mentioned in chapter 2) as:

<u>Outputs:</u>

1. Three current outputs (from the first three drives), A1 - A3.
2. Four loop heights, L1 - L4.
3. One D.C. drive speed (Drive eight), S1.

Therefore this plant has an eight by eight interaction matrix. An 'x', in figure 4.1, means that that output (shown by the row label) is affected by a change in that input (shown by the column label). No 'x' implies that the output indicated is not altered by a step in that input.

The matrices that follow illustrate a complete cycle. A cycle starts with no metal in the production line at all. The steel bar then enters the stands in sequential order until there is metal in all eight stands. The bar then leaves the stands one by one and passes completely through the production line to complete the cycle.

During this cycle the plant interaction depends on the position of the steel bar in the production line. Each matrix shows how the interaction changes when the bar enters or leaves one of the stands.

The reasons for a particular interaction occurring or not occurring in a stage of the cycle are fully explained below each matrix.

Figure 4.2 shows the B.I.M. when there is no steel bar in the production line. Since there is no metal, there will be no loop height outputs. Hence none of the inputs affect any of the loop height outputs, L1 - L4.

There is no interaction in the plant. This is due to there being no metal bar present. The three current outputs and the drive speed output are only affected by the inputs that are used to control them.

B.I.M. (No Metal)

Inputs:	Outputs:
	A : Ammeter
S : Speed	L : Loop Height
	S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2								
L3								
L4								
S1								x

Figure 4.2: The B.I.M. when line A is empty.

B.I.M. (Metal in Drive 1)

Inputs:	Outputs:
	A : Ammeter
S : Speed	L : Loop Height
	S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2								
L3								
L4								
S1								x

Figure 4.3: The B.I.M. when metal is in Drive #1.

Figure 4.3 illustrates the B.I.M. when the metal bar enters the first stand. The matrix does not change from the previous case, however the current consumption in the first drive , A1 in figure 4.3, increases. This is due to the increase in the load torque caused by the work required to roll the metal between the rollers of the first stand. The load torque is considered a disturbance to the accelerating torque, as is shown in figure 3.2. The transfer function, between input S1 and output A1, is indicated by means of a 'x' in the (1,1) element of the matrix.

There is still no interaction between any drives in the plant since the steel bar does not extend between any two of the stands.

B.I.M. (Metal in Drives 1&2)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x						
A2	x	x						
A3			x					
L1								
L2								
L3								
L4								
S1								x

Figure 4.4: The B.I.M. when metal is in Drives 1&2.

Figure 4.4 is the B.I.M. for the plant when the steel bar enters the second drive in the production line. As is shown, the matrix is different from the previous case when the metal is only in the first stand.

The current output, A1, of the first drive is now not only affected by the input that is used to control it, S1, but also by the input to the second drive, S2. This is because a step in the output speed of the second drive will change the load torque on the first drive, as explained in section 3.2.

The current required by the first drive to maintain a constant speed will, therefore, change as well. Thus a step in the speed setpoint (input) of the second drive affects the current consumption (output) of the first drive. This interaction is reflected in the matrix by a 'x' in the (1,2) element.

The speed of the first drive similarly affects the current in the second drive. An 'x' in the (2,1) element of the B.I.M. reflects this interaction. Hence, output A2 is also affected by both inputs, S1 and S2.

The metal bar then enters the third stand in the line. The interaction matrix is shown in figure 4.5. There is now interaction between the first three stands of rollers.

The current output of the first drive, A1, is altered by a change in the speed setpoint of the second or third drive. Similarly A2 and A3 are also affected by step changes in the inputs of the other two drives.

B.I.M. (Metal in Drives 1-3)

Inputs:	Outputs:
	A : Ammeter
S : Speed	L : Loop Height
	S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x					
A2	x	x	x					
A3	x	x	x					
L1								
L2								
L3								
L4								
S1								x

Figure 4.5: The B.I.M. when metal is in Drives 1-3.

This cascading interaction is due to the tension in the bar caused by a speed mismatch. As already discussed in section 3.2, the load torques of adjacent drives are affected by a change in the speed of a drive. Although the inner speed control loop, figure 2.2, will alter the current consumption in a drive in order to maintain a set speed, it will never be able to get the output speed exactly equal to the setpoint if an adjacent drive is kept at a mismatched speed. This is because the load torque on the drive will be changing continually.

Therefore a step change in the speed of the third drive causes the second drive's speed to be mismatched with respect to the first drive's speed. Thus, the load torque on the first drive is altered and the current output, A1, also changes.

Hence, speed changes in any of the three drives, in which there is metal, will disturb the current outputs of the other two drives. These interaction effects are shown in figure 4.5.

B.I.M. (Metal in Drives 1-4)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1								
L2								
L3								
L4								
S1								x

Figure 4.6: The B.I.M. when metal is in Drives 1-4.

Figure 4.6 is an illustration of the interaction when the steel bar enters the fourth stand. The three current outputs, A1 - A3, are all affected by the four speed inputs, S1 - S4, in the cascaded manner described above.

The steel does not extend to the fifth stand yet and thus there is no metal loop between these two stands. The height output after stand 4 is, therefore, not affected by any input change.

Once the metal enters the fifth drive, a loop is formed between stands four and five.

The loop height changes if the speeds of the stands either side of it are not matched. Thus, a step in either of the speed setpoints of drives four and five will affect the output of the first loop height, L1. Since a speed change in any of the first three drives alters the speed of the fourth drive, the loop height will be also be disturbed in this manner.

This added input-output interaction is shown in figure 4.7. The change from the previous B.I.M. is the interaction shown in the fourth row, labelled L1.

B.I.M. (Metal in Drives 1-5)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1	x	x	x	x	x			
L2								
L3								
L4								
S1								x

Figure 4.7: The B.I.M. when metal is in Drives 1-5.

The second metal loop is formed between stands five and six when the steel enters the sixth drive. This metal loop is only affected by the speeds of the two drives on either side of it. This interaction is shown in figure 4.8.

B.I.M. (Metal in Drives 1-6)

Inputs:	Outputs:
	A : Ammeter
S : Speed	L : Loop Height
	S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1	x	x	x	x	x			
L2					x	x		
L3								
L4								
S1								x

Figure 4.8: The B.I.M. when metal is in Drives 1-6.

The second loop height, L2, is not disturbed by a change in the speed of any of the first four drives. The height of the first loop changes when one of these speeds is altered. Thus, the speed of the fifth drive is not affected and hence nor is L2.

Figure 4.9 is the B.I.M. when steel is in drives 1-7. The third loop of steel is formed between stands six and seven. This loop height is also only disturbed by a change in the speed setpoints of the drives on either side of it. i.e S6 and S7.

Hence, the only alterations to the matrix from the previous situation are the interactions indicated in row six, representing the third loop height L3.

B.I.M. (Metal in Drives 1-7)

Inputs:

S : Speed

Outputs:

A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1	x	x	x	x	x			
L2					x	x		
L3						x	x	
L4								
S1								x

Figure 4.9: The B.I.M. when metal is in Drives 1-7.

The metal then enters the last stand. It is now in all eight stands in the production line. The fourth and last loop is formed between stands seven and eight.

This loop height is also only affected by the drive speeds on either side of it. The reasons for this are the same as those explained for the second and third loops.

The last output, the speed of the eighth drive, is only altered by a change in the speed setpoint to that drive. A step in the input to the seventh drive does not disturb this speed because the height of the fourth loop will vary and absorb any speed mismatch.

The interaction matrix for the situation where metal is in all eight stands is illustrated by figure 4.10. The only difference in this B.I.M. is that L4 is now affected by two inputs, S7 and S8.

B.I.M. (Metal in Drives 1-8)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x	x	x	x				
A2	x	x	x	x				
A3	x	x	x	x				
L1	x	x	x	x	x			
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.10: The B.I.M. when metal is in Drives 1-8.

The above matrix describes the interaction of the production line until the metal starts to leave the first stand. The steel will then stretch between drives two to eight. When there is no metal between two stands, a change in the speed output of one drive cannot have any effect on the speed or current of the other.

Thus, the current output of drive 1 is not altered by a step in any of the speed setpoints to the next three drives. Similarly, a step in the input to Stand #1 does not cause a change in any output other than it's own current consumption.

These alterations in the plant interaction are reflected in figure 4.11, the B.I.M. for the situation where there is metal in drives two upto eight.

B.I.M. (Metal in Drives 2-8)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x	x	x				
A3		x	x	x				
L1		x	x	x	x			
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.11: The B.I.M. when metal is in Drives 2-8.

The next change in the matrix occurs when the bar leaves the second stand. This change is illustrated by the B.I.M. in figure 4.12.

Since the bar is no longer in the second set of rollers, a step in the speed input to Stand #2 does not cause a change in any other output but A2, it's current.

For the same reason the output A2 remains undisturbed when the inputs to any of the other drives are stepped.

B.I.M. (Metal in Drives 3-8)

Inputs:

S : Speed

Outputs:

A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x	x				
L1			x	x	x			
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.12: The B.I.M. when metal is in Drives 3-8.

B.I.M. (Metal in Drives 4-8)

Inputs:

S : Speed

Outputs:

A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1				x	x			
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.13: The B.I.M. when metal is in Drives 4-8.

Figure 4.13 gives the changes in the interaction of the production line when the steel bar leaves the third stand. The height of the first loop is now only affected by the speeds of the fourth and fifth drives. The speed of the fourth stand, and hence L1, is no longer disturbed by a step change in any of the setpoints to the first three drives.

B.I.M. (Metal in Drives 5-8)

Inputs:	Outputs:
	A : Ammeter
S : Speed	L : Loop Height
	S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2					x	x		
L3						x	x	
L4							x	x
S1								x

Figure 4.14: The B.I.M. when metal is in Drives 5-8.

The height of the first loop is then slowly reduced to zero before the steel leaves the fourth stand. This is to prevent the end of the metal bar from whipping when it travels between the fourth and fifth stands.

After leaving the fourth stand, there is, once again, no height output from the first loop. This change is reflected in figure 4.14.

The second loop is reduced in a similar manner to the first to prevent whipping of the bar. This is done while the steel stretches between the Stands 5 - 8, and the speed setpoint, S5, still controls this output.

The bar then leaves Stand #5. The loop height, L2, does not exist any more and, therefore, remains unaffected by a step change in any of the eight speed setpoints. This is shown in figure 4.15.

B.I.M. (Metal in Drives 6-8)

Inputs: S : Speed

Outputs: A : Ammeter, L : Loop Height, S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2								
L3						x	x	
L4							x	x
S1								x

Figure 4.15: The B.I.M. when metal is in Drives 6-8.

The third loop, L3, is similarly reduced to zero before the steel leaves Stand #6. The binary interaction matrix becomes that shown in figure 4.16 when the metal is only in stands seven and eight. The fourth loop is the only loop in the process at this stage.

B.I.M. (Metal in Drives 7&8)

Inputs:
S : Speed

Outputs:
A : Ammeter
L : Loop Height
S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2								
L3								
L4							x	x
S1								x

Figure 4.16: The B.I.M. when metal is in Drives 7&8.

For a short time the steel will only be in the rollers of the eighth stand. In this situation the B.I.M. is the same as for when there is no metal in the production line at all. This is illustrated in figure 4.17.

The only difference between these two situations is that the load torque on the motor changes. This will alter the response of Drive #8, since the load torque is considered a disturbance to the acceleration torque of the motor, as is shown in figure 3.2.

A complete cycle of the rolling process has now been completed with the passage of the bar completely through the production line. The next cycle starts with the entrance of another billet into the first stand.

B.I.M. (Metal in Drive 8)

Inputs: S : Speed

Outputs: A : Ammeter, L : Loop Height, S : Speed

	S1	S2	S3	S4	S5	S6	S7	S8
A1	x							
A2		x						
A3			x					
L1								
L2								
L3								
L4								
S1								x

Figure 4.17: The B.I.M. when metal is in Drive 8.

The above interaction matrices show all the different forms that interaction can take in this plant when a bar goes through the rolling process.

These B.I.Ms are used in the following section to determine the best control structure to use on this plant. As is shown the required control structure changes according to where in the production line the bar is. The need for switching control structures becomes apparent in section 4.2.

4.2.: The Hille Mill Control Structures.

On systems with many inputs and outputs, it is often useful to analyse the structure of the process before deriving transfer function models.

Structural analysis is carried out on the binary interaction matrices derived in section 4.1. The inputs and outputs are re-ordered in such a way that the matrix becomes as lower triangular as possible. One of the ways of conducting the re-ordering is to apply the following steps to the matrices.

1. Count the number of 'x's in each row.
2. Re-arrange the matrix so that the output row with the highest number of 'x's is the bottom row of the matrix.
3. Continue to arrange all the output rows from the bottom in descending number of 'x's.
4. Count the number of 'x's in each column.
5. Re-arrange the matrix so that the columns with the largest number of 'x's are on the left of the matrix.

Once the matrix has been re-arranged into as lower triangular form as possible the matrix can be analysed in order to determine the best control structure to apply to the process.

The advantage of having the matrix completely lower triangular is that such a matrix does not present a multivariable problem. The mix of controllers in the structure for such a process would consist of single-variable and feedforward controllers.
This becomes apparent when analysing an interaction matrix. To determine the control structure, the matrix

is split anywhere along the diagonal into four quadrants. This can only be done if the upper right hand quadrant caused by the split contains no plant interactions (indicated by 'x's). A lower triangular matrix can be continually divided in this manner since all upper right hand quadrants will be empty. The resulting sub-matrices required in the control structure are all either single-variable or feedforward. This is only true for lower triangular matrices.

The control structures that make a particular plant matrix type as diagonal as possible are fully explained in Appendix H. The structures required for the rolling mill plant are determined by analysing all the B.I.M.s which make up a rolling cycle. The start of the cycle is when there is no metal in the production line at all.

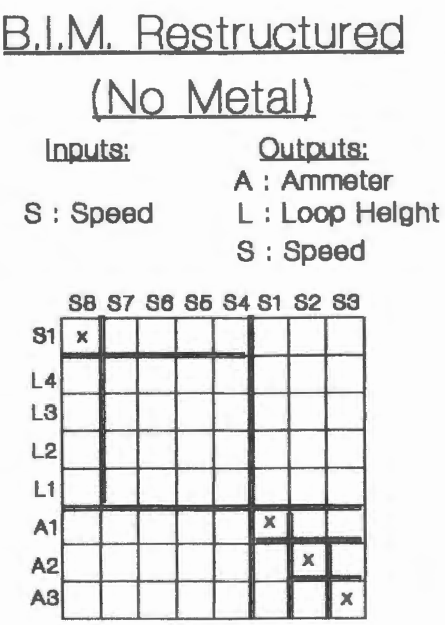


Figure 4.18: The restructured B.I.M. when there is no metal in the drives.

The B.I.M., shown in figure 4.2, is already lower triangular but the inputs and outputs have been re-ordered in order to maintain the arrangement required by the next eight restructured matrices. The re-arranged matrix is illustrated in figure 4.18.

Analysing this matrix the control structure required is:

4 single-variable controllers.

The next stage in the rolling cycle, when the metal enters the first stand, shown in figure 4.3, is similarly re-ordered and analysed. The required control structure is:

4 single-variable controllers.

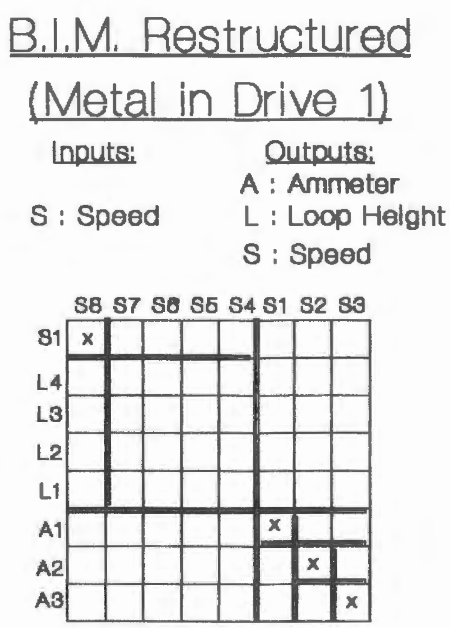


Figure 4.19: The restructured B.I.M. when metal is in Drive 1.

This is shown by figure 4.19. The control structure has not changed from the first stage in the cycle even though the steel is in the first stand.

Each stage in the process has been similarly analysed and the simplest control strategy determined.

__Metal in drives 1-2:__ (Figure 4.4) The matrix is restructured in figure 4.20.

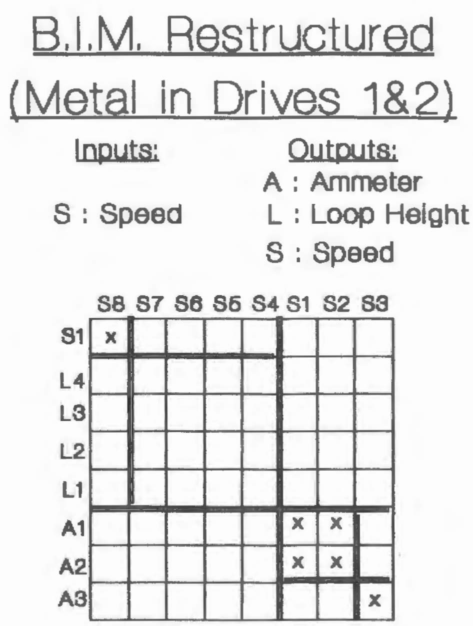


__**Figure 4.20:**__ The restructured B.I.M. when metal is in Drives 1&2.

Control Structure:

2 single-variable controllers

1 multivariable controller (2x2)

Metal in drives 1-3: (Figure 4.5) The matrix is restructured in figure 4.21.

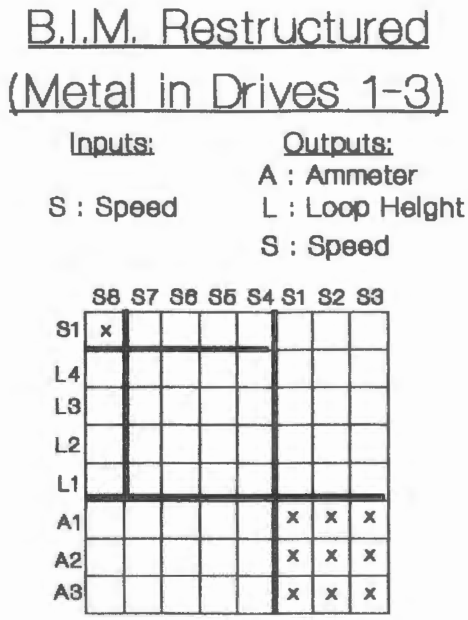


Figure 4.21: The restructured B.I.M. when metal is in Drives 1-3.

Control Structure:

1 single-variable controller

1 multivariable controller (3x3)

__Metal in drives 1-4:__ (Figure 4.6) The matrix is restructured in figure 4.22.

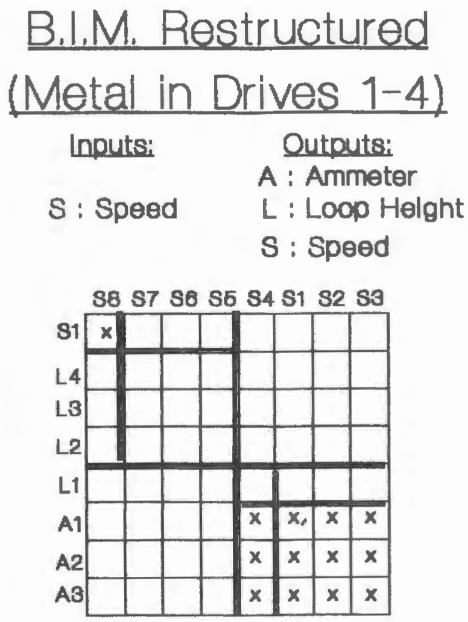


__Figure 4.22:__ The restructured B.I.M. when metal is in Drives 1-4.

Control Structure:

1 single-variable controller

1 multivariable controller (3x3)

1 feedforward controller (3x1)

All feedforward controllers are classified according to the overall matrix size even if the feedforward matrix is not completely full.

Metal in drives 1-5: (Figure 4.7) The matrix is restructured in figure 4.23

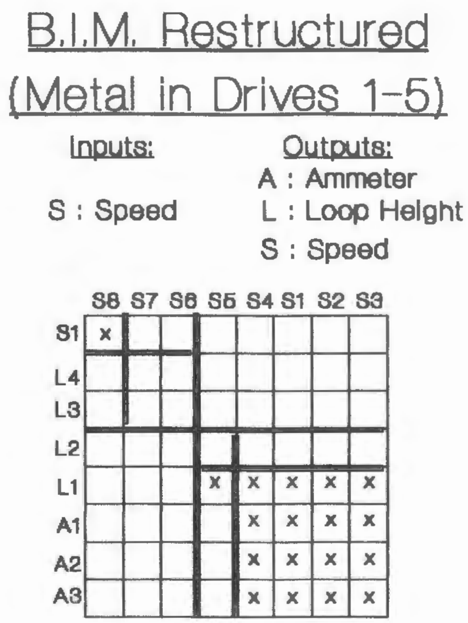


Figure 4.23: The restructured B.I.M. when metal is in Drives 1-5.

Control Structure:

1 single-variable controller

1 multivariable controller (4x4)

1 feedforward controller (4x1)

Metal in drives 1-6: (Figure 4.8) The matrix is restructured in figure 4.24.

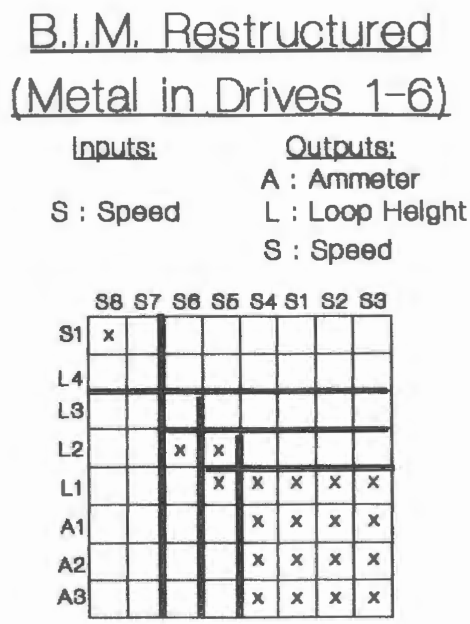


Figure 4.24: The restructured B.I.M. when metal is in Drives 1-6.

Control Structure:

2 single-variable controllers

1 multivariable controller (4x4)

2 feedforward controllers (5x1);(4x1)

Metal in drives 1-7: (Figure 4.9) The matrix is restructured in figure 4.25.

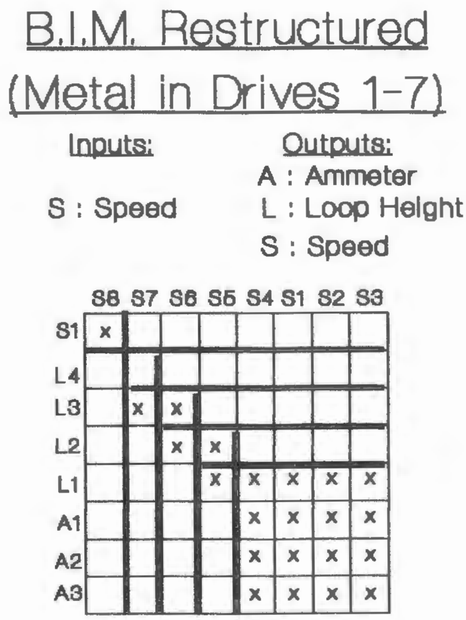


Figure 4.25: The restructured B.I.M. when metal is in Drives 1-7.

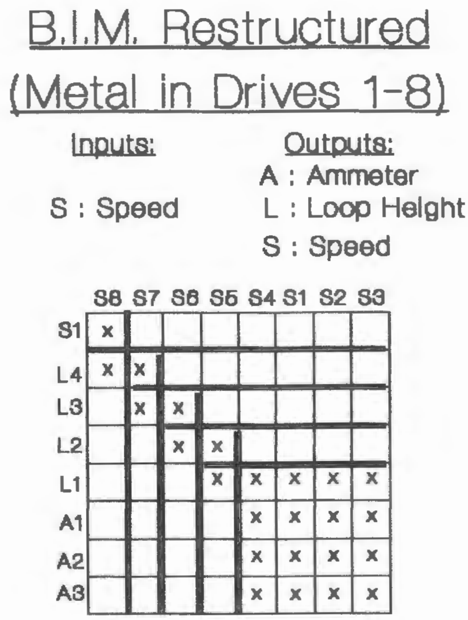
Control Structure:

3 single-variable controllers

1 multivariable controller (4x4)

3 feedforward controllers (6x1);(5x1);(4x1)

<u>Metal in drives 1-8:</u> (Figure 4.10) The matrix is restructured in figure 4.26



<u>Figure 4.26:</u> The restructured B.I.M. when metal is in Drives 1-8.

Control Structure:

4 single-variable controllers

1 multivariable controller (4x4)

4 feedforward controllers (7x1);(6x1);(5x1);(4x1)

<u>Metal in drives 2-8:</u> (Figure 4.11) The matrix is restructured in figure 4.27.

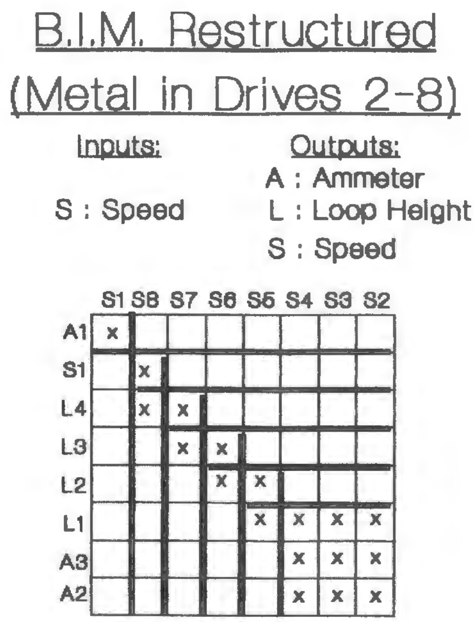


Figure 4.27: The restructured B.I.M. when metal is in Drives 2-8.

Control Structure:

5 single-variable controllers

1 multivariable controller (3x3)

4 feedforward controllers (6x1);(5x1);(4x1);(3x1)

Metal in drives 3-8: (Figure 4.12) The matrix is restructured in figure 4.28.

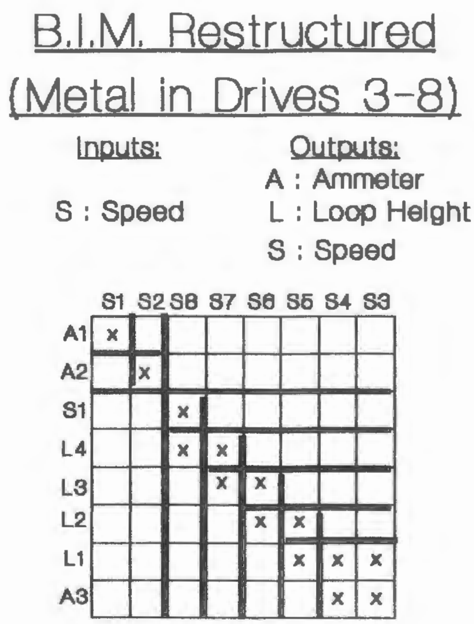


Figure 4.28: The restructured B.I.M. when metal is in Drives 3-8.

Control Structure:

6 single-variable controllers

1 multivariable controller (2x2)

4 feedforward controllers (5x1);(4x1);(3x1);(2x1)

__Metal in drives 4-8:__ (Figure 4.13) The matrix is restructured in figure 4.29.

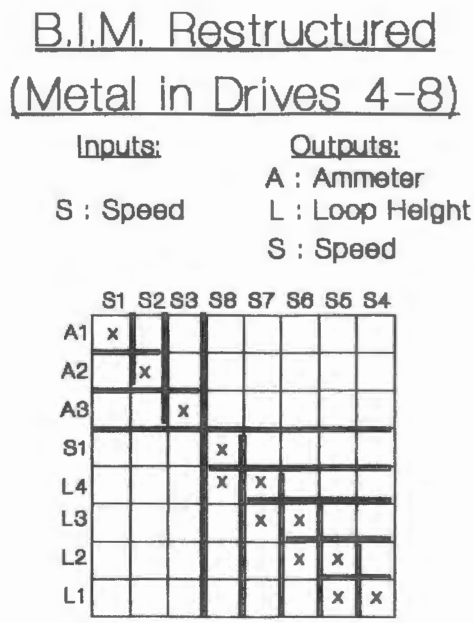


__Figure 4.29:__ The restructured B.I.M. when metal is in Drives 4-8.

Control Structure:

8 single-variable controllers

4 feedforward controllers (4x1);(3x1);(2x1);(1x1)

<u>Metal in drives 5-8:</u> (Figure 4.14) The matrix is restructured in figure 4.30.

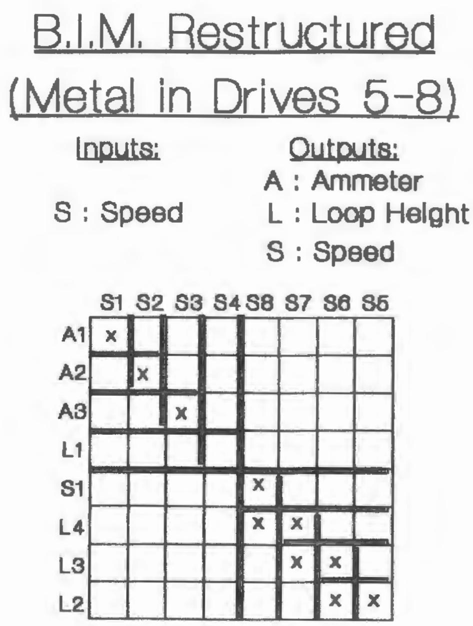


Figure 4.30: The restructured B.I.M. when metal is in Drives 5-8.

Control Structure:

7 single-variable controllers

3 feedforward controllers (3x1);(2x1);(1x1)

Metal in drives 6-8: (Figure 4.15) The matrix is restructured in figure 4.31.

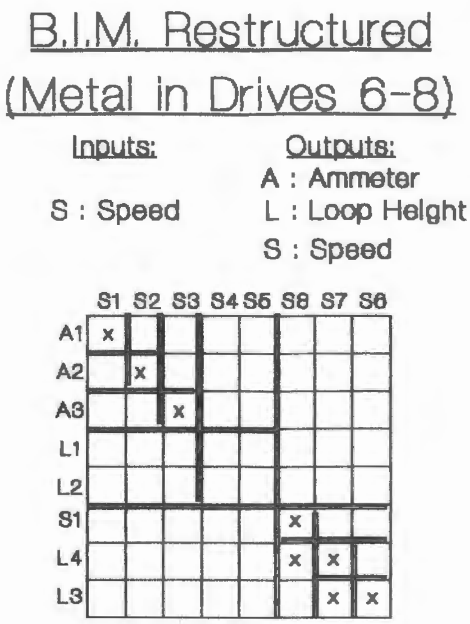


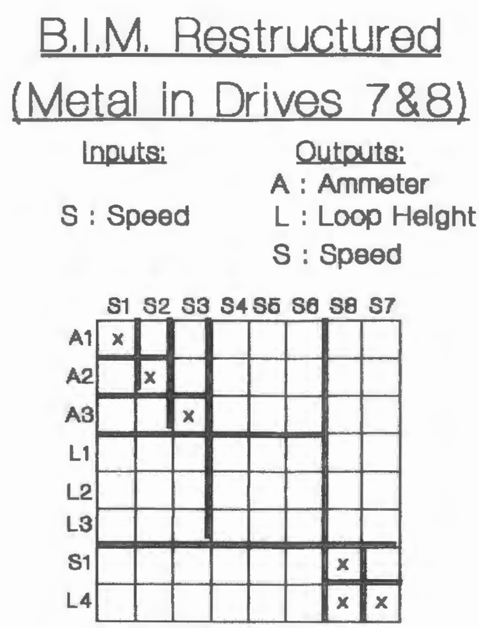
Figure 4.31: The restructured B.I.M. when metal is in Drives 6-8.

Control Structure:

6 single-variable controllers

2 feedforward controllers (2x1);(1x1)

<u>Metal in drives 7&8:</u> (Figure 4.16) The matrix is restructured in figure 4.32.



<u>Figure 4.32:</u> The restructured B.I.M. when metal is in Drives 7&8.

Control Structure:

5 single variable controllers

1 feedforward controller (1x1)

<u>Metal in drive 8:</u> (Figure 4.17) The matrix is restructured in figure 4.33.

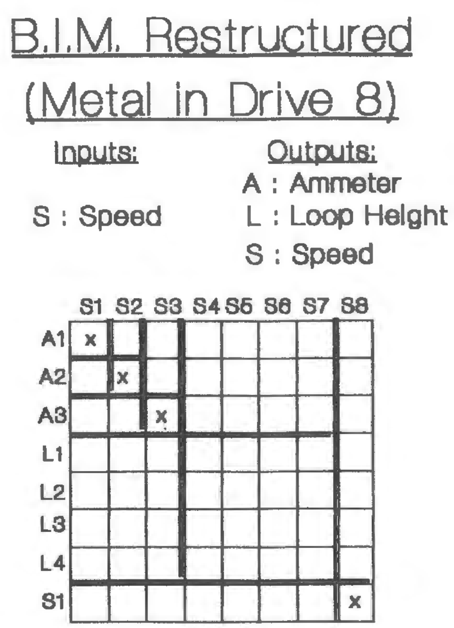


Figure 4.33: The restructured B.I.M. when metal is in Drive 8.

Control Structure:

4 single-variable controllers.

In this thesis the control structures are designed using the Inverse Nyquist Array design method where possible. Since in some situations during the cycle the loop heights are non-existent, the on-diagonals of the matrix which reflect these loop heights are zero. In these situations the inverse of the matrix is unobtainable because the determinant is zero. In such cases the Direct Nyquist Array design method is used. The design and implementation of these control structures is covered in Chapter 6.

The results of implementing all the control structures and switching between them at the correct times during the rolling cycle are also discussed in Chapter 6. These results are compared to those obtained when the control system switches in eight single-variable P.I. controllers.

However, before any of the controllers can be designed, all the transfer functions between the inputs and outputs must be derived. The derivation of transfer functions is covered in the next chapter.

References.

1.Gear, A.B.J.: *The Design of Decentralized Controllers for Large Scale Systems*, MSc Thesis, U.C.T., 1988.

CHAPTER 5: THE HILLE MILL TRANSFER FUNCTIONS

In order to design the control system for this production line using classical control theory, it is necessary to derive the transfer function matrix $\underline{G}(s)$. This matrix contains the transfer equations between the inputs (marked 'u' in figure 5.1) and outputs of the plant.

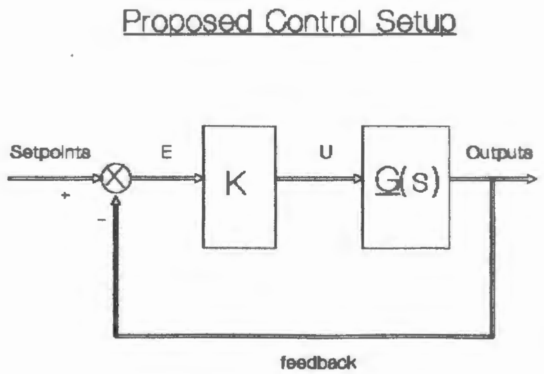


Figure 5.1: Block diagram of closed loop control.

The elements of this matrix change in the manner described in chapter 4. Some of the transfer functions describing these elements change during the rolling cycle. The reasons for this are discussed in section 5.2 and Appendix K.

The transfer function matrix for the situation where there is metal in all the stands is calculated in Appendix J and the results are analysed and explained in section 5.2.1. The matrices for all the other situations discussed in detail in the previous chapter, are derived in Appendix K. How the transfer functions change during the rolling cycle is summarised in section 5.2.2.

Actual data from the Hille Mill is analysed in the next section. This is ramp test data from the third stand in the production line (Stand #7 in figure 1.1). How this data is measured without disabling the present controller and how it is used to derive transfer functions is also explained.

The problems with the actual plant data are discussed and the reasons that transfer function equations from this data are not used to design the simulator in Chapter 3 are explained.

The simulator is then used in section 5.2 to obtain step test data. All the inputs are stepped, one at a time, and those outputs affected are logged. These output responses to the step inputs are analysed and all the transfer functions derived.

The transfer functions derived from the mill simulator package are then compared to those derived from the plant data in section 5.3.

Throughout this chapter the output and the input which is used to control it are plotted on the same graph against time. At the top of each graph these variables are stated. (i.e in figure 5.2, output current I7 vs Input speed setpoint Nr). Thus, even when the response of the first drive current, A1, is plotted for a step in the speed setpoint, S2, of the second drive, the graph illustrates the output current A1 versus the input speed setpoint S1. In this way the interaction from the plant inputs, other than the one used to control that output, is illustrated.

5.1: The Transfer Functions from Plant Data.

The plant data is taken from the third drive in line A, (figure 2.1). The results of the ramp tests conducted on this D.C. drive are presented in Appendix I. The current output of the drive is measured while the speed setpoint is ramped.

These are open loop ramp tests since the tension control part of the present controller, the Contritrol-P system, is no longer in operation at the Hille Mill. The tension controller was used to control the first three drives in the production line. Due to inefficient operation it is no longer in use and the tension is controlled by an operator. The first three drives are, therefore, no longer in closed loop and tests can be conducted.

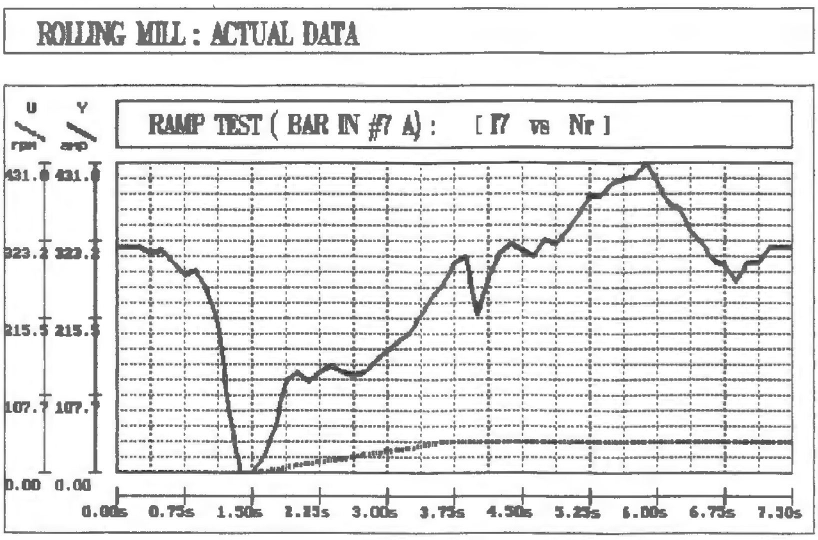


Figure 5.2: Ramp test data from drive #3.

In the first test there are two ramp changes in the speed input of the drive. This is shown in figure 5.2.

The first ramp is at approximately 1.5 seconds and the second at about 3.8 seconds. The ramp increase and decrease rate is 20 rpm/second. The original test data is given in figure I.1, Appendix I.

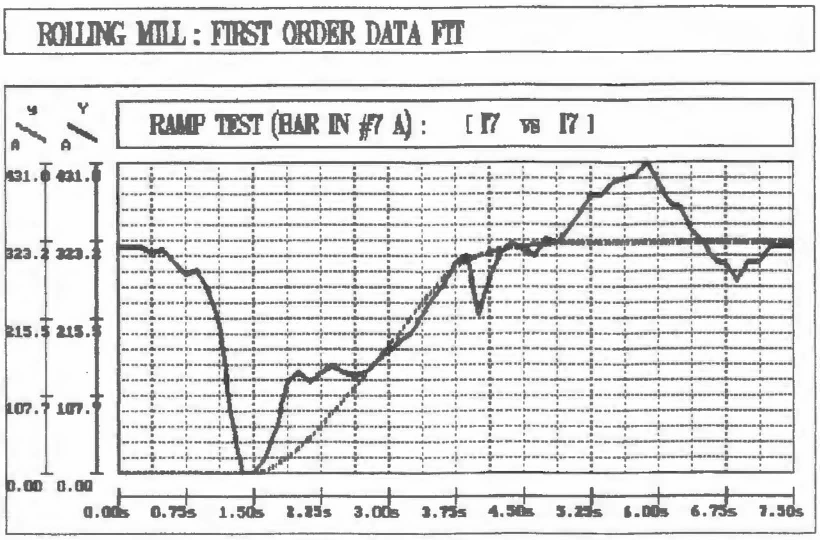


Figure 5.3: First order fit to ramp data.

Using the system identification program (Appendix L), a first order data fit is shown in figure 5.3. The first order equation used is given in equation 5.1.1, where K is the steady state gain, $1/\alpha$ is the time constant and τ is the time delay constant.

$$\frac{y}{u} = \frac{K * \alpha * e^{-s\tau}}{(s + \alpha)} \qquad \ldots(5.1.1)$$

In figure 5.3, K is 8.0, α is 2.5 and τ is 0. Although a satisfactory data fit is obtained using this equation, a pure integral response, given by equation 5.1.2 also gives a satisfactory first order fit.

$$\frac{y}{u} = \frac{K * e^{-s\tau}}{(s)} \qquad \ldots(5.1.2)$$

Using a value of one for K and zero for τ, the data fit shown in figure 5.4 is obtained.

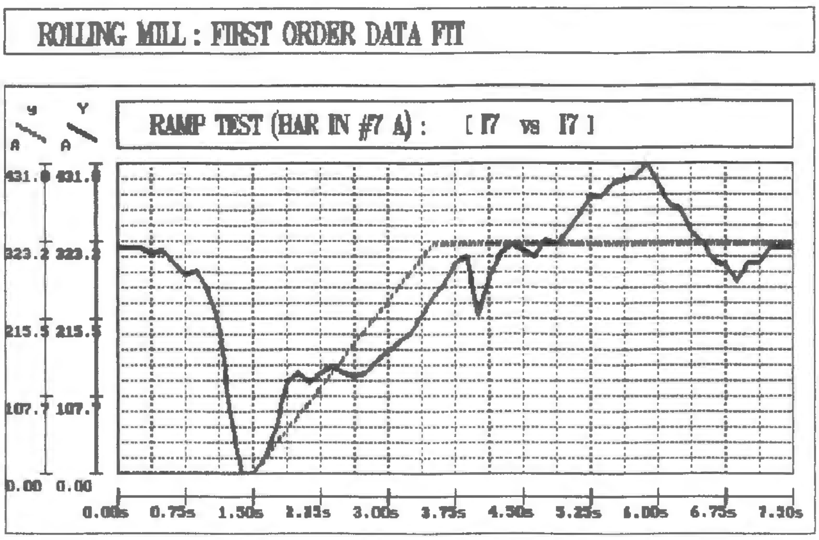


Figure 5.4: An integral first order fit to ramp data.

The second ramp test conducted on the third drive is similarly analysed using the two first order equations 5.1.1 and 5.1.2. The original test data is shown in figure I.2, Appendix I. The previous values used in the first ramp test for the equation constants are altered for this test in order to get the best line data fit.

The results of using equation 5.1.1 with K at 10, α equals to 0.1 and τ equal to 0, are shown in figure 5.5.

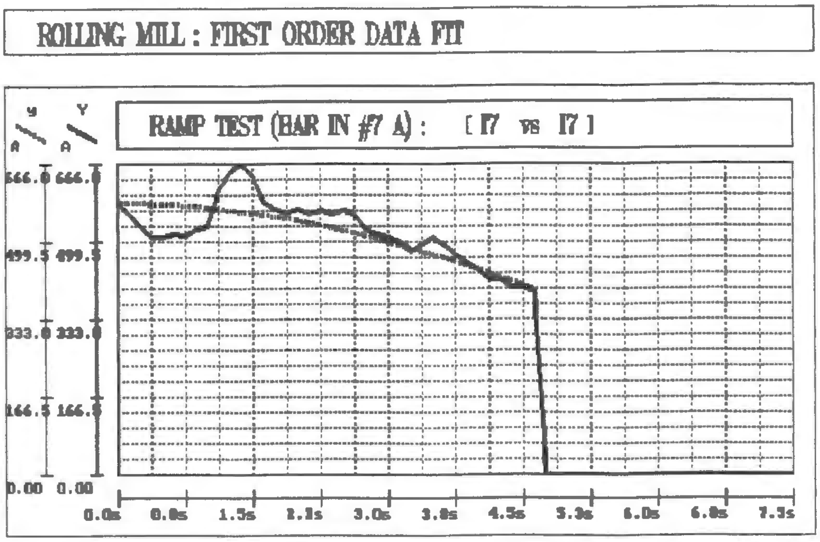


Figure 5.5: A first order fit to ramp data.

The results of fitting this data with a pure integral response, eq.5.1.2, are illustrated in figure 5.6. K is 0.15 in this application.

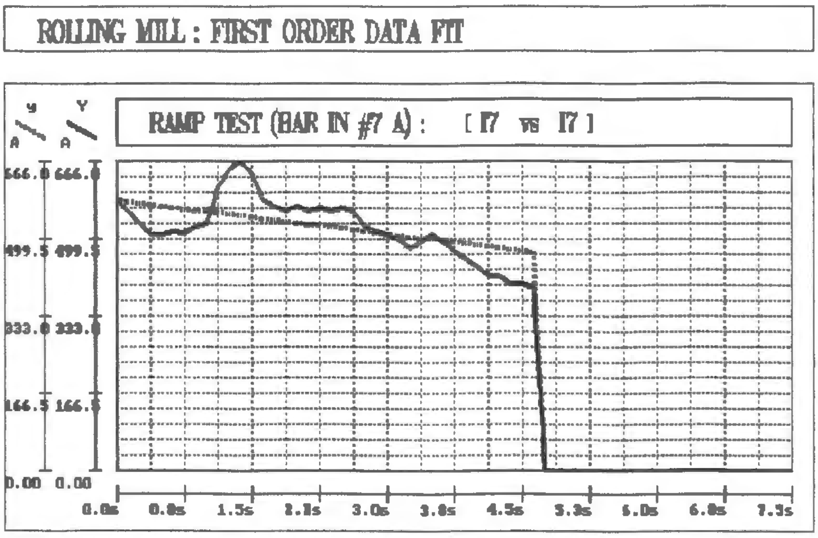


Figure 5.6: An integral first order fit to ramp data.

Using the data from the third ramp test (figure I.3, Appendix I), Similar values are obtained for the constants used in the two first order equations. The fit with K equal to 15, α equal to 2 and τ equal to zero in equation 5.1.1, is shown in figure 5.7

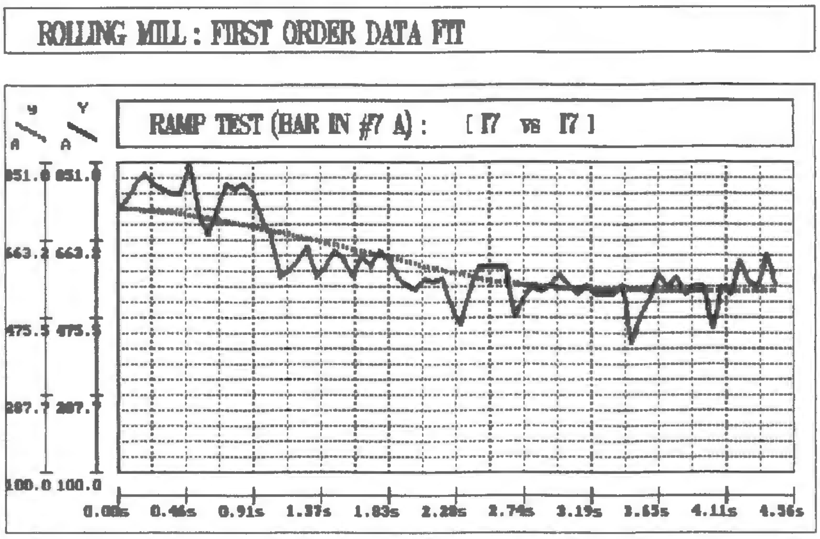


Figure 5.7: A first order fit to the third ramp test.

As in the first attempt at fitting a first order response to the plant data, there is a satisfactory fit with both equations. The results of fitting this data with an integral response are illustrated in figure 5.8. This is done using the value of 0.8 for K in equation 5.1.2.

Comparing figures 5.7 and 5.8, either first order modelling equation could be used for the transfer function between the speed setpoint and the current output of drive #3.

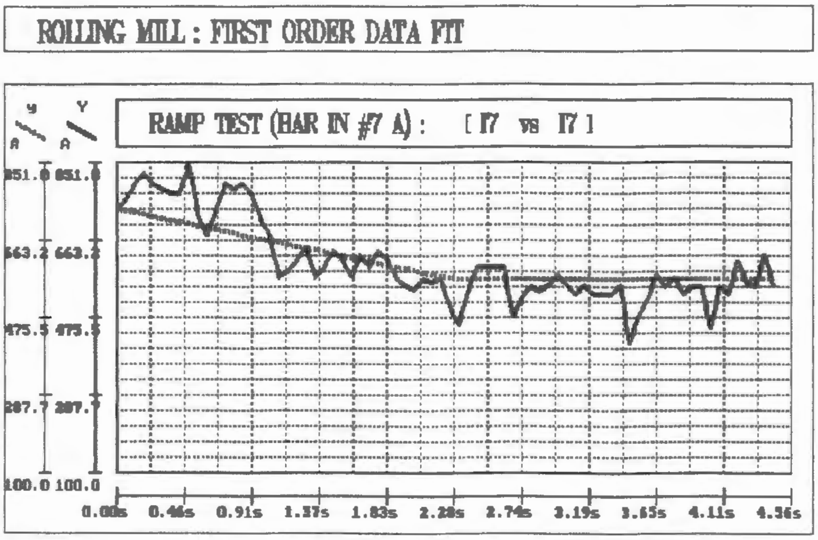


Figure 5.8: An integral fit to the third ramp test.

Table 5.1 is a summary of the constants used in both first order equations in the three ramp tests.

Table of Constants Used

TEST NO. \ OPTION	Eq. 5.1.1			Eq.5.1.2	
	K	α	τ	K	τ
TEST #1	10	0.1	0	0.15	0
TEST #2	8	2.5	0	1.0	0
TEST #3	15	2	0	0.8	0
MEAN	11	1.5	0	0.65	0
Std. Dev.	1.7	0.8	0	0.2	0

Table 5.1: The constants used in fitting the data.

Due to the quality of the plant data obtained, it is not possible to determine which of the two models to use as the transfer equation for the third drive. There is a large amount of noise on the current output signal and more exhaustive tests would have to be conducted on the plant to determine the correct model for this process.

Although the equations used to model the data in the second and third tests are similar, the constants that are used in the first set of equations are very different. This could be due to the fact that the first test is only conducted for a short time and hence it is less probable that the disturbances in the data can be filtered out by means of a best line data fit.

There are several reasons why adequate plant data cannot be obtained. Due to the economic constraints mentioned previously, exhaustive tests cannot be conducted on the Hille Mill itself. Another problem is that the interaction between the speed setpoints and the four loop heights cannot be investigated because these loops are in closed loop. They are at present controlled by the Contritrol-P control system discussed in section 2.4.

Therefore a complete matrix of accurate transfer functions cannot be derived and so the simulator written in Chapter 3 is based not on actual plant data but on mathematical models of the components of the rolling process. Plant constants (Appendix D) are used in the simulator wherever possible.

The transfer functions studied in this thesis are derived using the simulator. The procedure and final transfer matrix is detailed in the next section.

5.2: The Transfer Functions of the Simulator.

In order to derive the transfer functions from the simulator each input is stepped and the response of the outputs is recorded. The response of an output is then analysed to determine the transfer function equation between that output and the stepped input.

During the complete rolling mill cycle, shown in figure 5.9, the plant interaction changes as explained in Chapter 4.

Figure 5.9 illustrates the rolling process cycle graphically. This series of graphs all show the same time period. The eight output responses are plotted along with the inputs used to control them. The time when the bar enters the first stand is the start of the cycle. This is shown in the first graph of the series. Each successive graph shows the output of the next stand in the line. The entrance of the bar into the first three stands occurs at the point where the current is seen to suddenly increase. This simulates the real current action in the plant (see Appendix I).

The time at which the bar leaves the stands is marked by the sudden reduction in either the current in the stand (the first three drives) or the length of the loop after the stand (graphs 4 to 7).

The step changes in the speed setpoints of the first seven drives are in order to form the four metal loops. This is fully explained later in this section.

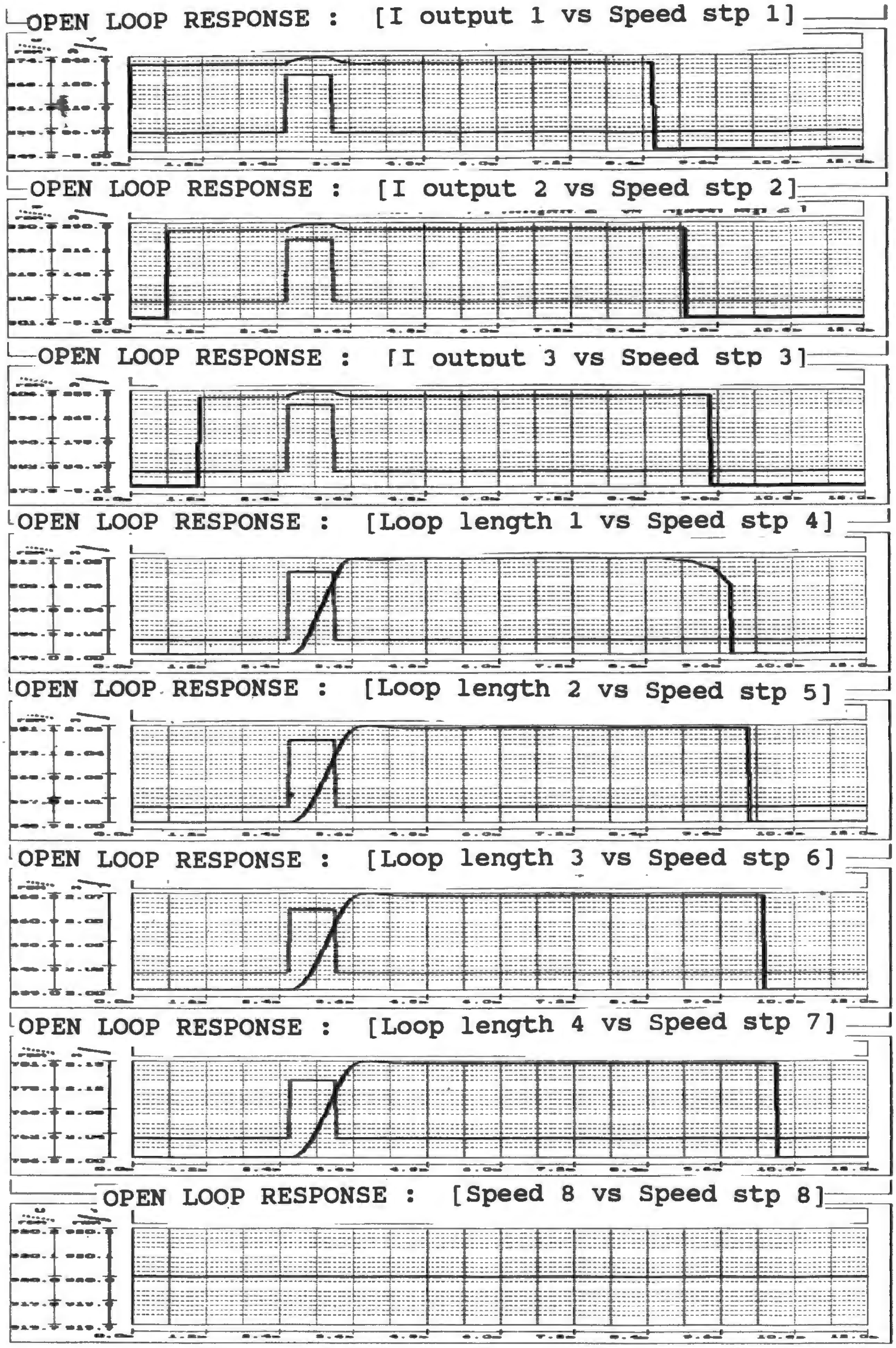


Figure 5.9: The output responses during the rolling of a steel billet.

As stated previously, the plant interaction changes during the rolling process cycle. The transfer function equations describing some of these interactions also change. Due to the different number of stands interacting in the cascading tension manner described previously, the transfer functions describing the outputs that are affected (A1, A2, A3 and L1) vary.

Although the interaction matrix does change with respect to the loop heights, L2, L3 and L4, the equations describing these output variables do not change. The transfer function describing the speed response of the eighth drive, S1, does not change either due to it being buffered by the fourth loop.

The output responses to step changes in the input variables when there is metal in all eight stands is described in the next section. The features of the responses are explained and the transfer functions presented. These functions are derived in Appendix J.

How the transfer functions change in the rolling process cycle is summarised in section 5.2.2. The transfer functions for all the other situations in the cycle are calculated and discussed in Appendix K.

5.2.1: Metal in all Eight Drives.

As is shown in the binary interaction matrix for the situation where there is metal in all eight drives (figure 4.10), a step change in the speed input of the first drive affects the current outputs of the first three drives, A1 - A3, and the height of the first loop, L1.

As is observable from the graphs in this section, the first seven drive inputs are stepped up after about 2.4 seconds and then stepped down 0.7 seconds later. The reason for this is to form loops between the last five drives.

If this is not done the responses of the loop heights are not observable. Therefore, the first four drives are stepped in the same ratio so that no loops or tensions build up between them. The speed setpoint of the fifth drive is stepped slightly less relative to the speed of Drive #4 and so on, thereby causing the extra metal between the stands to form a loop.

These loops are stabilised by stepping the seven speed setpoints down to their previous level which maintains the correct relative drive speeds and achieves steady state in the plant outputs. The step tests are then conducted on the plant. In this manner the open loop responses of the loops are observed.

As is illustrated by figure 5.9, the sudden changes in the current outputs of the drives occur when the bar enters or leaves the stand. This is also illustrated by the graphs showing the output responses of A1, A2 and A3.

Each speed setpoint is now stepped and the responses of all the affected plant outputs, as determined in Chapter 4, are logged. As explained, each graph still shows a plant output and the plant input that is used to control it, not the plant input that is stepped.

Output Responses to a Step in Speed setpoint S1.

Transfer Function G_{11}.

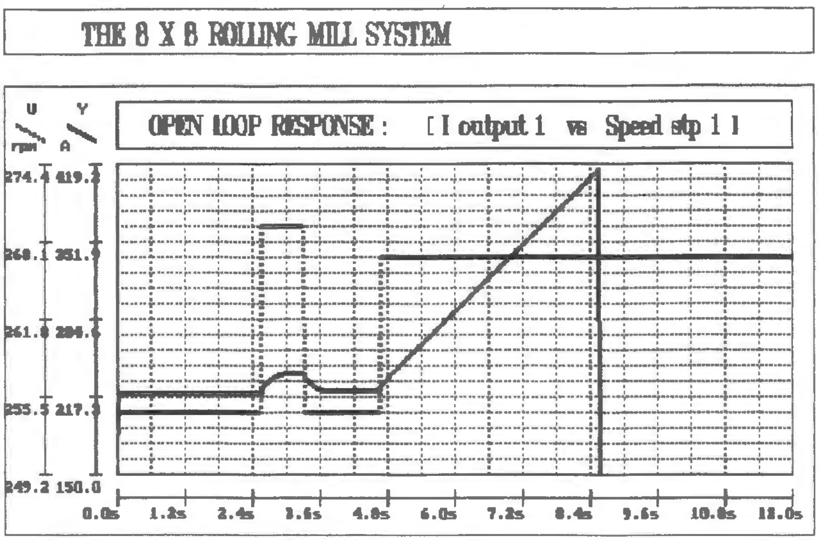


Figure 5.10: The current response of the first drive to a step in the input S1.

The response of the output current A1 to a step in the speed setpoint, S1, is shown in figure 5.10. Using this graph the transfer function G_{11} is derived. The full derivation of this and the other transfer function equations is detailed in Appendix J.

The initial step changes in the speed setpoint and the first order response of A1 observed are in order to form the loops, as previously explained. The reason that the response is first order is that the first four drive speeds are kept at the same relative speeds. No tension interaction results and the only response observed is the first order response of the drive's current to a step change in the speed setpoint. This first order response is calculated and explained in Appendix K for the situation where there is no metal in any of the stands and therefore no interaction. These initial steps are to be observed on all the graphs. The step in a input occurs once the loop heights are at steady state (after about 4.8 seconds).

The transfer function is a integral equation with a gain of 3.88.

$$G_{11}(s) = \frac{3.88}{s}$$

Transfer Function G_{21}.

The response of current A2 to the step in the input of the first drive is shown in figure 5.11. As is shown in this graph the current A2 decreases due to the decrease in the load torque on the second drive, as is previously explained in Chapter 4.

As the process cycle diagram (figure 5.9) and figure 5.10 show, the metal leaves the first stand after about 8.4 seconds. Since the first stand is the one to have been stepped in this case, the speed mismatch is due to the speed of this drive. Hence, when the bar leaves the first stand the tension caused by this speed mismatch will no longer affect the load torque of the second or the third drive. This is shown in figure 5.11 by the sudden recovery of the current in the second stand after 8.4 seconds. This effect is also observable in the other graphs.

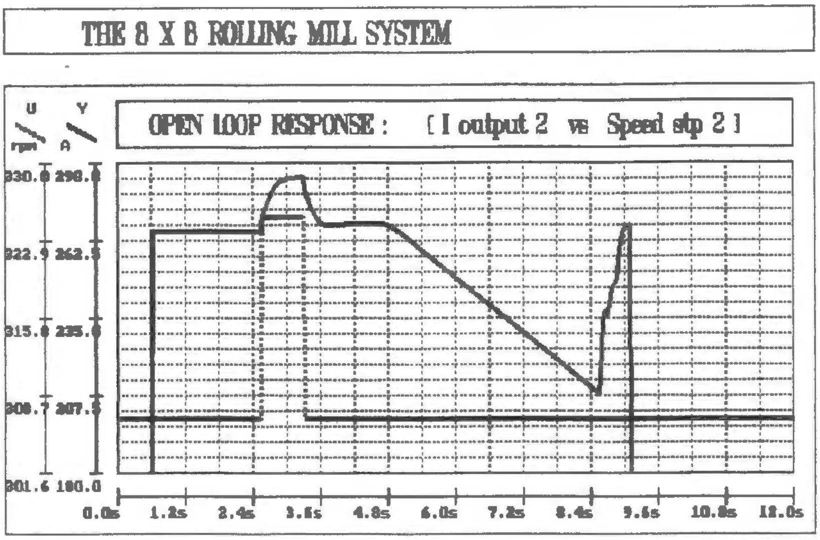


Figure 5.11: The current response of the second drive to a step in the input S1.

The first order responses in the current A2 shown in figure 5.11 are due to the step changes in the speed setpoint of the second drive required to form the loops. The reason that these responses are first order is explained at the beginning of this section. The integral response is due to the step change in the speed setpoint, S1. From this G_{21} is calculated and is an integral transfer function with a gain of -1.22.

$$G_{21}(s) = \frac{-1.22}{s}$$

Transfer Function G_{31}.

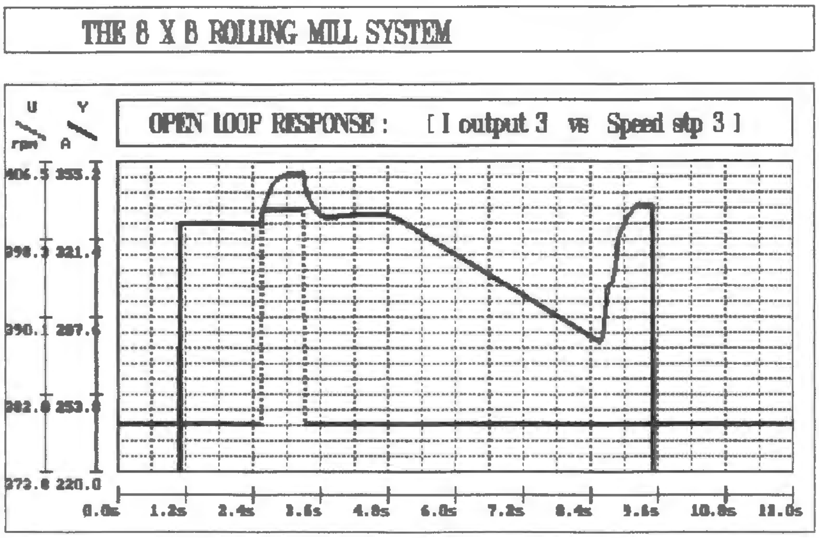


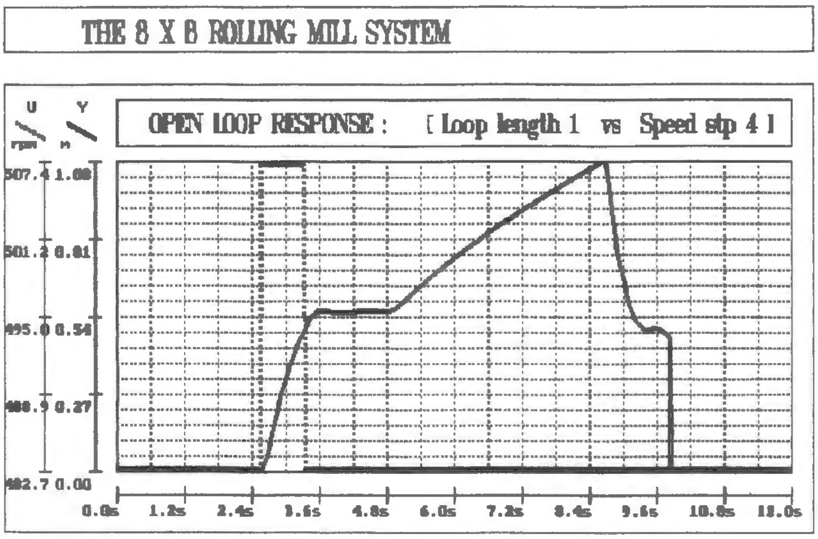
Figure 5.12: The current response of the third drive to a step in the input S1.

Figure 5.12 illustrates the response of the current output A3 to a step in the speed input S1. Using the above figure the transfer function G_{31} is calculated as:

$$G_{31}(s) = \frac{-1.10}{s}$$

Transfer Function G_{41}.

The response of the height of the first loop, formed between stands four and five, to a step in the speed input of any of the first four drives is non-linear. This is illustrated by figure 5.13, which shows the loop height response to a step in the first drive's input. Linear control theory cannot be applied to a non-linear function and it is therefore necessary to linearise the loop outputs, L1 - L4.



<u>Figure 5.13:</u> Loop height response to a step in input S1.

To linearise the outputs of the loop heights, they are converted back to an equivalent length. The height setpoint to the closed loop is then also converted to an equivalent length before it is compared to the output and the error fed into the controller. This is illustrated in figure 5.14, the block diagram of the closed loop setup that is used to linearise the control of the height outputs.

As is shown by this figure the output that is now controlled is the length of the loop and not the height. Therefore the controller K is designed to control the "new plant".

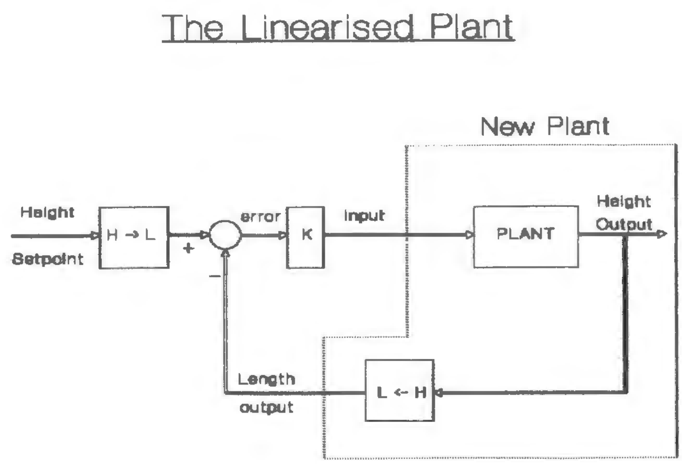


Figure 5.14: The linearisation of the loop height.

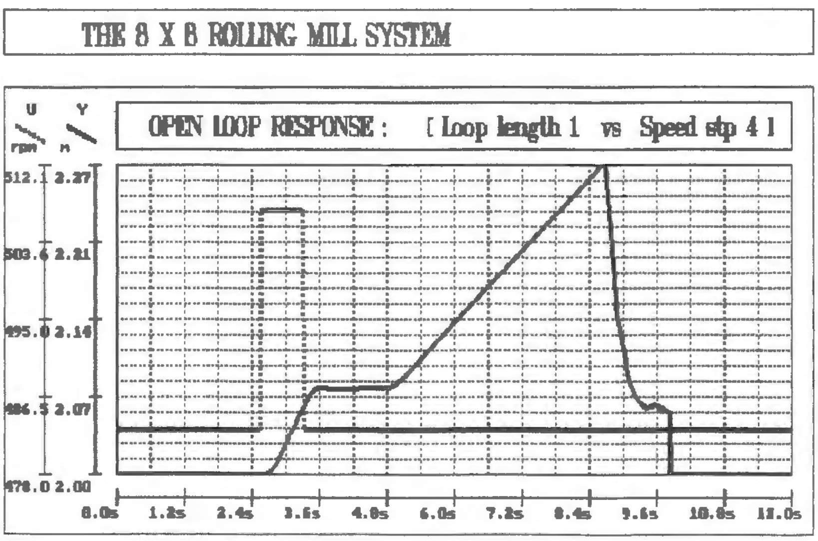


Figure 5.15: The length response of the first loop to a step in the input S1.

This is implemented in the simulator and the response of the loop length is analysed in order to determine the transfer function equations for the outputs L1 - L4. The response of L1 to a step in the speed setpoint, S1 is that observed in figure 5.15.

The transfer function G_{41} is calculated as:

$$G_{41}(s) = \frac{0.004}{s}$$

Output Responses to a Step in the Speed Setpoint S2.

Transfer Function G_{12}.

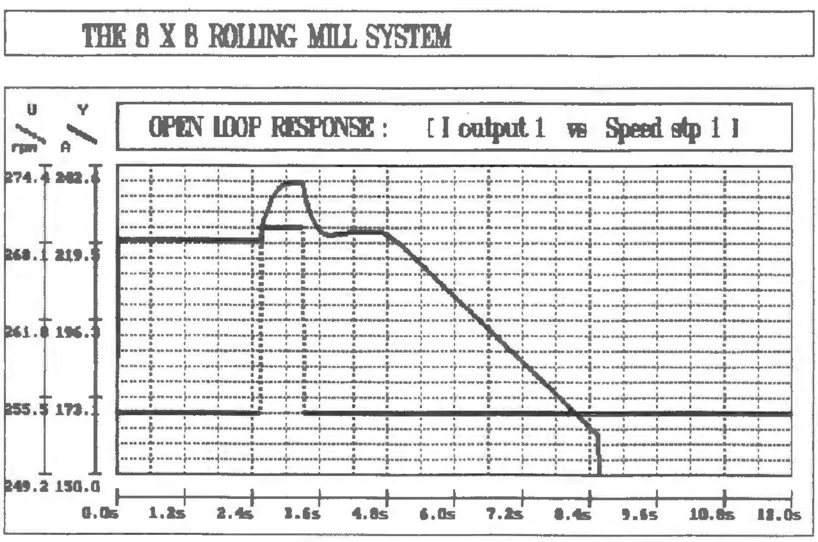


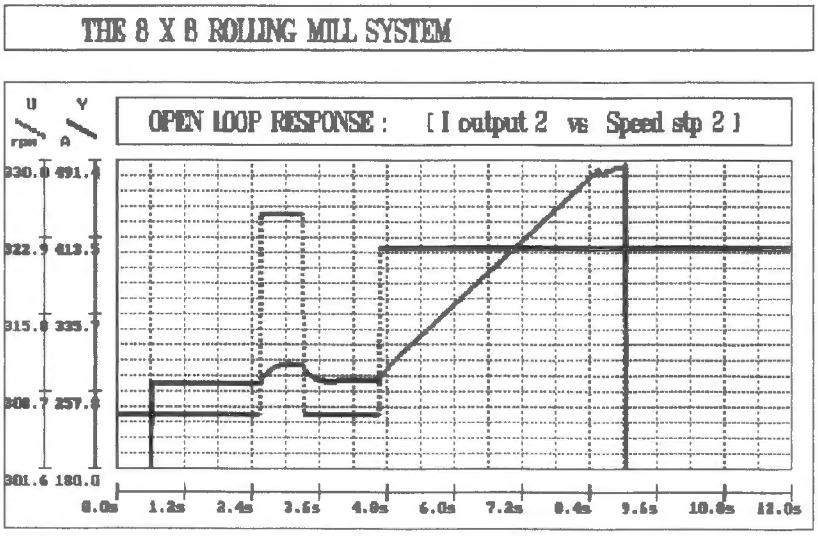
Figure 5.16: The current response of the first drive to a step in the input S2.

Figure 5.16 illustrates the response of the current A1 to a step in speed setpoint S2. G_{12} is calculated from this graph and is a integral equation with a gain constant of -1.04.

$$G_{12}(s) = \frac{-1.04}{s}$$

<u>Transfer Function G_{22}.</u>

G_{22} is calculated using the graph in figure 5.17, which illustrates the transfer function between input S2 and output A2.



<u>Figure 5.17:</u>The current response of the second drive to a step in the input S2.

The current increases when the speed setpoint is stepped up for two reasons. Firstly, it increases in order to provide the extra torque required to increase the drive's speed. It also increases, in an integral manner because the load torque on the motor increases due to the tension of the metal, as explained previously. The combined response is best approximated using a purely integral transfer function equation since the first response dies out within 0.6 seconds. This is illustrated by the step response of drive two in figure K.2, Appendix K. This figure shows the results of stepping the second drive's input when there is no metal in the stand. This allows the initial current response due to the speed setpoint increase to be observed. The on-diagonal elements G_{11}, G_{22} and G_{33} are all approximated in this manner at this stage of the cycle.

G_{22} is thus approximated by a integral equation with gain constant of 3.58.

$$G_{22}(s) = \frac{3.58}{s}$$

<u>Transfer Function G_{32}.</u>

The response of the current in the third drive, A3, to a step change in the speed setpoint of the second drive, S2, is illustrated in figure 5.18.

This graph is analysed and the equation describing G_{32} is also an integral response. The gain constant in this transfer function is -1.09.

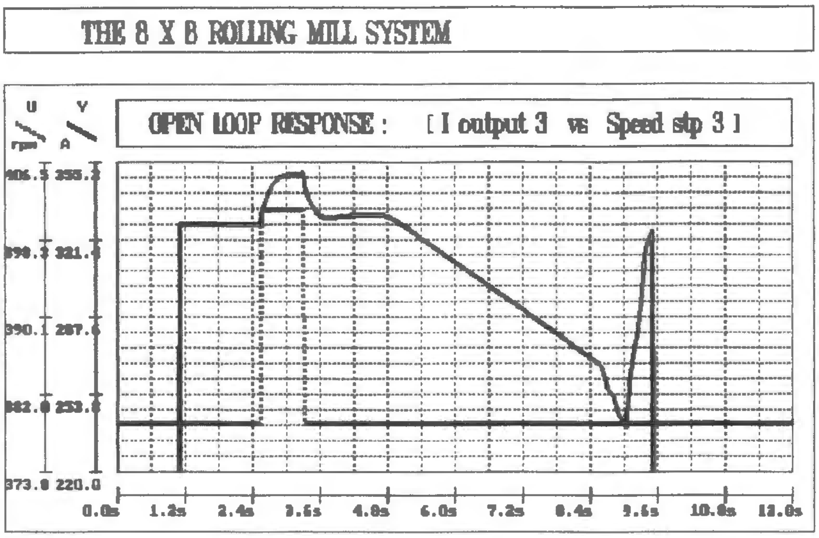


Figure 5.18: The current response of the third drive to a step in the input S2.

$$G_{32}(s) = \frac{-1.09}{s}$$

Transfer Function G_{42}.

Figure 5.19 shows the response of the length of the loop, L1, to a step increase in the speed setpoint of the second drive. The sudden change in the response after 8.7 seconds is caused by the change in the interaction when the steel bar leaves the first stand. After 9.0 seconds the bar leaves the second stand and the interaction decreases as expected. The bar then leaves the fourth stand after 9.9 seconds, causing the loop output to return to it's zero level.

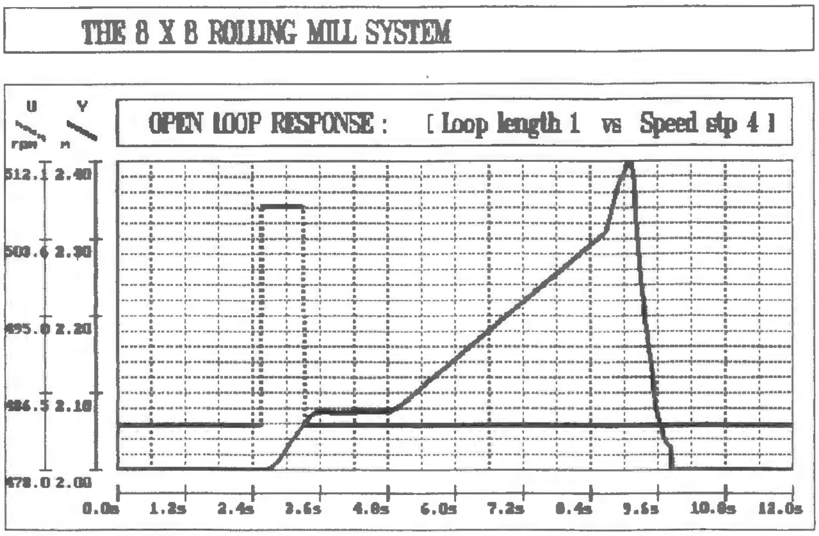


Figure 5.19: The length response of the first loop to a step in the input S2.

$$G_{42}(s) = \frac{0.004}{s}$$

Transfer Function G_{13}.

Due to the decrease in load torque on drive #1 caused by the metal being pulled through the third stand at a higher speed, the current in the first drive decreases integrally.

This is illustrated by the graph of figure 5.20. This graph is used to obtain the transfer equation G_{13} between A1 and S3.

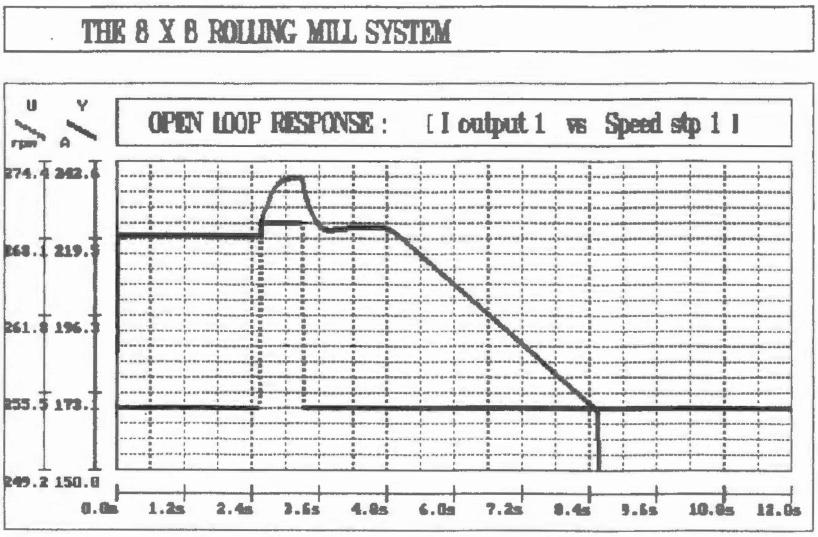


Figure 5.20: The current response of the first drive to a step in the input S3.

$$G_{13}(s) = \frac{-0.74}{s}$$

Transfer Function G_{23}.

The response of the current in the second drive to a step in the speed S3 when there is metal in all the drives is shown in figure 5.21.

This graph is analysed over the region 4.8 - ~8.4 seconds. After 8.4 seconds the metal leaves the first stand and the transfer function changes. All other forms of this transfer function are calculated and explained in Appendix K.

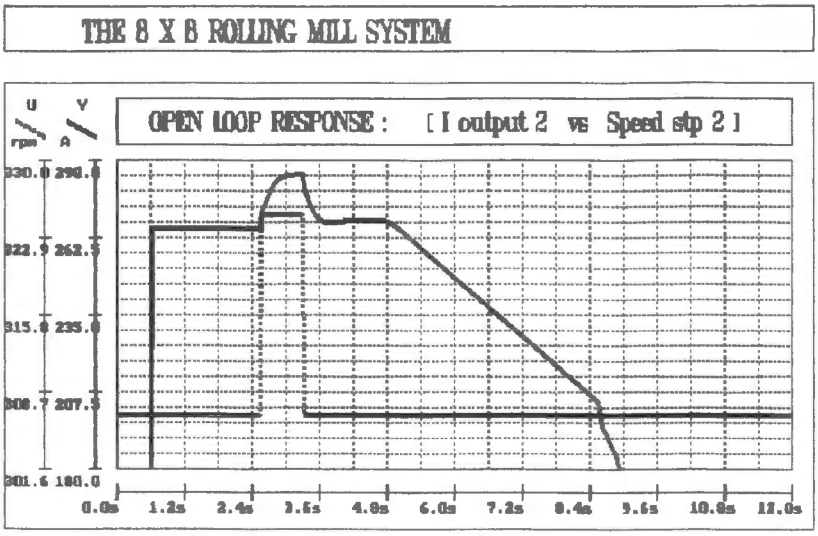


Figure 5.21: The current response of the second drive to a step in the input S3.

The resulting transfer function is given by the following equation:

$$G_{23}(s) = \frac{-0.89}{s}$$

Transfer Function G_{33}.

Analysing the response graph shown in figure 5.22 the transfer equation relating the current output, A3, of the third drive to the speed input, S3, of the same drive is obtained. As explained for G_{22} the current increases for the two reasons stated there. The shape of the output response in figure 5.22 illustrates that the first reaction is a first order response which dominates the response initially but then after approximately 0.6 seconds the integral response dominates.

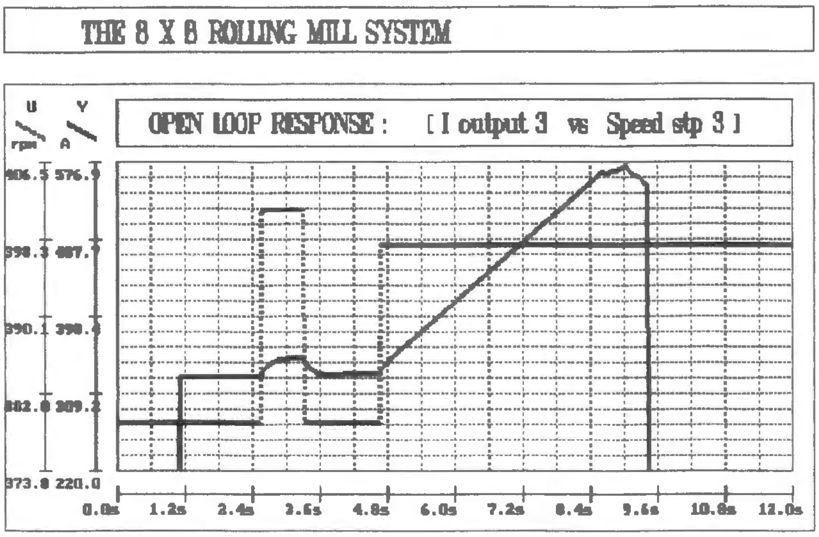


Figure 5.22: The current response of the third drive to a step in the input S3.

The transfer function is thus approximated by the integral response.

$$G_{33}(s) = \frac{3.15}{s}$$

Transfer Function G_{43}.

The length response of loop L1 to a step increase in S3 is illustrated by figure 5.23.

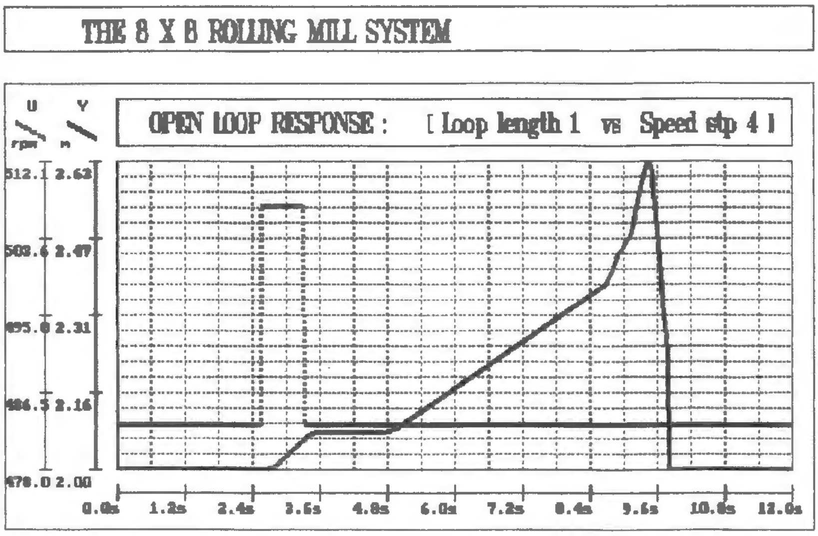


Figure 5.23: The length response of the first loop to a step in the input S3.

This response is analysed over the region before the steel leaves the first stand. This occurs after ~8.4 seconds. All other forms of G_{43} are explained in Appendix K. G_{43} for this situation is calculated in Appendix J as:

$$G_{43}(s) = \frac{0.004}{s}$$

Output Responses to a Step in Speed Setpoint S4.

Transfer Function G_{14}.

The output response of current A1 to a step in the speed setpoint of the fourth drive is shown in figure 5.24. It similarly decreases integrally due to the decrease in the load torque on the first drive.

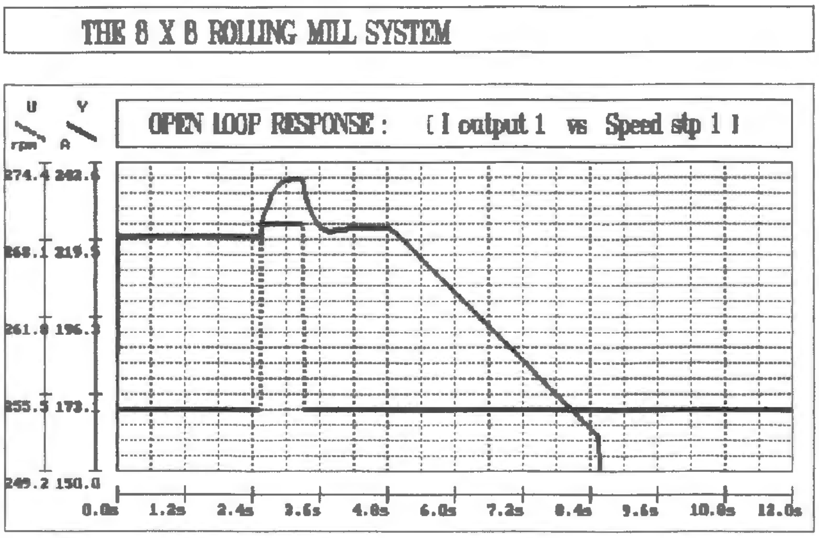


Figure 5.24: The current response of the first drive to a step in the input S4.

The transfer function is calculated as having a gain constant of -0.67.

$$G_{14}(s) = \frac{-0.67}{s}$$

Transfer Function G_{24}.

The response of the current A2 similarly decreases integrally when S4 is increased. This is shown by the graph in figure 5.25.

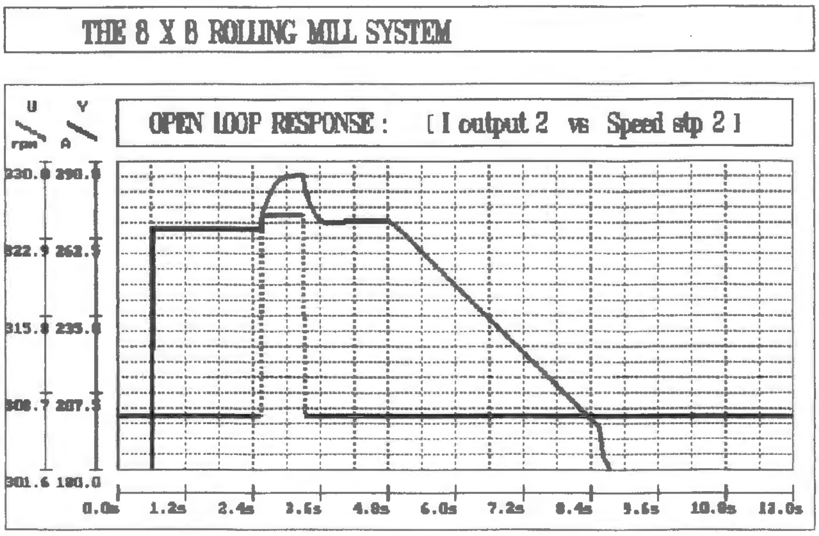


Figure 5.25: The current response of the second drive to a step in the input S4.

The transfer function has a gain constant of -0.79 and is given by the following expression

$$G_{24}(s) = \frac{-0.79}{s}$$

Transfer Function G_{34}.

Figure 5.26 shows how the current of the third drive declines when the input of the fourth drive is stepped up. The two steps in the speed setpoints of this drive that are necessary to form the four loops are also illustrated by this graph.

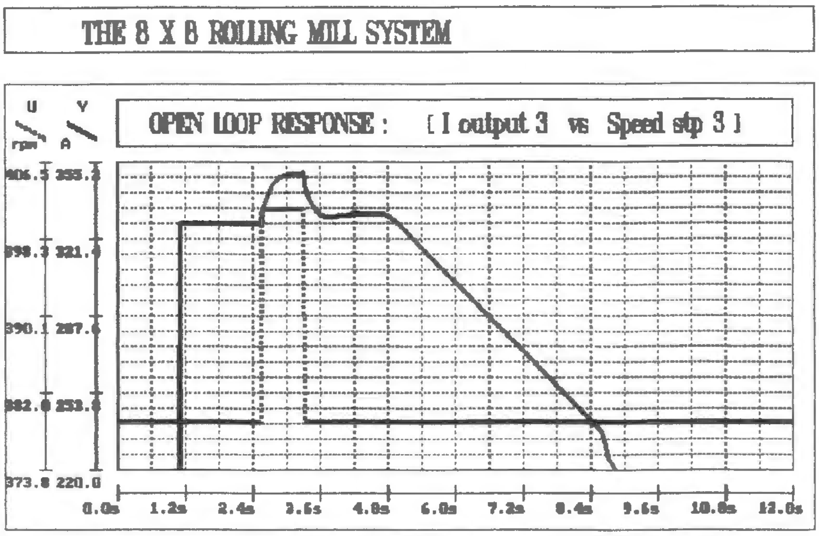


Figure 5.26: The current response of the third drive to a step in the input S4.

The integral transfer equation is calculated to have an a negative gain of 1.01.

$$G_{34}(s) = \frac{-1.01}{s}$$

Transfer Function G_{44}.

The response of the length of metal in the first loop to a step change in the speed of the fourth drive is illustrated by figure 5.27. The first loop is formed between the fourth and fifth stands in the production line (See figure 2.1). As explained in the previous chapter, increasing the speed of the drive before the loop increases the height and hence the length as well.

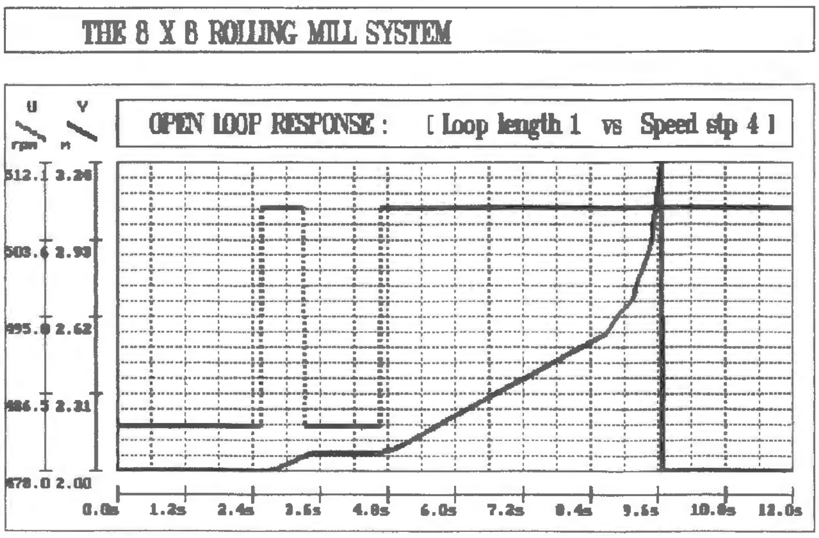


Figure 5.27: The length response of the first loop to a step in the input S4.

The linear transfer function is calculated to have a gain constant of 0.005 and is given by the following expression.

$$G_{44}(s) = \frac{0.005}{s}$$

Output Responses to a Step in Speed Setpoint S5.

Transfer function G_{45}.

The length of the first loop is affected by a step increase in the speed of the fifth drive as shown by the graph in figure 5.28.

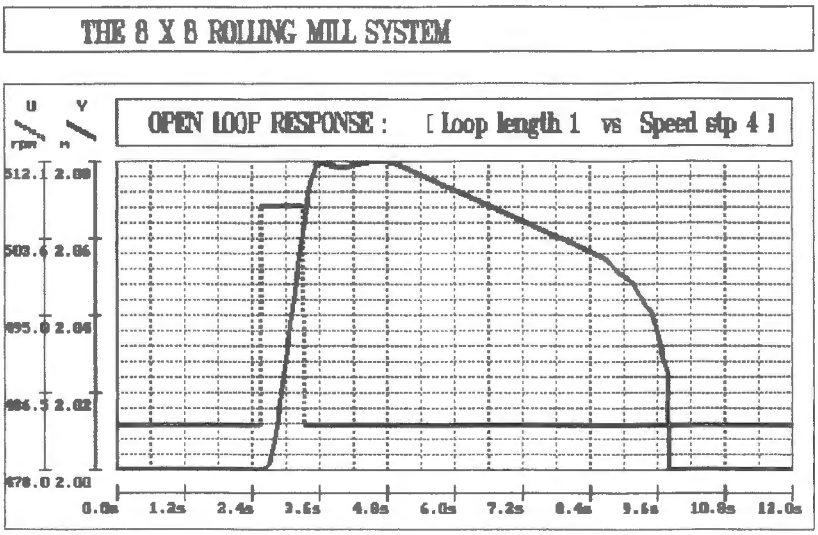


Figure 5.28: The length response of the first loop to a step in the input S5.

This loop length response is affected by the bar leaving the first stand after ~8.4 seconds. The reason for this is that when all the stands are stepped to form the loops, as previously explained, the transient responses of the first four stands are not exactly the same. This is due to the different drive characteristics (Appendix D) and to the fact that the PI controllers in the inner speed control loop (figure 2.2) are not exactly the same.

Therefore, even though their relative speeds are kept constant when these steps are performed, the slight difference in the transient responses causes these relative speeds to be slightly mismatched during the response. This causes a slight tension to build up in the metal. Hence when the steel leaves the first drive, this tension is relaxed slightly and it affects the speed of the fourth drive which in turn disturbs the loop length. This is seen in figure 5.28.

The following equation is calculated from the region of the graph during which only the step in the speed input of the fifth drive is affecting the loop height.

$$G_{45}(s) = \frac{-0.012}{s}$$

<u>Transfer Function G_{55}.</u>

The length of the second loop reacts to a step in the speed input of the first drive in the manner illustrated by the graph in figure 5.29.

As expected the loop length increases when the stand before it is increased in speed. The sudden drop in the loop height shown after ~10.2 seconds is caused by the metal leaving the fifth stand. If the end of the steel bar went into the loop before the loop height had been reduced to zero, it would whip when pulled down into the sixth stand. A major aim of the controller must be to reduce this loop height before the bar leaves the previous stand.

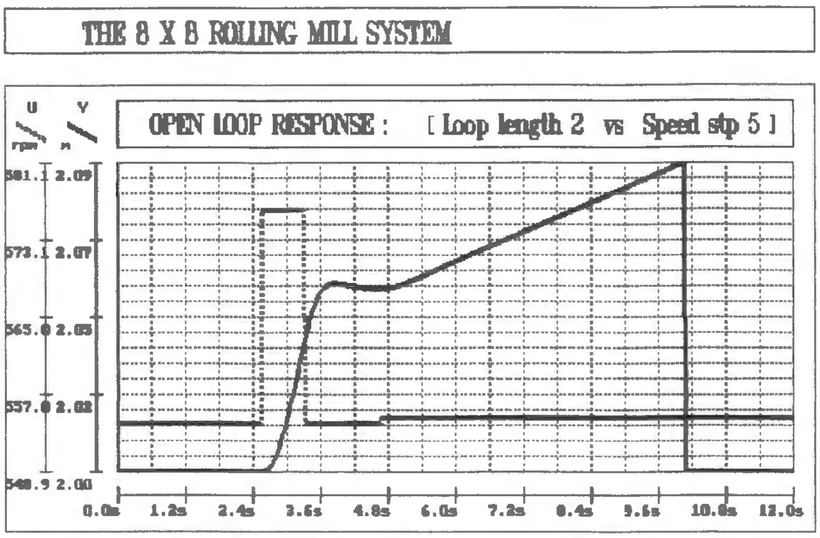


Figure 5.29: The length response of the second loop to a step in the input S5.

The response of the second loop length, L2, to this step is calculated to be:

$$G_{55}(s) = \frac{0.013}{s}$$

Output Responses to a Step in the Speed Setpoint S6.

Transfer Function G_{56}.

The step increase in the speed setpoint of the sixth drive pulls more metal through the stand causing the second loop length to decrease. This is illustrated by figure 5.30.

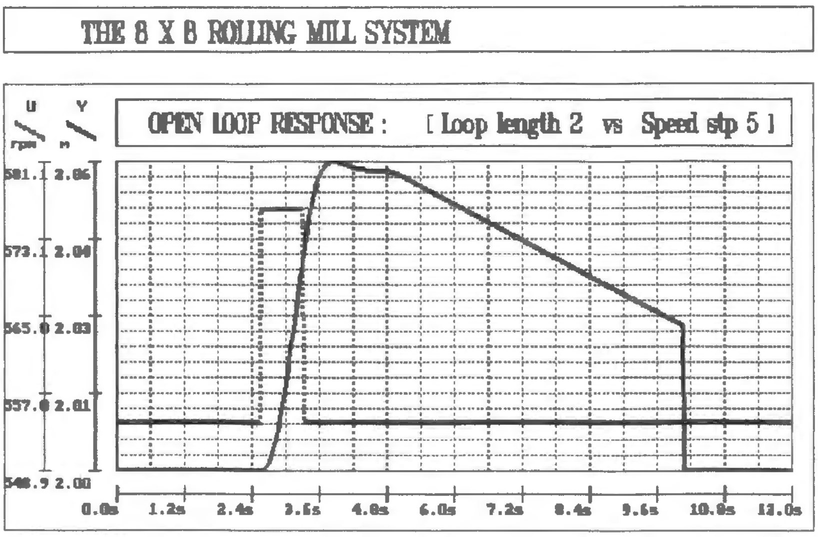


Figure 5.30: The length response of the second loop to a step in the input S6.

From this graph the transfer function equation is calculated to be a linear integral response with a gain constant of -0.012.

$$G_{56}(s) = \frac{-0.012}{s}$$

Transfer Function G_{66}.

Since there are no other factors affecting the second, third and fourth loops apart from the speeds of the two drives on either side of them, the response of the loop lengths should be the same. This fact is shown when figure 5.31 and the accompanying transfer equation are compared to figure 5.29 and the test results for the second loop.

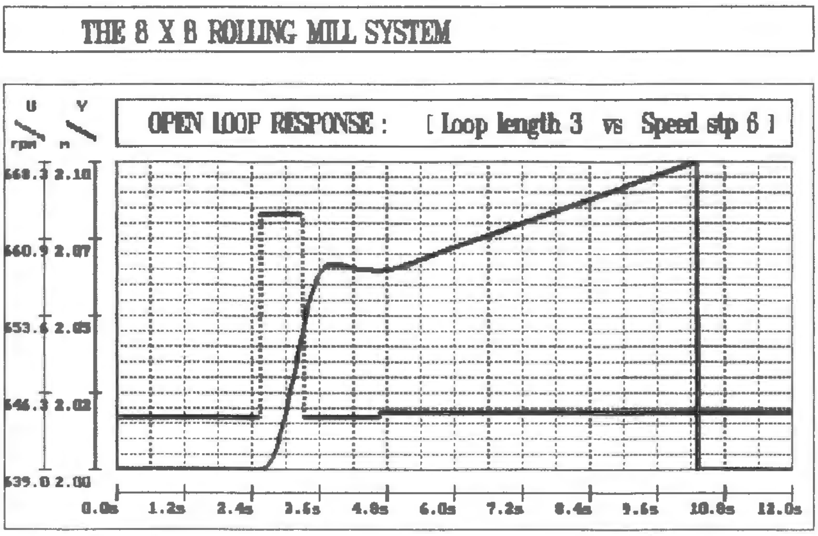


Figure 5.31: The length response of the third loop to a step in the input S6.

As for the second loop the transfer function is given by:

$$G_{66}(s) = \frac{0.013}{s}$$

Output Responses to a Step in Speed Setpoint S7.

Transfer Function G_{67}.

The response of the third loop length to a step in the speed of the drive after it, is the same as for the second loop. The transfer function G_{67} is shown graphically by figure 5.32.

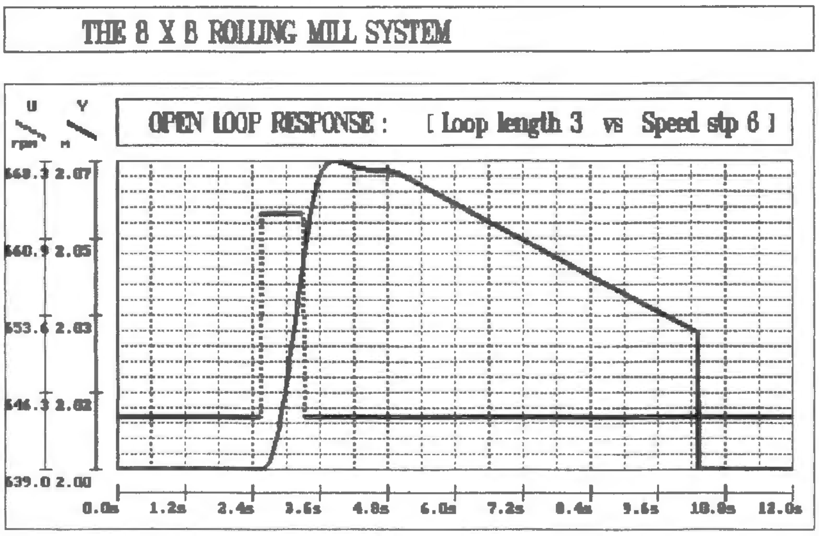


Figure 5.32: The length response of the third loop to a step in the input S7.

The equation describing this response confirms the transfer function G_{56} calculated for the second loop.

$$G_{67}(s) = \frac{-0.012}{s}$$

Transfer Function G_{77}.

The length of the fourth loop in the production process reacts in the manner illustrated by figure 5.33 when the input of the seventh drive is stepped.

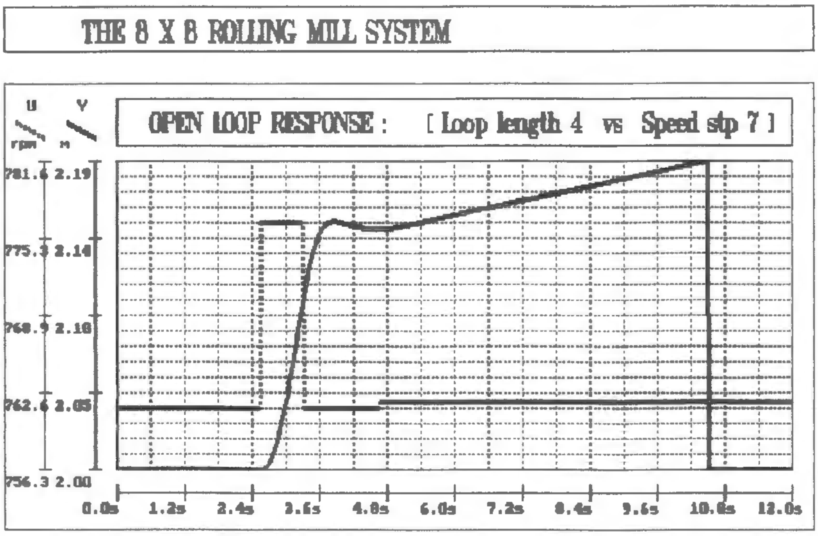


Figure 5.33: The length response of the fourth loop to a step in the input S7.

The transfer function derived from this graphic information is the same as those derived for the other two similar loops, L2 and L3. The gain constant in this case is calculated to be 0.013 whereas in G_{55} it is 0.012, and G_{66}, 0.013.

$$G_{77}(s) = \frac{0.013}{s}$$

Output Responses to a Step in Speed Setpoint S8.

Transfer Function G_{78}.

When the speed input of the eighth drive is stepped it causes the length of the fourth loop to decrease linearly in the manner shown by figure 5.34.

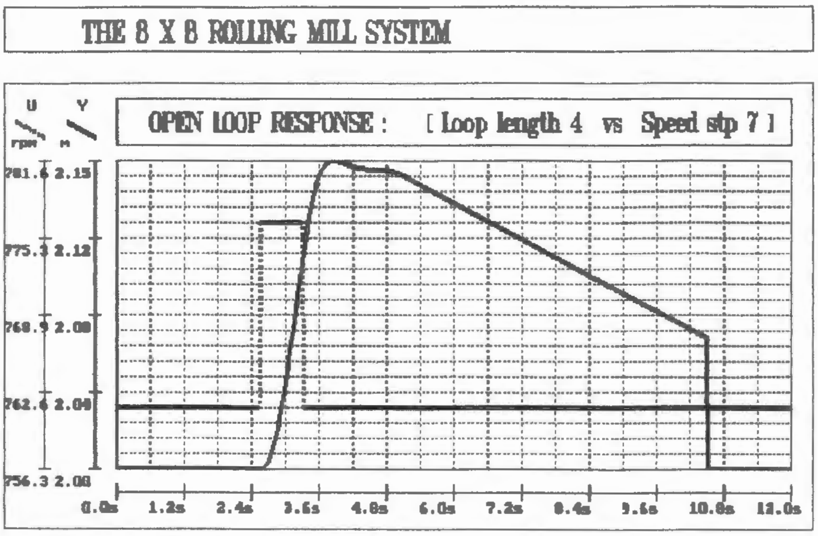


Figure 5.34: The length response of the fourth loop to a step in the input S8.

The transfer function equation that describes this performance is given by the following integral expression.

$$G_{78}(s) = \frac{-0.011}{s}$$

Transfer Function G_{88}.

As figure 5.35 shows, the speed-input to speed-output relation is a second order equation.

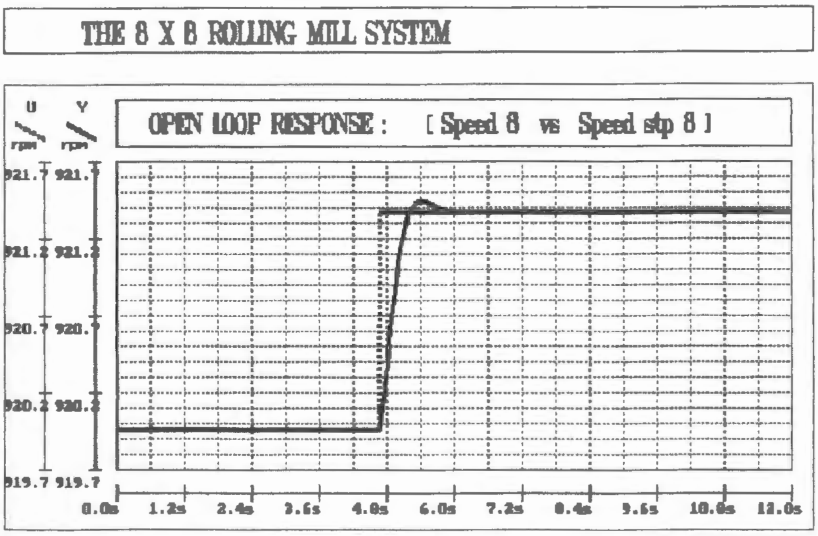


Figure 5.35: The speed response of the eighth drive to a step in the input S8.

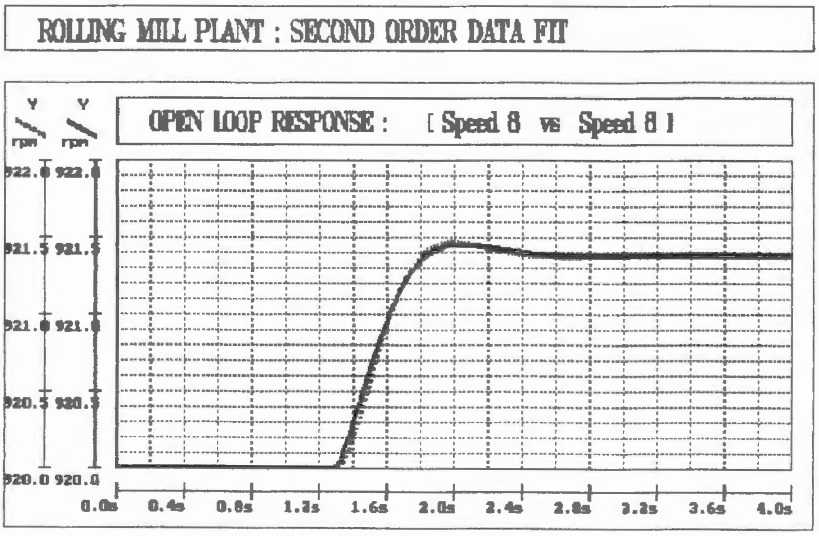


Figure 5.36: The second order data fit to G_{88}.

controller, K_D, is a diagonal matrix used to switch in the PI controllers at the right time.

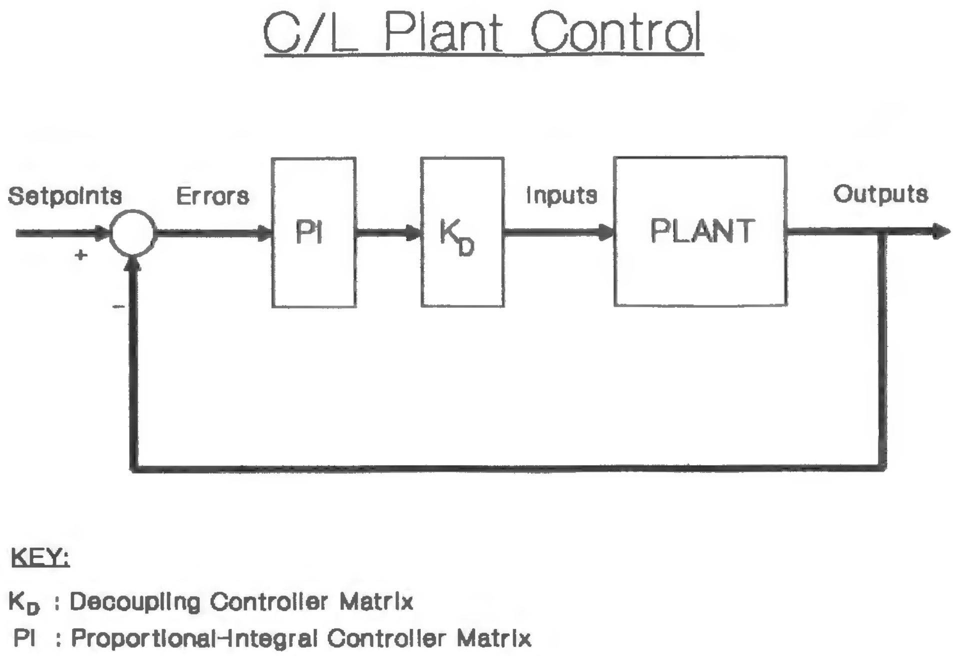


Figure 6.1: The control system implementation.

The Inverse and Direct Nyquist Array design methods are then used to design a decoupling control system. The decoupling control structures for all the transfer function matrices (which occur during the rolling cycle) are designed and presented in section 6.2.1.

The reasons why a single decoupler cannot be used for the control of this process and then the results of implementing a switching decoupling control system are presented in section 6.2.2.

The plant performance under the different control strategies adopted is then compared and discussed in section 6.3.

6.1. Single-Variable Proportional Integral Controllers.

The design of these controllers is done in the next section using the Characteristic Loci design method.[1] The results of implementing them on the plant simulator are then discussed.

6.1.1.Single-Variable Controller Design.

The eight proportional-integral controllers used in this single-variable control system are designed using the Generalized Nyquist Diagram. The eigenvalues of the matrix FQ(s) are plotted on this diagram to give the Characteristic Loci for the MIMO system. F is the feedback gain matrix and Q(s) is the open loop transfer function matrix. The feedback gain matrix in this system is a diagonal identity matrix.

The number of times that the loci encircle the critical point in a likewise direction to that taken by the Laplace operator along the Nyquist contour must be equal to the number of open loop unstable poles if the system is to be stable.

In this plant the number of unstable open loop poles is zero. This is determined by analysing the transfer function matrices derived in the previous chapter. Thus, the characteristic loci must not encircle the critical point.

Since the Generalized Nyquist Diagram gives no indication of the plant interaction, the generalized Bode plot is used to determine the interaction.

At low frequencies the magnitude of the eigenvalues is analysed and at high frequencies the interaction is determined from the corresponding eigenvectors.

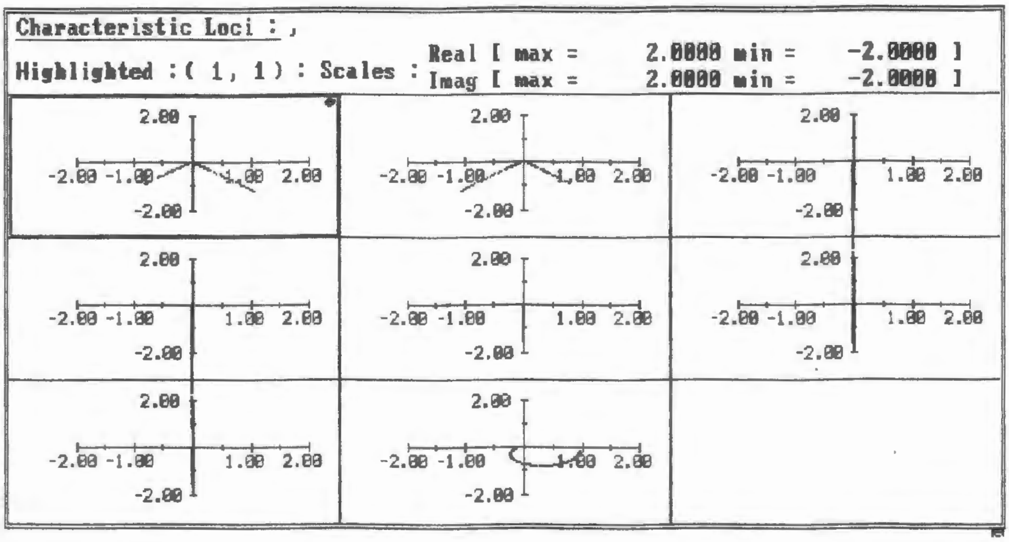


Figure 6.2: The plant characteristic loci.

The Characteristic Loci of the rolling mill transfer function matrix are shown in figure 6.2, for the situation where there is metal in all eight stands. The PI controllers are designed using the transfer functions derived for this situation since the metal bar spends 50% of the rolling cycle in all the stands. All other transfer function matrices are a variation of this case.

Figure 6.2 illustrates that the closed loop control system is stable for all feedback gains, since the loci do not encircle any point on the real axis.
In order to determine the plant interaction the generalized bode plot, shown in figure 6.3, is studied.

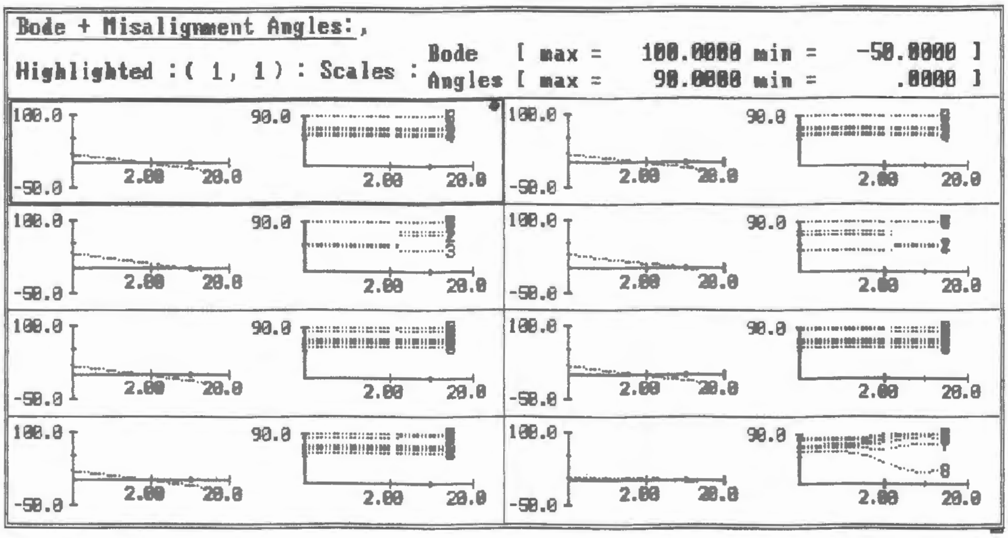


Figure 6.3: The plant generalized bode plot.

This figure illustrates that there is interaction at both low and high frequencies. The bode plots show that at low frequencies the eigenvalues are small and there is thus interaction. The misalignment angles are large at high frequency which indicates high frequency interaction.

The PI controllers are designed to get rid of these interactions while not making the system unstable. The PI control system matrix designed is given in figure 6.4.

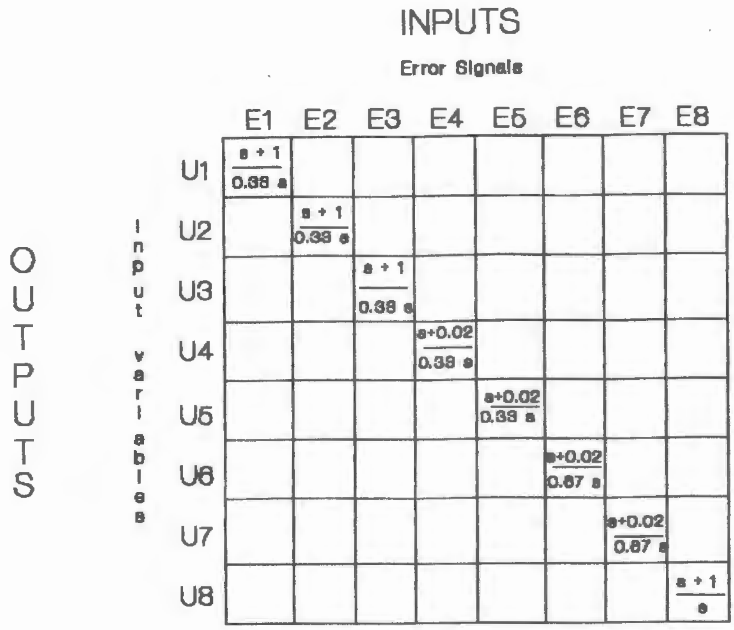


Figure 6.4: PI control system matrix.

The characteristic loci shown in figure 6.5 indicate that the closed loop system is still stable.

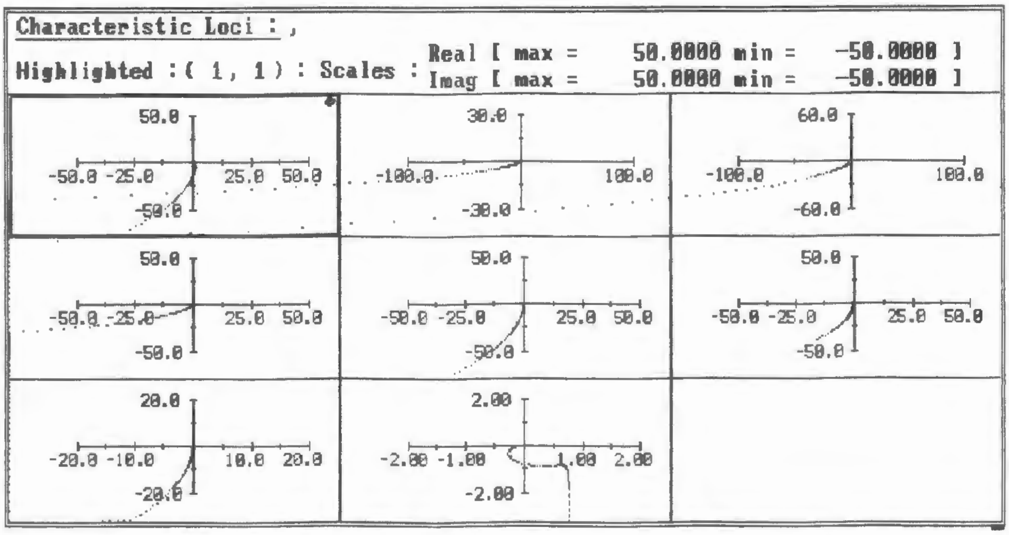


Figure 6.5: PI system characteristic loci.

The interaction at low frequencies is much reduced when using this PI control system. This is illustrated by the larger values of the eigenvalues at low frequencies on the bode plot shown in figure 6.6.

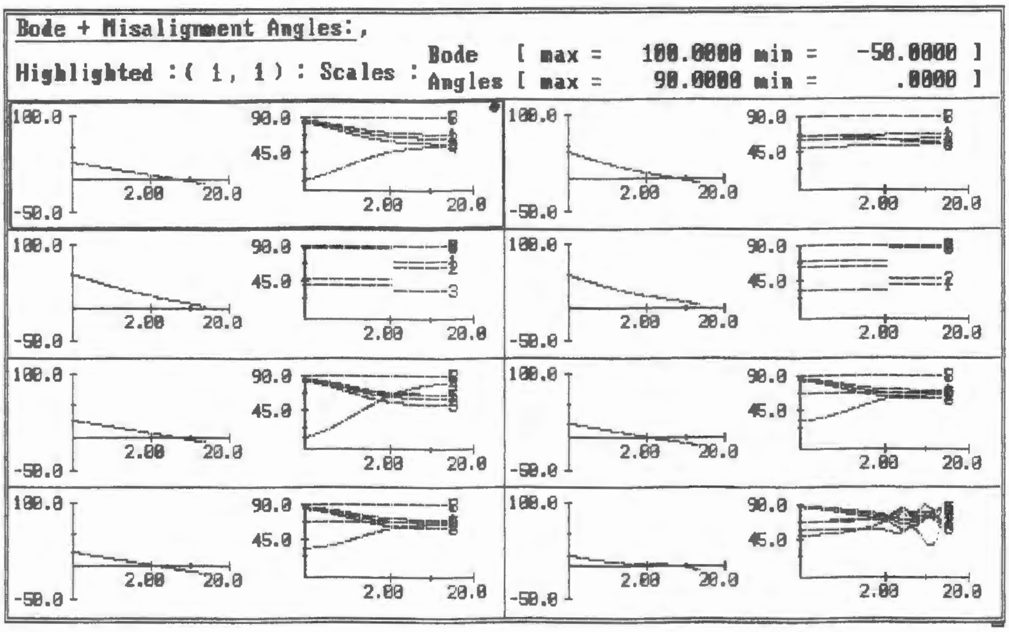


Figure 6.6: The PI system generalized bode plot.

If the gains of the PI controllers for the loop heights are increased, the system becomes unstable for the unity feedback gain used in this plant. The characteristic loci encircle the critical point (since the control of this process is done with unity feedback, the critical point is -1). This is shown in figure 6.7.

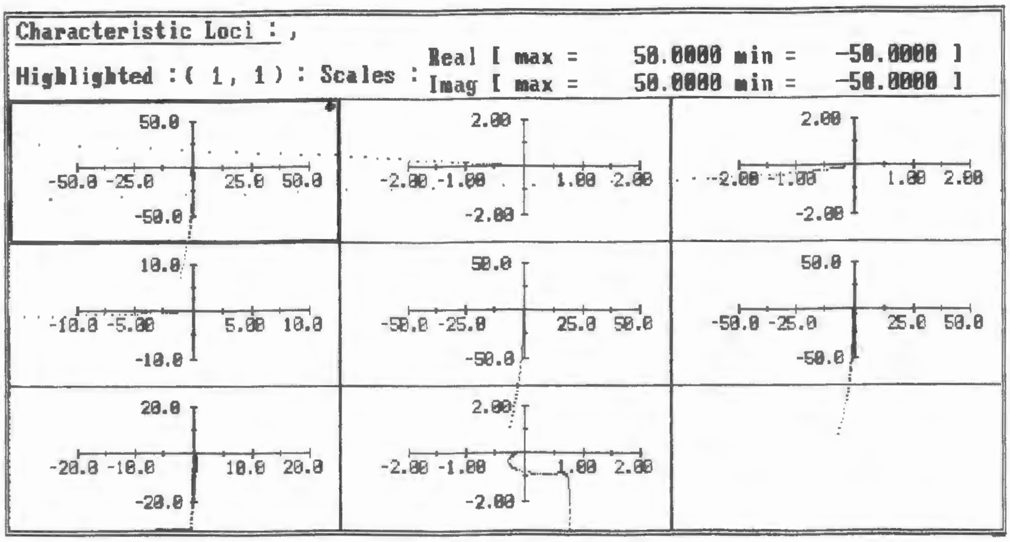


Figure 6.7: The effect of increasing the PI gains.

If the integral constant of these four loops is decreased the steady state interaction decreases. This is illustrated by the bode plots for this situation in figure 6.8.

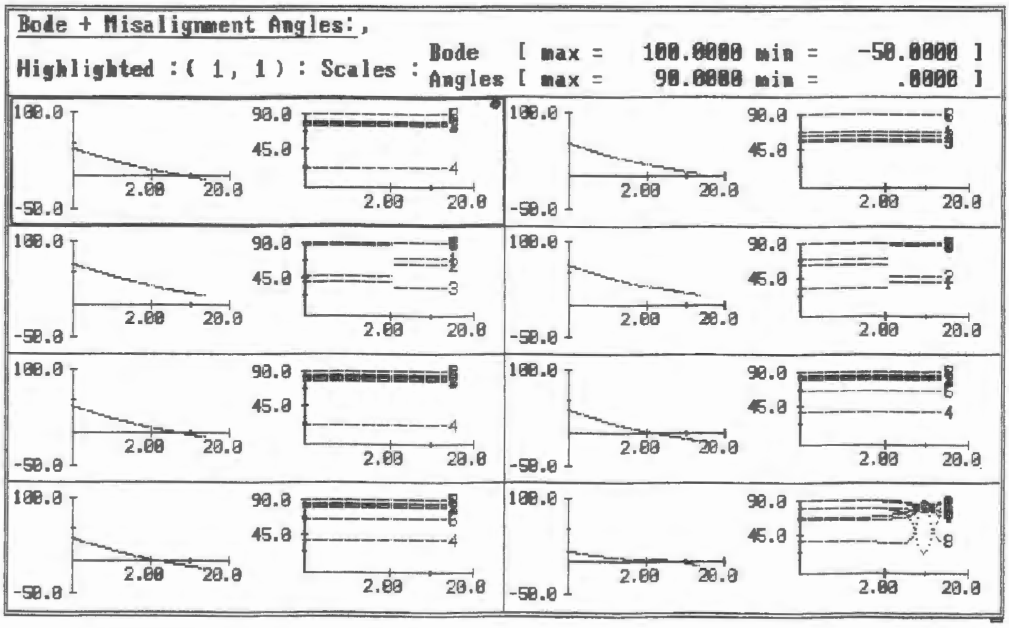


Figure 6.8: Effects of integral constant changes.

However, as discussed in section 6.1.2, the high frequency interaction is not improved at all. It is in fact worse although this is difficult to tell from the above misalignment angles.

The PI constants shown in figure 6.4 are the ones used in the single-variable control system initially. These constants are then tuned in an attempt to obtain the best plant performance during the rolling cycle. This tuning is done on the simulator and the best possible results using this control system are presented in the next section.

6.1.2: PI Control System Performance.

The PI control system designed in the previous section and shown in figure 6.4, is now used to control the rolling mill simulator.

The PI constants were tuned to obtain the best performance (the final values used are given in Appendix N).

6.1.2.(a) Production performance under PI control.

The rolling of one steel billet is shown in figure 6.9.1 The figure shows the output variables and their corresponding setpoints. (This figure is compared to the production results from the plant under multivariable control, shown in figure 6.9.2, in section 6.3).

Each loop is formed as soon as the metal is in the pair of stands between which it is formed. The loop setpoints are stepped up to the desired level. These setpoints are then later stepped down in order to remove the loop before the metal leaves the previous stand. This is to prevent the end of the bar from whipping.

Since the loop outputs do not exist during some of the stages in the rolling cycle, the corresponding PI controllers must be switched out of the system. This is to prevent a loop controller from speeding up a drive indefinitely in an attempt to drive the error signal to zero, while the closed loop output is non-existent. This switching is achieved by changing the corresponding diagonal element of the single-variable "decoupler" matrix (K_D) to either zero or one (depending on the stage of the process).

Figure 6.9.1 clearly shows the interaction effects during the rolling cycle. This is especially observable on the graph of the first loop. Each time another loop forms, the interaction is fed back and it disturbs the loops being formed before it. Therefore, the control of a loop improves the closer it is to the end of the production line.

Although good control of the last loop is achieved, the performance of the first three loops is poor.

Loops #1 and #3 are seen to be suddenly reduced to zero, when the bar leaves the previous stand. This indicates that the bar would whip under this control. However increasing the speed of response of the third loop in order to make the loop decrease to zero at the right time causes the first and second loops to become unstable. If the loop is removed before the metal leaves the previous stand it causes the metal to be stretched between the two stands. Since the relation between the speed and the state of the metal is completely different when the metal is in a loop to when undergoes tensional forces, the system becomes unstable. For that reason the loops are allowed to whip rather than to disappear too quickly.

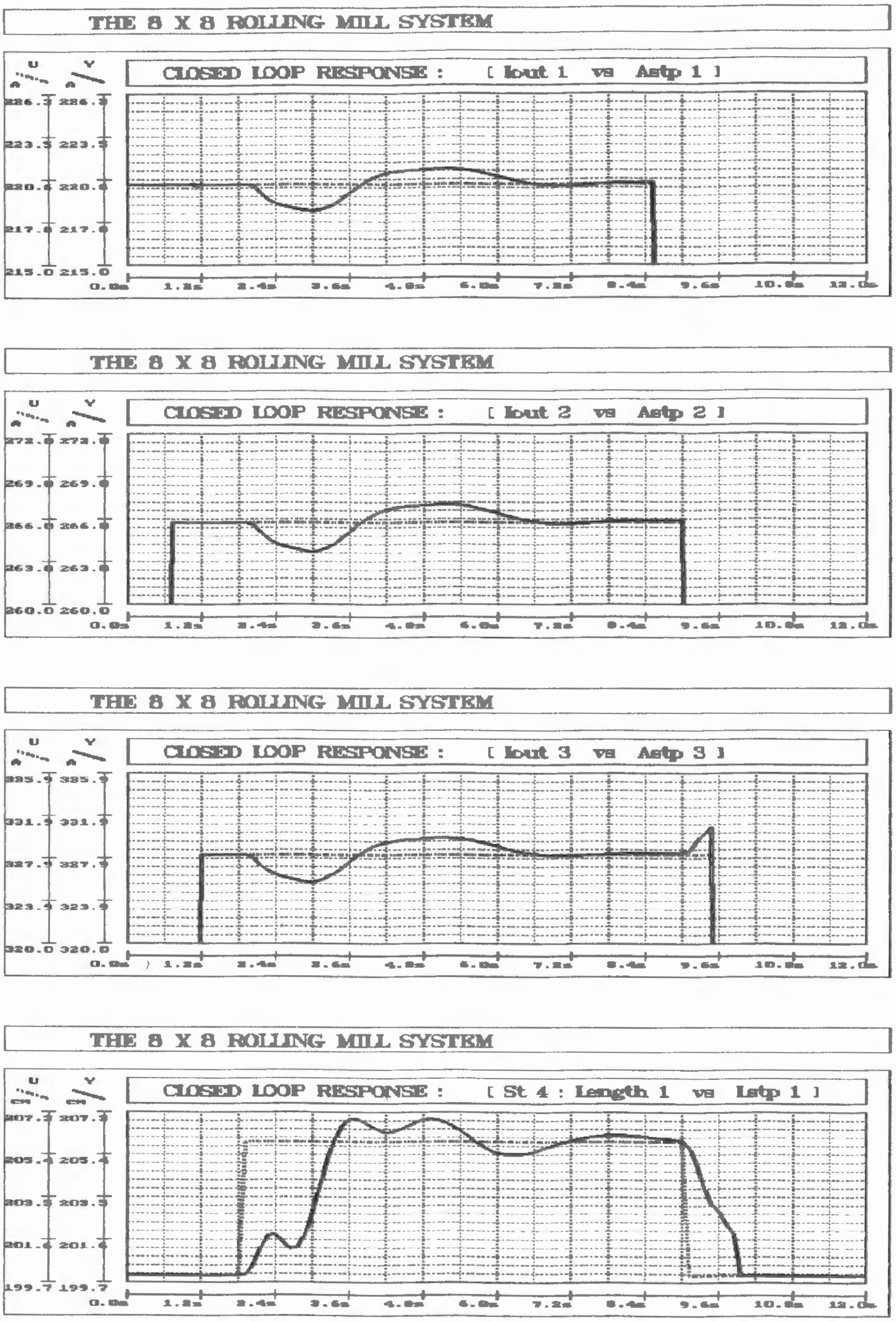


Figure 6.9.1a: The first four stands under PI control.

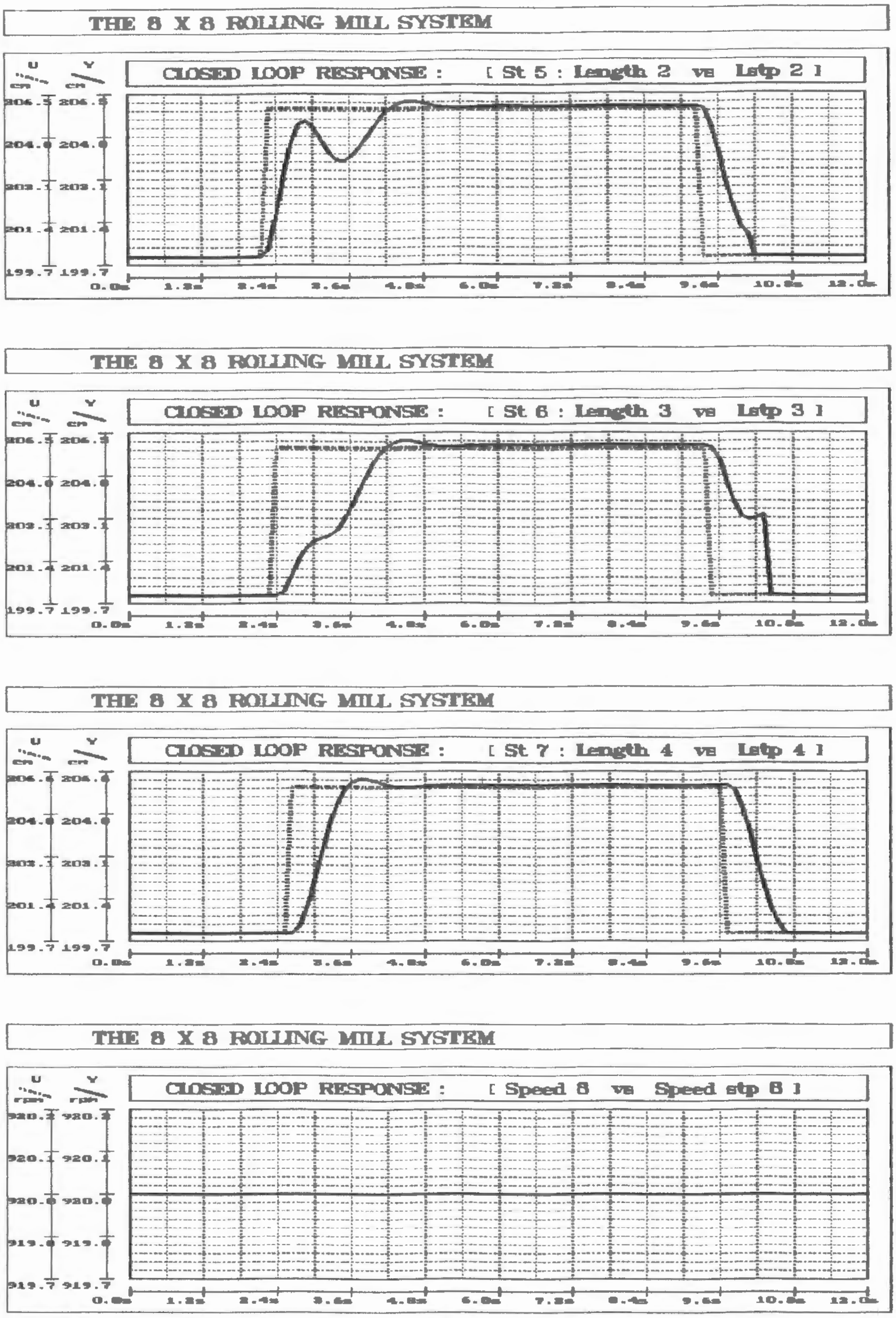


Figure 6.9.1b: The last four stands under PI control.

The tension control is seen to be satisfactory although the speed setpoints of these drives are not stepped during the normal rolling process cycle. (The maximum interaction of the first three outputs is calculated in section 6.3 to be 26.5 % of their setpoint value). The correct rolling tension would be set for the rolling of a particular industrial product. This would not then be altered during normal production. During commissioning, however, the tension required for a product would have to be tuned. The plant performance when any of the setpoints to the first three tension control drives is stepped is covered in section 6.1.2(b).

The speed of the last stand is also kept constant during production and is only affected if the fourth loop disappears too quickly, inducing a tension in the metal between the last two stands. As with the tension control of the plant, an optimum running speed would have to be found during commissioning. The results of changing the output speed of the production line are explained in section 6.1.2(c).

The effects on the production control, of a change in the metal model used in the simulator package (designed in Chapter 3) are investigated in section 6.1.2(d).

6.1.2.(b) The effects of tension setpoint changes.

The set of figures (6.10.1) show the response of the first four outputs (A1 - A3 and L1) to changes in the three tension setpoints. As is explained in chapter 4, the other three loop outputs and the speed of the eighth drive are not disturbed by changes in the running conditions of the first four drives. (The corresponding set of figures for the multivariable control system is 6.10.2)

The first tension setpoint is stepped in figure 6.10.1a. Although the first tension output, shown in the top graph, is satisfactory (8.4% overshoot etc. see section 6.3 for full details), the tension outputs of the other two drives do not settle down before the bar leaves the first stand.

The control of the first loop is not acceptable. This output increases to a value more than 140% of the setpoint and it then oscillates. (The length output of the loop is shown in the graphs in this chapter. A length of 200.0 cm indicates a height of 0.0 cm).

A step change in either of the other two setpoints also results in unacceptable control of the first loop output. The results of a step in the second tension setpoint are shown in figure 6.10.1b, while figure 6.10.1c shows the plant performance when the third tension setpoint is stepped.

This set of graphs shows that using eight single-variable PI controllers, a control system is unable to maintain the first loop output within

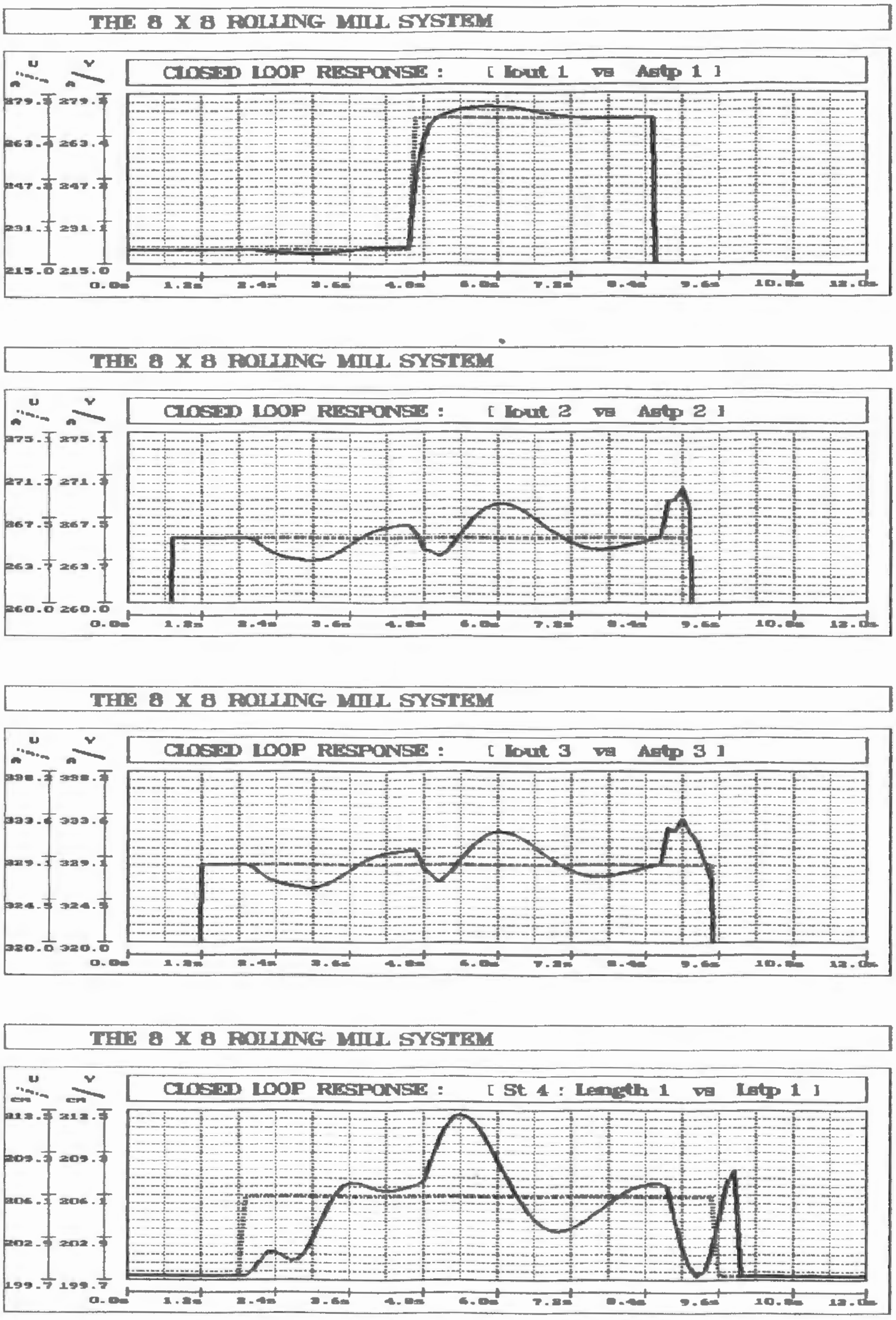


Figure 6.10.1a: The effects of a step in the first tension setpoint.

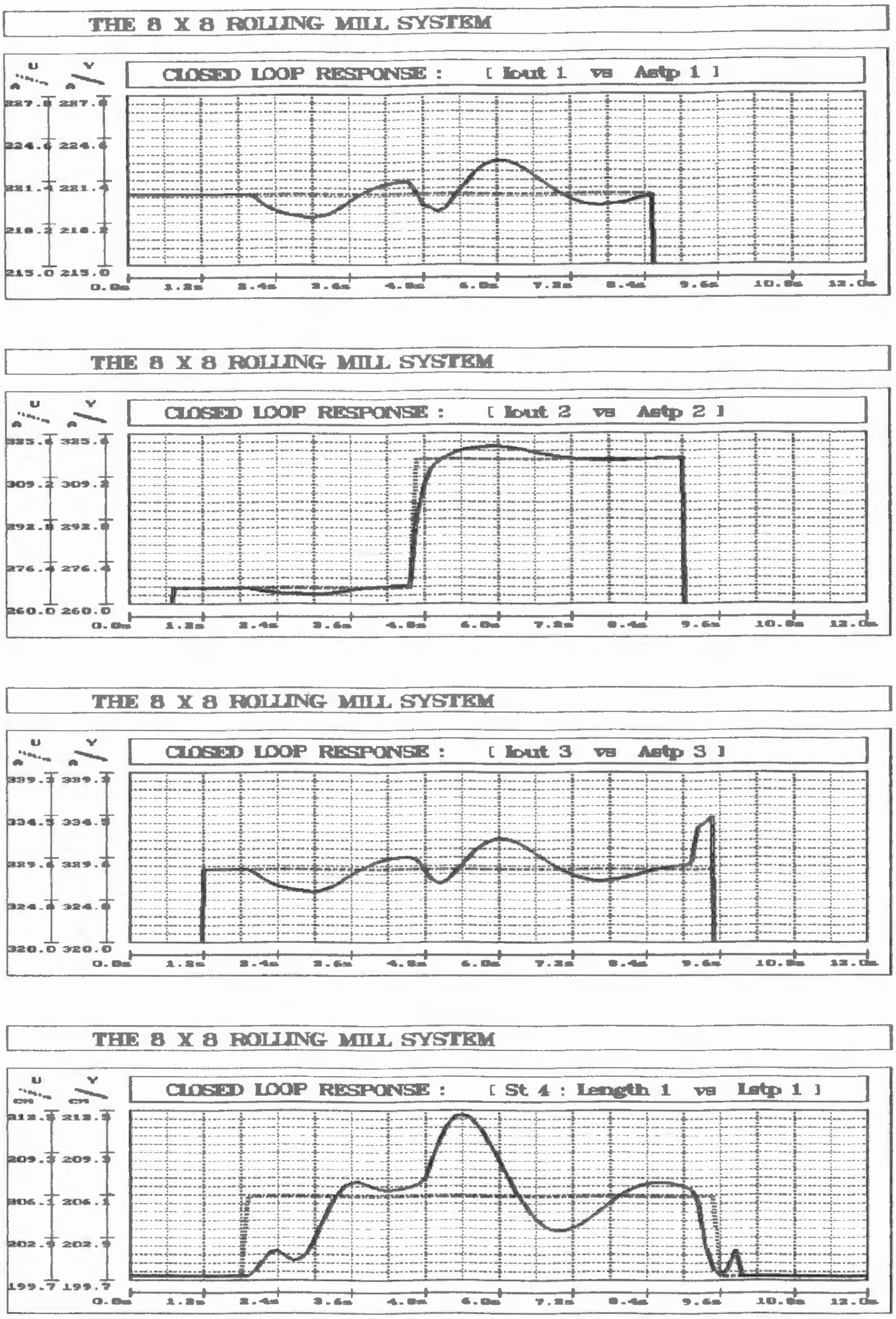


Figure 6.10.1b: The effects of a step in the second tension setpoint.

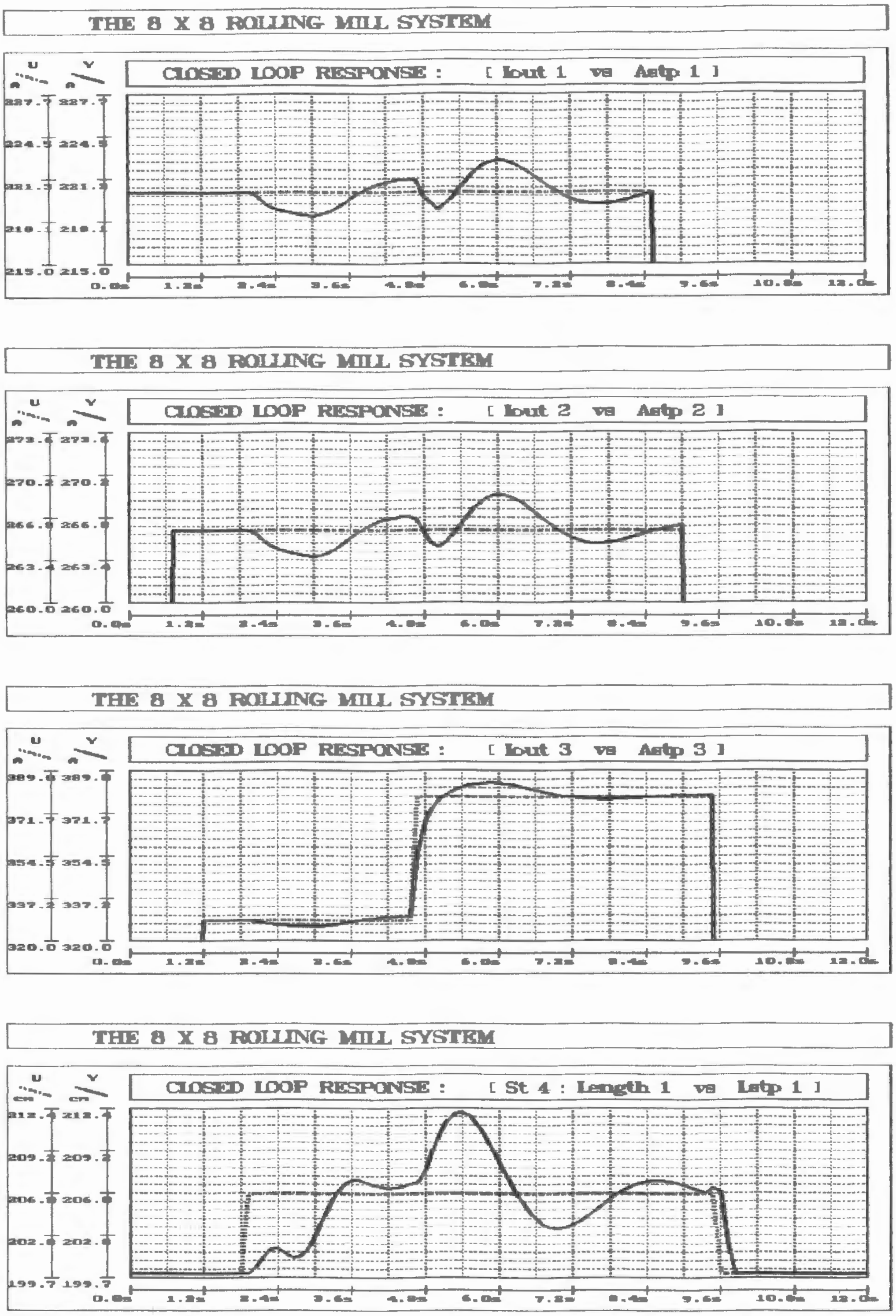


Figure 6.10.1c: The effects of a step in the third tension setpoint.

acceptable limits. These results are numerically analysed in section 6.3.

6.1.2.(c) The effects of a production speed change.

The speed of the production line is very important economically and, thus, the production line must run at an optimum speed. Not only must the output speed be high, but the production line is only part of the whole Hille Mill (as explained in the introduction), and therefore, it must fit in with the speed of operation of the rest of the rolling mill.

Although the speed setpoint will not normally be changed during normal production, it will be changed for different product sizes and if line A is running at the wrong speed compared to the rest of the mill.

The results of stepping the speed setpoint of the eighth drive (and, therefore, the production line) are shown in figure 6.11.1. The first four outputs (A1, A2, A3 and L4) are recorded in 6.11.1a, while the remaining outputs (L2, L3, L4 and S1) are illustrated in figure 6.11.1b.

This figure shows the poor output control when using single-variable PI controllers. This is shown when it is compared to the control under the multivariable switching decoupler system.

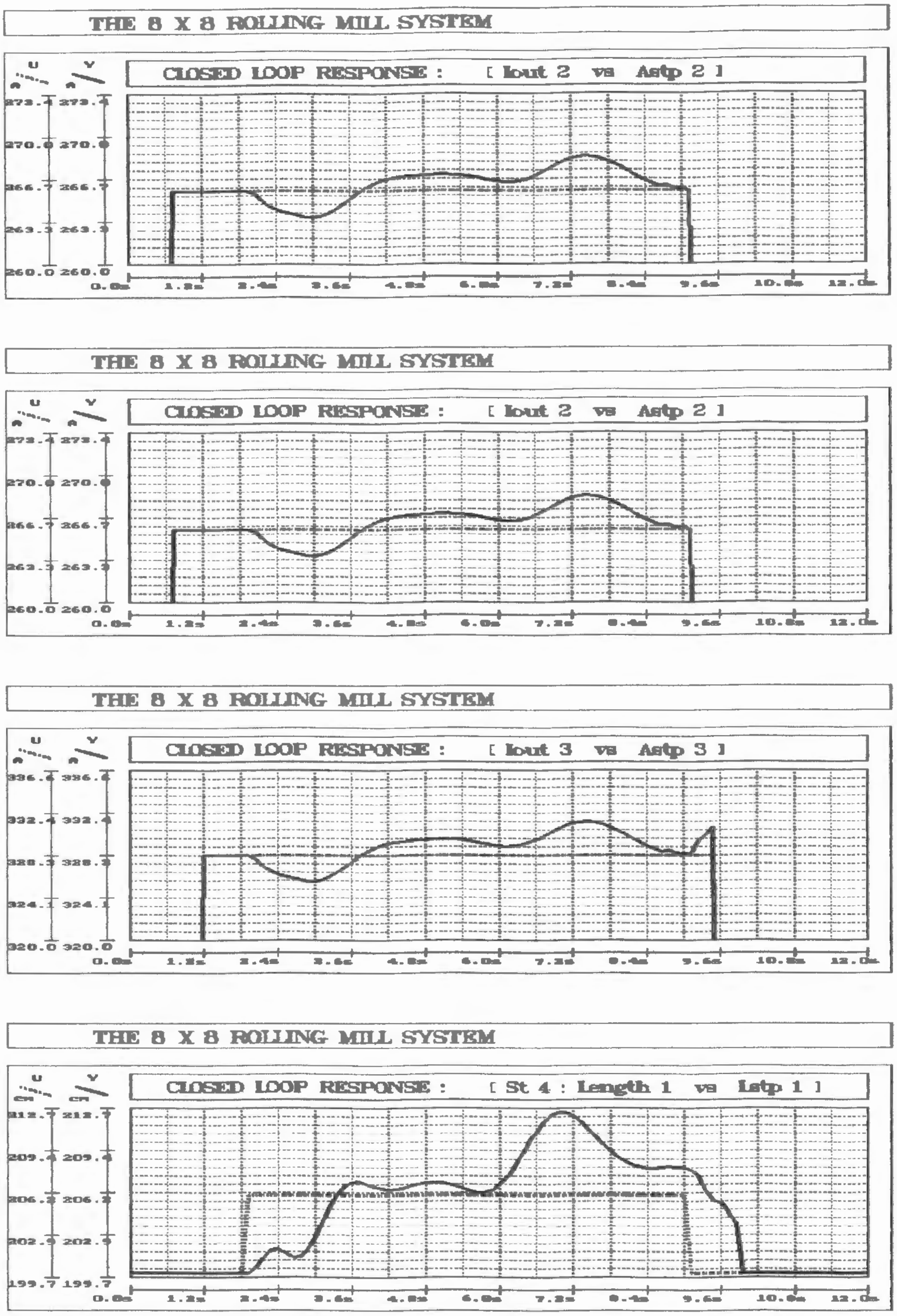


Figure 6.11.1a: The effects on the first four outputs of stepping the output speed of the production line.

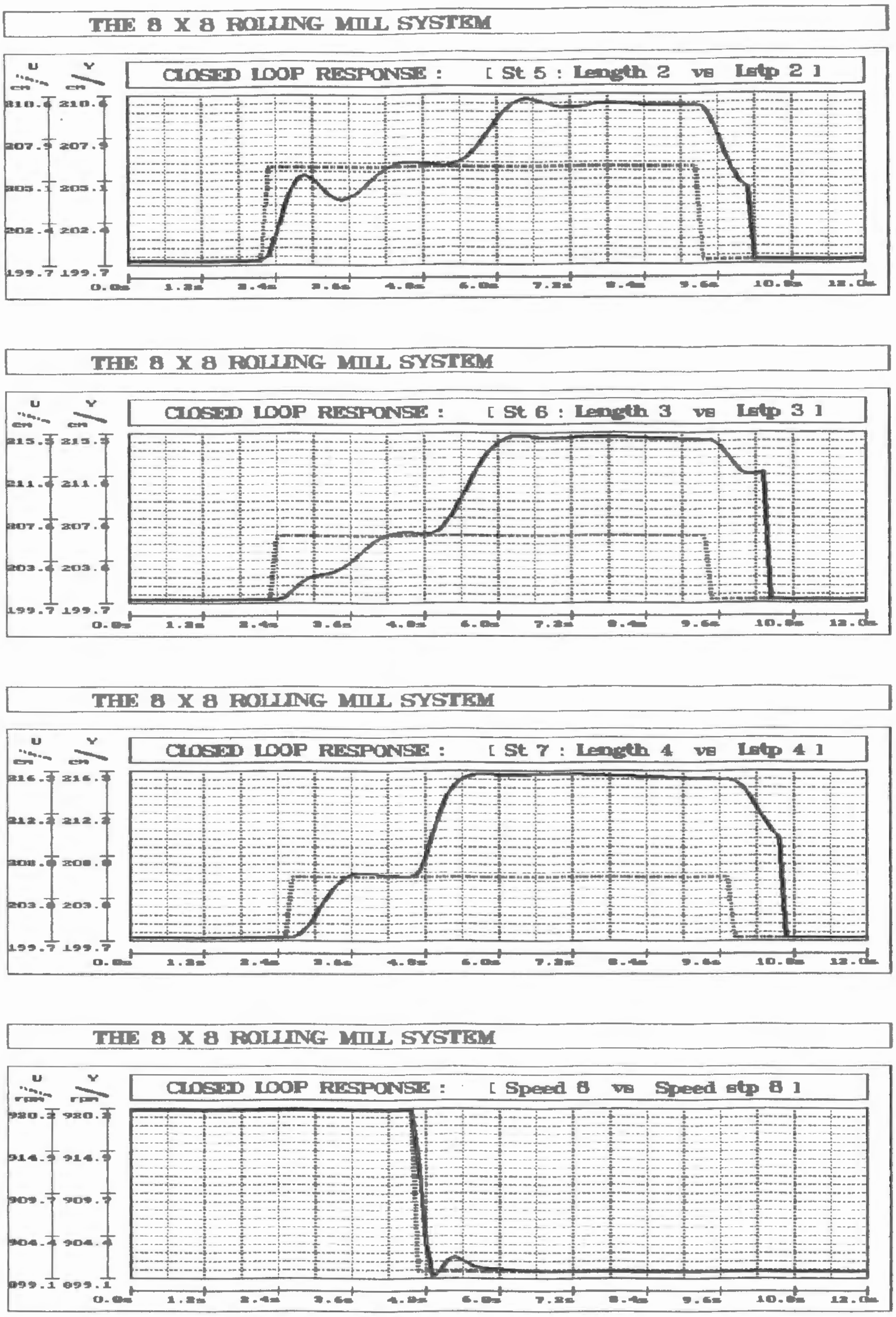


Figure 6.11.1b: The effects on the last four outputs of stepping the output speed of the production line.

In this situation, the tension outputs are kept within acceptable limits. However, the responses do not settle quickly and the bar leaves the stands before any steady state values are reached.

The responses of the four loop outputs are badly degraded by a step in the speed setpoint of the eighth stand.

The fourth loop is, obviously, affected first and by the largest amount. This is because when the speed of the eighth stand changes, the amount of metal in the fourth loop changes immediately. This loop's controller will change the speed of the seventh drive in order to make the loop output track the setpoint. This will then disturb the third loop and so on.

Although the response of these loops to this step change can be improved by re-tuning the PI controllers, the most important aim of the control system is to optimize normal production performance. Therefore, the single-variable controllers used are those that achieve this aim.

The performance of the multivariable system when the speed is changed is discussed in section 6.2.2.(c) and the corresponding output graphs are given in figure 6.11.2.

<u>6.1.2.(d) Performance effects due to a model change</u>.

Figures 6.12.1a and 6.12.1b show the effect of varying the value of Young's Modulus on the performance of the control system designed. Since there is only tension interaction between the first four stands (if the control system is working properly), the last three loops and the speed of the eighth drive are not affected by the change in Young's Modulus. For this reason only the three current outputs and the first loop are illustrated in these figures.

Figure 6.12.1a shows the effect of increasing the chosen value of 400 MPa by one hundred percent to 800 MPa.

As is illustrated by this figure, the closed loop system performance changes when the value of Young's Modulus is increased. The tension control in the first three drives does not change significantly. This is observed from the comparing the first three graphs in figures 6.9.1a and 6.12.1a. The percentage interaction in the three tension outputs is shown in section 6.3 to be almost exactly the same.

However, the control of the first loop is degraded by this change in the process model. The closed loop response is seen to be more oscillatory and have a larger overshoot of the setpoint (21.7% compared to 16.7%). The time to settle to within five percent of the setpoint is also larger.

The PI controller of the first loop output has been re-tuned in order to achieve the response illustrated here. The final values used for this controller under these conditions are given in Appendix N.

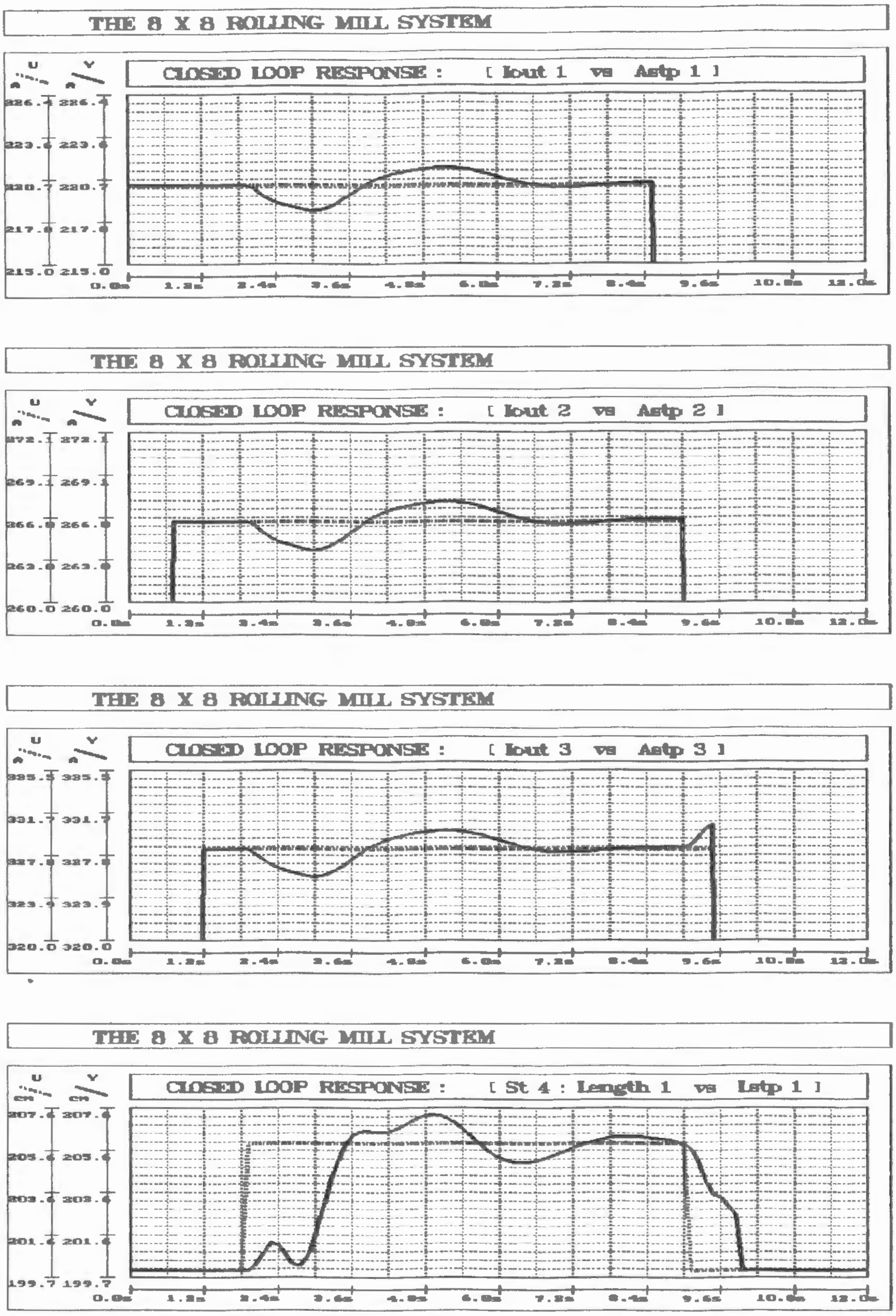


Figure 6.12.1a: The effects of an increase in the value of Young's Modulus.

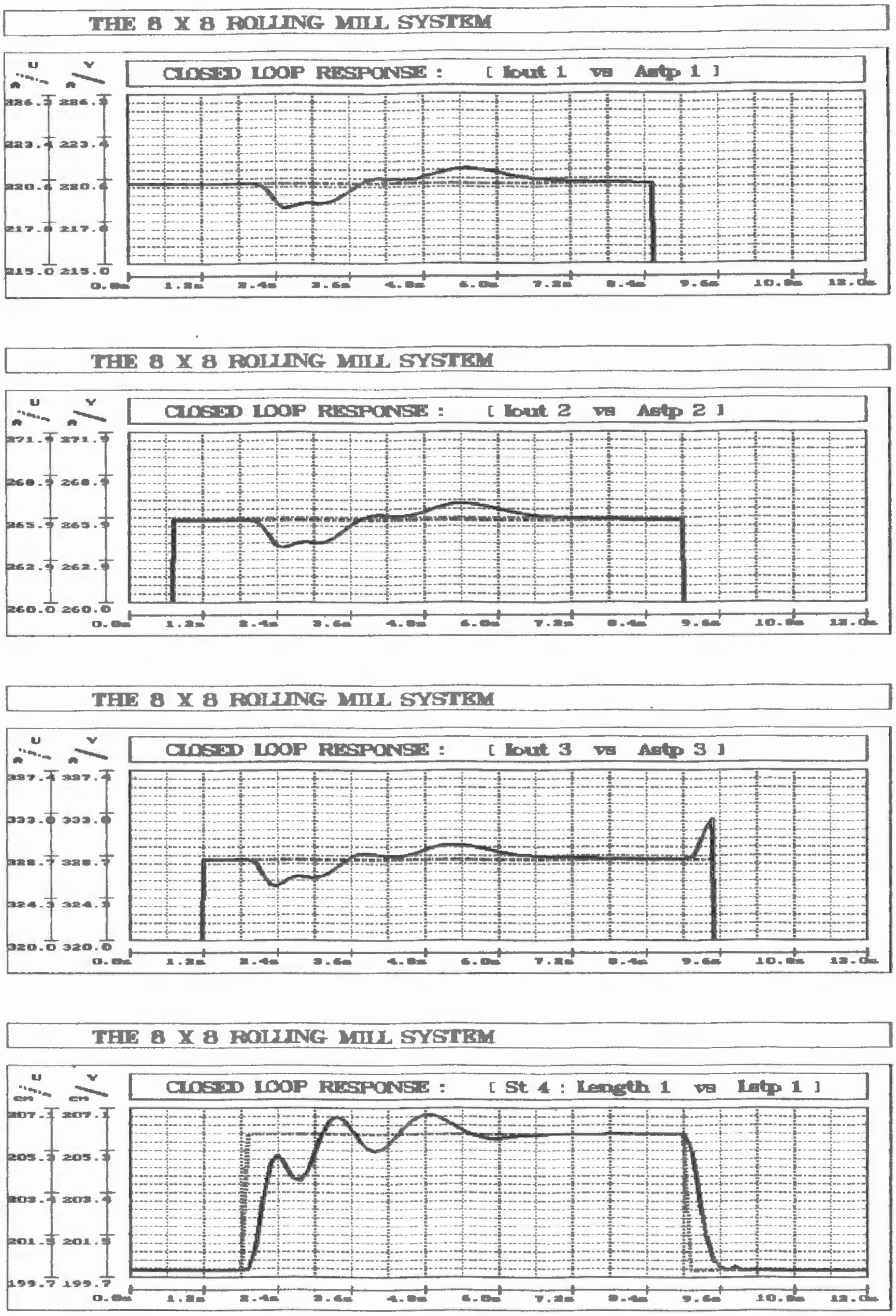


Figure 6.12.1b: The effects of decreasing the value of Young's Modulus.

Figure 6.12.1b shows the effects on controller performance if the value of Young's Modulus is decreased from 400 MPa to 100 MPa. This is a decrease of 75 % percent in the value used in the simulator. The transfer functions used in the controller design are calculated from the simulator when the value is assumed to be 400 MPa.

The effect of decreasing the value of this constant is calculated numerically in section 6.3, where it is compared to the effects on the other control system's performance. From figure 6.12.1b, however, it can be immediately seen that the controller performance improves. There is now less tension interaction between the first four drives. This is because the reduction in the value of Young's Modulus means that the metal is easier to work. Overall, the interaction in the three current outputs has decreased and the transient performance of the first loop is improved.

Although the PI controller values for this loop output have been tuned to obtain the best response, it is easier to push this loop response than before. Again, this is due to there being less tension interaction with the other drives. This is verified by the fact that the system becomes more sluggish when the value of this constant is increased to 800 MPa.

As shown in this section a large change in the value of Young's Modulus does not have a dramatic effect on the closed loop performance under PI control. The PI values for the first loop must be re-tuned to compensate for the change in the model of the metal tension.

Therefore, although this single-variable control system controls the plant outputs to an extent during normal production, the performance of the plant is not acceptable when any of the three tension setpoints or the speed setpoint of the last drive are stepped to a new value.

The percentage overshoot (or time to within 5% of the final value), the time constants, the percentage interaction and the effects of varying the process model on the performance of this control system are determined from the graphs shown in this section. The disturbance rejection of this system is illustrated in section 6.3.5. These results, along with the performance results of the multivariable system designed in section 6.2.1., are analysed and compared in section 6.3.

6.2. The Multivariable Switching Control System.

The control structures derived in Chapter 4 are designed in this section, using the Nyquist Array design methods[2]. As the loop outputs are zero in many of the situations during the rolling process, the Inverse Nyquist Array cannot be used in these cases. The Direct Nyquist Array is then used to design the required control structure. (The theory of these design metods is explained in Appendix O).

The transfer functions used to design the control structure at each step of the process cycle are summarised in Chapter 5.

The fact that the loop heights are not affected by any inputs, at some stages of the rolling process, means that all the decoupling elements to the input of the stand used to control the loop, must be zero. If this is not done then step changes in another loop's setpoint would lead to the stand being driven too fast or too slow, due to the feedforward elements in the decoupler. If the loop setpoint itself, is stepped during one of these stages, the motor will continue to change speed until the loop forms. Not only is this potentially dangerous to the state of the machine, but the drive will be running at the wrong speed when the bar enters it. This could lead to an unstable response.

Therefore, the decoupler that is required to control this process must switch in different structures at the correct time in the rolling process.

The results of implementing the multivariable control structures, designed in the next section, on the plant are illustrated in section 6.2.2.

6.2.1. The Design of the Control Structures.

Using the Inverse and Direct Nyquist Array design package available at the University of Cape Town[2], the required control structures are calculated.

The transfer functions for all the elements of the structures derived previously are presented and explained. Since the transfer functions describing the response of the loop heights do not change, the loop decoupling elements used in the control structures do not change either. Thus in many situations the only difference in the structure from the previous stage in the rolling process, is the addition or subtraction of a feedforward element.

The elements needed to cancel the interaction from the inputs which affect the tension, are however different under certain circumstances. This is because the transfer functions describing the plant tension interaction change in the manner described in section 5.2.2.

No Metal & Metal in Drive 1.

Figure 4.18 shows that the required control structure is four single-variable controllers in both of these situations. These single-variable controllers are PI controllers. The controllers used in the single-variable control system are tuned for the multivariable case. The final values used in the PI matrix are given in Appendix N.

Metal in Drives 1&2.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6								
S5								
S4								
S1						1.00	0.065	
S2						0.067	1.004	
S3								1.00

Figure 6.13: Control structure for metal in 1&2.

The structure required when the metal bar is only in the first two drives is given in figure 6.13. The rows and columns of this matrix (and all the others presented in this section) are presented in the same order as the corresponding reordered binary interaction matrix shown in figure 4.20. The control structure shown here is thus easily seen to be that required in figure 4.20. The empty matrix elements mean that that transfer function is zero.

The B.I.M. in figure 4.20 shows that there is no interaction between the loop outputs and any of the inputs. The control matrix, therefore, has a zero gain element between the loop drive error signal and their setpoints. The loop control drives will, thus, run at their previous speeds and not be affected by any of the inputs.

As expected, when the first drive speed setpoint is increased, the feedforward matrix increases the speed setpoint of the second drive in the correct ratio. The tension that would have resulted from the mismatch in the two drives' speeds does not occur. Similarly, the feedforward element S1.E2 compensates the first drive for any speed setpoint changes in the second drive.

The eighth drive speed and the current in the third drive are not disturbed by plant interaction at this stage and the control structure has unity gain elements between their error signals and their speed setpoints (inputs U in figure 6.13).

<u>Metal in Drives 1-3.</u>

The bar now enters the third drive and the plant interaction changes to that shown in figure 4.21.

The control structure elements needed in this situation are given in figure 6.14. The control structure is marked by the dividing matrix lines. As derived in chapter 4, this structure consists of one single-variable controller and a 3x3 multivariable matrix.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6								
S5								
S4								
S1						1.00	0.123	0.056
S2						0.146	1.018	0.076
S3						0.091	0.102	1.011

Figure 6.14: Control structure for metal in 1-3.

The four loop heights are still zero and therefore all the error signals are multiplied by zero gain elements.

Metal in Drives 1-4.

The steel bar then enters the fourth stand. Since the metal is still not between the fourth and fifth drives, the first loop is not formed yet. The drive speed of the fourth stand does however affect the currents in the first three drives.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6								
S5								
S4								
S1					0.448	1.00	0.480	0.383
S2					0.587	0.469	1.202	0.443
S3					0.661	0.612	0.686	1.280

Figure 6.15: Control structure for metal in 1-4.

The control structure required (figure 6.15) therefore contains feedforward elements which cancel out any speed setpoint changes in the fourth drive. The increase in the speed setpoint is multiplied by the correct ratios and the results added to the present speed setpoints of the other three drives. As explained before this reduces the plant interaction.

Metal in Drives 1-5.

The fifth stage of the process cycle is now entered. The first loop is formed during this stage. The loop output is disturbed by changes in the setpoints of any of the first five drives.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6								
S5								
S4				0.978	0.434	-.995	-.965	-.898
S1				0.433	0.192	1.00	0.122	0.057
S2				0.526	0.288	0.141	1.017	0.078
S3				0.647	0.287	0.079	0.065	1.009

Figure 6.16: Control structure for metal in 1-5.

Figure 6.16 shows the mix of controllers that make up the structure at this stage of the production cycle. This structure is derived and explained in Chapter 4. The difference (apart from the actual values of the controller elements) from the previous situation is that the multivariable controller is now a 4x4 matrix and there is a larger feedforward controller. This feedforward is from the speed setpoint of the fifth drive.

Metal in Drives 1-6.

With the metal in the first six drives of the production line, the control structure is given by figure 6.17. The second loop is formed during this stage.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6								
S5			0.944	1.00				
S4			0.924	0.978	0.434	-.995	-.965	-.896
S1			0.409	0.433	0.192	1.00	0.122	0.057
S2			0.496	0.526	0.288	0.141	1.017	0.073
S3			0.610	0.647	0.287	0.079	0.065	1.009

Figure 6.17: Control structure for metal in 1-6.

The bar's entrance into the sixth stand means that a feedforward controller from the speed setpoint of this drive must now be included in the structure. Practically, this is because the second loop is disturbed by changes in the speed of the sixth drive. In closed loop, the inner speed setpoint and thus the speed of the fifth drive will change. This in turn affects the first loop output. The interaction is fed backwards in this manner to all the previous outputs. The feedforward controller from the sixth setpoint is used to prevent this.

Metal in Drives 1-7.

Figure 6.18 is an illustration of the control system decoupler required at this stage of the process. The steel is now in the seventh drive of the production line.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7								
S6		0.881	1.00					
S5		0.831	0.944	1.00				
S4		0.813	0.924	0.978	0.434	-.995	-.965	-.896
S1		0.360	0.409	0.433	0.192	1.00	0.122	0.057
S2		0.437	0.496	0.526	0.233	0.141	1.017	0.073
S3		0.538	0.610	0.647	0.287	0.079	0.065	1.009

Figure 6.18: Control structure for metal in 1-7.

The third loop is formed during this stage. It is therefore necessary to add a feedforward controller from the seventh closed loop error signal (for the reasons previously discussed).

Metal in Drives 1-8.

The metal is now in all eight stands of the production line. This situation makes up 50% of the total rolling process cycle. The control structure, shown in figure 6.19, consists of four single-variable controllers, four feedforward controllers and one multivariable controller. This is the largest structure that is switched in by this system.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E8	E7	E6	E5	E4	E1	E2	E3
S8	1.00							
S7	0.863	1.00						
S6	0.761	0.881	1.00					
S5	0.709	0.831	0.944	1.00				
S4	0.693	0.813	0.924	0.978	0.434	-.995	-.965	-.896
S1	0.307	0.360	0.409	0.433	0.192	1.00	0.122	0.057
S2	0.373	0.437	0.496	0.526	0.233	0.141	1.017	0.073
S3	0.458	0.536	0.610	0.647	0.287	0.079	0.065	1.009

Figure 6.19: Control structure for metal in 1-8.

The addition of the one feedforward controller from the eighth stand is the only difference in the control structure from the previous stage of the cycle.

Metal in Drives 2-8.

The bar now starts to exit from the stands. After leaving the first stand, this stand is no longer disturbed by the rest of the plant. Similarly, the speed of this drive no longer affects the other plant outputs.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E1	E8	E7	E6	E5	E4	E3	E2
S1	1.00							
S8		1.00						
S7		0.853	1.00					
S6		0.751	0.881	1.00				
S5		0.709	0.831	0.944	1.00			
S4		0.712	0.835	0.949	1.005	0.498	-.975	-.934
S3		0.441	0.517	0.587	0.622	0.808	1.004	0.080
S2		0.358	0.420	0.477	0.508	0.251	0.048	1.00

Figure 6.20: Control structure for metal in 2-8.

The control structure thus becomes that shown in figure 6.20. This figure has been similarly rearranged and divided in order to illustrate the mix of controllers that make up the control system at this stage.

Metal in Drives 3-8.

After the bar leaves the second stand, the plant interaction is best decoupled by the use of the multivariable controller shown in figure 6.21.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E1	E2	E8	E7	E6	E5	E4	E3
S1	1.00							
S2		1.00						
S8			1.00					
S7			0.853	1.00				
S6			0.751	0.881	1.00			
S5			0.709	0.831	0.944	1.00		
S4			0.478	0.560	0.636	0.674	0.782	-.644
S3			0.321	0.376	0.426	0.453	0.492	1.00

Figure 6.21: Control structure for metal in 3-8.

Since there are now only two stands (#3 and #4) which both interact with each other, the multivariable matrix is now 2x2. The elements of all the feedforward matrices which were used to change the speed setpoint of the second drive have been removed.

Metal in Drives 4-8.

During this stage, there is no tension interaction. The binary interaction matrix can now be reordered so that it becomes completely lower triangular (see figure 4.29). Therefore, as explained in Appendix H, there is now no multivariable problem and the control structure is lower triangular itself. The structure only consists of single-variable and feedforward controllers.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E1	E2	E3	E8	E7	E6	E5	E4
S1	1.00							
S2		1.00						
S3			1.00					
S8				1.00				
S7				0.853	1.00			
S6				0.751	0.881	1.00		
S5				0.709	0.881	0.944	1.00	
S4				0.652	0.765	0.869	0.920	1.00

Figure 6.22: Control structure for metal in 4-8.

Figure 6.22 shows the required single-variable and feedforward controllers.

Metal in Drives 5-8.

The first loop is reduced to zero during the previous stage. All the row elements affecting this loop's controlling drive again return to zero in this situation.

Inputs: E : Error

Outputs: S : Speed Setpoint

	E1	E2	E3	E4	E8	E7	E6	E5
S1	1.00							
S2		1.00						
S3			1.00					
S4								
S8					1.00			
S7					0.868	1.00		
S6					0.761	0.881	1.00	
S5					0.709	0.831	0.944	1.00

Figure 6.23: Control structure for metal in 5-8.

Figure 6.23 illustrates that there is also no multivariable problem and that the control system consists of only single-variable and feedforward controllers.

5.2.2: The Complete Rolling Mill Cycle.

The transfer functions that change when the plant interaction changes are calculated in Appendix K. The transfer function equations of all the situations in the cycle described by the changing interaction matrices in the previous chapter are explained below.

No Metal and Metal in Drive #1.

INPUTS

Speed Setpoints

OUTPUTS		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{8.68}{s + 10}$							
Current	A2		$\frac{8.68}{s + 10}$						
Current	A3			$\frac{51.78}{s^2+9s \; 59.73}$					
Loop Lth	L1								
Loop Lth	L2								
Loop Lth	L3								
Loop Lth	L4								
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.38: The equations for metal in Drive #1.

Figure 5.38 shows the transfer function equations when there is no plant interaction.

Metal in Drives 1&2.

INPUTS

Speed Setpoints

OUTPUTS

		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{4.63}{s}$	$\frac{-0.8}{s}$						
	A2	$\frac{-0.3}{s}$	$\frac{4.5}{s}$						
	A3			$\frac{51.76}{s^2+9s+59.76}$					
Loop Lth	L1								
	L2								
	L3								
	L4								
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.39: The equations for metal in Drives 1&2.

There is now plant interaction between the first two stands. G_{11}, G_{12}, G_{21} and G_{22} have changed. This is illustrated by figure 5.39.

Metal in Drives 1-3.

In this stage of the cycle the first three drives interact. The new transfer functions of $G_{(1-3)(1-3)}$ are shown in figure 5.40.

INPUTS

Speed Setpoints

OUTPUTS		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{4.41}{s}$	$\frac{-0.51}{s}$	$\frac{-0.21}{s}$					
	A2	$\frac{-0.59}{s}$	$\frac{4.22}{s}$	$\frac{-0.25}{s}$					
	A3	$\frac{-0.31}{s}$	$\frac{-0.32}{s}$	$\frac{3.93}{s}$					
Loop Lth	L1								
	L2								
	L3								
	L4								
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.40: The equations for metal in Drives 1-3.

Metal in Drives 1-4.

The interaction matrix is shown in figure 4.6. The active functions are given in section 5.1.2.

Metal in Drives 1-5.

The B.I.M. is shown in figure 4.7. The active equations are given in section 5.1.2.

Metal in Drives 1-6.

The B.I.M. is shown in figure 4.8. The active equations are given in section 5.1.2.

Metal in Drives 1-7.

The binary interaction matrix is shown in figure 4.9. The active equations are given by their equivalent calculated in 5.1.2.

Metal in Drives 1-8.

These functions are explained in section 5.1.2.

Metal in Drives 2-8.

INPUTS

Speed Setpoints

OUTPUTS		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{8.68}{s+10}$							
Current	A2		$\frac{3.28}{s}$	$\frac{-1.10}{s}$	$\frac{-0.97}{s}$				
Current	A3		$\frac{-1.32}{s}$	$\frac{2.95}{s}$	$\frac{-1.16}{s}$				
Loop Lth	L1		$\frac{0.005}{s}$	$\frac{0.005}{s}$	$\frac{0.006}{s}$	$\frac{-0.012}{s}$			
Loop Lth	L2					$\frac{0.012}{s}$	$\frac{-0.012}{s}$		
Loop Lth	L3						$\frac{0.013}{s}$	$\frac{-0.012}{s}$	
Loop Lth	L4							$\frac{0.013}{s}$	$\frac{-0.011}{s}$
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.41: The equations for metal in Drives 2-8.

Figure 5.41 shows that the first stand no longer interacts with the rest of the plant. The equations $G_{(2-4)(2-4)}$ change from their previous form.

Metal in Drives 3-8.

INPUTS

Speed Setpoints

OUTPUTS (Current / Loop Lth / Speed)

	S1	S2	S3	S4	S5	S6	S7	S8
A1	$\frac{8.68}{s+10}$							
A2		$\frac{8.68}{s+10}$						
A3			$\frac{2.44}{s}$	$\frac{-1.64}{s}$				
L1			$\frac{0.007}{s}$	$\frac{0.008}{s}$	$\frac{-0.012}{s}$			
L2					$\frac{0.012}{s}$	$\frac{-0.012}{s}$		
L3						$\frac{0.013}{s}$	$\frac{-0.012}{s}$	
L4							$\frac{0.019}{s}$	$\frac{-0.011}{s}$
S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.42: The equations for metal in Drives 3-8.

In this stage of the cycle the second stand also becomes independent of the plant. G_{33}, G_{34}, G_{43} and G_{44} are different from the previous stage.

Metal in Drives 4-8.

Figure 5.43 illustrates that the first loop behaves like the other loops, now that there is no tension interaction with the first three drives.

INPUTS

Speed Setpoints

OUTPUTS		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{8.68}{s + 10}$							
Current	A2		$\frac{8.68}{s + 10}$						
Current	A3			$\frac{61.78}{s^2+9s}$ 59.73					
Loop Lth	L1				$\frac{0.012}{s}$	$\frac{-0.012}{s}$			
Loop Lth	L2					$\frac{0.012}{s}$	$\frac{-0.012}{s}$		
Loop Lth	L3						$\frac{0.013}{s}$	$\frac{-0.012}{s}$	
Loop Lth	L4							$\frac{0.013}{s}$	$\frac{-0.011}{s}$
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.43: The equations for metal in Drives 4-8.

Metal in Drives 5-8.

The B.I.M. matrix is shown in figure 4.14. The active transfer functions illustrated there are given by their equivalent when the metal is in all the drives, calculated in section 5.1.2.

Metal in Drives 6-8.

The interaction matrix is shown in figure 4.15. The active equations are given by their equivalent when the metal is in all the drives, calculated in 5.1.2.

Metal in Drives 7&8.

The binary interaction matrix is that shown in figure 4.16. The active transfer functions of this matrix are given by their equivalent calculated in section 5.1.2.

Metal in Drive 8.

The transfer functions are the same as when there is metal in the first drive. This is illustrated in figure 5.38. The bar then leaves this stand and the cycle will begin again when another billet enters the production line.

5.3: The Relevance of the Simulator Functions.

As is shown in section 5.1 the transfer function G_{33} when there is metal in all the stands can be approximated using an integral function with a gain of 0.65. This is done using the actual plant data. The transfer function G_{33} is calculated from the simulator as an integral function with a gain of 3.15.

Although the gain of the integral function derived from the simulator is different, the integral form is correct. This gain difference does not change the form of the controller, it will only change it's steady state gain.

The simulator is, thus, an relevant model of this transfer function.

The plant data is shown, in section 5.1, to be inadequate for use in the design of the simulator package. The simulation is, therefore, designed using mathematical models as described in Chapter 3.

The transfer functions are derived from the simulator in section 5.2. They are shown to be of the correct form in section 5.3 and indicate the relevance of the simulator for this control study.

Using the transfer function matrix calculated in section 5.1.2 and those illustrated in section 5.2.2, the control structures that are required in each situation, as laid out in Chapter 4, are designed using the Nyquist Array design methods. The designs are explained and presented in the next chapter. The results of switching between the control structures when controlling the plant are then compared to the plant performance under P.I. control.

CHAPTER 6: CONTROL SYSTEM DESIGN AND IMPLEMENTATION

The control system's main aims are to control the speed of production and the quality of the product. Therefore, as discussed in Chapter 2, the plant outputs that are controlled are the three current outputs (A1, A2 and A3), the four loop height outputs (L1 - L4) and the speed of the last drive (S1).

The inputs that are used to control these plant output variables are the eight drive speed setpoints. The plant interaction between these input and output variables changes according to the position of the bar in the production line. This changing interaction is fully explained in Chapter 4.

In order to design the control systems, the transfer function matrix is derived. However, due to the changing plant interaction, not only do the active transfer function elements change, but the transfer functions describing some elements also change. All the possible transfer function matrices are presented in Chapter 5. These transfer functions are calculated from the rolling mill simulator designed in Chapter 3.

The transfer function matrix in figure 5.37 is used to design the eight single-variable PI controllers. This matrix gives the transfer functions for the situation when there is metal in all eight stands. The design is explained in section 6.1.1. and the need to switch in the four loop height PI controllers is discussed. This switching single-variable control system is then implemented on the simulator as shown in figure 6.1 and the results are recorded. In the single-variable control situation the decoupling

Using the same data fitting program, Appendix L, as in section 5.1, to analyse this second order function, the response is modelled, as shown in figure 5.36, by the following equation.

$$G_{88}(s) = \frac{36.75}{s^2 + 8s + 36.75}$$

The final transfer function matrix for the situation where the steel bar is in all the drives is thus that shown in figure 5.37.

INPUTS

Speed Setpoints

OUTPUTS		S1	S2	S3	S4	S5	S6	S7	S8
Current	A1	$\frac{3.88}{s}$	$\frac{-1.04}{s}$	$\frac{-0.74}{s}$	$\frac{-0.67}{s}$				
Current	A2	$\frac{-1.22}{s}$	$\frac{3.68}{s}$	$\frac{-0.89}{s}$	$\frac{-0.79}{s}$				
Current	A3	$\frac{-1.10}{s}$	$\frac{-1.09}{s}$	$\frac{3.15}{s}$	$\frac{-1.01}{s}$				
Loop Lth	L1	$\frac{0.004}{s}$	$\frac{0.004}{s}$	$\frac{0.004}{s}$	$\frac{0.005}{s}$	$\frac{-0.012}{s}$			
Loop Lth	L2					$\frac{0.012}{s}$	$\frac{-0.012}{s}$		
Loop Lth	L3						$\frac{0.013}{s}$	$\frac{-0.012}{s}$	
Loop Lth	L4							$\frac{0.013}{s}$	$\frac{-0.011}{s}$
Speed	S1								$\frac{36.75}{s^2+8s+36.75}$

Figure 5.37: The complete transfer function matrix.

Metal in Drives 6-8.

Figure 6.24 is the control structure that is designed using the INA design package. The second loop is again inactive after the previous stage.

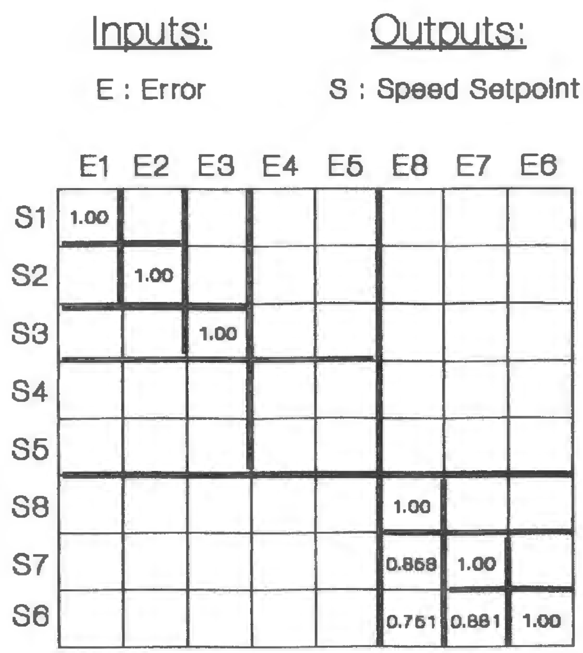


Figure 6.24: Control structure for metal in 6-8.

Metal in Drives 7-8.

The metal is now only between the last two stands. The only plant interaction is from the eighth drive, which affects the fourth loop output. A single feedforward element is therefore used to cancel this interaction.

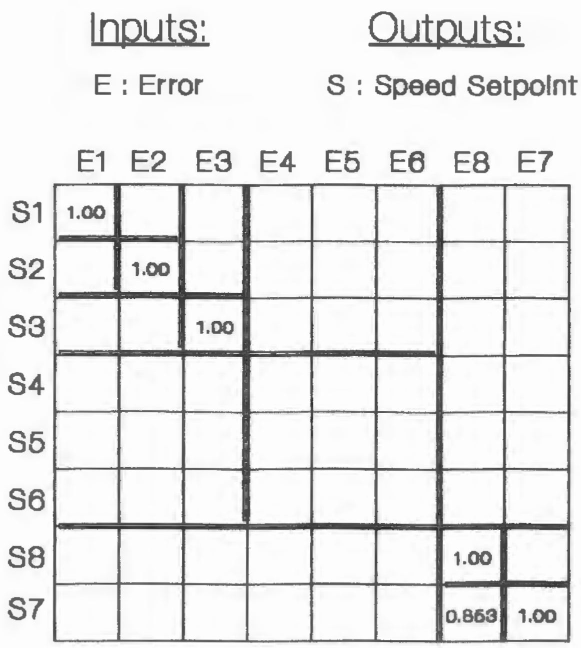


Figure 6.25: Control structure for metal in 7&8.

This feedforward element is shown in figure 6.25.

Metal in Drive 8.

The control structure in this case reduces to only single-variable controllers. This is shown in figure 4.33.

The control structures designed and explained in this section are used in the following section to control the plant simulator. The control system, which is a composition of both controller matrices shown in figure 6.1, switches between these decoupling structures in order to obtain the best closed loop plant performance.

6.2.2. Implementing the Multivariable Control System.

The results of implementing this switching control system on the production line are recorded and explained in the following sections. As with the single-variable control system, the production performance is dealt with first. Then, the effects on this performance of stepping the tension and speed setpoints to the closed loop plant are discussed. The change in the control system performance due to a large model change are then investigated in section 6.2.2(d).

These results are compared in sections 6.3.1 - 6.3.4 with the results achieved under single-variable control.

6.2.2.(a) Production performance under MV control.

The performance of the control system during a complete rolling cycle is shown in figures 6.9.2a and b. The PI controllers have been tuned to obtain the optimum plant performance during normal production.

The loops are formed and reduced in the manner described in the previous section. Whenever, the steel bar enters another stand the control system switches in the required control structure. The structure is then used by the controller to decouple the plant interaction by feeding forward the correct amounts to the other plant inputs, whenever a one of the plant input's error signal is different from zero.

When comparing these results to the performance under single-variable control, shown in figure 6.9.1a and b, the advantage of using this decoupling system are apparent.

Since the speed of the eighth drive is not changed during normal production, the responses of the speed of the eighth stand and the fourth loop are the same under both control methods.

The response of the other loops is seen to be significantly improved when using a switching decoupling system. These loop outputs are within 5 % of the setpoint in less than half the time taken before, and the oscillations round this point are also much reduced. The last three loops, see figure 6.9.2b, are completely decoupled from the rest of the plant. However, the most noticeable improvement in performance is in the control of the first loop. Under PI control only, it is seen to be highly oscillatory and does not reach steady state before the loop setpoint is stepped down. With the switching decoupling system, the loop output response is comparable to that of the other loops. It remains within the 5 % band as quickly as the other loops (0.9 seconds) and reaches steady state after 3.5 seconds.

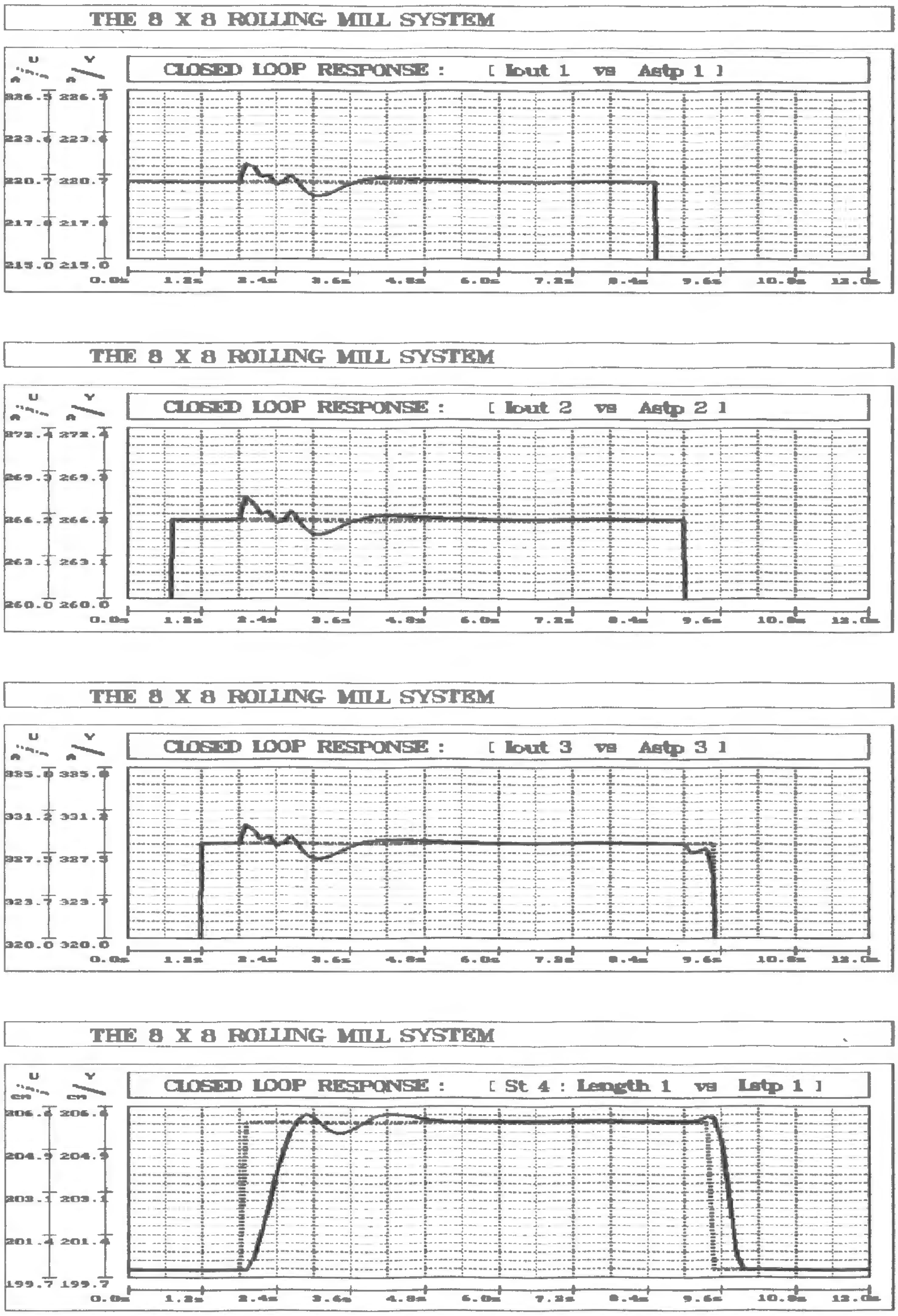


Figure 6.9.2a: The first four stands under MV control.

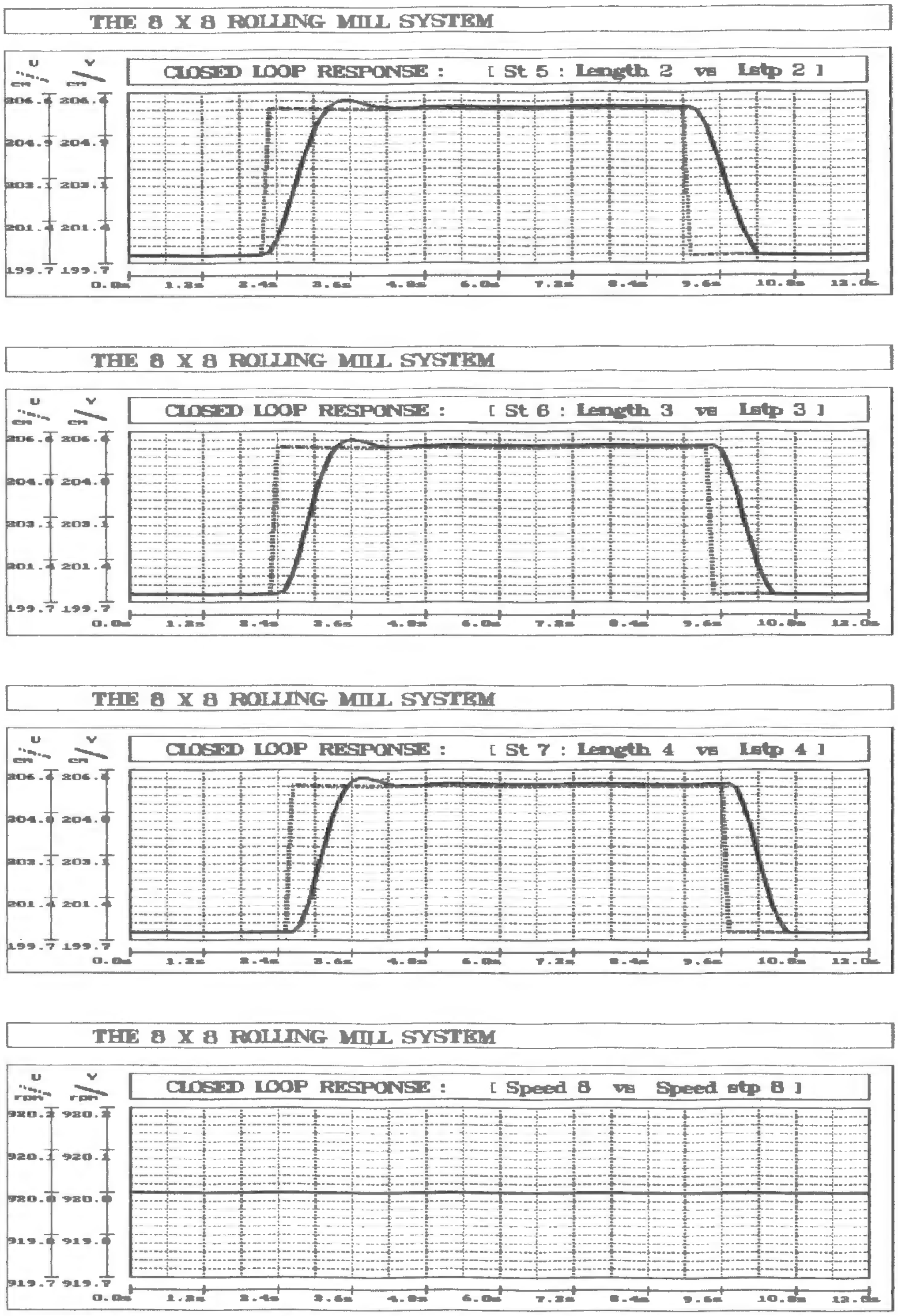


Figure 6.9.2b: The last four stands under MV control.

The tension interaction in the first three drives has also been significantly improved. Not only has the percentage variation in the tension outputs been reduced but the time for these oscillations to decay is much less (for the exact figures of comparison between the two control systems see section 6.3).

6.2.2.(b) The effects of tension setpoint changes.

The effects on the performance due to step changes in the tension setpoints of the plant are shown in figures 6.10.2(a),(b) and (c).

Comparing these graphs to their equivalents for single-variable control in figures 6.10.1(a),(b) and (c), the improvement in the performance is seen to be very large. As calculated in the next section, the maximum initial variation of the fourth loop output from it's setpoint is reduced from 13.3% to 3.2% by implementing this control strategy. (The second variation occurs after the bar's exit from the first drive, where the interaction is reduced from 12.8% to 7.4%).

Other improvements in the performance that are immediately apparent are the reductions in both the percentage variation and duration of the interaction in the first three drives and the fact that the first loop no longer whips. Only the first four outputs are shown in both these sets of graphs since, as the plant interaction matrices derived in Chapter 4 show, the other plant outputs are not affected by the tension inputs.

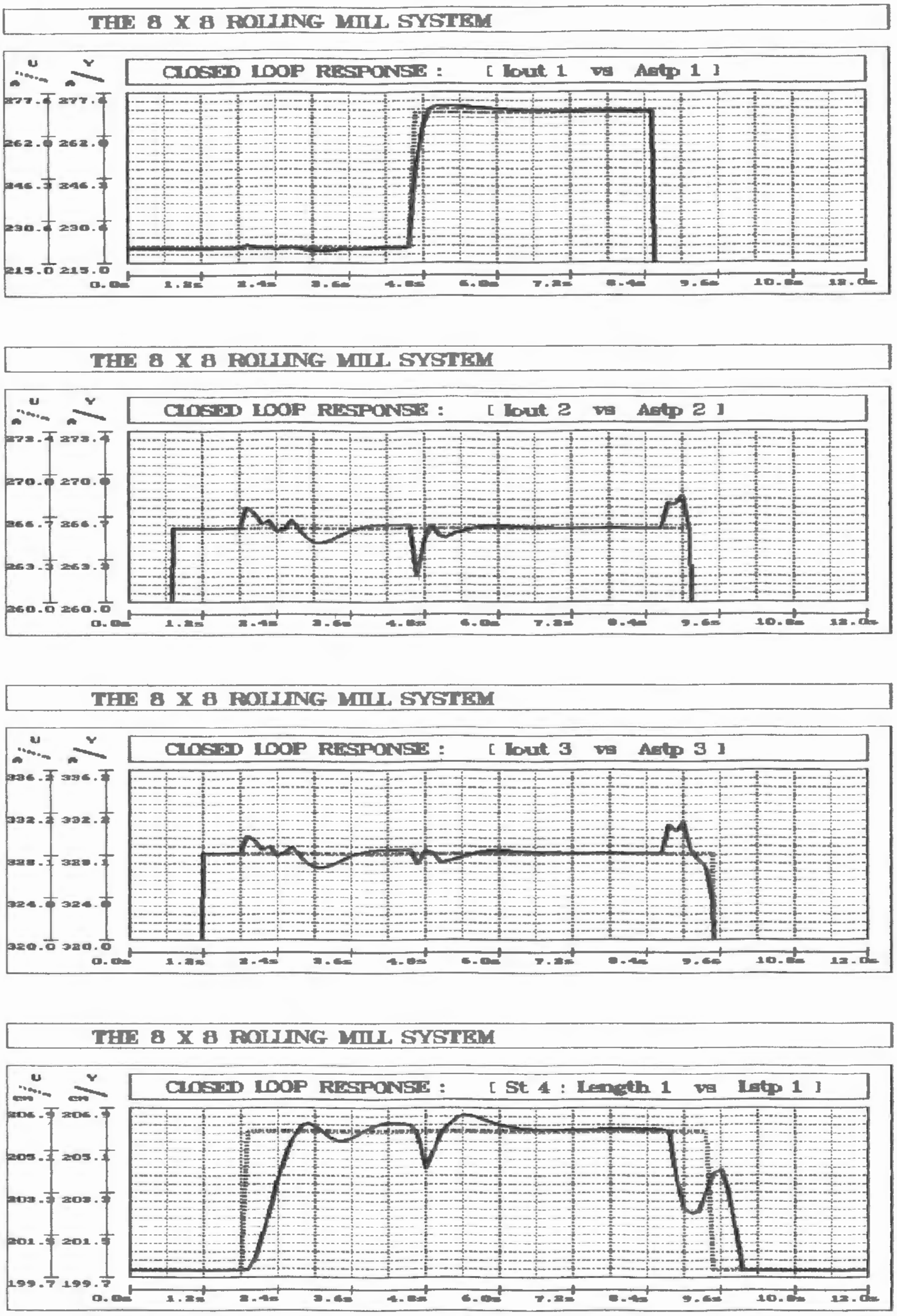


Figure 6.10.2a: The effects of a step in the first tension setpoint.

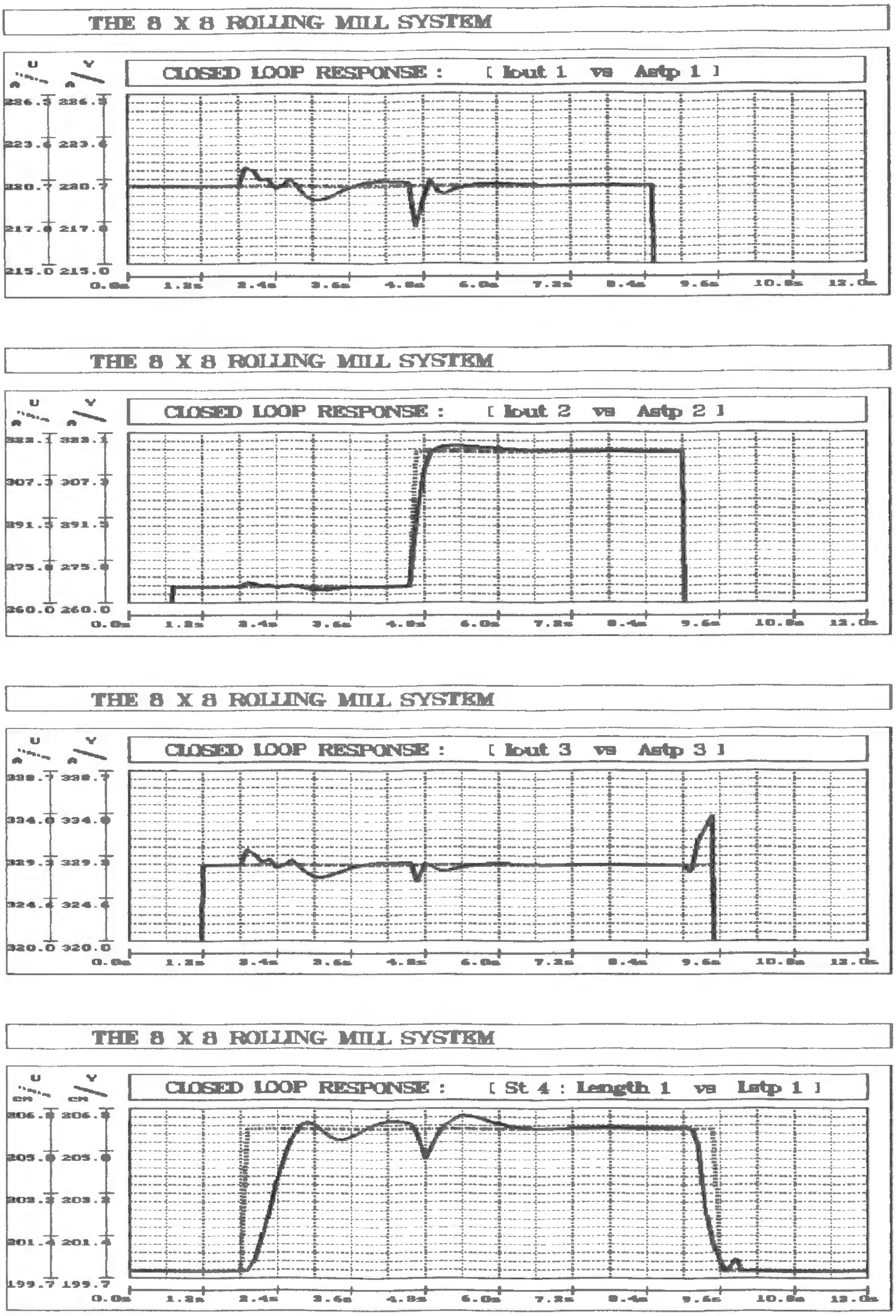


Figure 6.10.2b: The effects of a step in the second tension setpoint.

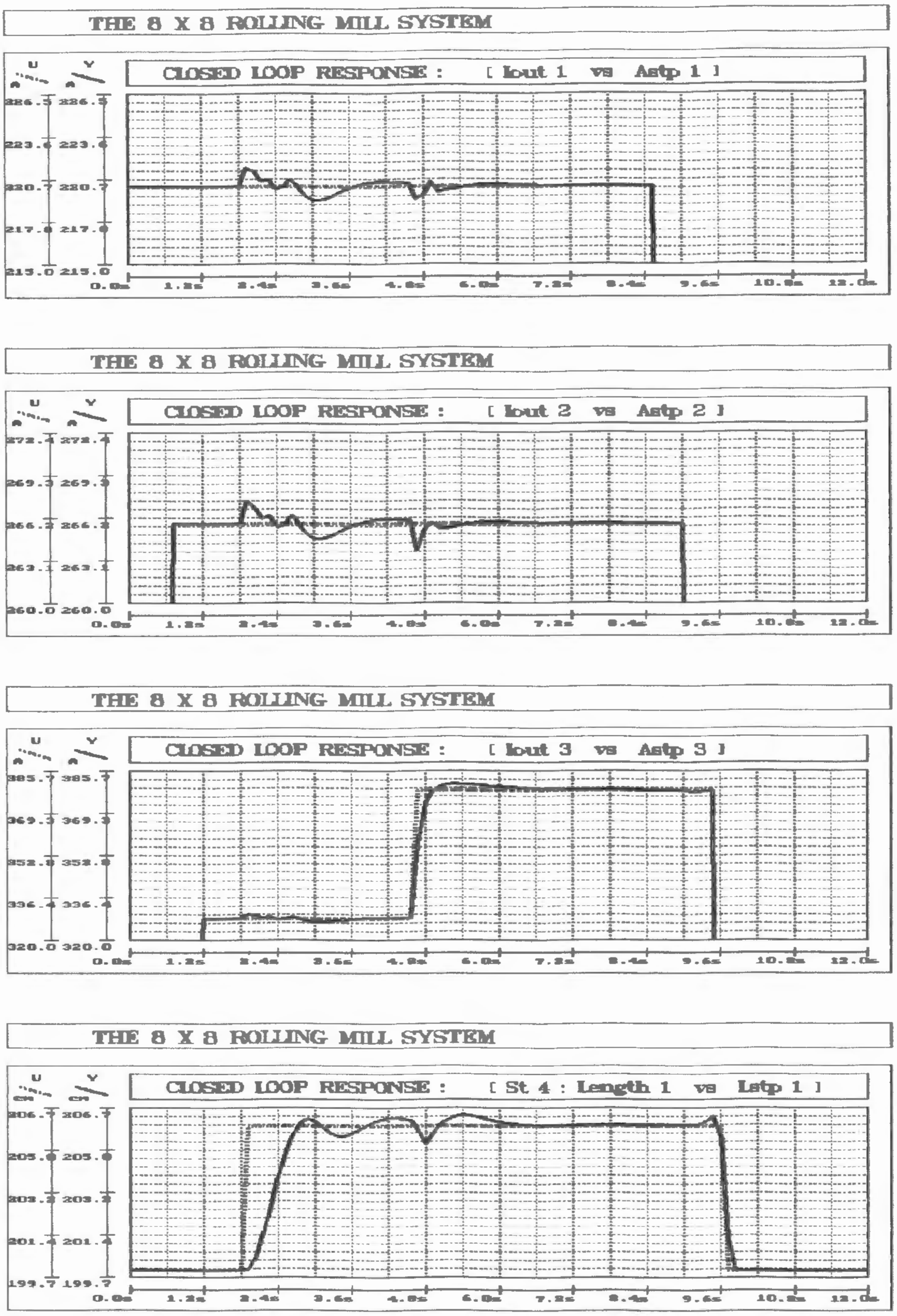


Figure 6.10.2c: The effects of a step in the third tension setpoint.

6.2.2.(c) The effects of a production speed change.

Figure 6.11.2 shows all eight output responses when the speed of the production line is stepped down. All the output responses are significantly improved from when a single-variable control system is implemented.

This is shown graphically by comparing figures 6.11.1 and 6.11.2 and numerically in table 6.3 in the next section.

Although this change in production speed causes errors between the last three loops and their setpoints for the remainder of the production cycle, these loops are stable and the quality of the product is maintained. There is no whipping or pulling of the metal in the loops and there is not significantly more tension interaction in the first four stands.

Figure 6.11.2(a) shows that the first loop is not significantly affected by this step change and that it reaches steady state again after 0.9 seconds.

The importance of implementing the multivariable switching control system designed in section 6.2.1, is illustrated by the significant improvement in the closed loop control under all situations, but especially when the speed of the eighth drive is changed. The effects on the control system performance due to a change in the model of the metal are discussed in the next section.

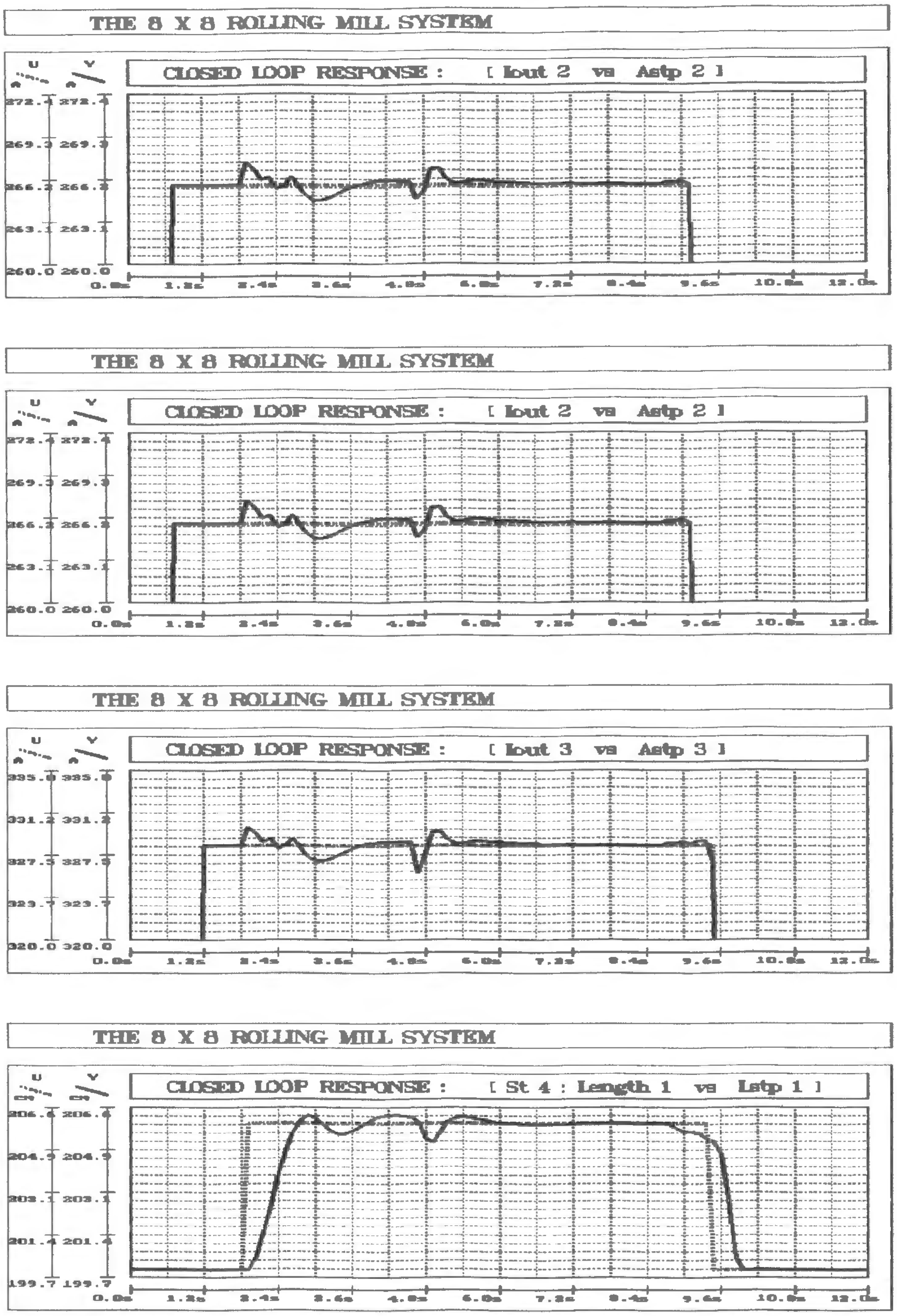


Figure 6.11.2a: The effects on the first four outputs of stepping the output speed of the production line.

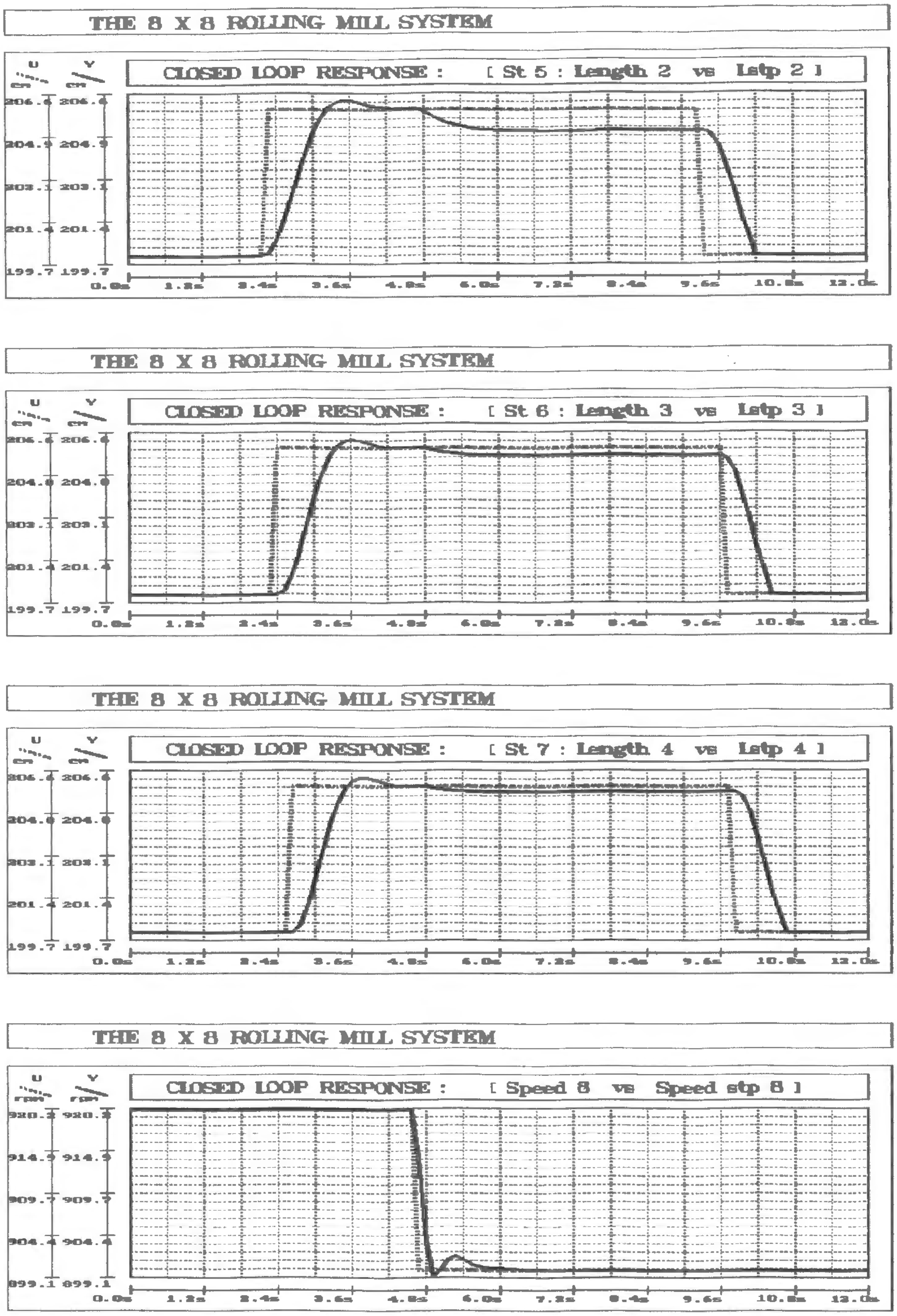


Figure 6.11.2b: The effects on the last four outputs of stepping the output speed of the production line.

6.2.2.(d) Performance effects due to a model change.

The value of Young's Modulus measures the force required to extend the length of a metal bar. A higher value indicates that it requires more force to extend the bar.

Variations in Young's modulus could be due to changes in the constituents of the metal or variations in the rolling temperature of the steel bar.

Therefore, as is explained in the single-variable section 6.1.2(d), when this value is increased, the tension interaction increases.

In order to test the robustness of this control system to changes in the metal's Young's Modulus, the value is first increased by 100 % and the performance of the control system investigated. Then the nominal value of 400 MPa used in the simulator, is reduced by 75 % to 100 MPa.

The results of increasing the value are shown in figure 6.12.2a. Although the response of the first loop is now more sluggish and it's overshoot is larger than before, the system performance is not significantly degraded by this large change.

Figure 6.12.2b shows the graphs of the four output responses affected by tension interaction, when Young's Modulus is reduced to 100 MPa. Again, the control system performance is degraded. The control of the process is still acceptable, though, and the quality of the product is not degraded by whipping or excessive tension forces.

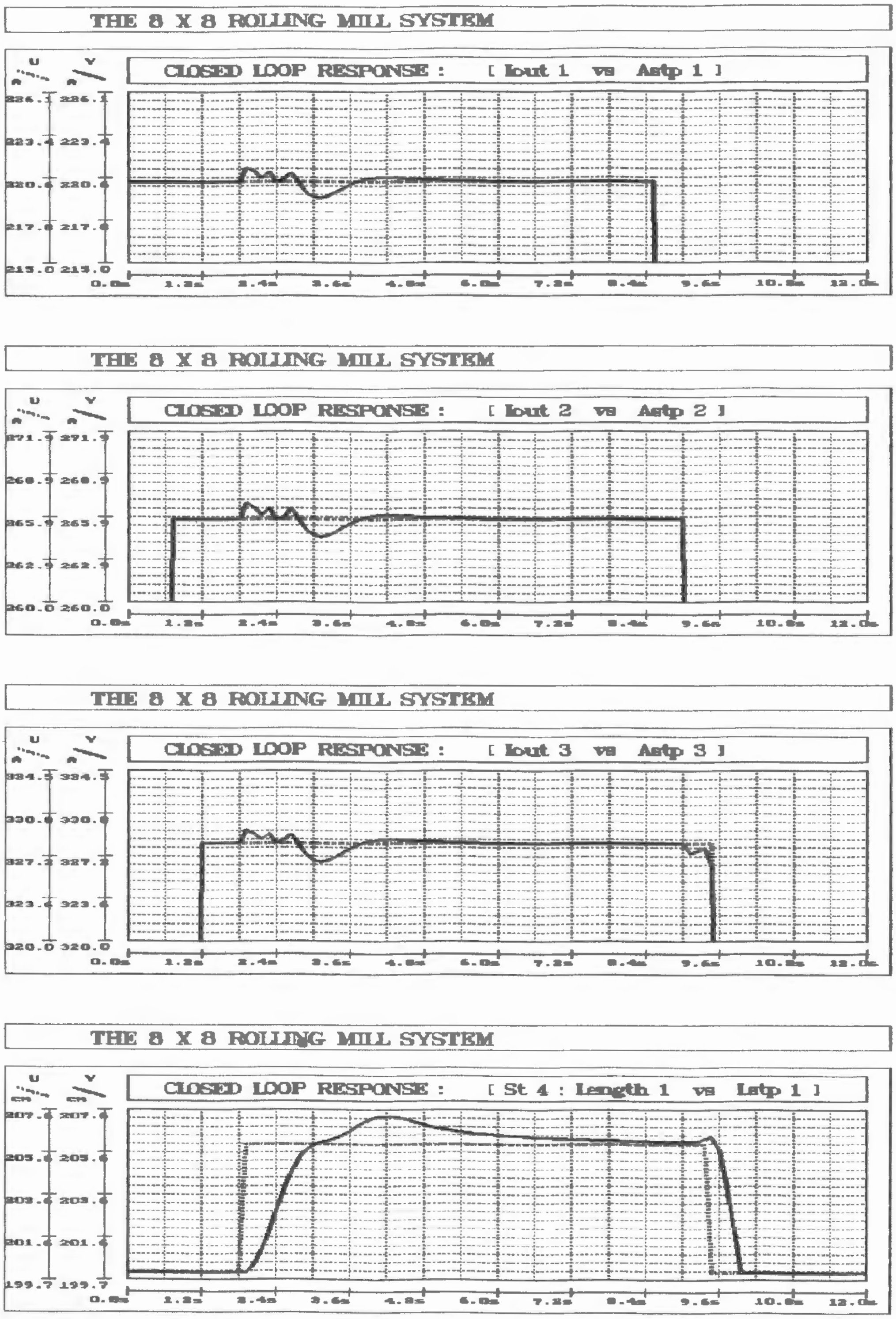


Figure 6.12.2a: The effects of an increase in the value of Young's Modulus.

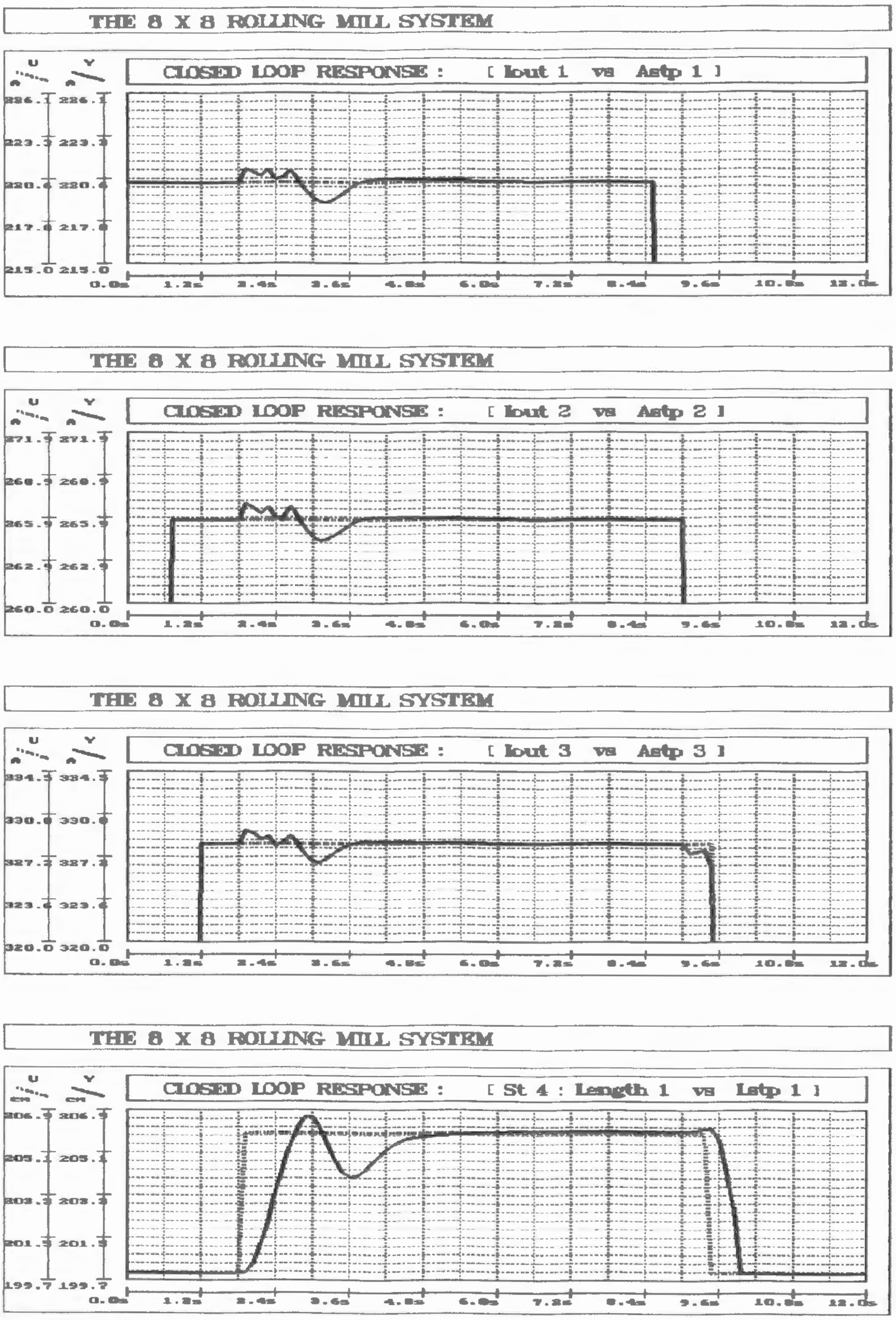


Figure 6.12.2b: The effects of decreasing the value of Young's Modulus.

Large changes in the metal model do, therefore degrade the performance of the control system, but the controller performance is still acceptable with changes of 100 %.

The significant improvements in closed loop plant performance (when a multivariable switching control strategy is adopted) shown graphically in this section, are numerically analysed and tabled in the next section.

6.3. Performance Comparison of the Control Systems.

The results discussed in sections 6.1.2 and 6.2.2 for the different control systems are analysed and tabulated in this section. The major improvements in the closed loop performance when the multivariable controller is implemented, are discussed in each section.

The normal plant operation is covered in 6.3.1. Then, the plant performance when the tension and speed setpoints are stepped is covered in sections 6.3.2 and 6.3.3. The effect on each controller's performance due to a change in the metal model is then discussed in 6.3.4 and, finally, the disturbance rejection of both closed loop systems is compared in section 6.3.5.

The interaction for each output is calculated as it's percentage variation from it's setpoint divided by the percentage step in the setpoint that causes the interaction.(This is then converted to a percentage). In other words if an output doubles in value when another setpoint is stepped to twice it's value, then this is considered to be 100 % interaction.

The *Time to 5%* quoted in these tables is the time the output takes to settle to within 5% of it's setpoint value after a setpoint change.

6.3.1 Normal Plant Operation.

Table 6.1 shows the percentage interaction or the overshoot of all the outputs and the time each output takes to settle to within 5 % of the setpoint, during normal plant production.

The tension control is seen to be significantly better when using the switching multivariable control system. Not only is there about a 10.5% improvement in all the levels of interaction of the tension outputs but the output is made to track the setpoint in 40% of the time.

		PI Control	MV Control
Tension 1	Interaction [%]	26.5	16.5
	Time to 5 % [s]	4.3	1.7
Tension 2	Interaction [%]	25.9	13.4
	Time to 5 % [s]	4.4	1.7
Tension 3	Interaction [%]	26.4	15.8
	Time to 5 % [s]	4.4	1.8
Loop 1	Overshoot [%]	16.7	5.0
	Time to 5 % [s]	6.6	0,9
Loop 2	Overshoot [%]	3.3	5.0
	Time to 5 % [s]	2.0	0.9
Loop 3	Overshoot [%]	3.3	5.0
	Time to 5 % [s]	1.8	0.9
Loop 4	Overshoot [%]	5.0	5.0
	Time to 5 % [s]	0.9	0.9
Speed 1	Interaction [%]	0.0	0.0
	Time to 5 % [s]	0.0	0.0

Table 6.1: Performance comparison during normal production.

As discussed in section 6.2.2, the first three loop outputs are also controlled far better by the MV control strategy. This is especially true of the first loop where the overshoot is reduced from 16.7%

to 5%, and the time to settle to 5% has been improved from 6.6 seconds to 0.9 seconds. Since the speed of the eighth drive is not normally stepped during production or affected by any of the other plant setpoints it remains constant under both control systems.

6.3.2. The Effects of a Tension Setpoint Step.

Tables 6.2.1, 6.2.2 and 6.2.3 show the performance results of the four outputs (affected by a change in tension) when the first, second and third tension setpoints are stepped, respectively.

		PI Control	MV Control
Tension 1	Overshoot [%]	8.4	4.6
	Time to 5 % [s]	2.3	0.3
Tension 2	Interaction [%]	6.9	5.9
	Time to 5 % [s]	--	0.3
Tension 3	Interaction [%]	6.3	3.7
	Time to 5 % [s]	--	0.3
Loop 1	Interaction [%]	13.3	3.2
	Time to 5 % [s]	--	1.3

Table 6.2.1: Table of results when tension setpoint #1 is stepped.

In all three tables the percentage interaction, the overshoot and the settling time have been improved using multivariable control. Table 6.2.1 shows that using single-variable control, the closed loop responses take much longer to settle. The interaction quoted for loop #1 in this table is for the initial interaction affect. The second effect is caused by the bar exiting from the first stand. The interaction is then reduced to 7.4 % from 12.8 % by the switching control system.

		PI Control	MV Control
Tension 1	Interaction [%]	6.1	6.2
	Time to 5 % [s]	1.9	0.2
Tension 2	Overshoot [%]	9.6	4.6
	Time to 5 % [s]	2.2	0.3
Tension 3	Interaction [%]	9.7	8.4
	Time to 5 % [s]	--	--
Loop 1	Interaction [%]	16.0	3.1
	Time to 5 % [s]	--	1.2

Table 6.2.2: Table of results when tension setpoint #2 is stepped.

The dashed entries in these charts indicate that the bar left the stand (used to control this output) before the output had settled. Tables 6.2.2 and

6.2.3 show the similar improvement in closed loop performance when the multivariable control system is implemented to control a step in the other two tension setpoints.

		PI Control	MV Control
Tension 1	Interaction [%]	7.2	2.4
	Time to 5 % [s]	1.9	0.0
Tension 2	Interaction [%]	7.0	4.8
	Time to 5 % [s]	1.9	0.0
Tension 3	Overshoot [%]	11.4	4.8
	Time to 5 % [s]	2.2	0.3
Loop 1	Interaction [%]	19.5	2.1
	Time to 5 % [s]	2.9	0.0

Table 6.2.3: Table of results when tension setpoint #3 is stepped.

6.3.3. The Effects of a Production Speed Change.

Table 6.3 is a comparison of the closed loop performances of the two control systems when the speed of the eighth stand is decreased. This table shows the extremely poor performance of the single-variable control system when trying to control the four loop outputs.

		PI Control	MV Control
Tension 1	Interaction [%]	44.5	24.5
	Time to 5 % [s]	--	1.4
Tension 2	Interaction [%]	45.4	20.1
	Time to 5 % [s]	4.4	1.4
Tension 3	Interaction [%]	46.0	28.6
	Time to 5 % [s]	4.3	1.4
Loop 1	Interaction [%]	141.2	15.8
	Exit Error [%]	49.5	0.0
Loop 2	Interaction [%]	94.9	19.7
	Exit Error [%]	87.1	18.4
Loop 3	Interaction [%]	203.0	6.6
	Exit Error [%]	195.4	6.6
Loop 4	Interaction [%]	220.7	5.3
	Exit Error [%]	208.4	5.3
Speed 1	Overshoot [%]	4.5	4.5
	Time to 5 % [s]	0.9	0.9

Table 6.3: Performance comparison after a production speed change.

Although the multivariable control system does not achieve a perfect control of the plant, it does keep the effects of this step change to an acceptable level. The exit errors describe the output error when the loop setpoints are decreased.

6.3.4. Performance Effects of a Metal Model Change.

The effects of changing the value of the Young's Modulus value used in the simulator on the closed loop performances of the two control systems is shown in tables 6.4.1 and 6.4.2.

		PI Control	MV Control
Tension 1	Interaction [%]	7.2	2.4
	Time to 5 % [s]	1.9	0.0
Tension 2	Interaction [%]	7.0	4.8
	Time to 5 % [s]	1.9	0.0
Tension 3	Overshoot [%]	11.4	4.8
	Time to 5 % [s]	2.2	0.3
Loop 1	Interaction [%]	19.5	2.1
	Time to 5 % [s]	2.9	0.0

Table 6.4.1: The effects of increasing Young's Modulus by 100 %.

These tables show that both of the control systems do not control the plant as well as when the model is what they are designed for. The multivariable switching control system still performs better in all aspects of the control in closed loop.

Therefore, a large change in the metal's properties does not render the control systems unstable nor make the multivariable system perform worse than the simple PI system.

		PI Control	MV Control
Tension 1	Interaction [%]	7.2	2.4
	Time to 5 % [s]	1.9	0.0
Tension 2	Interaction [%]	7.0	4.8
	Time to 5 % [s]	1.9	0.0
Tension 3	Overshoot [%]	11.4	4.8
	Time to 5 % [s]	2.2	0.3
Loop 1	Interaction [%]	19.5	2.1
	Time to 5 % [s]	2.9	0.0

Table 6.4.2: The effects of decreasing Young's Modulus by 75 %.

6.3.5. The Disturbance Rejection Performance.

The singular value plots of the sensitivity function matrix S(s) are used to assess the disturbance rejection of the closed loop systems. S(s) should tend to a matrix of zeros at low frequencies for good disturbance rejection[3].

On the other hand, the singular value plots of the complementary sensitivity function T(s) are an indication of the measurement noise attenuation at high frequencies and of setpoint tracking at low frequencies[3]. Therefore T(s) must tend to the identity matrix at low frequencies and to a matrix of zeros at high frequencies.

The singular value plots of S(s) and T(s) are plotted, for both closed loop control systems designed in this thesis, in figures 6.26 and 6.27.

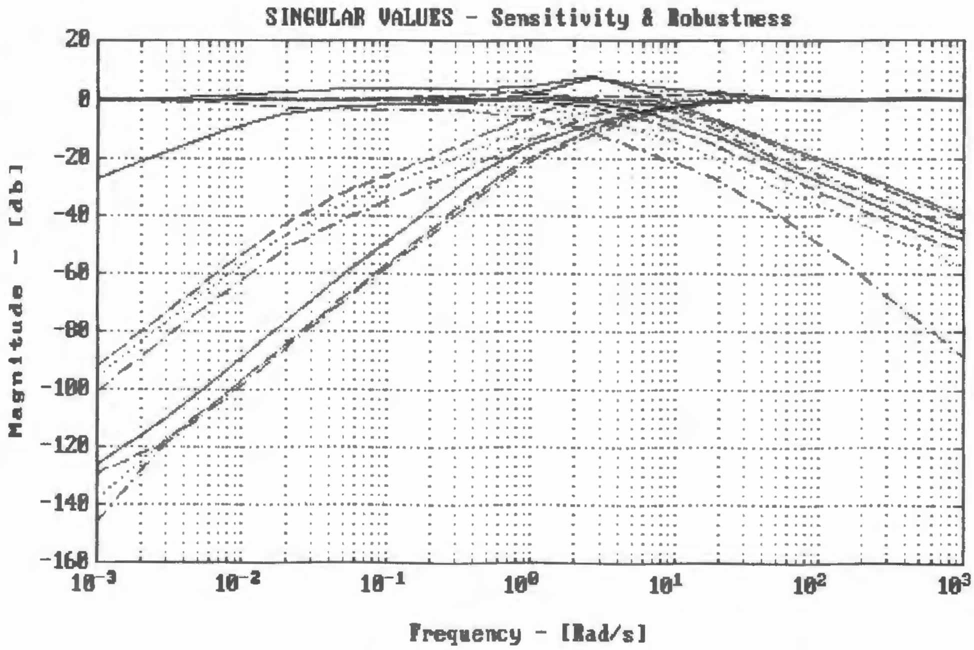


Figure 6.26: S(s) and T(s) for the single-variable control system.

As these two graphs show, the multivariable control system reduces the maximum singular value of the sensitivity function by 20 dB at low frequencies. This shows that the disturbance rejection at low frequencies is significantly better when using the multivariable decoupler.

The multivariable controller also improves the measurement noise attenuation of the closed loop plant. This is illustrated by the fact that T(s) is smaller at high frequencies.

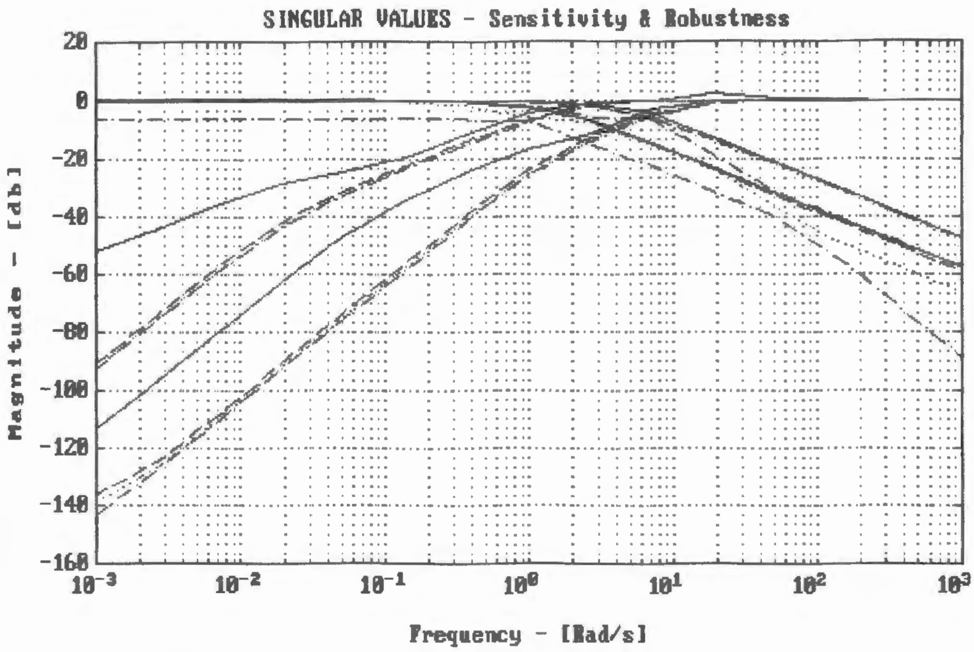


Figure 6.27: S(s) and T(s) for the multivariable control system.

As is illustrated in the graphs obtained from the simulator in section 6.2.2, the setpoint tracking (T(s) at low frequencies) of the multivariable controller is slightly degraded. This occurs because the multivariable PI controller matrix (see figure 6.1) has been tuned to optimise normal plant production. Thus, when the speed of the production line is stepped, the multivariable controller reduces the interaction effect well but is slow in returning the loop heights to the setpoint. The minimum singular value of T(s) at low frequencies, in figure 6.27 indicates that this should happen.

The singular value plots shown in this sub-section indicate the overall performance improvement of the multivariable control method.

Using the transfer functions derived in Chapter 5, the single-variable PI and the switching multivariable control systems are designed. These control systems are then implemented on the rolling mill simulator, in the manner indicated by figure 6.1. The graphic results obtained from the implementation of the different control systems in sections 6.1.2 and 6.2.2. are analysed and tabulated in the previous section. The tables indicate that the closed loop performance of the production line is better in all aspects, when the switching decoupler control system is used.

The singular value plots of the sensitivity function and it's complementary function confirm the responses obtained when the different control systems are implemented.

Conclusions as to the aims of the control system and which is the best control strategy to adopt for this particular problem are drawn in the following chapter.

References.

1. Fisher, I.P.: *Multivariable Control of a Flotation Plant Simulator*, MSc Thesis, U.C.T, 1988.
2. Venzke, R.H.E.: *Comparison of INA Technique to the Pole Assignment Technique*, MSc Thesis, U.C.T, 1988.
3. Maciejowski, J.M.: *Multivariable Feedback Design*, Addison -Wesley Publishing, Great Britain, 1989.

CHAPTER 7: CONCLUSIONS

The control of the Hille Mill production line A was studied in this thesis. The type of industrial product was determined and the main control aims were ascertained.

The rolling mill was studied and the input and output variables were established. The open loop plant structure was then derived using this information.

Since, it was not economically viable to conduct numerous tests on the actual plant, or to implement the control systems designed in practice, a simulation package of the rolling process was written.

The plant interaction was then studied to determine the control structure required. All the transfer functions for the plant were then derived from the simulator and these were used to design the single-variable and multivariable control systems.

The closed loop performances of the rolling mill simulator when the control systems were implemented, was recorded, analysed and discussed.

From the results obtained in this dissertation the following conclusions can be drawn.

Production Process.

The steel billet exits the furnace at a temperature of approximately 1200°C. This bar is then coarsely rolled down by a number of stands before it passes to the beginning of either production line A or B. The control of production line A was considered in this study.

The steel bar enters the eight stands that make up line A, one by one. The metal is formed into four loops between the last five stands. This is to prevent the metal from being stretched or compressed at this stage, where it is thin.

While the steel bar is still thick (between the first four stands) the tension in the metal is controlled by using the first three drives. The current consumption in the first three drives is used to measure this tension.

The speed of the production line is determined by the speed of the last drive.

Production Aims.

The main aims of the control system were determined to be to control the quality of the final product and the output rate of the production line.

Input and Output Variables.

The output variables of the plant that are used to measure the state (and, therefore, the quality) of the metal are the three current outputs and the four loop height outputs.

The speed of the eighth drive is the output variable indicating the speed of the production line.

These outputs are controlled by using the eight speed inputs to the stands.

Rolling Mill Simulator.

The eight drives were mathematically modelled using basic machine theory. The constants used in the derivation were obtained from the drives used in the Hille Mill.

The effects of tensional forces on the state of the metal was simulated using the stress/strain relation of materials. This stress/strain relation is described by a constant known as Young's Modulus.

The value of Young's Modulus depends on the constituents of the steel, the temperature and the speed at which the metal is worked. Although, the exact figure for the type of steel rolled at Scaw Metals was not available, the Young's Modulus of a similar steel was found to be 400 MPa at 1000°C. The value of this constant was found not to be critical to the control systems' performance.

The tension in a section of steel was found to be related to the difference in the speeds of the drives on either side of it.

The formation of the metal loops between the drives was simulated to take the approximate shape of an isosceles triangle. The relation between the height to the difference in the speeds of the two drives on either side of it was found to be non-linear. This was linearised by converting the height output to an equivalent length output.

Plant Interaction.

The plant interaction depends on the position of the bar in the production line. All the different plant interaction matrices derived during the rolling of a steel bar make up the rolling process cycle.

The Control Structures.

The analysis of the different plant interaction matrices which make up the process cycle, shows that different control structures are required by a multivariable control system during the rolling of a steel billet.

The multivariable control system therefore switches between the required control structures at the correct times in the process.

Plant Transfer Functions.

The transfer functions describing the loop height transient responses do not change during the rolling process. This is because there is no multivariable interaction between the loop outputs.

However, the equations describing the tension outputs of the metal do change, depending on the position of the bar. This is because all the stands between which there is tension control interact with each other. Therefore, the transfer functions describing these outputs depend on the number of stands interacting in this manner.

The transfer functions matrices were derived for all the plant interaction situations, from step tests conducted on the simulator. The ramp test data obtained from the plant was inadequate for this purpose.

Control System Design and Implementation.

The eight single-variable PI controllers used in the first control system were designed using the Characteristic Loci method. This was done for the dominant plant situation, where the steel is in all eight stands.

All the control structures required in the multivariable system were designed using the transfer functions from the simulator. This was done using the Nyquist Array design methods. The multivariable control system then switched between these structures during the process cycle.

The closed loop plant performance of the switching multivariable control system was found to be superior in all aspects. These aspects were normal plant operation, production speed changes, tension setpoint changes, metal model effects and disturbance rejection of the closed loop plant.

Therefore, from the results obtained in this thesis, it can be concluded that the best control strategy to adopt for this rolling mill process is a multivariable control system which switches between different control structures depending on the position of the bar in the production line.

BIBLIOGRAPHY

American Society of Metals (1978), *Metals Handbook: Vol. 1: Properties and Selections: Irons and Steels.*, Ninth Edition, ASM printers.

American Society of Metals (1978), *Metals Handbook: Vol. 3: Properties and Selections: Stainless Steel, Tool Materials and Special-Purpose Metals* , ASM printers.

Avitzur, B. (1968), *Metal Forming: Processes and Analysis.*, McGraw-Hill, New York.

Brown Boveri Corporation. Operating and Maintenance Manual for Bar and Section Mill - Scaw Metals, contract DV 11000, 1980.

Davies, R. and Austin, E.R. (1970), *Developments in High Speed Metal Forming.*, The Machinery Publishing Co., Sussex.

Fisher, I.P.: *Multivariable Control of a Flotation Plant Simulator*, MSc Thesis, U.C.T, 1988.

Fitzgerald, A.E., Kingsley, C.Jr. and Umans, S.D. (1985), *Electric Machinery.*, 4th ed., McGraw-Hill, Singapore.

Gear, A.B.J.: *The Design of Decentralized Controllers for Large Scale Systems*, MSc Thesis, U.C.T., 1988.

Gear, A.B.J. and Braae, M.: *Structure of Large Scale Systems,* SACAC 3rd Workshop on Multivariabel Control, Mintek, Randburg, 1986.

GEC Machines Ltd., Warwickshire, England, *Technical Information for the D.C. Mill Drive Motors of the Rod Mill at Lancashire Steel (Irlam) Ltd.*, Ref: LMSD/PJS/J52393, May 1979.

Halliday, D. and Resnick, R. (1988), *Fundamentals of Physics.*, 3rd ed., Wiley, New York.

Harris, J.N. (1983), *Mechanical Working of Metals: Theory and Practice.*, Pergamon Press, Oxford.

Hill, R.J. (1990), *Mathematical Modelling in Undergraduate D.C. Machine Drive Projects*, Int. J. Elect. Enging. Educ., Vol. 27, pp 197 - 217.

Hill, R.J. and Lou, F.L. (1989), *Microprocessor-Based Control of Steel Rolling Mill Digital DC Drives.*, IEEE Trans. Power Electronics, Vol. 4, No. 2, pp 289 - 297.

Hindmarsh, J. (1970), *Electrical Machines and their Applications.*, Pergamon Press, Oxford.

Honeycombe, R.W.K. (1984), *The Plastic Deformation of Metals.*, 2nd ed., Edward Arnold, London.

Maciejowski, J.M.: *Multivariable Feedback Design*, Addison -Wesley Publishing, Great Britain, 1989.

Rosenbrock, H.H.: *State Space and Multivariable Theory,* Nelson, London, 1970.

Rowe, G.W. (1977), *Principles of Industrial Metalworking Processes.*, Edward Arnold, London.

Say, M.G. and Taylor, E.O. (1980), *Direct Current Machines.*, Pitman Publishing, London.

Steven, R.E. (1983), *Electrical Machines and Power Electronics.*, Van Nostrand Reinhold, England.

Venzke, R.H.E.: *Comparison of INA Technique to the Pole Assignment Technique*, MSc Thesis, U.C.T, 1988.

APPENDIX A: THE STEEL TEMPERATURE DURING ROLLING

The temperature of the metal at different stages in the rolling process is illustrated in figure A.1. These temperatures were measured on production line A in the Hille Mill. The tests were conducted by the Test Department of Scaw Metals[1].

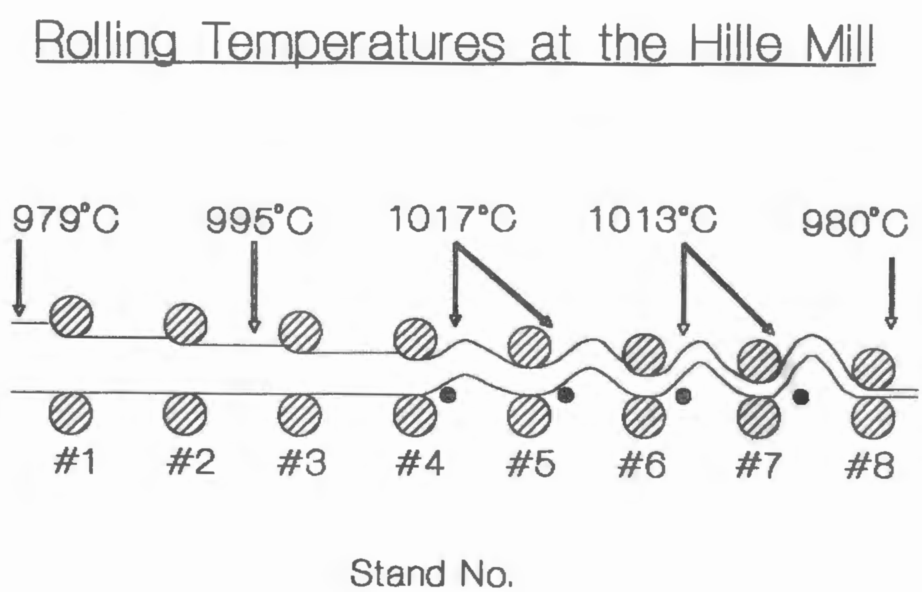


Figure A.1: The rolling temperatures.

The average temperature of the metal in this production line is approximately 1000°C.

References.

1. Welgemoed, K. *Private discussion*, Electrical Engineer, Hille Mill, Scaw Metals Ltd., Germiston, Transvaal.

APPENDIX B: THE D.C. MOTOR DYNAMICAL EQUATION

The D.C. motor dynamical equation is derived in the following manner. Equations B.1-B.4[1-6] govern the machine's behaviour. They describe the effects of the armature current, the back EMF, the drag torque and the acceleration torque.

$$E_a + L_a * \frac{di_a}{dt} + R_a * i_a = v_a \qquad \text{...(B.1)}$$

$$E_a = C_e * \Phi * n \qquad \text{...(B.2)}$$

In equation B.2, C_e is the back EMF constant and Φ is the total field flux.

$$M_d = M_l + M_a = C_m * \Phi * i_a \qquad \text{...(B.3)}$$

In the above equation M_d is the drag torque, M_l the load torque, M_a the acceleration torque and C_m is the load constant. Friction is included in the load torque.

$$M_a = J * \frac{dn}{dt} \qquad \text{...(B.4)}$$

J is the machine mechanical moment of inertia.

Rearranging equations B.1 and B.3 in terms of i_a

$$i_a = \frac{v_a}{R_a} - \frac{L_a}{R_a} * \frac{di_a}{dt} - \frac{E_a}{R_a} \qquad \text{...(B.5)}$$

$$i_a = \frac{M_l + M_a}{C_m * \Phi} \qquad ...(B.6)$$

Equating the two expressions B.5 and B.6.

$$\frac{M_l}{C_m\Phi} + \frac{M_a}{C_m\Phi} = \frac{v_a}{R_a} - \frac{L_a}{R_a} * \frac{di_a}{dt} - \frac{E_a}{R_a} \qquad ...(B.7)$$

Substituting in for M_a from equation B.4 and for E_a from equation B.2 the following equation is obtained.

$$\frac{M_l}{C_m\Phi} + \frac{J}{C_m\Phi} * \frac{dn}{dt} = \frac{v_a}{R_a} - \frac{L_a}{R_a} * \frac{di_a}{dt} - \frac{C_e\Phi}{R_a} * n$$

$$...(B.8)$$

Replacing M_a in equation B.6 with equation B.4, the following equation results.

$$i_a = \frac{M_l}{C_m\Phi} + \frac{J}{C_m\Phi} * \frac{dn}{dt} \qquad ...(B.9)$$

Substituting for i_a in equation B.8 and rearranging produces equation B.10:

$$\frac{J}{C_m\Phi} * \frac{dn}{dt} + \frac{C_e\Phi}{R_a} * n = \frac{v_a}{R_a} - \frac{M_l}{C_m\Phi} - \frac{L_a}{R_a C_m\Phi} * \frac{dM_l}{dt} - \frac{L_a J}{R_a C_m\Phi} * \frac{d^2n}{dt^2}$$

$$...(B.10)$$

Rewriting equation B.10 into a differential equation in terms of n, the result is the motor dynamic equation.

$$\frac{JL_a}{C_m\Phi R_a}*\frac{d^2n}{dt^2} + \frac{J}{C_m\Phi}*\frac{dn}{dt} + \frac{C_e\Phi}{R_a}*n = \frac{v_a}{R_a} - \frac{1}{C_m\Phi}\left[M_1 + \frac{L_a}{R_a}\frac{dM_1}{dt}\right]$$

Substituting the following constants into this equation, the final form of the motor dynamic equation is obtained in equation B.11.

Mechanical time constant:

$$T_m = \frac{J*R_a}{C_e\Phi C_m\Phi}$$

Electrical time constant:

$$T_a = \frac{L_a}{R_a}$$

Machine constants:

$$k = \frac{1}{C_e\Phi}$$

$$k' = \frac{R_a}{C_e\Phi C_m\Phi}$$

$$T_mT_a*\frac{d^2n}{dt^2} + T_m*\frac{dn}{dt} + n = k*v_a - k'*\left[M_1 + T_a*\frac{dM_1}{dt}\right] \quad ...(B.11)$$

References.

1. Say, M.G. and Taylor, E.O. (1980), *Direct Current Machines.*, Pitman Publishing, London.
2. Steven, R.E. (1983), *Electrical Machines and Power Electronics.*, Van Nostrand Reinhold, England.
3. Hindmarsh, J. (1970), *Electrical Machines and their Applications.*, Pergamon Press, Oxford.
4. Fitzgerald, A.E., Kingsley, C.Jr. and Umans, S.D. (1985) , *Electric Machinery.*, 4th ed., McGraw-Hill, Singapore.
5. Hill, R.J. and Lou, F.L. (1989), *Microprocessor-Based Control of Steel Rolling Mill Digital DC Drives.* , IEEE Trans. Power Electronics, Vol. 4, No. 2, pp 289 - 297.
6. Hill, R.J. (1990), *Mathematical Modelling in Undergraduate D.C. Machine Drive Projects.*, Int. J. Elect. Enging. Educ., Vol. 27, pp 197 - 217.

APPENDIX C: THE MOTOR TORQUE-CURRENT EQUATION

The equation describing the electrical current response of the D.C. drive is derived in the following manner. Equations C.1 - C.4[1-6] describe the effects of the armature current, the back EMF, the drag torque and the acceleration torque.

$$E_a + L_a * \frac{di_a}{dt} + R_a * i_a = v_a \qquad \ldots(C.1)$$

$$E_a = C_e * \Phi * n \qquad \ldots(C.2)$$

In equation C.2, C_e is the back EMF constant and Φ is the total field flux.

$$M_d = M_l + M_a = C_m * \Phi * i_a \qquad \ldots(C.3)$$

In the above equation M_d is the drag torque, M_l the load torque, M_a the acceleration torque and C_m is the load constant. Friction is included in the load torque.

$$M_a = J * \frac{dn}{dt} \qquad \ldots(C.4)$$

J is the machine mechanical moment of inertia.

Equation C.1 can be expressed in terms of E_a.

$$E_a = v_a - L_a * \frac{di_a}{dt} - R_a * i_a \qquad \ldots(C.5)$$

Substituting for E_a with it's equivalent from equation C.2 and rearranging in terms of n:

$$n = \frac{v_a}{C_e\Phi} - \frac{L_a}{C_e\Phi} * \frac{di_a}{dt} - \frac{R_a}{C_e\Phi} * i_a \qquad ...(C.6)$$

Taking the derivatives of equations C.6 and C.4, expressions are found for the first and second derivative of n.

$$\frac{dn}{dt} = \frac{1}{C_e\Phi} * \frac{dv_a}{dt} - \frac{L_a}{C_e\Phi} * \frac{d^2 i_a}{dt^2} - \frac{R_a}{C_e\Phi} * \frac{di_a}{dt} \qquad ...(C.7)$$

$$\frac{d^2 n}{dt^2} = \frac{C_m\Phi}{J} * \frac{di_a}{dt} - \frac{1}{J} * \frac{dM_l}{dt} \qquad ...(C.8)$$

Substituting equations C.7 and C.8 into the motor dynamical equation C.9 (derived in Appendix B).

$$T_m T_a * \frac{d^2 n}{dt^2} + T_m * \frac{dn}{dt} + n = k * v_a - k' * \left[M_l + T_a * \frac{dM_l}{dt} \right]$$

...(C.9)

After re-ordering C.9 into a differential form in terms of i, the result is the electrical current torque equation. This is represented in equation C.10.

$$T_m T_a * \frac{d^2 i_a}{dt^2} + T_m * \frac{di_a}{dt} + i_a = \frac{T_m}{R_a} * \frac{dv_a}{dt} + \frac{M_l}{C_m \phi} \quad \ldots (C.10)$$

References.

1. Say, M.G. and Taylor, E.O. (1980), *Direct Current Machines.*, Pitman Publishing, London.
2. Steven, R.E. (1983), *Electrical Machines and Power Electronics.*, Van Nostrand Reinhold, England.
3. Hindmarsh, J. (1970), *Electrical Machines and their Applications.*, Pergamon Press, Oxford.
4. Fitzgerald, A.E., Kingsley, C.Jr. and Umans, S.D. (1985) , *Electric Machinery.*, 4th ed., McGraw-Hill, Singapore.
5. Hill, R.J. and Lou, F.L. (1989), *Microprocessor-Based Control of Steel Rolling Mill Digital DC Drives.* , IEEE Trans. Power Electronics, Vol. 4, No. 2, pp 289 - 297.
6. Hill, R.J. (1990), *Mathematical Modelling in Undergraduate D.C. Machine Drive Projects.* , Int. J. Elect. Enging. Educ., Vol. 27, pp 197 - 217.

APPENDIX D: THE D.C. DRIVE CHARACTERISTICS

The characteristics of the D.C. drives used in production line A at the Hille Mill are presented in the following table. They are taken from the data sheets[1] for these two types of motors.

D.C. Drive Characteristics

		Drives 1 & 2	Drives 3 - 8
Speed		900 rpm	900 rpm
Power		373 kW	447 kW
Armature	Voltage	600 V	600 V
	Current	670 A	800 A
	Resistance	0.028 Ohm	0.022 Ohm
	Inductance	0.00164 H	0.00137 H
Field	Voltage	200 V	200 V
	Current	12 A	11 A

Table D.1: D.C. Machine Characteristics.

References.

1.GEC Machines Ltd., Warwickshire, England, *Technical Information for the D.C. Mill Drive Motors of the Rod Mill at Lancashire Steel (Irlam) Ltd.*, Ref: LMSD/PJS/J52393, May 1979.

APPENDIX E: THE MILD STEEL CONSTITUENTS

The main constituent of steel is iron. Iron (Fe) makes up approximately 95 % of a mild steel. The other components of mild steel determine the name of the steel. Table E.1 shows the different types of steels rolled at the Hille Mill and what percentage of the weight of the steel is made up by the other elements.

These elements are carbon (C), silicon (Si), manganese (Mn), sulphur (S), phosphorous (P), chromium (Cr), nickel (Ni), molybdenum (Mo), copper (Cu), vanadium (V) and tin (Sn).

Mild Steel Constituents

Steel Type

Element % of steel Weight

	SABS 14312 40WA	SABS 14312 00WA	SABS 920 1969 Grade 415 RHTA	RHTB	RHTC	RHTD	RHTE	RHTF	RHTG
C	0.011	0.14	0.38	0.38	0.38	0.38	0.38	0.38	0.38
Si	0.20	0.28							
Mn	0.60	0.70	1.25	1.25	1.25	1.20	1.20	1.25	1.15
S	0.05	0.06							
P	0.04	0.06							
Cr	0.4	0.3							
Ni	0.4	0.4							
Mo	0.05	0.05							
Cu	0.4	0.3							
V	0.04	•	0.033	0.030	0.025	0.023	0.02	0.00	0.00
Sn	0.06	0.05							

Table E.1: Mild steel constituents[1].

References.

1. Welgemoed, K. *Private discussion*, Electrical engineer, Hille Mill, Scaw Metals Ltd., Germiston, Transvaal.

APPENDIX F: THE BAR DIMENSIONS DURING ROLLING

The cross-sectional area of the steel bar is reduced in the rolling process. The measurements [m^2] chosen for the simulation are as illustrated in figure F.1. These are based on having a final product[1] of 70 mm X 30 mm. Using different areas, to those shown here, would just alter the relative speeds that the drives would have to run at.

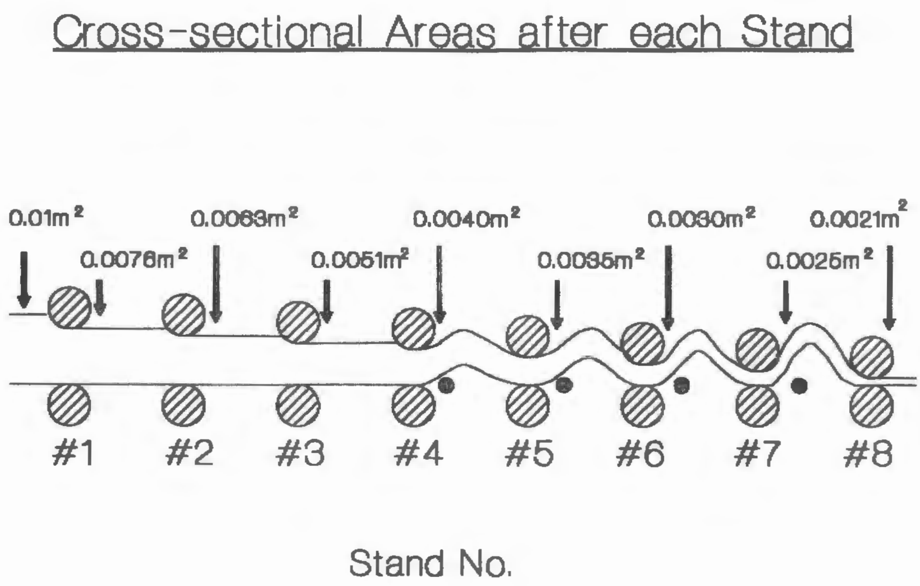


Figure F.1: The areas used in the simulation.

References.

1.Welgemoed, K. *Private discussion*, Electrical Engineer, Hille Mill, Scaw Metals Ltd., Germiston, Transvaal.

APPENDIX G: THE SIMULATION D.C. DRIVE SPEEDS

In order that the volume of metal passing between all the stands during a unit of time is the same, the drives must have the correct relative speeds. The smaller the cross-sectional area (Appendix F), the faster the drive's speed must be. Setting the output speed of the production line at 30 m/s (see section 3.4.2) and using the bar areas quoted in Appendix F, the correct drive speeds are calculated using equation 3.2.11. These are shown in figure G.1.

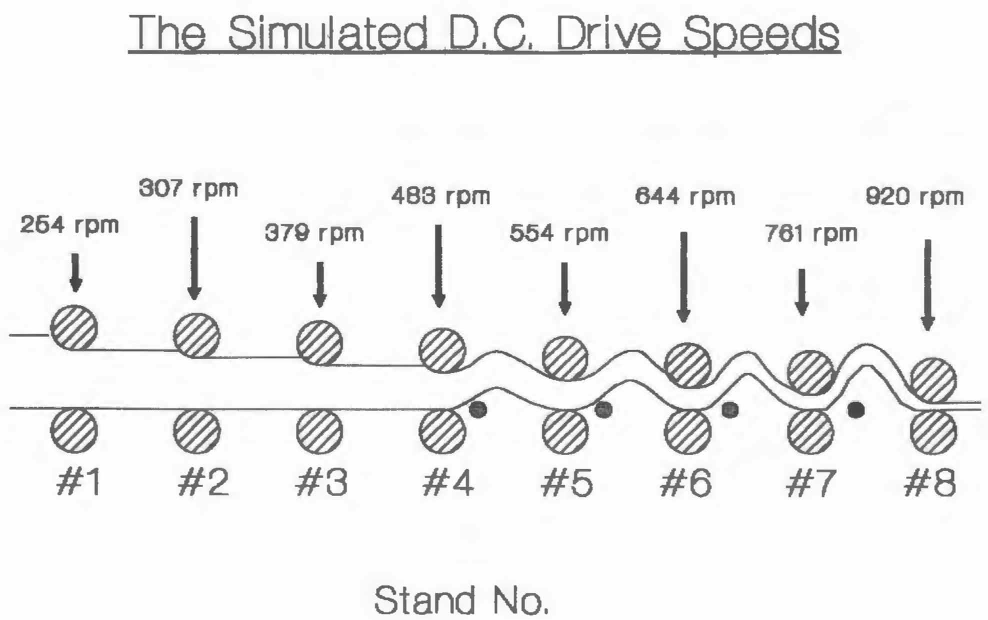


Figure G.1: The drive speeds used in the simulation.

APPENDIX H: THE THEORY OF CONTROL STRUCTURES

The control system required for a particular plant depends on it's structure. The three different structures, diagonal, triangular and full, are analysed and explained for the simple two by two plant matrix[1]. The required control systems are derived for each type of structure.

Firstly consider the case where there is no interaction.

Diagonal Plant Structure.

G(s)

$$\begin{bmatrix} g_{11} & 0 \\ 0 & g_{22} \end{bmatrix}$$

where g_{11} and g_{22} are square submatrices. The above plant matrix is known as a diagonal structure. Using the diagonal control matrix K below.

K(s)

$$\begin{bmatrix} k_{11} & 0 \\ 0 & k_{22} \end{bmatrix}$$

The OPEN LOOP TRANSFER FUNCTION MATRIX is

$Q(s) = G(s).K(s)$

$$= \begin{bmatrix} g_{11}\,k_{11} & 0 \\ 0 & g_{22}\,k_{22} \end{bmatrix}$$

Then the SENSITIVITY function is

$$[I + Q(s)]^{-1} = \begin{bmatrix} 1 + q_{11} & 0 \\ 0 & 1 + q_{22} \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} (1 + q_{11})^{-1} & 0 \\ 0 & (1 + q_{22})^{-1} \end{bmatrix}$$

The CLOSED LOOP TRANSFER FUNCTION MODEL is

$$H(s) = [I + Q(s)]^{-1} Q(s)$$

$$= \begin{bmatrix} [1 + q_{11}]^{-1} q_{11} & 0 \\ 0 & [1 + q_{22}]^{-1} q_{22} \end{bmatrix}$$

$$= \begin{bmatrix} h_{11} & 0 \\ 0 & h_{22} \end{bmatrix}$$

The following observations are made for a diagonal plant structure from the above matrix manipulation.

1. There is no interaction and no multivariable problem

2. A process with a diagonal structure requires a controller with a diagonal structure.

3. The original multivariable controller design problem (for $\underline{G(s)}$) which is inherently very complex has been split into two independent single-variable designs (for g_{11} and g_{22}) that are simpler.

Triangular Plant Structure.

There is some interaction in the following process.

$$\begin{bmatrix} y_2 \\ y_1 \end{bmatrix} = \begin{bmatrix} g_{22} & g_{21} \\ 0 & g_{22} \end{bmatrix} \begin{bmatrix} u_2 \\ u_1 \end{bmatrix}$$

The above plant does not have an idealized TRIANGULAR structure but by re-ordering the inputs and outputs this can be obtained.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} g_{11} & 0 \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Using the simple diagonal controller

$$\underline{K(s)} \qquad \begin{bmatrix} k_{11} & 0 \\ 0 & k_{22} \end{bmatrix}$$

The OPEN LOOP TRANSFER FUNCTION model is

$$Q(s) = G(s).K(s)$$

$$= \begin{bmatrix} g_{11}\,k_{11} & 0 \\ g_{21}\,k_{11} & g_{22}\,k_{22} \end{bmatrix} = \begin{bmatrix} q_{11} & 0 \\ q_{21} & q_{22} \end{bmatrix}$$

The CLOSED LOOP TRANSFER FUNCTION MODEL is

$$H(s) = [I + Q(s)]^{-1} Q(s)$$

$$= \begin{bmatrix} [1 + q_{11}]^{-1} q_{11} & 0 \\ [1+q_{22}]^{-1} q_{21} [1+q_{11}]^{-1} & [1 + q_{22}]^{-1} q_{22} \end{bmatrix}$$

$$= \begin{bmatrix} h_{11} & 0 \\ h_{21} & h_{22} \end{bmatrix}$$

The following observations are made about the control of a triangular structure with a diagonal controller.

1. There is interaction but it does not constitute a multivariable problem.

2. The diagonal terms of the closed loop matrix do not depend on the interaction but only on the corresponding open loop models.

3. A disturbance term, h_{21}, has been created between the first and second loop. The second loop however does not affect the first.

4. With good designs for the two diagonal terms of the closed loop, the disturbance term, h_{21}, is reduced by the factors $[1 + q_{ii}]^{-1}$.

5. If the two single-variable control loops are stable, the total system will be stable if q_{21} is.

6. The disturbance term can be further reduced by using a feedforward controller, discussed in the next section.

Feedforward Controllers.

The following plant has some interaction

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} g_{11} & 0 \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Using a triangular controller.

K(s) $$\begin{bmatrix} k_{11} & 0 \\ k_{21} & k_{22} \end{bmatrix}$$

The OPEN LOOP TRANSFER FUNCTION model is

$Q(s) = G(s).K(s)$

$$= \begin{bmatrix} g_{11}\ k_{11} & 0 \\ g_{21}\ k_{11} + g_{22}\ k_{21} & g_{22}\ k_{22} \end{bmatrix} = \begin{bmatrix} q_{11} & 0 \\ q_{21} & q_{22} \end{bmatrix}$$

The disturbance term q_{21} can be reduced by designing the feedforward controller k_{21} so that

$$g_{21}\ k_{11} + g_{22}\ k_{21} = 0$$

Rearranging for k_{21}

$$k_{21} = -g_{22}^{-1} g_{21} k_{11}$$

If k_{21} is stable and realizable, the open loop transfer function model reduces to a diagonal structure

$$Q(s) = \begin{bmatrix} q_{11} & 0 \\ 0 & q_{22} \end{bmatrix}$$

The structure is now controlled using two single variable designs as explained under diagonal structures.

The following observations concerning feedforward controllers are made from the above manipulation.

1. In practice the controller k_{21} may not be obtainable so that the disturbance term is exactly zero. In this case an approximation to this controller is implemented.

2. The matrix K(s) is usually designed by a number of column operations on G(s)[1].

Full Plant Structure.

In this case all the plant variables interact.

$$\underline{G(s)} \qquad \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}$$

Normally these processes cannot be controlled by single-variable control structures. This is because both loops are affected by the other. Thus, when installing a controller for g_{11} and then closing that loop the relationship g_{22} changes. This is due to a second path through which u_2

affects y_2 being formed by the feedback of the first closed loop.

Similarly, when closing the second loop g_{11} will change. Hence a number of interactive single-variable controller designs would have to be performed. In systems with a large plant matrix this method of controller design becomes very complicated. Multivariable design methods, such as I.N.A. are, therefore, often used for the full plant structure.

Summary of Control Structures.

G(s) Structure	K(s) Structure	Open Loop Transfer Function	Closed Loop Transfer Function	Controller Design
Diagonal	Diagonal	Diagonal	Diagonal	Single-variable
Triangular	Diagonal	Triangular	Triangular	Single-variable
Triangular	Triangular	Diagonal	Diagonal	S-variable & feedforward
Full	Diagonal*	Full	Full	Multi-variable
Full	Triangular	Triangular	Triangular	Multi-variable
Full	Full	Diagonal	Diagonal	Multi-variable

*if stable

Table H.1: Summary of control structures.

The table H.1 illustrates all the possible combinations of plant and controller structures and the resulting controller design method.

References.

1. Gear, A.B.J. and Braae, M.: *Structure of Large Scale Systems,* SACAC 3rd Workshop on Multivariabel Control, Mintek, Randburg, 1986.
2. Rosenbrock, H.H.: *State Space and Multivariable Theory,* Nelson, London, 1970.

APPENDIX I: HILLE MILL RAMP TESTS

The results of the ramp tests conducted on Drive #7, figure 1.1, are recorded in the figures discussed below. The data taken from these graphs is used in Chapter 5 to determine transfer function models for the actual Hille Mill. These tests are done while the metal bar is in the stand.

In the following graphs, the reference speed (1), the drive current (2) and the actual speed (3) are recorded. The zero line for each output is marked by the encircled number. The time scale in the first two graphs is 1 second per division, while in the third it is half this. The outputs scales are as follows. Figures I.1 and I.2: (1) 110 rpm/div. (2) 140 A/div. (3) 150 rpm/div. Figure I.3: (1) & (3) 55 rpm/div. (2) 200 A/div.

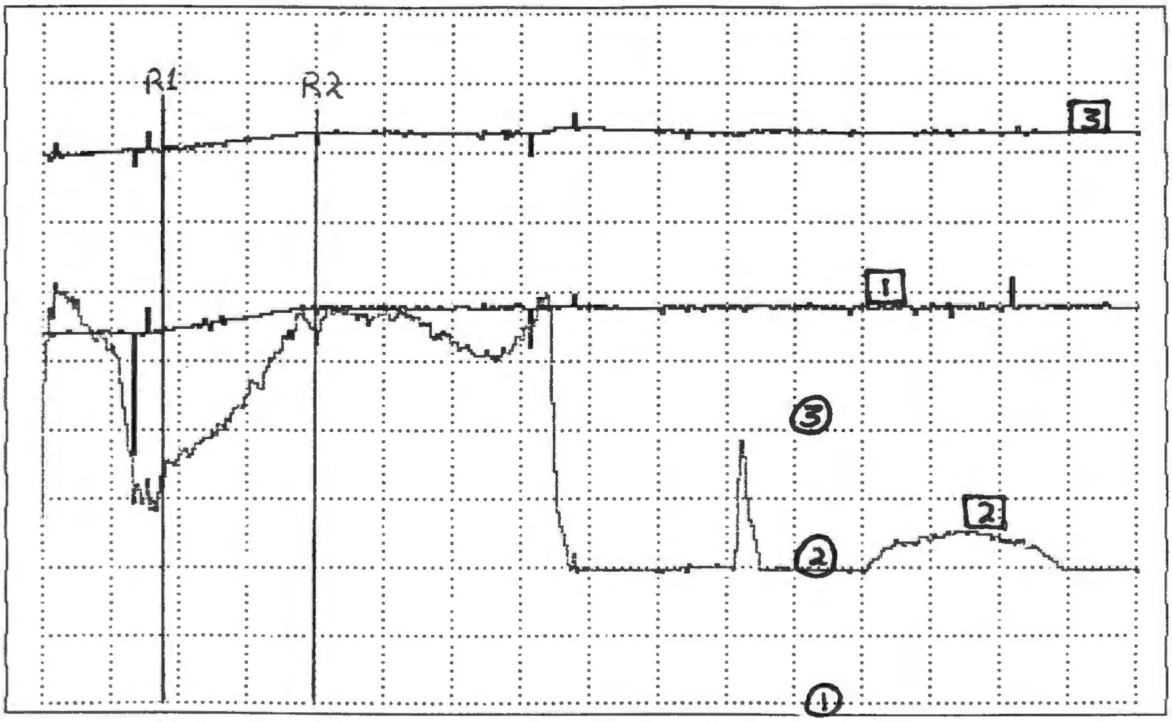


Figure I.1: Ramp test data from Stand #7.

Figure I.1 shows the results of two ramp tests. These are marked R1 and R2. The speed setpoint of the seventh drive (Drive #3 in production line A) is first ramped up at 20 rpm per second. Two seconds later it is kept at 640 rpm by imposing another ramp of -20 rpm per second. The speed output of the drive is seen to closely follow the speed setpoint.

Figure I.2 illustrates a ramp decrease in the speed setpoint of the seventh drive. The current output is seen to decline in response to this ramp. The sudden changes in the current are caused by the bar entering and leaving the stand.

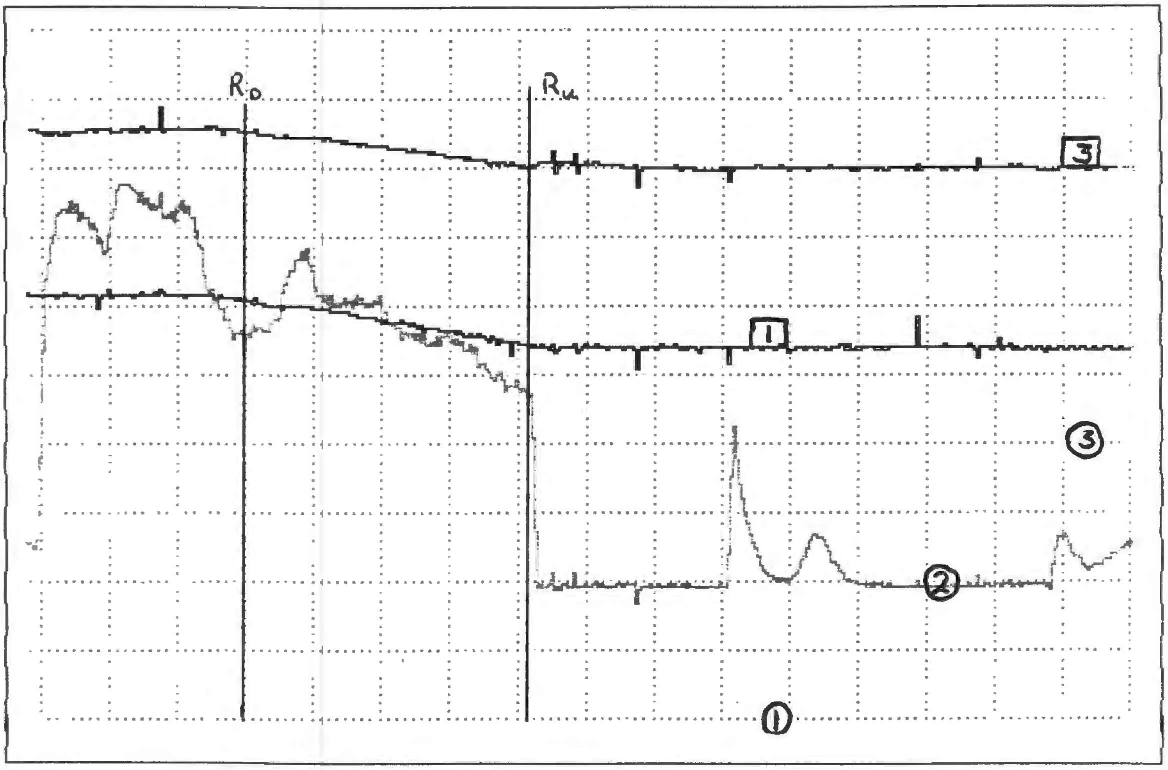


Figure I.2: Decline ramp test data from Stand #7.

The third ramp test conducted on this drive is illustrated in figure I.3. There are two tests during this time period. The first occurs at R3 and the second at R4. The amplitudes of these ramps are -5.64 rpm/second and +5.64 rpm/second respectively.

The point at which the bar enters or leaves the stand is marked by the sudden change in the drive current explained previously. This is due to the immediate increase or decrease in the load torque placed on the motor.

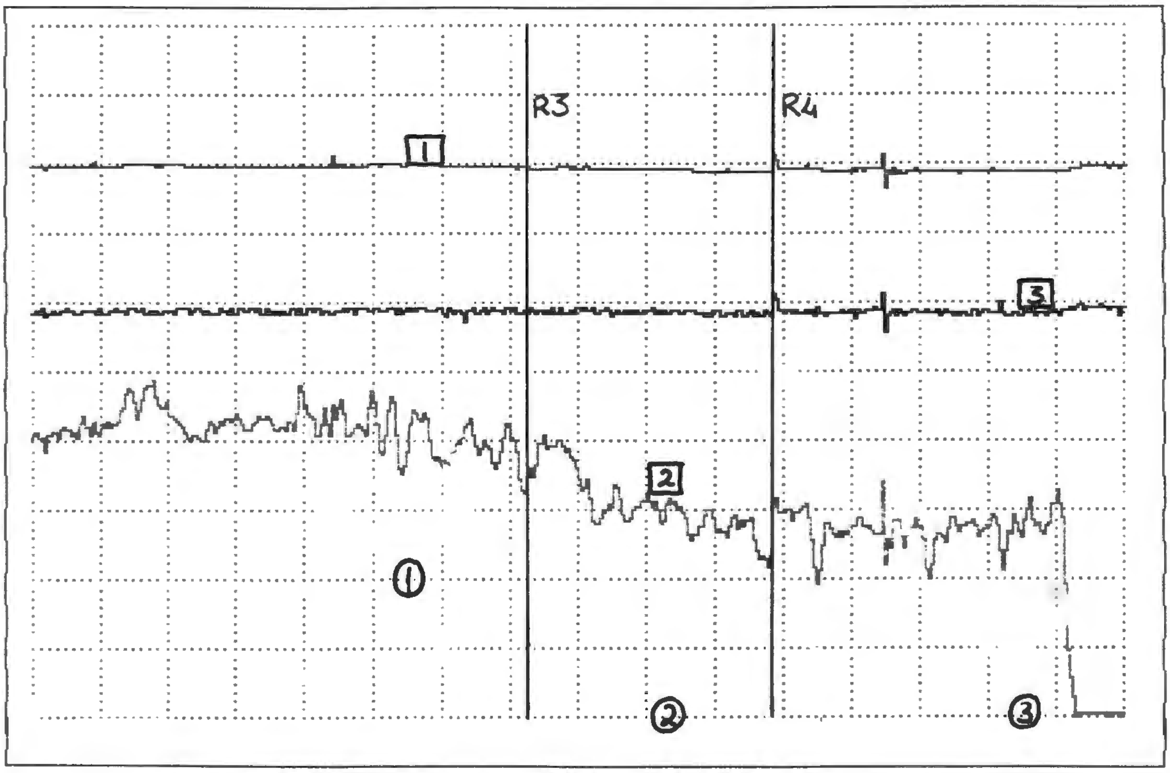


Figure I.3: Ramp test data from Stand #7.

The current in the drive decreases due to the first ramp down in the speed setpoint. It then levels off again after the second ramp change. There is however a large amount of noise on the current signal and some of the change in the response could be due to noise disturbances.

APPENDIX J: TRANSFER FUNCTION DERIVATION

The transfer function equations quoted in Chapter 5 are derived in this appendix. These equations are for the situation where the metal bar is in all eight stands of the production line. The other situations described in Chapter 4 are investigated in Appendix K.

All these transfer function equations are calculated from the graphs presented in Chapter 5.

Transfer Function G_{11}.

From figure 5.10:

Change in output $dy := 414.2 - 222.9 = 191.3$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = 49.3 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{11} := \frac{slope}{step}$$

$$G_{11}(s) := \frac{3.88}{s} \text{ [A/rpm]}$$

Transfer Function G_{21}.

From figure 5.11:

Change in output dy := 208.77 -268.81 = -60.04 [A]

Time period dt := 3.88 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -15.47 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{21} := \frac{\text{slope}}{\text{step}}$$

$$G_{21}(s) := \frac{-1.22}{s} \text{ [A/rpm]}$$

Transfer Function G_{31}.

From figure 5.12:

Change in output dy := 278.14 - 332.16 = -54.02 [A]

Time period dt := 3.88 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -13.92 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{31} := \frac{\text{slope}}{\text{step}}$$

$$G_{31}(s) := \frac{-1.10}{s} \text{ [A/rpm]}$$

Transfer Function G_{41}.

From figure 5.15:

Change in output dy := 2.248 - 2.137 = 0.1108 [m]

Time period dt := 2.1 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.053 \text{ [m/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{41} := \frac{\text{slope}}{\text{step}}$$

$$G_{41}(s) := \frac{0.0042}{s} \text{ [m/rpm]}$$

Transfer Function G_{12}.

From figure 5.16:

Change in output dy := 161.26 - 222.9 = -61.64 [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -15.89 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{12} := \frac{\text{slope}}{\text{step}}$$

$$G_{12}(s) := \frac{-1.04}{s} \text{ [A/rpm]}$$

Transfer Function G_{22}.

From figure 5.17:

Change in output $dy := 481.49 - 268.81 = 212.68$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 54.81 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{22} := \frac{\text{slope}}{\text{step}}$$

$$G_{22}(s) := \frac{3.57}{s} \quad [A/rpm]$$

Transfer Function G_{32}.

From figure 5.18:

Change in output $dy := 267.32 - 332.16 = -64.84$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = -16.71 \quad [A/s]$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{32} := \frac{slope}{step}$$

$$G_{32}(s) := \frac{-1.09}{s} \quad [A/rpm]$$

Transfer Function G_{42}.

From figure 5.19:

Change in output $dy := 2.28 - 2.10 = 0.18$ [m]

Time period $dt := 2.95$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.061 \text{ [m/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{42} := \frac{\text{slope}}{\text{step}}$$

$$G_{42}(s) := \frac{0.004}{s} \text{ [m/rpm]}$$

Transfer Function G_{13}.

From figure 5.20:

Change in output dy := 168.48 - 222.91 = -54.43 [A]

Time period dt := 3.88 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -14.03 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{13} := \frac{\text{slope}}{\text{step}}$$

$$G_{13}(s) := \frac{-0.74}{s} \text{ [A/rpm]}$$

Transfer Function G_{23}.

From figure 5.21:

Change in output $dy := 203.5 - 268.81 = -65.31$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -16.83 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{23} := \frac{\text{slope}}{\text{step}}$$

$$G_{23}(s) := \frac{-0.89}{s} \text{ [A/rpm]}$$

Transfer Function G_{33}.

From figure 5.22:

Change in output $dy := 563.93 - 332.16 = 231.77$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 59.73 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{33} := \frac{\text{slope}}{\text{step}}$$

$$G_{33}(s) := \frac{3.15}{s} \quad [A/rpm]$$

Transfer Function G_{43}.

From figure 5.23:

Change in output dy := 2.342 - 2.168 = 0.184 [m]

Time period dt := 2.44 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.075 \ [m/s]$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{43} := \frac{\text{slope}}{\text{step}}$$

$$G_{43}(s) := \frac{0.004}{s} \quad [m/rpm]$$

Transfer Function G_{14}.

From figure 5.24:

Change in output dy := 160.5 - 222.9 = -62.4 [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = -16.09 \text{ [A/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{14} := \frac{slope}{step}$$

$$G_{14}(s) := \frac{-0.67}{s} \text{ [A/rpm]}$$

Transfer Function G_{24}.

From figure 5.25:

Change in output $dy := 194.5 - 268.8 = -74.3$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = -19.15 \text{ [A/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{24} := \frac{slope}{step}$$

$$G_{24}(s) := \frac{-0.8}{s} \quad [A/rpm]$$

Transfer Function G_{34}.

From figure 5.26:

Change in output dy := 237.82 - 332.16 = -94.34 [A]

Time period dt := 3.88 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -24.31 \ [A/s]$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{34} := \frac{\text{slope}}{\text{step}}$$

$$G_{34}(s) := \frac{-1.01}{s} \quad [A/rpm]$$

Transfer Function G_{44}.

From figure 5.27:

Change in output dy := 0.3125 [m]

Time period dt := 2.54 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.123 \text{ [m/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{44} := \frac{\text{slope}}{\text{step}}$$

$$G_{44}(s) := \frac{0.005}{s} \text{ [m/rpm]}$$

Transfer Function G_{45}.

From figure 5.28:

Change in output dy := -0.016 [m]

Time period dt := 2.517 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -0.0064 \text{ [m/s]}$$

The step in S5 := 0.554 [rpm]

Therefore since

$$G_{45} := \frac{\text{slope}}{\text{step}}$$

$$G_{45}(s) := \frac{-0.012}{s} \text{ [m/rpm]}$$

Transfer Function G_{55}.

From figure 5.29:

Change in output dy := 0.023 [m]

Time period dt := 3.25 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.0069 \text{ [m/s]}$$

The step in S5 := 0.554 [rpm]

Therefore since

$$G_{55} := \frac{\text{slope}}{\text{step}}$$

$$G_{55}(s) := \frac{0.012}{s} \text{ [m/rpm]}$$

Transfer Function G_{56}.

From figure 5.30:

Change in output dy := 0.015 [m]

Time period dt := 2.64 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.0057 \text{ [m/s]}$$

The step in S6 := 0.483 [rpm]

Therefore since

$$G_{56} := \frac{\text{slope}}{\text{step}}$$

$$G_{56}(s) := \frac{-0.012}{s} \; [\text{m/rpm}]$$

Transfer Function G_{66}.

From figure 5.31:

Change in output $dy := 0.025$ [m]

Time period $dt := 3.86$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.0065 \; [\text{m/s}]$$

The step in S6 := 0.483 [rpm]

Therefore since

$$G_{66} := \frac{\text{slope}}{\text{step}}$$

$$G_{66}(s) := \frac{0.013}{s} \; [\text{m/rpm}]$$

Transfer Function G_{67}.

From figure 5.32:

Change in output $dy := 0.028$ [m]

Time period dt := 4.17 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.0067 \text{ [m/s]}$$

The step in S7 := 0.571 [rpm]

Therefore since

$$G_{67} := \frac{\text{slope}}{\text{step}}$$

$$G_{67}(s) := \frac{-0.012}{s} \text{ [m/rpm]}$$

<u>Transfer Function G_{77}.</u>

From figure 5.33:

Change in output dy := 0.029 [m]

Time period dt := 3.86 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.0074 \text{ [m/s]}$$

The step in S7 := 0.571 [rpm]

Therefore since

$$G_{77} := \frac{\text{slope}}{\text{step}}$$

$$G_{77}(s) := \frac{0.013}{s} \text{ [m/rpm]}$$

<u>Transfer Function G_{78}.</u>

From figure 5.34:

Change in output dy := 0.072 [m]

Time period dt := 4.75 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -0.015 \text{ [m/s]}$$

The step in S8 := 1.38 [rpm]

Therefore since

$$G_{78} := \frac{\text{slope}}{\text{step}}$$

$$G_{78}(s) := \frac{-0.011}{s} \text{ [m/rpm]}$$

<u>Transfer Function G_{88}.</u>

This transfer function is calculated in Chapter 5.

APPENDIX K: THE CYCLE OF TRANSFER FUNCTIONS

Some of the transfer function equations describing the relationships between the plant outputs and the inputs change during the rolling process cycle described in Chapter 4. The reasons for the plant interaction occuring as it does at different stages is also explained in Chapter 4.

Certain features about the output responses obtained and analysed in this appendix are explained in Chapter 5. This is done for the situation where there is metal in all the rolling stands.

The transfer functions for all the other stages in the process cycle are derived in this appendix. Only those outputs that change are covered in the following derivations. The other transfer functions used in Chapter 5 are derived in Appendix J.

The transfer functions describing the loop length responses, L2, L3 and L4, and the response of the speed output of the eighth stand, S8, remain constant. This is because the interaction effects on the drives on either side of these loops do not occur in a cascaded manner as they do with the tension interaction (Chapter 4).

The outputs affected in a cascaded manner, A1, A2, A3 and L1 react differently to changes in the inputs, S1 - S4, depending on the position of the bar. This is illustrated by the following graphs.

Metal in Stand #1.

Transfer Function G_{11}.

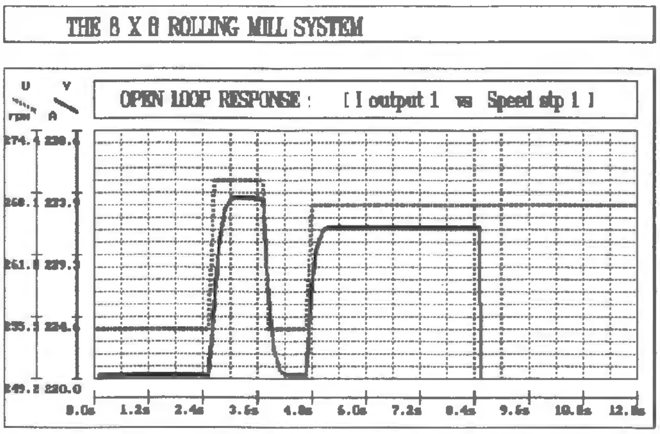


Figure K.1: The response of A1 to a step in S1.

From figure K.1:

Change in steady state value

$$dy := 231.33 - 220.30 = 11.03 \text{ [A]}$$

Step in input S1 := 12.71 [rpm]

Therefore gain $$K := \frac{dy}{step} := 0.868 \text{ [A/rpm]}$$

Time constant $T \approx 0.1$ [seconds]

Therefore since:

$$G_{11}(s) = \frac{K}{1 + s*T}$$

$$G_{11}(s) = \frac{8.68}{s + 10} \text{ [A/rpm]}$$

Transfer Function G_{22}.

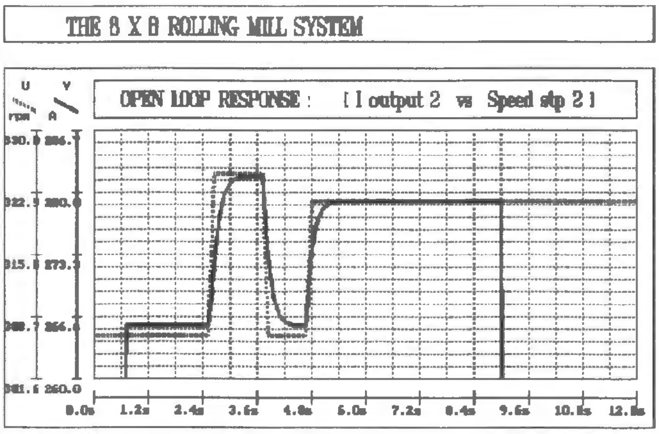


Figure K.2: The response of A2 to a step in S2.

From figure K.2:

Change in steady state value
$$dy := 279.07 - 265.8 = 13.27 \text{ [A]}$$

Step in input S2 := 15.34 [rpm]

Therefore gain $$K := \frac{dy}{step} := 0.868 \text{ [A/rpm]}$$

Time constant $T \approx 0.1$ [seconds]

Therefore since:

$$G_{22}(s) = \frac{K}{1 + s*T}$$

$$G_{22}(s) = \frac{8.68}{s + 10} \text{ [A/rpm]}$$

Transfer Function G_{33}.

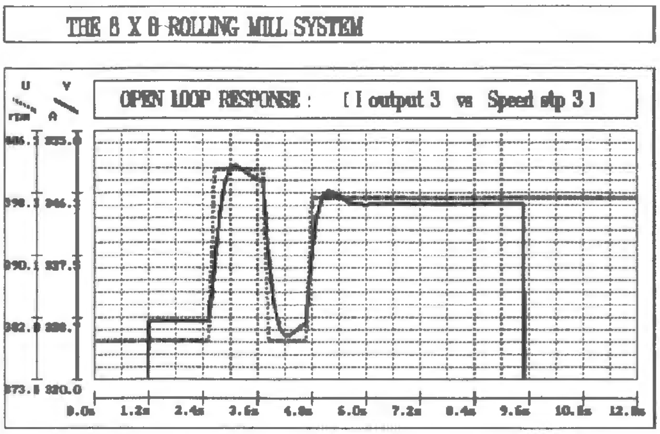


Figure K.3.a: The response of A3 to a step in S3.

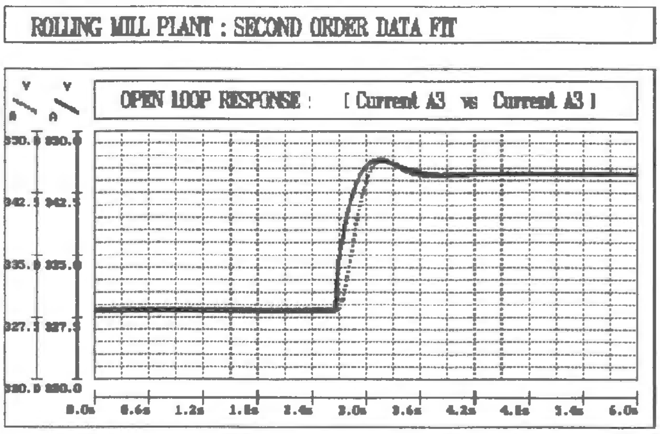


Figure K.3.b: The best fit curve (dotted line).

Using the function fit program (Appendix K), the best fit second order response is shown in figure K.3.b. Therefore G_{33} is approximated by:

$$G_{33}(s) = \frac{51.78}{s^2 + 9s + 59.73} \quad [A/rpm]$$

Metal in Stands 1&2.

Transfer Function G_{11}.

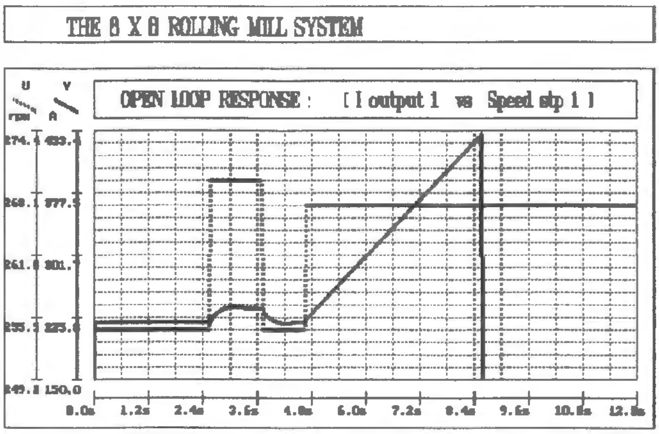


Figure K.4: The response of A1 to a step in S1.

From figure K.4:

Change in output $dy := 448.4 - 220.3 = 228.1$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 58.79 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{11} := \frac{\text{slope}}{\text{step}}$$

$$G_{11}(s) := \frac{4.63}{s} \text{ [A/rpm]}$$

Transfer Function G_{21}.

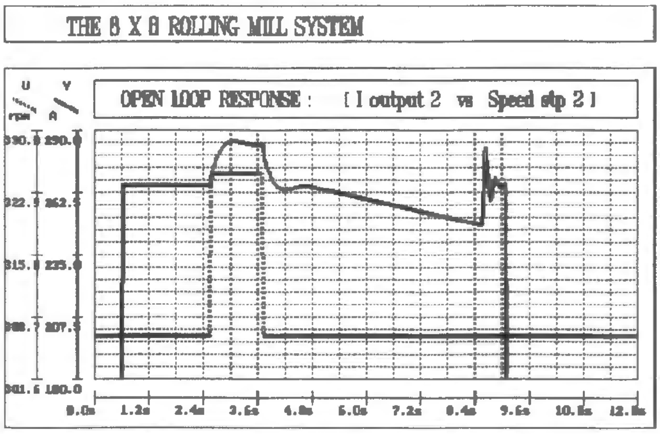


Figure K.5: The response of A2 to a step in S1.

From figure K.5:

Change in output $dy := 247.77 - 265.8 = -17.97$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -4.63 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{21} := \frac{\text{slope}}{\text{step}}$$

$$G_{21}(s) := \frac{-0.3}{s} \text{ [A/rpm]}$$

Transfer Function G_{12}.

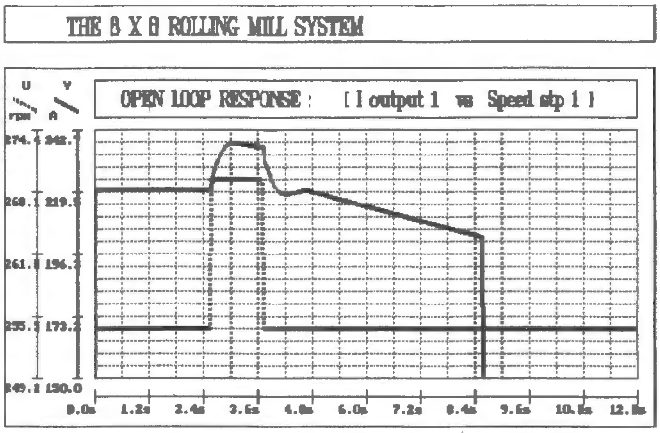


Figure K.6: The response of A1 to a step in S2.

From figure K.6:

Change in output $dy := 202.56 - 220.3 = -17.74$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -4.57 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{12} := \frac{\text{slope}}{\text{step}}$$

$$G_{12}(s) := \frac{-0.3}{s} \text{ [A/rpm]}$$

Transfer Function G_{22}.

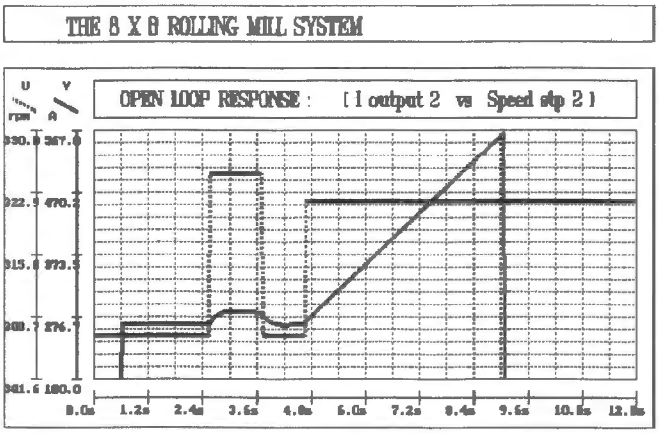


Figure K.7: The response of A2 to a step in S2.

From figure K.7:

Change in output $dy := 533.3 - 265.8 = 267.5$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 68.95 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{22} := \frac{\text{slope}}{\text{step}}$$

$$G_{22}(s) := \frac{4.5}{s} \text{ [A/rpm]}$$

Metal in Stands 1-3.

Transfer Function G_{11}.

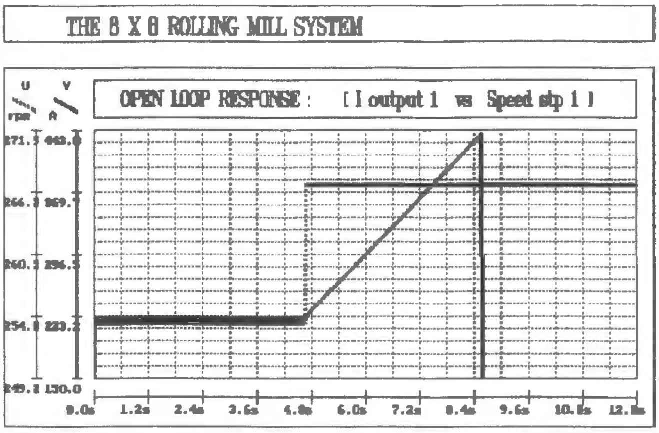


Figure K.8: The response of A1 to a step in S1.

From figure K.8:

Change in output dy := 438.0 - 220.3 = 217.7 [A]
Time period dt := 3.88 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 56.11 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{11} := \frac{\text{slope}}{\text{step}}$$

$$G_{11}(s) := \frac{4.41}{s} \text{ [A/rpm]}$$

Transfer Function G_{21}.

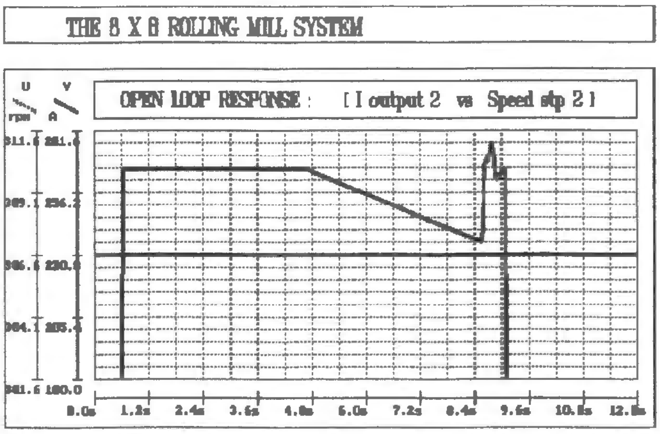


Figure K.9: The response of A2 to a step in S1.

From figure K.9:

Change in output $dy := 236.73 - 265.8 = -29.07$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -7.49 \text{ [A/s]}$$

The step in S1 $:= 12.71$ [rpm]

Therefore since

$$G_{21} := \frac{\text{slope}}{\text{step}}$$

$$G_{21}(s) := \frac{-0.59}{s} \text{ [A/rpm]}$$

Transfer Function G_{31}.

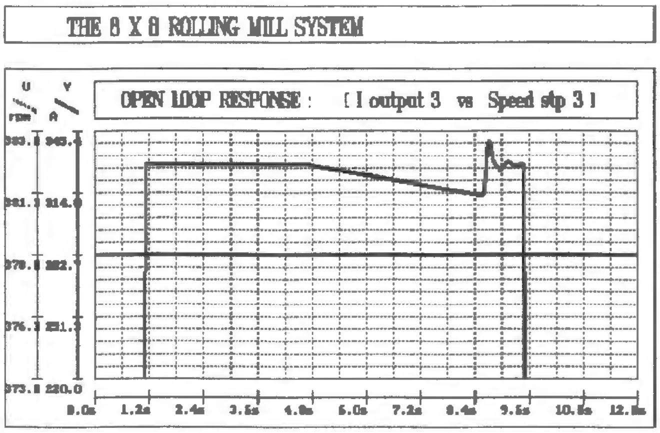


Figure K.10: The response of A3 to a step in S1.

From figure K.10:

Change in output $dy := 312.96 - 328.3 = -15.34$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -3.95 \text{ [A/s]}$$

The step in S1 := 12.71 [rpm]

Therefore since

$$G_{31} := \frac{\text{slope}}{\text{step}}$$

$$G_{31}(s) := \frac{-0.31}{s} \text{ [A/rpm]}$$

Transfer Function G_{12}.

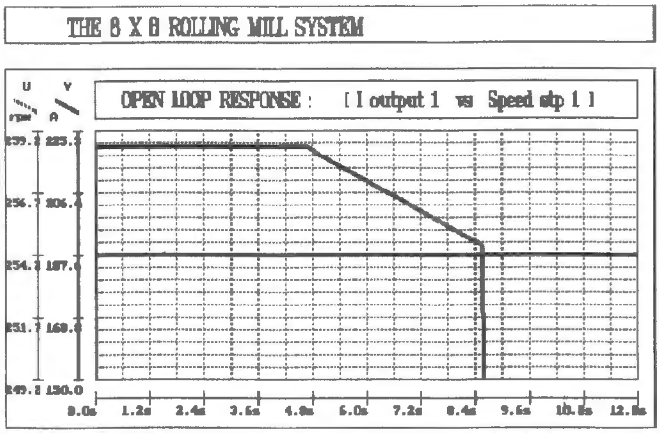


Figure K.11: The response of A1 to a step in S2.

From figure K.11:

Change in output $dy := 190.1 - 220.3 = -30.2$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -7.78 \text{ [A/s]}$$

The step in S2 $:= 15.34$ [rpm]

Therefore since

$$G_{12} := \frac{\text{slope}}{\text{step}}$$

$$G_{12}(s) := \frac{-0.51}{s} \text{ [A/rpm]}$$

Transfer Function G_{22}.

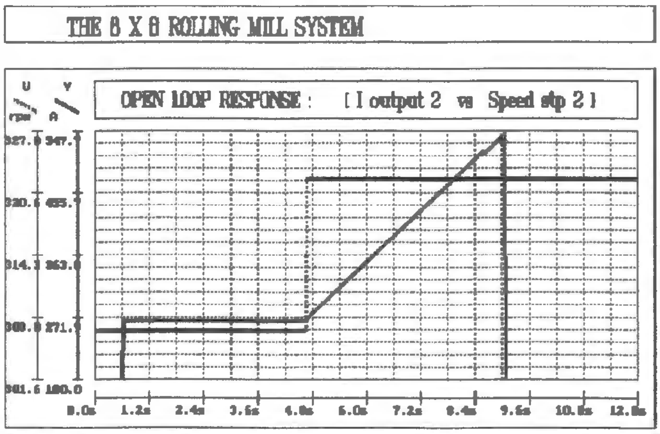


Figure K.12: The response of A2 to a step in S2.

From figure K.12:

Change in output $dy := 517.03 - 265.8 = 251.23$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 64.75 \text{ [A/s]}$$

The step in S2 $:= 15.34$ [rpm]

Therefore since

$$G_{22} := \frac{\text{slope}}{\text{step}}$$

$$G_{22}(s) := \frac{4.22}{s} \text{ [A/rpm]}$$

Transfer Function G_{32}.

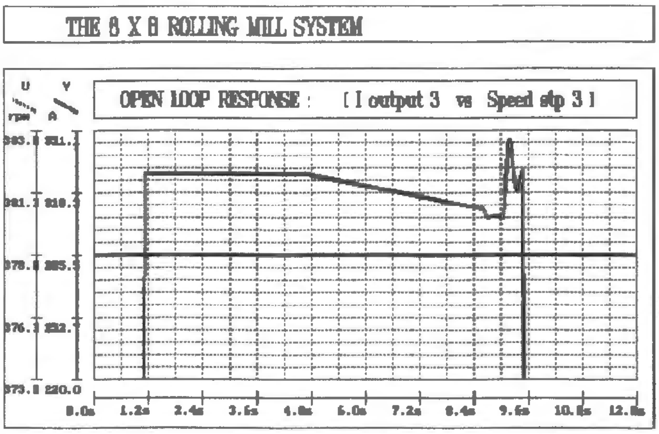


Figure K.13: The response of A3 to a step in S2.

From figure K.13:

Change in output	$dy := 309.6 - 328.3 = -18.75$ [A]
Time period	$dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -4.83 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{32} := \frac{\text{slope}}{\text{step}}$$

$$G_{32}(s) := \frac{-0.32}{s} \text{ [A/rpm]}$$

Transfer Function G_{13}.

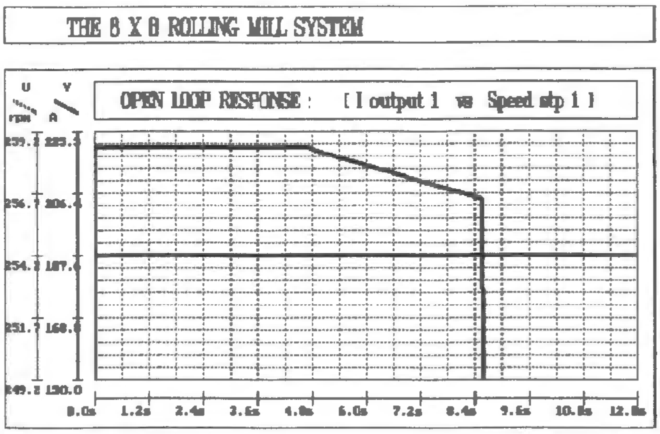


Figure K.14: The response of A1 to a step in S3.

From figure K.14:

Change in output $dy := 205.15 - 220.3 = -15.15$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -3.91 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{13} := \frac{\text{slope}}{\text{step}}$$

$$G_{13}(s) := \frac{-0.21}{s} \text{ [A/rpm]}$$

Transfer Function G_{23}.

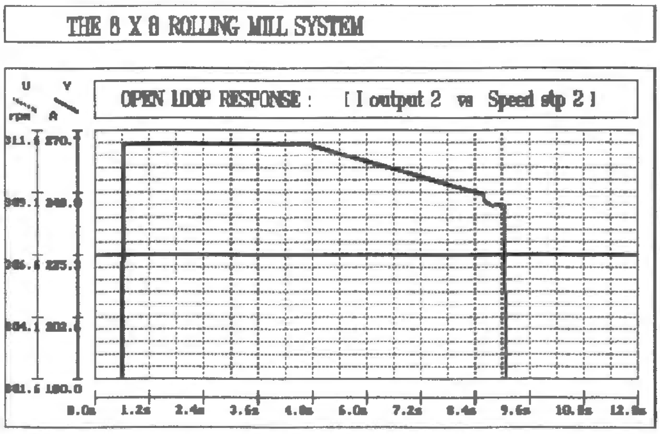


Figure K.15: The response of A2 to a step in S3.

From figure K.15:

Change in output $dy := 247.7 - 265.8 = -18.1$ [A]

Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -4.67 \text{ [A/s]}$$

The step in S3 $:= 18.94$ [rpm]

Therefore since

$$G_{23} := \frac{\text{slope}}{\text{step}}$$

$$G_{23}(s) := \frac{-0.25}{s} \text{ [A/rpm]}$$

Transfer Function G_{33}.

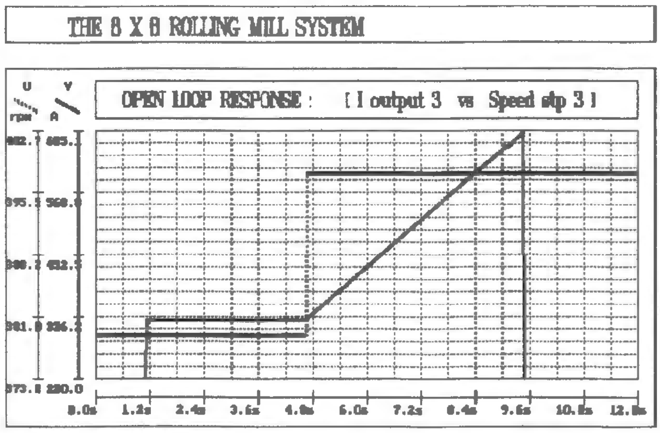


Figure K.16: The response of A3 to a step in S3.

From figure K.16:

Change in output $dy := 616.87 - 328.3 = 288.57$ [A]
Time period $dt := 3.88$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 74.37 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{33} := \frac{\text{slope}}{\text{step}}$$

$$G_{33}(s) := \frac{3.93}{s} \text{ [A/rpm]}$$

Metal in Stands 1-4.

The transfer equations in this situation in the cycle are those calculated for the case where metal is in all the drives. These are covered in Chapter 4.

Metal in Stands 1-5.

Refer to Chapter 4. The transfer equations for this situation are those for when the metal bar is in all the drives.

Metal in Stands 1-6.

Refer to Chapter 4. The transfer equations for this situation are those for when the metal bar is in all the drives.

Metal in Stands 1-7.

Refer to Chapter 4. The transfer equations for this situation are those for when the metal bar is in all the drives.

Metal in Stands 1-8.

Refer to Chapter 4. The transfer equations for this situation are calculated in Appendix I.

Metal in Stands 2-8.

Transfer Function G_{22}.

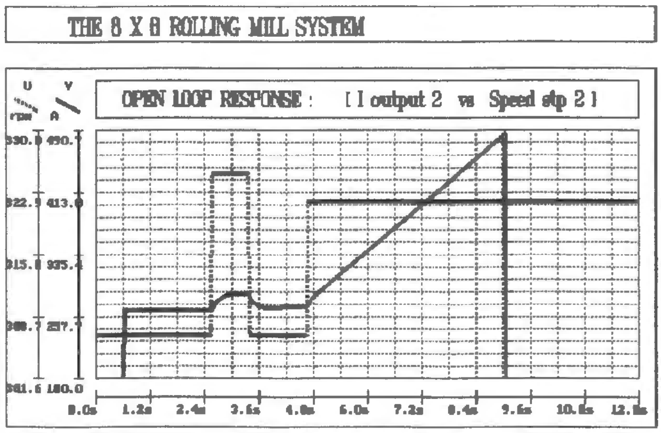


Figure K.17: The response of A2 to a step in S2.

From figure K.17:

Change in output $dy := 485.8 - 265.8 = 220.0$ [A]

Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 50.23 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{22} := \frac{\text{slope}}{\text{step}}$$

$$G_{22}(s) := \frac{3.28}{s} \text{ [A/rpm]}$$

Transfer Function G_{32}.

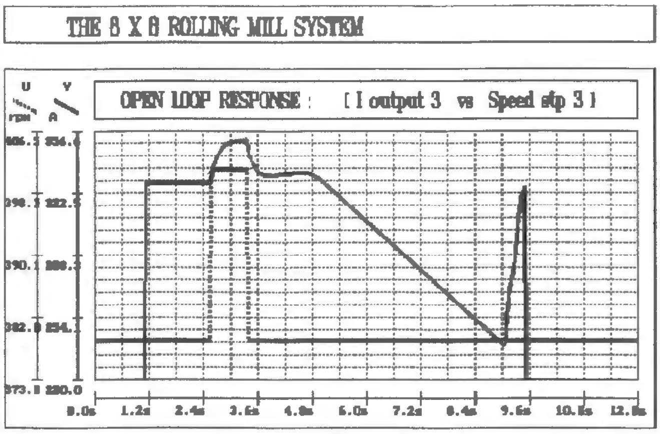


Figure K.18: The response of A3 to a step in S2.

From figure K.18:

Change in output dy := 239.55 -328.3 = -88.75 [A]
Time period dt := 4.38 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -20.26 \text{ [A/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{32} := \frac{\text{slope}}{\text{step}}$$

$$G_{32}(s) := \frac{-1.32}{s} \text{ [A/rpm]}$$

Transfer Function G_{42}.

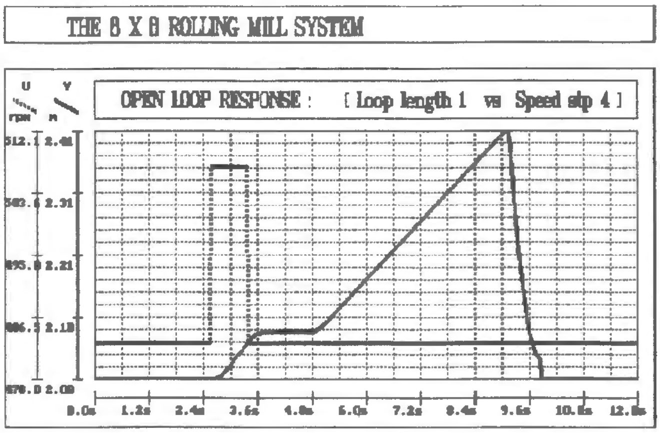


Figure K.19: The response of L1 to a step in S2.

From figure K.19:

Change in output $dy := 2.41 - 2.079 = 0.331$ [m]
Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.076 \text{ [m/s]}$$

The step in S2 := 15.34 [rpm]

Therefore since

$$G_{42} := \frac{\text{slope}}{\text{step}}$$

$$G_{42}(s) := \frac{0.005}{s} \text{ [m/rpm]}$$

Transfer Function G_{23}.

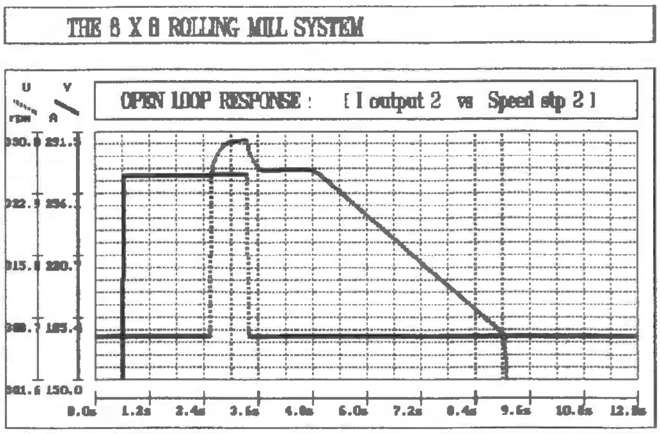


Figure K.20: The response of A2 to a step in S3.

From figure K.20:

Change in output $dy := 174.78 - 265.8 = -91.02$ [A]

Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -20.78 \text{ [A/s]}$$

The step in S3 $:= 18.94$ [rpm]

Therefore since

$$G_{23} := \frac{\text{slope}}{\text{step}}$$

$$G_{23}(s) := \frac{-1.10}{s} \text{ [A/rpm]}$$

Transfer Function G_{33}.

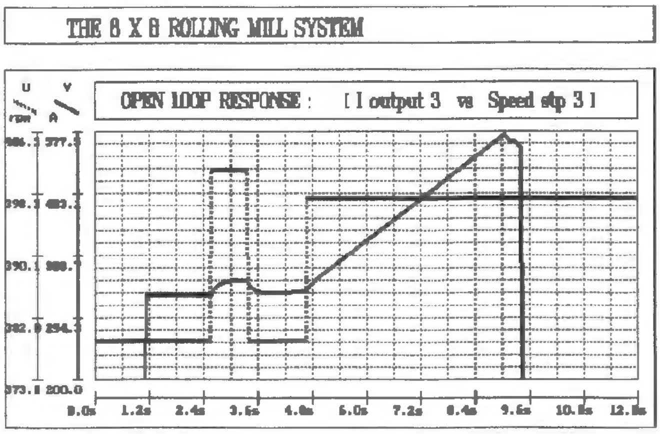


Figure K.21: The response of A3 to a step in S3.

From figure K.21:

Change in output $dy := 572.6 - 328.3 = 244.3$ [A]
Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 55.78 \text{ [A/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{33} := \frac{\text{slope}}{\text{step}}$$

$$G_{33}(s) := \frac{2.95}{s} \text{ [A/rpm]}$$

Transfer Function G_{43}.

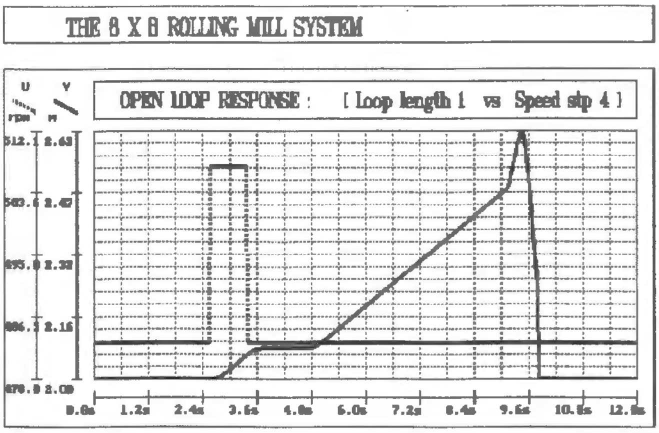


Figure K.22: The response of L1 to a step in S3.

From figure K.22:

Change in output dy := 2.52 - 2.08 = 0.44 [m]

Time period dt := 4.38 [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.101 \text{ [m/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{43} := \frac{\text{slope}}{\text{step}}$$

$$G_{43}(s) := \frac{0.005}{s} \text{ [m/rpm]}$$

Transfer Function G_{24}.

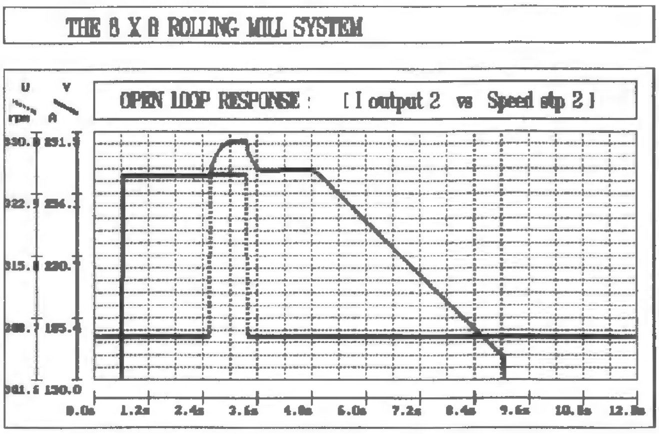


Figure K.23: The response of A3 to a step in S4.

From figure K.23:

Change in output $dy := 163.2 - 265.8 = -102.58$ [A]
Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -23.42 \text{ [A/s]}$$

The step in S4 $:= 24.15$ [rpm]

Therefore since

$$G_{24} := \frac{\text{slope}}{\text{step}}$$

$$G_{24}(s) := \frac{-0.97}{s} \text{ [A/rpm]}$$

Transfer Function G_{34}.

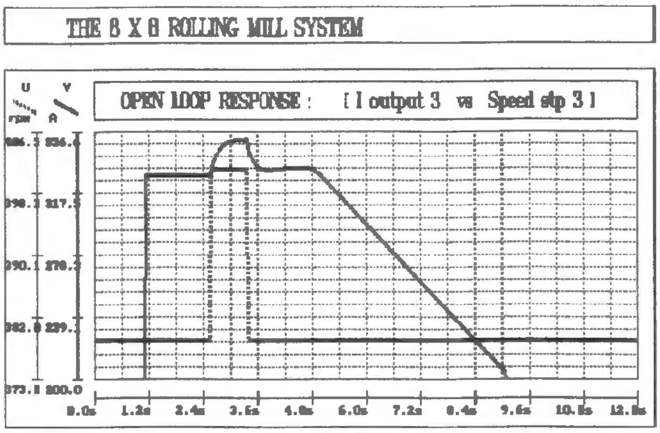


Figure K.24: The response of A2 to a step in S4.

From figure K.24:

Change in output $dy := 205.2 - 328.3 = -123.09$ [A]

Time period $dt := 4.38$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = -28.1 \text{ [A/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{34} := \frac{\text{slope}}{\text{step}}$$

$$G_{34}(s) := \frac{-1.16}{s} \text{ [A/rpm]}$$

Transfer Function G_{44}.

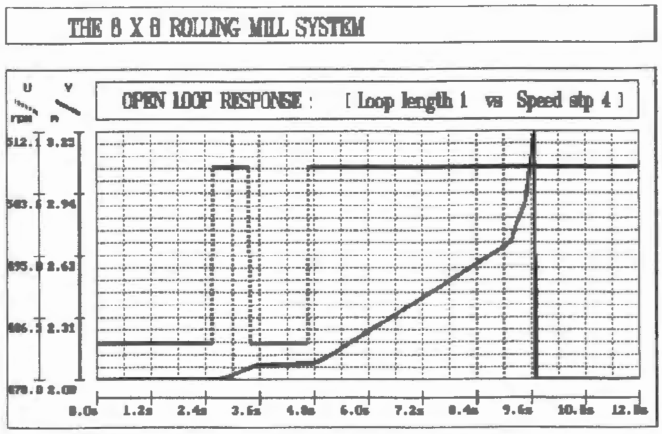


Figure K.25: The response of L1 to a step in S4.

From figure K.25:

Change in output $dy := 2.68 - 2.079 = 0.603$ [m]

Time period $dt := 4.38$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = 0.14 \text{ [m/s]}$$

The step in S4 $:= 24.15$ [rpm]

Therefore since

$$G_{44} := \frac{slope}{step}$$

$$G_{44}(s) := \frac{0.006}{s} \text{ [m/rpm]}$$

Metal in Stands 3-8.

Transfer Function G_{33}.

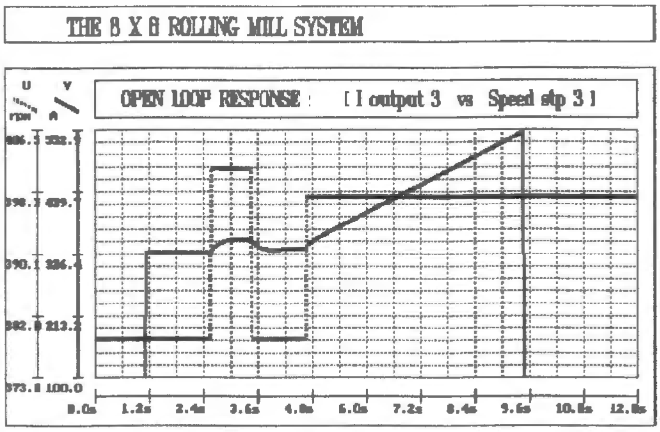


Figure K.26: The response of A3 to a step in S3.

From figure K.26:

Change in output $dy := 548.0 - 328.3 = 219.7$ [A]

Time period $dt := 4.76$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 46.16 \text{ [A/s]}$$

The step in S3 $:= 18.94$ [rpm]

Therefore since

$$G_{33} := \frac{\text{slope}}{\text{step}}$$

$$G_{33}(s) := \frac{2.44}{s} \text{ [A/rpm]}$$

Transfer Function G_{43}.

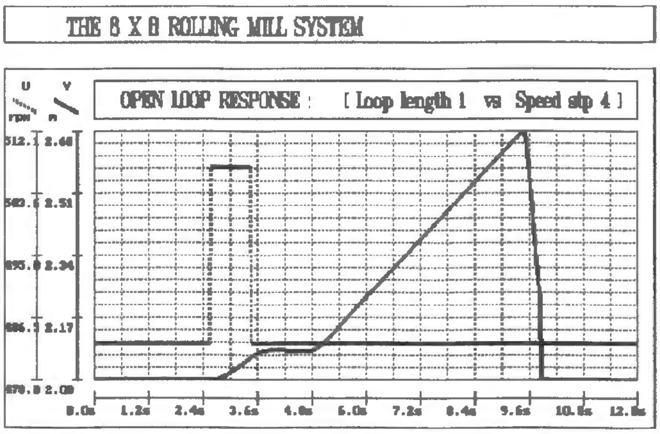


Figure K.27: The response of L1 to a step in S3.

From figure K.27:

Change in output $dy := 2.69 - 2.079 = 0.611$ [m]
Time period $dt := 4.76$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.13 \text{ [m/s]}$$

The step in S3 := 18.94 [rpm]

Therefore since

$$G_{43} := \frac{\text{slope}}{\text{step}}$$

$$G_{43}(s) := \frac{0.007}{s} \text{ [m/rpm]}$$

Transfer Function G_{34}.

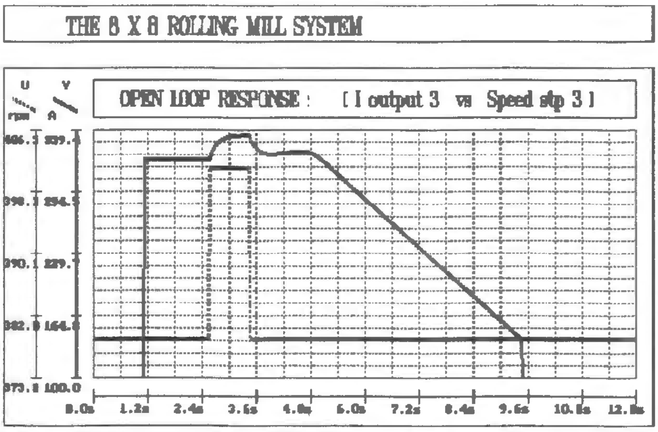


Figure K.28: The response of A3 to a step in S4.

From figure K.28:

Change in output $dy := 139.96 - 328.3 = -188.34$ [A]

Time period $dt := 4.76$ [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = -39.57 \text{ [A/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{34} := \frac{slope}{step}$$

$$G_{34}(s) := \frac{-1.64}{s} \text{ [A/rpm]}$$

Transfer Function G_{44}.

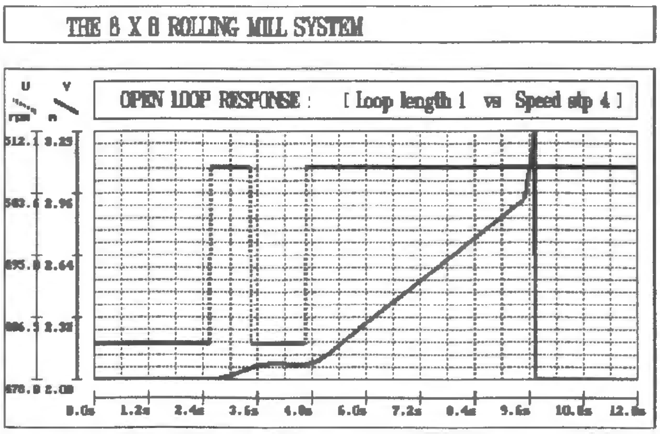


Figure K.29: The response of L1 to a step in S4.

From figure K.29:

Change in output $dy := 2.937 - 2.079 = 0.858$ [m]

Time period $dt := 4.76$ [seconds]

Therefore:

$$\text{Slope} := \frac{dy}{dt} = 0.18 \text{ [m/s]}$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{44} := \frac{\text{slope}}{\text{step}}$$

$$G_{44}(s) := \frac{0.008}{s} \text{ [m/rpm]}$$

Metal in Stands 4-8.

Transfer Function G_{44}.

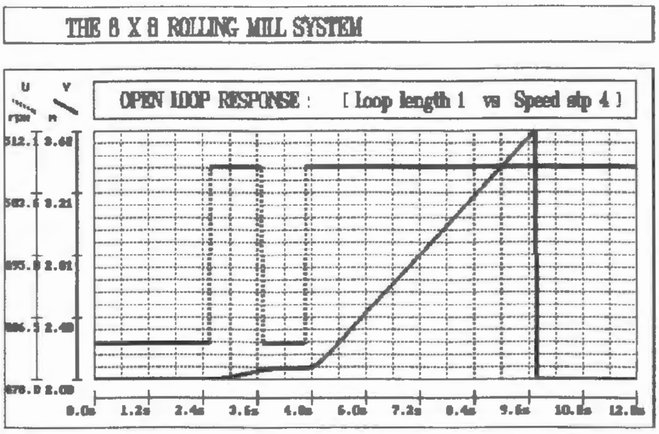


Figure K.30: The response of L1 to a step in S4.

From figure K.30:

Change in output dy := 3.62 - 2.079 = 1.541 [m]

Time period dt := 5.12 [seconds]

Therefore:

$$Slope := \frac{dy}{dt} = 0.3 \ [m/s]$$

The step in S4 := 24.15 [rpm]

Therefore since

$$G_{44} := \frac{slope}{step}$$

$$G_{44}(s) := \frac{0.012}{s} \ [m/rpm]$$

Metal in Stands 5-8.

The transfer function equations describing this situation are those given in Chapter 5. The active transfer functions are described in Chapter 4.

Metal in Stands 6-8.

The transfer function equations describing this situation are those given in Chapter 5. The active transfer functions are described in Chapter 4.

Metal in Stands 7&8.

The transfer function equations describing this situation are those given in Chapter 5. The active transfer functions are described in Chapter 4.

Metal in Stand 8.

The transfer function equations describing this situation are those given in Chapter 5. The active transfer functions are described in Chapter 4.

APPENDIX L: INDENTIFICATION SOFTWARE LISTING

```
PROGRAM ident;

{$M 65520,0,655360}

USES
    dos,crt,graph;

CONST
     max_point = 600;

     valid0 : set of '0'..'Z' = ['1','2','X'];

TYPE
    data_var = record
     u : array [0..max_point] of double;
     y : array [0..max_point] of double;
     END;
    data_ptr = ^data_var;

    labl_var = record
     head1 : string;
     head2 : string;
     utitl : string;
     ytitl : string;
     uname : string;
     yname : string;
     ulabl : string;
     ylabl : string;
     xlabl : string;
     END;
    labl_ptr = ^labl_var;

VAR
   datclc : data_ptr;
   datlog : data_ptr;
   lables : labl_ptr;

   search : pathstr;
   inputname : string;
   inputfile : text;

   maxpts : integer;
   dx     : double;

   x0,x1       : real;
   p1,p2,p3,p4 : real;
   z1,z2,z3,z4 : real;
```

```
   exit0  : boolean;
   ch0    : char;



PROCEDURE data_plot (d:data_ptr;l:labl_ptr);
VAR
   y_maxi,y_mini : real;
   y_maxo,y_mino : real;
   i_fact,o_fact : real;
   i_diff,o_diff : real;
   x_pos,y_pos   : integer;
   i,gm,gd : integer;
   s : string;
BEGIN
 gd := detect;
 initgraph (gd,gm,'');
 setviewport (0,0,getmaxx,getmaxy,clipon);clearviewport;

 setviewport  (0,0,719,29,clipon);rectangle (0,0,719,29);
 settextstyle (triplexfont,horizdir,2);
 outtextxy    (30,5,l^.head1);
 setviewport  (0,50,719,339,clipon);rectangle (0,0,719,289);

 { Heading }
 setlinestyle (solidln,1,normwidth);
 setviewport  (100,60,700,90,clipon);clearviewport;rectangle (0,0,600,30);
 settextstyle (triplexfont,horizdir,1);
 outtextxy (30,5,l^.head2+' :    [ '+l^.ytitl+'  vs  '+l^.utitl+' ]');

 { Determine y_maxi,y_mini,y_maxo,y_mino }
 y_maxi := 0;
 y_mini := 10000;
 y_maxo := 0;
 y_mino := 10000;
 for i := 0 to maxpts do
  BEGIN
   if d^.u[i] > y_maxi then y_maxi := d^.u[i];
   if d^.u[i] < y_mini then y_mini := d^.u[i];
   if d^.y[i] > y_maxo then y_maxo := d^.y[i];
   if d^.y[i] < y_mino then y_mino := d^.y[i];
  END;

 i_diff := y_maxi - y_mini;
 i_fact := 1E-12;
 REPEAT i_fact := i_fact*10;
 UNTIL (i_diff*i_fact > 1);
 y_maxi := (trunc(y_maxi*i_fact)+1.0)/i_fact;
 y_mini := (trunc(y_mini*i_fact)+0.0)/i_fact;

 o_diff := y_maxo - y_mino;
 o_fact := 1E-12;
 REPEAT o_fact := o_fact*10;
 UNTIL (o_diff*o_fact > 1);
```

```
y_maxo := (trunc(y_maxo*o_fact)+1.0)/o_fact;
y_mino := (trunc(y_mino*o_fact)+0.0)/o_fact;

{ Plot x-axis }
setviewport  (100,305,718,340,clipoff);
settextstyle (defaultfont,horizdir,1);
line (0,10,600,10);
for i := 0 to 10 do
 BEGIN
  line (i*60,5,i*60,15);
  str (i/10*dx*maxpts:3:1,s);outtextxy (i*60-30,17,s+l^.xlabl);
 END;

{ Plot y-axis : input }
setviewport  (1,60,45,320,clipon);clearviewport;
settextstyle (defaultfont,horizdir,1);
outtextxy    (20,15,l^.uname);
outtextxy    (5,25,l^.ulabl);
line (35,40,35,240);
for i := 0 to 4 do
 BEGIN
  line (30,40+i*50,40,40+i*50);
  str  (y_maxi-i*0.25*(y_maxi-y_mini):4:2,s);
  outtextxy ( 0,45+i*50,s);
 END;

{ Plot y-axis : output }
setviewport  (46,60,90,320,clipon);clearviewport;
settextstyle (defaultfont,horizdir,1);
outtextxy    (20,15,l^.yname);
outtextxy    (5,25,l^.ylabl);
line (35,40,35,240);
for i := 0 to 4 do
 BEGIN
  line (30,40+i*50,40,40+i*50);
  str  (y_maxo-i*0.25*(y_maxo-y_mino):4:2,s);
  outtextxy ( 0,45+i*50,s);
 END;

{ Plot graph }
setviewport  (100,100,700,300,clipon);clearviewport;
rectangle (0,0,600,200);

{ Plot grids }
setlinestyle (dottedln,1,normwidth);
for i := 1 to 19 do line (0,10*i,600,10*i);
for i := 1 to 19 do line (i*30,0,i*30,200);

{ Plot input }
x_pos := 0;
y_pos := 200 - round(200*((d^.u[0]-y_mini)/(y_maxi-y_mini)));
moveto (x_pos,y_pos);
setlinestyle (userbitln,$9999,thickwidth);
for i := 1 to maxpts do
```

```
  BEGIN
   x_pos := i * (600 div maxpts);
   y_pos := 200 - round(200*((d^.u[i]-y_mini)/(y_maxi-y_mini)));
   lineto (x_pos,y_pos);
  END;

 { Plot output }
 x_pos := 0;
 y_pos := 200 - round(200*((d^.y[0]-y_mino)/(y_maxo-y_mino)));
 moveto (x_pos,y_pos);
 setlinestyle (userbitln,$FFFF,thickwidth);
 for i := 1 to maxpts do
  BEGIN
   x_pos := i * (600 div maxpts);
   y_pos := 200 - round(200*((d^.y[i]-y_mino)/(y_maxo-y_mino)));
   lineto (x_pos,y_pos);
  END;

 REPEAT
 UNTIL keypressed;

 setviewport (0,0,getmaxx,getmaxy,clipon);clearviewport;
 closegraph;
END;

{**************************************************}
{*****   PROCEDURE TO FIT DESIRED FUNCTION   *****}
{**************************************************}
PROCEDURE func_fit (ordr,tipe:char);
VAR
   loop : integer;
   tstp : integer;
   step : real;
   mean : real;

   FUNCTION estimate (t:integer) : real;
   BEGIN
    CASE ordr OF
    '1' : CASE tipe OF
          '1' : estimate := step * x1*(1-exp(-p1*(t*dx)));
          '2' : estimate := step * x1*(1-(1-p1/z1)*exp(-p1*(t*dx)));
          '3' : estimate := step * x1*(1-(1+p1/z1)*exp(-p1*(t*dx)));
          '4' : estimate := step * x1*(t*dx);
          END;
    '2' : CASE tipe OF
          '1' : estimate := step * x1*(1-(p2*exp(-p1*(t*dx))-p1*exp(-p2*(t*dx)))/(p2-p1));
          '2' : estimate := step * x1*(1-sqrt(p1*p1+p2*p2*pi*pi)/(p2*pi)*exp(-
p1*(t*dx))*sin(p2*pi*(t*dx)-arctan(-p2*pi/p1)));
          '3' : estimate := step * x1*(1-(1+p1*(t*dx))*exp(-p1*(t*dx)));
          '4' : estimate := step * x1*(1-p2/z1*(p1-z1)/(p1-p2)*exp(-p1*(t*dx))-p1/z1*(p2-z1)/(p2-
p1)*exp(-p2*(t*dx)));
          END;
    END;
   END;
```

```
BEGIN
 step := (datlog^.u[maxpts] - datlog^.u[0]);
 tstp := 0;
 REPEAT
  inc (tstp);
 UNTIL (abs(datlog^.u[maxpts]-datlog^.u[tstp]) < abs(step/3));
 mean := 0;
 FOR loop := 0 to tstp+round(x0/dx) DO
  BEGIN
   mean := mean + datlog^.y[loop];
  END;
 mean := mean / (tstp+round(x0/dx)+1);

 FOR loop := 0 to tstp+round(x0/dx) DO
  BEGIN
   datclc^.u[loop] := mean;
   datclc^.y[loop] := datlog^.y[loop];
  END;
 FOR loop := tstp+round(x0/dx)+1 to maxpts DO
  BEGIN
   datclc^.u[loop] := mean + estimate (loop-(tstp+round(x0/dx)));
   datclc^.y[loop] := datlog^.y[loop];
  END;
END;

{***********************************************}
{*****      1 : FIRST ORDER RESPONSES      *****}
{***********************************************}
PROCEDURE order_1;
CONST
     valid1 : set of '0'..'Z' = ['1','2','3','4','X'];
VAR
   exit1 : boolean;
   ch1   : char;

 {***********************************************}
 {*****   1.1 : FIRST ORDER                *****}
 {*****           Real Poles : α           *****}
 {*****           Comp Poles : ---         *****}
 {*****           LHP  Zeros : ---         *****}
 {*****           RHP  Zeros : ---         *****}
 {***********************************************}
 PROCEDURE selct_1;
 CONST
      valid11 : set of '0'..'Z' = ['1','2','3','P','D','X'];
 VAR
    exit11 : boolean;
    ch11   : char;
 BEGIN
  lables^.head1 := 'ROLLING MILL PLANT : FIRST ORDER DATA FIT';
  lables^.util1 := lables^.ytitl;
  lables^.ulabl := lables^.ylabl;
  lables^.uname := lables^.yname;
```

```
x0 := 0;
x1 := 1;
p1 := 1;

exitll := false;
REPEAT
 func_fit (ch0,ch1);

 clrscr;
 gotoxy (20,01);write ('┌────────────────────────────┐');
 gotoxy (20,02);write ('│ 1  :  FIRST ORDER RESPONSE │');
 gotoxy (20,03);write ('│                            │');
 gotoxy (20,04);write ('│         ┌   α   ┐  -sτ    │');
 gotoxy (20,05);write ('│ 1  :  K │───────│ e        │');
 gotoxy (20,06);write ('│         └ s + α ┘          │');
 gotoxy (20,07);write ('└────────────────────────────┘');
 gotoxy (20,09);write ('        ┌────────────────────┐');
 gotoxy (20,10);write ('        │     1  :  K =      │');
 gotoxy (20,11);write ('        │     2  :  α =      │');
 gotoxy (20,12);write ('        │     3  :  τ =      │');
 gotoxy (20,13);write ('        │                    │');
 gotoxy (20,14);write ('        │     P  :  Plot     │');
 gotoxy (20,15);write ('        │     D  :  Display  │');
 gotoxy (20,16);write ('        │     X  :  Exit     │');
 gotoxy (20,17);write ('        └────────────────────┘');
 gotoxy (20,23);write ('        ┌────────────────────┐');
 gotoxy (20,24);write ('        │      OPTION :      │');
 gotoxy (20,25);write ('        └────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch11 := upcase(readkey);
 UNTIL ch11 in valid11;
 CASE ch11 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (x0);
       END;
 'P' : data_plot (datclc,lables);
 'D' : BEGIN
        clrscr;

        writeln ('NUMERATOR');
        IF (x0 = 0) THEN
```

```
          BEGIN
           writeln ('s^0 = ',(x1*p1):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*x1*p1/x0):20:12);
           writeln ('s^1 = ',(-x1*p1):20:12);
          END;
         writeln;

         writeln ('DENOMINATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(p1):20:12);
           writeln ('s^1 = ',(1.0):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*p1/x0):20:12);
           writeln ('s^1 = ',(p1+2/x0):20:12);
           writeln ('s^2 = ',(1.0):20:12);
          END;
         writeln;

         writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
         REPEAT UNTIL keypressed;
        END;
  'X' : exit11 := true;
  END;
 UNTIL exit11;
END;

{**************************************************}
{*****   1.2 : FIRST ORDER                    *****}
{*****          Real Poles : α                *****}
{*****          Comp Poles : ---              *****}
{*****          LHP  Zeros : a                *****}
{*****          RHP  Zeros : ---              *****}
{**************************************************}
PROCEDURE selct_2;
CONST
     valid12 : set of '0'..'Z' = ['1','2','3','4','P','D','X'];
VAR
   exit12 : boolean;
   ch12   : char;
BEGIN
 lables^.head1 := 'ROLLING MILL PLANT : FIRST ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
 lables^.uname := lables^.yname;
 x0 := 0;
 x1 := 1;
 p1 := 1;
 z1 := 1;
```

```
exit12 := false;
REPEAT
 func_fit (ch0,ch1);

 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────┐');
 gotoxy (20,02);write ('│ 1  :  FIRST ORDER RESPONSE   │');
 gotoxy (20,03);write ('│                              │');
 gotoxy (20,04);write ('│          ┌ α   s + a ┐  -sτ  │');
 gotoxy (20,05);write ('│ 2  : K  │ -   ─────── │ e    │');
 gotoxy (20,06);write ('│          └ a   s + α ┘       │');
 gotoxy (20,07);write ('└──────────────────────────────┘');
 gotoxy (20,09);write ('        ┌──────────────────────┐');
 gotoxy (20,10);write ('        │      1  :  K =       │');
 gotoxy (20,11);write ('        │      2  :  α =       │');
 gotoxy (20,12);write ('        │      3  :  a =       │');
 gotoxy (20,13);write ('        │      4  :  τ =       │');
 gotoxy (20,14);write ('        │                      │');
 gotoxy (20,15);write ('        │      P  :  Plot      │');
 gotoxy (20,16);write ('        │      D  :  Display   │');
 gotoxy (20,17);write ('        │      X  :  Exit      │');
 gotoxy (20,18);write ('        └──────────────────────┘');
 gotoxy (20,23);write ('        ┌──────────────────────┐');
 gotoxy (20,24);write ('        │      OPTION :        │');
 gotoxy (20,25);write ('        └──────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (z1:10:6);
 gotoxy (44,13);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch12 := upcase(readkey);
 UNTIL ch12 in valid12;
 CASE ch12 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (z1);
       END;
 '4' : BEGIN
        gotoxy (44,13);write ('          ');
        gotoxy (44,13);readln (x0);
       END;
 'P' : data_plot (datclc,lables);
 'D' : BEGIN
        clrscr;
```

```
        writeln ('NUMERATOR');
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(x1*p1):20:12);
          writeln ('s^1 = ',(x1*p1/z1):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*x1*p1/x0):20:12);
          writeln ('s^1 = ',(x1*p1*(2/(z1*x0)-1)):20:12);
          writeln ('s^2 = ',(-x1*p1/z1):20:12);
         END;
        writeln;

        writeln ('DENOMINATOR');
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(p1):20:12);
          writeln ('s^1 = ',(1.0):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*p1/x0):20:12);
          writeln ('s^1 = ',(p1+2/x0):20:12);
          writeln ('s^2 = ',(1.0):20:12);
         END;
        writeln;

        writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
        REPEAT UNTIL keypressed;
       END;
  'X' : exit12 := true;
  END;
 UNTIL exit12;
END;

{************************************************}
{*****   1.3 : FIRST ORDER                  *****}
{*****          Real Poles : α              *****}
{*****          Comp Poles : ---            *****}
{*****          LHP  Zeros : ---            *****}
{*****          RHP  Zeros : a              *****}
{************************************************}
PROCEDURE selct_3;
CONST
     valid13 : set of '0'..'Z' = ['1','2','3','4','P','D','X'];
VAR
   exit13 : boolean;
   ch13   : char;
BEGIN
 lables^.head1 := 'ROLLING MILL PLANT : FIRST ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
```

```
lables^.uname := lables^.yname;
x0 := 0;
x1 := 1;
p1 := 1;
z1 := 1;

exit13 := false;
REPEAT
 func_fit (ch0,ch1);

 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────┐');
 gotoxy (20,02);write ('│ 1 :  FIRST ORDER RESPONSE    │');
 gotoxy (20,03);write ('│                              │');
 gotoxy (15,04);write ('│        ┌ α   s - a ┐  -sτ    │');
 gotoxy (15,05);write ('│ 3 : K  │ - -  ─────│ e       │');
 gotoxy (15,06);write ('│        └ a   s + α ┘         │');
 gotoxy (20,07);write ('└──────────────────────────────┘');
 gotoxy (20,09);write ('         ┌────────────────────┐');
 gotoxy (20,10);write ('         │    1  :  K =       │');
 gotoxy (20,11);write ('         │    2  :  α =       │');
 gotoxy (20,12);write ('         │    3  :  a =       │');
 gotoxy (20,13);write ('         │    4  :  τ =       │');
 gotoxy (20,14);write ('         │                    │');
 gotoxy (20,15);write ('         │    P  :  Plot      │');
 gotoxy (20,16);write ('         │    D  :  Display   │');
 gotoxy (20,17);write ('         │    X  :  Exit      │');
 gotoxy (20,18);write ('         └────────────────────┘');
 gotoxy (20,23);write ('         ┌────────────────────┐');
 gotoxy (20,24);write ('         │     OPTION :       │');
 gotoxy (20,25);write ('         └────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (z1:10:6);
 gotoxy (44,13);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch13 := upcase(readkey);
 UNTIL ch13 in valid13;
 CASE ch13 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (z1);
       END;
 '4' : BEGIN
        gotoxy (44,13);write ('          ');
```

```
        gotoxy (44,13);readln (x0);
       END;
  'P' : data_plot (datclc,lables);
  'D' : BEGIN
        clrscr;

        writeln ('NUMERATOR');
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(x1*p1):20:12);
          writeln ('s^1 = ',(-x1*p1/z1):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*x1*p1/x0):20:12);
          writeln ('s^1 = ',(-x1*p1*(2/(z1*x0)+1)):20:12);
          writeln ('s^2 = ',(x1*p1/z1):20:12);
         END;
        writeln;

        writeln ('DENOMINATOR');
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(p1):20:12);
          writeln ('s^1 = ',(1.0):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*p1/x0):20:12);
          writeln ('s^1 = ',(p1+2/x0):20:12);
          writeln ('s^2 = ',(1.0):20:12);
         END;
        writeln;

        writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
        REPEAT UNTIL keypressed;
       END;
  'X' : exit13 := true;
 END;
 UNTIL exit13;
END;

{************************************************}
{*****   1.4 : FIRST ORDER                  *****}
{*****          Real Poles : 0              *****}
{*****          Comp Poles : ---            *****}
{*****          LHP  Zeros : ---            *****}
{*****          RHP  Zeros : ---            *****}
{************************************************}
PROCEDURE selct_4;
CONST
    valid14 : set of '0'..'Z' = ['1','2','P','D','X'];
VAR
   exit14 : boolean;
```

```
   ch14   : char;
BEGIN
 lables^.head1 := 'ROLLING MILL PLANT : FIRST ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
 lables^.uname := lables^.yname;
 x0 := 0;
 x1 := 1;

 exit14 := false;
 REPEAT
  func_fit (ch0,ch1);

  clrscr;
  gotoxy (20,01);write ('┌──────────────────────────────┐');
  gotoxy (20,02);write ('│ 1 :  FIRST ORDER RESPONSE    │');
  gotoxy (20,03);write ('│                              │');
  gotoxy (15,04);write ('│          ┌ 1 ┐  -sτ          │');
  gotoxy (15,05);write ('│ 4 : K    │ - │ e             │');
  gotoxy (15,06);write ('│          └ s ┘               │');
  gotoxy (20,07);write ('└──────────────────────────────┘');
  gotoxy (20,09);write ('        ┌──────────────────────┐');
  gotoxy (20,10);write ('        │     1  :  K =        │');
  gotoxy (20,11);write ('        │     2  :  τ =        │');
  gotoxy (20,14);write ('        │                      │');
  gotoxy (20,15);write ('        │     P  :  Plot       │');
  gotoxy (20,16);write ('        │     D  :  Display    │');
  gotoxy (20,17);write ('        │     X  :  Exit       │');
  gotoxy (20,18);write ('        └──────────────────────┘');
  gotoxy (20,23);write ('        ┌──────────────────────┐');
  gotoxy (20,24);write ('        │      OPTION :        │');
  gotoxy (20,25);write ('        └──────────────────────┘');
  gotoxy (44,10);write (x1:10:6);
  gotoxy (44,11);write (x0:10:6);
  REPEAT
   gotoxy (44,24);
   ch14 := upcase(readkey);
  UNTIL ch14 in valid14;
  CASE ch14 OF
  '1' : BEGIN
         gotoxy (44,10);write ('          ');
         gotoxy (44,10);readln (x1);
        END;
  '2' : BEGIN
         gotoxy (44,11);write ('          ');
         gotoxy (44,11);readln (x0);
        END;
  'P' : data_plot (datclc,lables);
  'D' : BEGIN
         clrscr;

         writeln ('NUMERATOR');
         IF (x0 = 0) THEN
          BEGIN
```

```
            writeln ('s^0 = ',(x1):20:12);
           END
          ELSE
           BEGIN
            writeln ('s^0 = ',(2*x1/x0):20:12);
            writeln ('s^1 = ',(-x1):20:12);
           END;
          writeln;

          writeln ('DENOMINATOR');
          IF (x0 = 0) THEN
           BEGIN
            writeln ('s^0 = ',(0.0):20:12);
            writeln ('s^1 = ',(1.0):20:12);
           END
          ELSE
           BEGIN
            writeln ('s^0 = ',(0.0):20:12);
            writeln ('s^1 = ',(2/x0):20:12);
            writeln ('s^2 = ',(1.0):20:12);
           END;
          writeln;

          writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
          REPEAT UNTIL keypressed;
         END;
   'X' : exit14 := true;
   END;
  UNTIL exit14;
 END;

BEGIN
 exit1 := false;
 REPEAT
  clrscr;
  gotoxy (15,01);write ('┌─────────────────────────────────┐');
  gotoxy (15,02);write ('│ 1  :  FIRST ORDER RESPONSE      │');
  gotoxy (15,03);write ('└─────────────────────────────────┘');
  gotoxy (15,06);write ('┌─────────────────────────────────┐');
  gotoxy (15,07);write ('│           ┌  α  ┐  -sτ          │');
  gotoxy (15,08);write ('│1 :  K     │─────│ e             │');
  gotoxy (15,09);write ('│           └ s + α ┘             │');
  gotoxy (15,10);write ('│           ┌ α   s + a ┐  -sτ    │');
  gotoxy (15,11);write ('│2 :  K     │ -   ───── │ e       │');
  gotoxy (15,12);write ('│           └ a   s + α ┘         │');
  gotoxy (15,13);write ('│           ┌ α   s - a ┐  -sτ    │');
  gotoxy (15,14);write ('│3 :  K     │- -  ───── │ e       │');
  gotoxy (15,15);write ('│           └ a   s + α ┘         │');
  gotoxy (15,16);write ('│           ┌ 1 ┐  -sτ            │');
  gotoxy (15,17);write ('│4 :  K     │ ─ │ e               │');
  gotoxy (15,18);write ('│           └ s ┘                 │');
  gotoxy (15,19);write ('│                                 │');
  gotoxy (15,20);write ('│X :  Exit                        │');
  gotoxy (15,21);write ('└─────────────────────────────────┘');
```

```
  gotoxy (15,23);write ('      ┌──────────────────────────┐');
  gotoxy (15,24);write ('      │      OPTION :            │');
  gotoxy (15,25);write ('      └──────────────────────────┘');
  REPEAT
   gotoxy (37,24);
   ch1 := upcase(readkey);
  UNTIL ch1 in valid1;
  CASE ch1 OF
  '1' : selct_1;
  '2' : selct_2;
  '3' : selct_3;
  '4' : selct_4;
  'X' : exit1 := true;
  END;
 UNTIL exit1;
END;

{************************************************}
{*****      2 : FIRST ORDER RESPONSES       *****}
{************************************************}
PROCEDURE order_2;
CONST
     valid2 : set of '0'..'Z' = ['1','2','3','4','X'];
VAR
   exit2 : boolean;
   ch2   : char;

 {***********************************************}
 {*****   2.1 : SECOND ORDER              *****}
 {*****           Real Poles : α,β        *****}
 {*****           Comp Poles : ---        *****}
 {*****           LHP  Zeros : ---        *****}
 {*****           RHP  Zeros : ---        *****}
 {***********************************************}
 PROCEDURE selct_1;
 CONST
      valid21 : set of '0'..'Z' = ['1','2','3','4','P','D','X'];
 VAR
    exit21 : boolean;
    ch21   : char;
 BEGIN
  lables^.head1 := 'ROLLING MILL PLANT : SECOND ORDER DATA FIT';
  lables^.utitl := lables^.ytitl;
  lables^.ulabl := lables^.ylabl;
  lables^.uname := lables^.yname;
  x0 := 0;
  x1 := 1;
  p1 := 0.5;
  p2 := 1.0;

  exit21 := false;
  REPEAT
   func_fit (ch0,ch2);
```

```
clrscr;
gotoxy (20,01);write ('┌─────────────────────────────────────┐');
gotoxy (20,02);write ('│ 2 :  SECOND ORDER RESPONSE          │');
gotoxy (20,03);write ('│                                     │');
gotoxy (20,04);write ('│          ┌      αβ       ┐ -sτ      │');
gotoxy (20,05);write ('│1: K      │───────────────│e         │');
gotoxy (20,06);write ('│          └ (s + α)(s + β)┘          │');
gotoxy (20,07);write ('└─────────────────────────────────────┘');
gotoxy (20,09);write ('             ┌───────────────────────┐');
gotoxy (20,10);write ('             │      1  :  K =        │');
gotoxy (20,11);write ('             │      2  :  α =        │');
gotoxy (20,12);write ('             │      3  :  β =        │');
gotoxy (20,13);write ('             │      4  :  τ =        │');
gotoxy (20,14);write ('             │                       │');
gotoxy (20,15);write ('             │      P  :  Plot       │');
gotoxy (20,16);write ('             │      D  :  Display    │');
gotoxy (20,17);write ('             │      X  :  Exit       │');
gotoxy (20,18);write ('             └───────────────────────┘');
gotoxy (20,23);write ('             ┌───────────────────────┐');
gotoxy (20,24);write ('             │       OPTION :        │');
gotoxy (20,25);write ('             └───────────────────────┘');
gotoxy (44,10);write (x1:10:6);
gotoxy (44,11);write (p1:10:6);
gotoxy (44,12);write (p2:10:6);
gotoxy (44,13);write (x0:10:6);
REPEAT
 gotoxy (44,24);
 ch21 := upcase(readkey);
UNTIL ch21 in valid21;
CASE ch21 OF
'1' : BEGIN
       gotoxy (44,10);write ('          ');
       gotoxy (44,10);readln (x1);
      END;
'2' : BEGIN
       gotoxy (44,11);write ('          ');
       gotoxy (44,11);readln (p1);
      END;
'3' : BEGIN
       gotoxy (44,12);write ('          ');
       gotoxy (44,12);readln (p2);
      END;
'4' : BEGIN
       gotoxy (44,13);write ('          ');
       gotoxy (44,13);readln (x0);
      END;
'P' : data_plot (datclc,lables);
'D' : BEGIN
       clrscr;

       writeln ('NUMERATOR');
       IF (x0 = 0) THEN
        BEGIN
         writeln ('s^0 = ',(x1*p1*p2):20:12);
```

```
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*x1*p1*p2/x0):20:12);
           writeln ('s^1 = ',(-x1*p1*p2):20:12);
          END;
         writeln;

         writeln ('DENOMINATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(p1*p2):20:12);
           writeln ('s^1 = ',(p1+p2):20:12);
           writeln ('s^2 = ',(1.0):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*p1*p2/x0):20:12);
           writeln ('s^1 = ',(p1*p2+2*(p1+p2)/x0):20:12);
           writeln ('s^2 = ',(p1+p2+2/x0):20:12);
           writeln ('s^3 = ',(1.0):20:12);
          END;
         writeln;

         writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
         REPEAT UNTIL keypressed;
        END;
  'X' : exit21 := true;
  END;
 UNTIL exit21;
END;

{***********************************************}
{*****   2.2 : SECOND ORDER               *****}
{*****          Real Poles : α ± jß       *****}
{*****          Comp Poles : ---          *****}
{*****          LHP  Zeros : ---          *****}
{*****          RHP  Zeros : ---          *****}
{***********************************************}
PROCEDURE selct_2;
CONST
     valid22 : set of '0'..'Z' = ['1','2','3','4','P','D','X'];
VAR
   exit22 : boolean;
   ch22   : char;
BEGIN
 lables^.head1 := 'ROLLING MILL PLANT : SECOND ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
 lables^.uname := lables^.yname;
 x0 := 0;
 x1 := 1;
 p1 := 1;
 p2 := 1;
```

```
exit22 := false;
REPEAT
 func_fit (ch0,ch2);

 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────────┐');
 gotoxy (20,02);write ('│ 2 :  SECOND ORDER RESPONSE       │');
 gotoxy (20,03);write ('│                                  │');
 gotoxy (20,04);write ('│       ┌      α²+β²      ┐   -sτ  │');
 gotoxy (20,05);write ('│2: K   │─────────────────│  e     │');
 gotoxy (20,06);write ('│       └ (s + α)² + β²   ┘        │');
 gotoxy (20,07);write ('└──────────────────────────────────┘');
 gotoxy (20,09);write ('         ┌─────────────────────────┐');
 gotoxy (20,10);write ('         │      1  :  K =          │');
 gotoxy (20,11);write ('         │      2  :  α =          │');
 gotoxy (20,12);write ('         │      3  :  β =          │');
 gotoxy (20,13);write ('         │      4  :  τ =          │');
 gotoxy (20,14);write ('         │                         │');
 gotoxy (20,15);write ('         │      P  :  Plot         │');
 gotoxy (20,16);write ('         │      D  :  Display      │');
 gotoxy (20,17);write ('         │      X  :  Exit         │');
 gotoxy (20,18);write ('         └─────────────────────────┘');
 gotoxy (20,23);write ('         ┌─────────────────────────┐');
 gotoxy (20,24);write ('         │       OPTION :          │');
 gotoxy (20,25);write ('         └─────────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (p2:10:6,'π');
 gotoxy (44,13);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch22 := upcase(readkey);
 UNTIL ch22 in valid22;
 CASE ch22 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (p2);
       END;
 '4' : BEGIN
        gotoxy (44,13);write ('          ');
        gotoxy (44,13);readln (x0);
       END;
 'P' : data_plot (datclc,lables);
 'D' : BEGIN
        clrscr;
```

```
         writeln ('NUMERATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(x1*(p1*p1+p2*p2*pi*pi)):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*x1*(p1*p1+p2*p2*pi*pi)/x0):20:12);
           writeln ('s^1 = ',(-x1*(p1*p1+p2*p2*pi*pi)):20:12);
          END;
         writeln;

         writeln ('DENOMINATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(p1*p1+p2*p2*pi*pi):20:12);
           writeln ('s^1 = ',(2*p1):20:12);
           writeln ('s^2 = ',(1.0):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*(p1*p1+p2*p2*pi*pi)/x0):20:12);
           writeln ('s^1 = ',(p1*p1+p2*p2*pi*pi+4*p1/x0):20:12);
           writeln ('s^2 = ',(2*p1+2/x0):20:12);
           writeln ('s^3 = ',(1.0):20:12);
          END;
         writeln;

         writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
         REPEAT UNTIL keypressed;
        END;
   'X' : exit22 := true;
   END;
  UNTIL exit22;
END;

{************************************************}
{*****   2.3 : SECOND ORDER                *****}
{*****           Real Poles : α , α        *****}
{*****           Comp Poles : ---          *****}
{*****           LHP  Zeros : ---          *****}
{*****           RHP  Zeros : ---          *****}
{************************************************}
PROCEDURE selct_3;
CONST
     valid23 : set of '0'..'Z' = ['1','2','3','P','D','X'];
VAR
   exit23 : boolean;
   ch23   : char;
BEGIN
 lables^.headl := 'ROLLING MILL PLANT : SECOND ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
```

```
lables^.uname := lables^.yname;
x0 := 0;
x1 := 1;
p1 := 1;

exit23 := false;
REPEAT
 func_fit (ch0,ch2);

 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────┐');
 gotoxy (20,02);write ('│ 2 :  SECOND ORDER RESPONSE   │');
 gotoxy (20,03);write ('│                              │');
 gotoxy (20,04);write ('│       ┌       α²      ┐  -sτ │');
 gotoxy (20,05);write ('│3: K   │───────────────│ e    │');
 gotoxy (20,06);write ('│       └    (s + α)²   ┘      │');
 gotoxy (20,07);write ('└──────────────────────────────┘');
 gotoxy (20,09);write ('        ┌──────────────────────┐');
 gotoxy (20,10);write ('        │      1  :  K =       │');
 gotoxy (20,11);write ('        │      2  :  α =       │');
 gotoxy (20,12);write ('        │      3  :  τ =       │');
 gotoxy (20,13);write ('        │                      │');
 gotoxy (20,14);write ('        │      P  :  Plot      │');
 gotoxy (20,15);write ('        │      D  :  Display   │');
 gotoxy (20,16);write ('        │      X  :  Exit      │');
 gotoxy (20,17);write ('        └──────────────────────┘');
 gotoxy (20,23);write ('        ┌──────────────────────┐');
 gotoxy (20,24);write ('        │       OPTION :       │');
 gotoxy (20,25);write ('        └──────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch23 := upcase(readkey);
 UNTIL ch23 in valid23;
 CASE ch23 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (x0);
       END;
 'P' : data_plot (datclc,lables);
 'D' : BEGIN
        clrscr;

        writeln ('NUMERATOR');
```

```
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(x1*p1*p1):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*x1*p1*p1/x0):20:12);
          writeln ('s^1 = ',(-x1*p1*p1):20:12);
         END;
        writeln;

        writeln ('DENOMINATOR');
        IF (x0 = 0) THEN
         BEGIN
          writeln ('s^0 = ',(p1*p1):20:12);
          writeln ('s^1 = ',(2*p1):20:12);
          writeln ('s^2 = ',(1.0):20:12);
         END
        ELSE
         BEGIN
          writeln ('s^0 = ',(2*p1*p1/x0):20:12);
          writeln ('s^1 = ',(p1*p1+4*p1/x0):20:12);
          writeln ('s^2 = ',(2*p1+2/x0):20:12);
          writeln ('s^3 = ',(1.0):20:12);
         END;
        writeln;

        writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
        REPEAT UNTIL keypressed;
       END;
  'X' : exit23 := true;
  END;
 UNTIL exit23;
END;

{************************************************}
{*****   2.4 : SECOND ORDER                 *****}
{*****           Real Poles : α,β           *****}
{*****           Comp Poles : ---           *****}
{*****           LHP  Zeros : a             *****}
{*****           RHP  Zeros : ---           *****}
{************************************************}
PROCEDURE selct_4;
CONST
     valid24 : set of '0'..'Z' = ['1','2','3','4','5','P','D','X'];
VAR
   exit24 : boolean;
   ch24   : char;
BEGIN
 lables^.head1 := 'ROLLING MILL PLANT : SECOND ORDER DATA FIT';
 lables^.utitl := lables^.ytitl;
 lables^.ulabl := lables^.ylabl;
 lables^.uname := lables^.yname;
 x0 := 0;
```

```
x1 := 1;
p1 := 0.5;
p2 := 1.0;
z1 := 2.0;

exit24 := false;
REPEAT
 func_fit (ch0,ch2);

 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────────┐');
 gotoxy (20,02);write ('│ 2 :  SECOND ORDER RESPONSE        │');
 gotoxy (20,03);write ('│                                   │');
 gotoxy (20,04);write ('│     ┌ αβ      (s + a)      ┐ -sτ  │');
 gotoxy (20,05);write ('│4:K  │ --  ─────────────────│e     │');
 gotoxy (20,06);write ('│     └ a   (s + α)(s + β)   ┘      │');
 gotoxy (20,07);write ('└──────────────────────────────────┘');
 gotoxy (20,09);write ('         ┌─────────────────────────┐');
 gotoxy (20,10);write ('         │       1  :  K =          │');
 gotoxy (20,11);write ('         │       2  :  α =          │');
 gotoxy (20,12);write ('         │       3  :  β =          │');
 gotoxy (20,13);write ('         │       4  :  a =          │');
 gotoxy (20,14);write ('         │       5  :  τ =          │');
 gotoxy (20,15);write ('         │                          │');
 gotoxy (20,16);write ('         │       P  :  Plot         │');
 gotoxy (20,17);write ('         │       D  :  Display      │');
 gotoxy (20,18);write ('         │       X  :  Exit         │');
 gotoxy (20,19);write ('         └─────────────────────────┘');
 gotoxy (20,23);write ('         ┌─────────────────────────┐');
 gotoxy (20,24);write ('         │       OPTION :           │');
 gotoxy (20,25);write ('         └─────────────────────────┘');
 gotoxy (44,10);write (x1:10:6);
 gotoxy (44,11);write (p1:10:6);
 gotoxy (44,12);write (p2:10:6);
 gotoxy (44,13);write (z1:10:6);
 gotoxy (44,14);write (x0:10:6);
 REPEAT
  gotoxy (44,24);
  ch24 := upcase(readkey);
 UNTIL ch24 in valid24;
 CASE ch24 OF
 '1' : BEGIN
        gotoxy (44,10);write ('          ');
        gotoxy (44,10);readln (x1);
       END;
 '2' : BEGIN
        gotoxy (44,11);write ('          ');
        gotoxy (44,11);readln (p1);
       END;
 '3' : BEGIN
        gotoxy (44,12);write ('          ');
        gotoxy (44,12);readln (p2);
       END;
```

```
  '4' : BEGIN
         gotoxy (44,13);write ('          ');
         gotoxy (44,13);readln (z1);
        END;
  '5' : BEGIN
         gotoxy (44,14);write ('          ');
         gotoxy (44,14);readln (x0);
        END;
  'P' : data_plot (datclc,lables);
  'D' : BEGIN
         clrscr;

         writeln ('NUMERATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(x1*p1*p2):20:12);
           writeln ('s^1 = ',(x1*p1*p2/z1):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*x1*p1*p2/x0):20:12);
           writeln ('s^1 = ',((2*x1*p1*p2/(z1*x0)-x1*p1*p2)):20:12);
           writeln ('s^2 = ',(-x1*p1*p2/p1):20:12);
          END;
         writeln;

         writeln ('DENOMINATOR');
         IF (x0 = 0) THEN
          BEGIN
           writeln ('s^0 = ',(p1*p2):20:12);
           writeln ('s^1 = ',(p1+p2):20:12);
           writeln ('s^2 = ',(1.0):20:12);
          END
         ELSE
          BEGIN
           writeln ('s^0 = ',(2*p1*p2/x0):20:12);
           writeln ('s^1 = ',(p1*p2+2*(p1+p2)/x0):20:12);
           writeln ('s^2 = ',(p1+p2+2/x0):20:12);
           writeln ('s^3 = ',(1.0):20:12);
          END;
         writeln;

         writeln ('<<< PRESS ANY KEY TO CONTINUE >>>');
         REPEAT UNTIL keypressed;
        END;
  'X' : exit24 := true;
  END;
 UNTIL exit24;
END;

BEGIN
 exit2 := false;
 REPEAT
  clrscr;
```

```
   gotoxy (15,01);write ('┌──────────────────────────────┐');
   gotoxy (15,02);write ('│ 2  :  SECOND ORDER RESPONSE  │');
   gotoxy (15,03);write ('└──────────────────────────────┘');
   gotoxy (15,06);write ('┌──────────────────────────────┐');
   gotoxy (15,07);write ('│      ┌       αβ       ┐ -sτ  │');
   gotoxy (15,08);write ('│1: K  │───────────────│ e    │');
   gotoxy (15,09);write ('│      └ (s + α)(s + β) ┘      │');
   gotoxy (15,10);write ('│      ┌     α²+β²      ┐ -sτ  │');
   gotoxy (15,11);write ('│2: K  │───────────────│ e    │');
   gotoxy (15,12);write ('│      └ (s + α)² + β²  ┘      │');
   gotoxy (15,13);write ('│      ┌      α²        ┐ -sτ  │');
   gotoxy (15,14);write ('│3: K  │───────────────│ e    │');
   gotoxy (15,15);write ('│      └    (s + α)²    ┘      │');
   gotoxy (15,16);write ('│     ┌ αβ    (s + a)    ┐ -sτ │');
   gotoxy (15,17);write ('│4: K│ --  ──────────────│e    │');
   gotoxy (15,18);write ('│     └ a  (s + α)(s + β) ┘    │');
   gotoxy (15,19);write ('│                              │');
   gotoxy (15,20);write ('│X: Exit                       │');
   gotoxy (15,21);write ('└──────────────────────────────┘');
   gotoxy (15,23);write ('      ┌──────────────────────┐');
   gotoxy (15,24);write ('      │       OPTION :       │');
   gotoxy (15,25);write ('      └──────────────────────┘');
   REPEAT
    gotoxy (37,24);
    ch2 := upcase(readkey);
   UNTIL ch2 in valid2;
   CASE ch2 OF
   '1' : selct_1;
   '2' : selct_2;
   '3' : selct_3;
   '4' : selct_4;
   'X' : exit2 := true;
   END;
  UNTIL exit2;
END;

{**************************************************}
{*****     M A I N     P R O G R A M M E     *****}
{**************************************************}
BEGIN
 new (datclc);
 new (datlog);
 new (lables);

 REPEAT
  clrscr;
  write  ('NAME OF INPUT FILE : ');
  readln (inputname);
  search := fsearch (inputname,'c:');
 UNTIL (search <> '');
 assign (inputfile,fexpand(inputname));
 reset  (inputfile);
 readln (inputfile,lables^.head2);
 readln (inputfile,lables^.utitl);
```

```
readln (inputfile,lables^.ulabl);
readln (inputfile,lables^.ytitl);
readln (inputfile,lables^.ylabl);
readln (inputfile,dx);
maxpts := -1;
WHILE NOT eof(inputfile) DO
 BEGIN
  inc (maxpts);
  read   (inputfile,datlog^.u[maxpts]);
  read   (inputfile,datlog^.y[maxpts]);
  readln (inputfile);
 END;
close  (inputfile);

lables^.head1 := 'ROLLING MILL SIMULATOR : ACTUAL DATA';
lables^.uname := 'U';
lables^.yname := 'Y';
lables^.xlabl := 's';
data_plot (datlog,lables);

exit0 := false;
REPEAT
 clrscr;
 gotoxy (20,01);write ('┌──────────────────────────────┐');
 gotoxy (20,02);write ('│ CURVE FITTING TO LOGGED DATA │');
 gotoxy (20,03);write ('└──────────────────────────────┘');
 gotoxy (20,08);write ('┌──────────────────────────────┐');
 gotoxy (20,09);write ('│                              │');
 gotoxy (20,10);write ('│      1 : First Order         │');
 gotoxy (20,11);write ('│                              │');
 gotoxy (20,12);write ('│      2 : Second Order        │');
 gotoxy (20,13);write ('│                              │');
 gotoxy (20,14);write ('│                              │');
 gotoxy (20,15);write ('│      X : Exit                │');
 gotoxy (20,16);write ('│                              │');
 gotoxy (20,17);write ('└──────────────────────────────┘');
 gotoxy (20,20);write ('┌──────────────────────────────┐');
 gotoxy (20,21);write ('│      OPTION :                │');
 gotoxy (20,22);write ('└──────────────────────────────┘');
 REPEAT
  gotoxy (36,21);
  ch0 := upcase(readkey);
 UNTIL ch0 in valid0;
 CASE ch0 OF
 '1' : order_1;
 '2' : order_2;
 'X' : exit0 := true;
 END;
UNTIL exit0;
clrscr;
END.
```

APPENDIX M: ROLLING MILL SIMULATOR LISTING

PASCAL UNITS USED BY MAIN PROGRAM.

```
PROGRAM roller;

{$M 65520,0,655360}

USES
    plant,crt,matrix,
    decoup,data;

VAR
 loggd                                  : boolean;
 state,stepd                            : boolean;
 exit1,exit2,exit3,exit4,exit5          : boolean;
 exit6,exit7,exit8,exit9,exit10,exit11  : boolean;
 exit12,exit13,exit14,exit15,exit16     : boolean;
 h1inc,h2inc,h3inc,h4inc                : boolean;
 h1dec,h2dec,h3dec,h4dec                : boolean;
 time,time1,dis1,ymch     : real;
 k1,k2,k3,k4,k5,k6,k7,k8  : real;
 I1,I2,I3,I4,I5,I6,I7,I8  : real;
 n,pts,answer,row         : integer;
 esc,pplot,exit,drivno    : char;
 stepno,choice,choice1    : char;
 chpi1,chpi2,chpi3,chpi4  : char;
 chpi5,chpi6,chpi7,chpi8  : char;
 chdc,chdcc,timch,ymchc   : char;
 outfile                  : text;
 filename                 : string;


 setp  : vect_ptr;
 eror  : vect_ptr;
 vint  : vect_ptr;
 oupt  : vect_ptr;
 pout  : vect_ptr;

 compos: mtrx_ptr;
 comput: mtrx_ptr;
 intatn: mtrx_ptr;

 stores: data_ptr;
 lables: labl_ptr;
 Chfile: crtx_ptr;
 roll  : stat_ptr;

 astor : temp_ptr;
```

```
 nstor : temp_ptr;
 m1str : temp_ptr;
 m2str : temp_ptr;
 ltim1 : temp_ptr;
 ltim2 : temp_ptr;




(******************************************************************)
procedure setup_pntr;
(******************************************************************)

BEGIN
 new (setp); vect_initl (setp);
 new (eror); vect_initl (eror);
 new (vint); vect_initl (vint);
 new (oupt); vect_initl (oupt);
 new (pout); vect_initl (pout);

 new (stores); data_init (stores);
 new (Chfile);
 new (roll);

 new(astor);
 new(nstor);
 new(m1str);
 new(m2str);
 new(ltim1);
 new(ltim2);

 new (lables);

 new(compos); mtrx_initl(compos);
 new(comput); mtrx_initl(comput);
 new(intatn); mtrx_initl(intatn);

 END;


(******************************************************************)
procedure clear_pntr;
(******************************************************************)

BEGIN
 vect_clear (setp);
 vect_clear (eror);
 vect_clear (vint);
 vect_clear (oupt);
 vect_clear (pout);

 file_initl (Chfile);
 stat_initl (roll);
```

```
 temp_initl(astor);
 temp_initl(nstor);
 temp_initl(m1str);
 temp_initl(m2str);
 temp_initl(1timl);

END;



(******************************************************************)
procedure cloop_labl;
(******************************************************************)

BEGIN
 lables^.uname[1] := 'Astp 1';
 lables^.uname[2] := 'Astp 2';
 lables^.uname[3] := 'Astp 3';
 lables^.uname[4] := 'Lstp 1';
 lables^.uname[5] := 'Lstp 2';
 lables^.uname[6] := 'Lstp 3';
 lables^.uname[7] := 'Lstp 4';
 lables^.uname[8] := 'Speed stp 8';

 lables^.yname[1] := 'Iout 1';
 lables^.yname[2] := 'Iout 2';
 lables^.yname[3] := 'Iout 3';
 lables^.yname[4] := 'St 4 : Length 1';
 lables^.yname[5] := 'St 5 : Length 2';
 lables^.yname[6] := 'St 6 : Length 3';
 lables^.yname[7] := 'St 7 : Length 4';
 lables^.yname[8] := 'Speed 8';

 lables^.ulabl[1] := 'A';
 lables^.ulabl[2] := 'A';
 lables^.ulabl[3] := 'A';
 lables^.ulabl[4] := 'cm';
 lables^.ulabl[5] := 'cm';
 lables^.ulabl[6] := 'cm';
 lables^.ulabl[7] := 'cm';
 lables^.ulabl[8] := 'rpm';

 lables^.ylabl[1] := 'A';
 lables^.ylabl[2] := 'A';
 lables^.ylabl[3] := 'A';
 lables^.ylabl[4] := 'cm';
 lables^.ylabl[5] := 'cm';
 lables^.ylabl[6] := 'cm';
 lables^.ylabl[7] := 'cm';
 lables^.ylabl[8] := 'rpm';

 lables^.xlabl := 's';
 lables^.headl := '    THE 8 X 8 ROLLING MILL SYSTEM';
```

```
 lables^.head2 := 'CLOSED LOOP RESPONSE';

END;



(*****************************************************************)
procedure oloop_labl;
(*****************************************************************)

BEGIN
 lables^.uname[1] := 'Speed stp 1';
 lables^.uname[2] := 'Speed stp 2';
 lables^.uname[3] := 'Speed stp 3';
 lables^.uname[4] := 'Speed stp 4';
 lables^.uname[5] := 'Speed stp 5';
 lables^.uname[6] := 'Speed stp 6';
 lables^.uname[7] := 'Speed stp 7';
 lables^.uname[8] := 'Speed stp 8';

 lables^.yname[1] := 'I output 1';
 lables^.yname[2] := 'I output 2';
 lables^.yname[3] := 'I output 3';
 lables^.yname[4] := 'Loop length 1';
 lables^.yname[5] := 'Loop length 2';
 lables^.yname[6] := 'Loop length 3';
 lables^.yname[7] := 'Loop length 4';
 lables^.yname[8] := 'Speed 8';

 lables^.ulabl[1] := 'rpm';
 lables^.ulabl[2] := 'rpm';
 lables^.ulabl[3] := 'rpm';
 lables^.ulabl[4] := 'rpm';
 lables^.ulabl[5] := 'rpm';
 lables^.ulabl[6] := 'rpm';
 lables^.ulabl[7] := 'rpm';
 lables^.ulabl[8] := 'rpm';

 lables^.ylabl[1] := 'A';
 lables^.ylabl[2] := 'A';
 lables^.ylabl[3] := 'A';
 lables^.ylabl[4] := 'm';
 lables^.ylabl[5] := 'm';
 lables^.ylabl[6] := 'm';
 lables^.ylabl[7] := 'm';
 lables^.ylabl[8] := 'rpm';

 lables^.xlabl := 's';
 lables^.head1 := '    THE 8 X 8 ROLLING MILL SYSTEM';
 lables^.head2 := 'OPEN LOOP RESPONSE';

END;
```

```
(*******************************************************************)
procedure nipi_cnt;
(*******************************************************************)

BEGIN

 comput^[1,1]^.nsz := 1;
 comput^[1,1]^.dsz := 1;
 comput^[1,1]^.num[0] := k1/I1;
 comput^[1,1]^.num[1] := k1;
 comput^[1,1]^.den[0] := 0.0;
 comput^[1,1]^.den[1] := 1;

 comput^[2,2]^.nsz := 1;
 comput^[2,2]^.dsz := 1;
 comput^[2,2]^.num[0] := k2/I2;
 comput^[2,2]^.num[1] := k2;
 comput^[2,2]^.den[0] := 0.0;
 comput^[2,2]^.den[1] := 1;

 comput^[3,3]^.nsz := 1;
 comput^[3,3]^.dsz := 1;
 comput^[3,3]^.num[0] := k3/I3;
 comput^[3,3]^.num[1] := k3;
 comput^[3,3]^.den[0] := 0.0;
 comput^[3,3]^.den[1] := 1;

 comput^[4,4]^.nsz := 1;
 comput^[4,4]^.dsz := 1;
 comput^[4,4]^.num[0] := k4/I4;
 comput^[4,4]^.num[1] := k4;
 comput^[4,4]^.den[0] := 0.0;
 comput^[4,4]^.den[1] := 1;

 comput^[5,5]^.nsz := 1;
 comput^[5,5]^.dsz := 1;
 comput^[5,5]^.num[0] := k5/I5;
 comput^[5,5]^.num[1] := k5;
 comput^[5,5]^.den[0] := 0.0;
 comput^[5,5]^.den[1] := 1;

 comput^[6,6]^.nsz := 1;
 comput^[6,6]^.dsz := 1;
 comput^[6,6]^.num[0] := k6/I6;
 comput^[6,6]^.num[1] := k6;
 comput^[6,6]^.den[0] := 0.0;
 comput^[6,6]^.den[1] := 1;

 comput^[7,7]^.nsz := 1;
 comput^[7,7]^.dsz := 1;
 comput^[7,7]^.num[0] := k7/I7;
 comput^[7,7]^.num[1] := k7;
```

```
 comput^[7,7]^.den[0] := 0.0;
 comput^[7,7]^.den[1] := 1;

 comput^[8,8]^.nsz := 1;
 comput^[8,8]^.dsz := 1;
 comput^[8,8]^.num[0] := k8/I8;
 comput^[8,8]^.num[1] := k8;
 comput^[8,8]^.den[0] := 0.0;
 comput^[8,8]^.den[1] := 1;

END;




(******************************************************************)
procedure initial;
(******************************************************************)

var  i,j : integer;

BEGIN

  Chfile^[1].yget:=        0;
  Chfile^[1].ipr :=        0;
  Chfile^[1].Aip :=     7600;
  Chfile^[1].npr :=    1932000/Chfile^[1].Aip;
  Chfile^[1].Kpi :=      8.0;
  Chfile^[1].Ipi :=     0.04;
  Chfile^[1].Cm0 :=    73.06;
  Chfile^[1].Bv  :=        0;
  Chfile^[1].J   :=      206;

  Chfile^[2].yget:=        0;
  Chfile^[2].ipr :=        0;
  Chfile^[2].Aip :=     6300;
  Chfile^[2].npr :=    1932000/Chfile^[2].Aip;
  Chfile^[2].Kpi :=      9.0;
  Chfile^[2].Ipi :=     0.05;
  Chfile^[2].Cm0 :=    73.06;
  Chfile^[2].Bv  :=        0;
  Chfile^[2].J   :=      206;

  Chfile^[3].yget:=        0;
  Chfile^[3].ipr :=        0;
  Chfile^[3].Aip :=     5100;
  Chfile^[3].npr :=    1932000/Chfile^[3].Aip;
  Chfile^[3].Kpi :=      8.0;
  Chfile^[3].Ipi :=     0.05;
  Chfile^[3].Cm0 :=    73.06;
  Chfile^[3].Bv  :=        0;
  Chfile^[3].J   :=      411;

  Chfile^[4].yget:=        0;
```

```
Chfile^[4].ipr :=        0;
Chfile^[4].Aip :=     4000;
Chfile^[4].npr :=    1932000/Chfile^[4].Aip;
Chfile^[4].Kpi :=      8.0;
Chfile^[4].Ipi :=     0.05;
Chfile^[4].Cm0 :=    73.06;
Chfile^[4].Bv  :=        0;
Chfile^[4].J   :=      411;

Chfile^[5].yget:=        0;
Chfile^[5].ipr :=        0;
Chfile^[5].Aip :=   3487.5;
Chfile^[5].npr :=   1932000/Chfile^[5].Aip;
Chfile^[5].Kpi :=      8.0;
Chfile^[5].Ipi :=     0.05;
Chfile^[5].Cm0 :=    73.06;
Chfile^[5].Bv  :=        0;
Chfile^[5].J   :=      411;

Chfile^[6].yget:=        0;
Chfile^[6].ipr :=        0;
Chfile^[6].Aip :=     3000;
Chfile^[6].npr :=    1932000/Chfile^[6].Aip;
Chfile^[6].Kpi :=      8.0;
Chfile^[6].Ipi :=     0.05;
Chfile^[6].Cm0 :=    73.06;
Chfile^[6].Bv  :=        0;
Chfile^[6].J   :=      411;

Chfile^[7].yget:=        0;
Chfile^[7].ipr :=        0;
Chfile^[7].Aip :=   2537.5;
Chfile^[7].npr :=   1932000/Chfile^[7].Aip;
Chfile^[7].Kpi :=      8.0;
Chfile^[7].Ipi :=     0.05;
Chfile^[7].Cm0 :=    73.06;
Chfile^[7].Bv  :=        0;
Chfile^[7].J   :=      411;

Chfile^[8].yget:=        0;
Chfile^[8].ipr :=        0;
Chfile^[8].Aip :=     2100;
Chfile^[8].npr :=    1932000/Chfile^[8].Aip;
Chfile^[8].Kpi :=      8.0;
Chfile^[8].Ipi :=     0.05;
Chfile^[8].Cm0 :=    73.06;
Chfile^[8].Bv  :=        0;
Chfile^[8].J   :=      411;

FOR j := 1 to max_cntrl do
FOR i := 0 to ypnts do
 BEGIN
  Chfile^[j].ytt[i] := Chfile^[j].ipr;
  Chfile^[j].B[i] := Chfile^[j].Bv;
```

```
    astor^[j] := dist/2.0;
   END;

 IF (choice = 'C') then
  BEGIN
   FOR i := 1 to 7 do setp^.n[0]^[i] := 0;
   FOR i := 4 to 7 do setp^.n[0]^[i] := 100*SQRT(SQR(setp^.n[0]^[i]) + SQR(dist/2));
   setp^.n[0]^[8] := 920.0;

   FOR i := 1 to 8 do vint^.o[0]^[i] := 1932000/Chfile^[i].Aip;

   FOR i := 1 to 7 do oupt^.n[0]^[i] := 0;
   FOR i := 4 to 7 do oupt^.n[0]^[i] := 100*SQRT(SQR(oupt^.n[0]^[i]) + SQR(dist/2));
   oupt^.n[0]^[8] := 920.0;
  END;

 IF (choice = 'O') then
  BEGIN
   FOR i := 1 to 8 do setp^.n[0]^[i] := 1932000/Chfile^[i].Aip;
  END;
END;


(*******************************************************************)
procedure dely_fnc;
(*******************************************************************)

var   r : integer;

BEGIN
 FOR r := 1 to max_cntrl do
  BEGIN
   Chfile^[r].yput := 0;
  END;
END;


(********************************************************************)
procedure position(var relay1,relay2 : boolean; var w1,w2 : real);
(********************************************************************)

BEGIN

IF (relay1 = true) and (relay2 = false) then
   dis1 := dis1 + ((w1*0.25*2*3.14159)/60)* dt;

IF (dis1 > dist) then dis1 := dist;

IF (relay1 = true) and (dis1 >= dist) then relay2 := true;

IF (dis1 > 0) then
```

```
 BEGIN
   IF (relay1 = false) and (relay2 = true) then
    BEGIN
      dis1 := dis1 -((w2*0.25*2*3.14159)/60)* dt;
      If (dis1 < 0) then dis1 := 0;
    END;
 END;

IF (dis1 > dist) then dis1 := dist;

IF (dis1 = 0) then
 BEGIN
   relay2 := false;
 END;

gotoXY(30,6);
write('Position:',dis1:6:2,'m');
END;



(*******************************************************************)
procedure step(var i : integer;f:crtx_ptr);
(*******************************************************************)

VAR
 j : integer;

BEGIN
 IF (state = true) then
  BEGIN
     f^[i].Bv      := 2418.6;
     f^[i].ipr     := (f^[i].npr*2*3.14159*0.25*f^[i].Bv)/(60*f^[i].Cm0);

     FOR j := 0 to ypnts do
      BEGIN
       f^[i].ytt[j] := f^[i].ipr;
       f^[i].B[j] := f^[j].Bv;
      END;

     IF (choice = 'C') then
      BEGIN
       IF (i = 1) or (i = 2) or (i = 3) then
        BEGIN
         setp^.n[0]^[i] := f^[i].ipr;
         oupt^.n[0]^[i] := f^[i].ipr;
        END;
       IF (i = 4) or (i = 5) or (i = 6) or (i = 7) then
        BEGIN
         setp^.n[0]^[i] := 100*dist/2;
         oupt^.n[0]^[i] := 100*dist/2;
        END;
       IF (i = 8) then
        BEGIN
```

```
        setp^.n[0]^[8] := 920.0;
        oupt^.n[0]^[8] := 920.0;
       END;
     END;
  END
 ELSE
  BEGIN
   f^[i].ipr     := 0;
   f^[i].Bv      := 0;
   IF (choice = 'C') then
    BEGIN
     IF (i = 1) or (i = 2) or (i = 3) then
      BEGIN
       setp^.n[0]^[i] := 0;
       oupt^.n[0]^[i] := 0;
      END;
    END;
  END;
END;



(******************************************************************)
procedure hght_gain;
(******************************************************************)

VAR
  i : integer;

BEGIN
 IF (oupt^.n[0]^[4] < 2.06) and (h1inc = false) then
  BEGIN
   setp^.n[0]^[1] := setp^.n[0]^[1] * 1.06;
   setp^.n[0]^[2] := setp^.n[0]^[2] * 1.06;
   setp^.n[0]^[3] := setp^.n[0]^[3] * 1.06;
   setp^.n[0]^[4] := setp^.n[0]^[4] * 1.05;
   h1inc := true;
  END;
 IF (oupt^.n[0]^[5] < 2.06) and (h2inc = false) then
  BEGIN
   setp^.n[0]^[5] := setp^.n[0]^[5] * 1.04;
   h2inc := true;
  END;
 IF (oupt^.n[0]^[6] < 2.06) and (h3inc = false) then
  BEGIN
   setp^.n[0]^[6] := setp^.n[0]^[6] * 1.03;
   h3inc := true;
  END;
 IF (oupt^.n[0]^[7] < 2.06) and (h4inc = false) then
  BEGIN
   setp^.n[0]^[7] := setp^.n[0]^[7] * 1.02;
   h4inc := true;
  END;
```

```
 IF (oupt^.n[0]^[4] > 2.06) then
  BEGIN
   IF (h1dec = false) then
    BEGIN
     setp^.n[0]^[1] := setp^.n[0]^[1] / 1.06;
     setp^.n[0]^[2] := setp^.n[0]^[2] / 1.06;
     setp^.n[0]^[3] := setp^.n[0]^[3] / 1.06;
     setp^.n[0]^[4] := setp^.n[0]^[4] / 1.05;
     h1dec := true;
    END;
   IF (h2dec = false) then
    BEGIN
     setp^.n[0]^[5] := setp^.n[0]^[5] / 1.04;
     h2dec := true;
    END;
   IF (h3dec = false) then
    BEGIN
     setp^.n[0]^[6] := setp^.n[0]^[6] / 1.03;
     h3dec := true;
    END;
   IF (h4dec = false) then
    BEGIN
     setp^.n[0]^[7] := setp^.n[0]^[7] / 1.02;
     h4dec := true;
    END;
  END;
END;




(*****************************************************************)
procedure hght_setp;
(*****************************************************************)

VAR
  i  : integer;

BEGIN
 FOR i := 4 to 7 do
  BEGIN
   IF (roll^[i] = true) and (roll^[i+1] = true) then
    BEGIN
     IF ((i = 4) and (roll^[i-3] = true)) or (roll^[i-4] = true) then
      BEGIN
       IF (setp^.n[0]^[i] < 206.0) then
        BEGIN
         setp^.n[0]^[i] := 206.0;
        END;
      END
     ELSE IF ((i = 4) and (roll^[i-3] = false)) or (roll^[i-4] = false) then
      BEGIN
```

```
       ltiml^[i] := ltiml^[i] + dt;
       IF (setp^.n[0]^[i] > 200.0) and (ltiml^[i] > ltim2^[i]) then
        BEGIN
         setp^.n[0]^[i] := 200.0;
        END;
      END;
    END
   ELSE setp^.n[0]^[i] := 100*dist/2;

   IF (setp^.n[0]^[i] < 100*dist/2) then setp^.n[0]^[i] := 100*dist/2;
  END;
END;



(******************************************************************)
procedure mast_cntrl;
(******************************************************************)

BEGIN
 nipi_cnt;
 IF (chdc = '1') then
  BEGIN
   i1_8_cnt (intatn);
  END
 ELSE IF (chdc = '2') then
  BEGIN
   IF (state = true) then
    BEGIN
     IF (roll^[1] = true) then nint_cnt (intatn);
     IF (roll^[2] = true) then i1_2_cnt (intatn);
     IF (roll^[3] = true) then i1_3_cnt (intatn);
     IF (roll^[4] = true) then i1_4_cnt (intatn);
     IF (roll^[5] = true) then i1_5_cnt (intatn);
     IF (roll^[6] = true) then i1_6_cnt (intatn);
     IF (roll^[7] = true) then i1_7_cnt (intatn);
     IF (roll^[8] = true) then i1_8_cnt (intatn);
    END
   ELSE
    BEGIN
     IF (roll^[1] = false) then i2_8_cnt (intatn);
     IF (roll^[2] = false) then i3_8_cnt (intatn);
     IF (roll^[3] = false) then i4_8_cnt (intatn);
     IF (roll^[4] = false) then i5_8_cnt (intatn);
     IF (roll^[5] = false) then i6_8_cnt (intatn);
     IF (roll^[6] = false) then i7_8_cnt (intatn);
     IF (roll^[7] = false) then i8_8_cnt (intatn);
    END;
  END
 ELSE nint_cnt (intatn);

 mtrx_multp (compos,intatn,comput);
END;
```

```
(*****************************************************************)
procedure logs_data;
(*****************************************************************)

VAR
  i  : integer;

BEGIN
 IF (pts <= max_point) and (round(time1/dx) = pts) then
  BEGIN
   data_logg (pts,stores,setp^.n[0],oupt^.n[0]);
   inc(pts);
  END;

 IF (time1 > dx*max_point) then
  loggd := true;

END;



(*****************************************************************)
procedure calclt;
(*****************************************************************)

VAR
  i : integer;

BEGIN
 IF (loggd = false) and (pplot = 'Y') then
  BEGIN
   logs_data;
   time1 := time1 + dt;
  END;

IF (choice = 'C') then
 BEGIN
  mast_cntrl;
  vect_subtr(setp,oupt,eror);
  mtrx_ip2op(eror,compos,vint);
  mtrx_ip2p2(roll,vint,Chfile,oupt,nstor,m1str,m2str,ymch);
  hght_ip2op(roll,astor,m1str,m2str,oupt,Chfile,ymch);
  FOR i := 4 to 7 do oupt^.n[0]^[i] := 100*SQRT(SQR(oupt^.n[0]^[i]) + SQR(dist/2));
 END;

IF (choice = 'O') then
 BEGIN
  mtrx_ip2p2(roll,setp,Chfile,oupt,nstor,m1str,m2str,ymch);
  hght_ip2op(roll,astor,m1str,m2str,oupt,Chfile,ymch);
  FOR i := 4 to 7 do oupt^.n[0]^[i] := SQRT(SQR(oupt^.n[0]^[i]) + SQR(dist/2));
 END;

  gotoXY(10,08);write('Stand 1 active:',roll^[1] ,' ');
```

```
  gotoXY(30,08);write('        Current in Drive 1: ',oupt^.n[0]^[1]:8:3,' A');
  gotoXY(10,10);write('Stand 2 active:',roll^[2] ,' ');
  gotoXY(30,10);write('        Current in Drive 2: ',oupt^.n[0]^[2]:8:3,' A');
  gotoXY(10,12);write('Stand 3 active:',roll^[3] ,' ');
  gotoXY(30,12);write('        Current in Drive 3: ',oupt^.n[0]^[3]:8:3,' A');
  gotoXY(10,14);write('Stand 4 active:',roll^[4] ,' ');
  gotoXY(30,14);write('        Height of Loop 1  : ',oupt^.n[0]^[4]:8:3,' cm');
  gotoXY(10,16);write('Stand 5 active:',roll^[5] ,' ');
  gotoXY(30,16);write('        Height of Loop 2  : ',oupt^.n[0]^[5]:8:3,' cm');
  gotoXY(10,18);write('Stand 6 active:',roll^[6] ,' ');
  gotoXY(30,18);write('        Height of Loop 3  : ',oupt^.n[0]^[6]:8:3,' cm');
  gotoXY(10,20);write('Stand 7 active:',roll^[7] ,' ');
  gotoXY(30,20);write('        Height of Loop 4  : ',oupt^.n[0]^[7]:8:3,' cm');
  gotoXY(10,22);write('Stand 8 active:',roll^[8] ,' ');
  gotoXY(30,22);write('        Speed of Drive 8  : ',oupt^.n[0]^[8]:8:3,' rpm');

 END;


(*********************************************************************)
procedure standno(var rolln,rollo :boolean ; var speedn,speedo :real);
(*********************************************************************)

BEGIN
   repeat
    position(rolln,rollo,speedn,speedo);
    IF (choice = 'C') then hght_setp;
    calclt;
   until (rollo = state);
END;


(*****************************************************************)
procedure startup;
(*****************************************************************)

BEGIN
 roll^[1] := true;
 state := true;
 loggd := false;
END;


(*****************************************************************)
procedure exiting;
(*****************************************************************)

BEGIN
 repeat
```

```
  IF (choice = 'C') then hght_setp;
  IF (choice = 'O') then hght_gain;
  calclt;
  time := time + dt;


  IF (time > 2.0) and (choice = 'C') then
   BEGIN
    IF (stepno = '1' ) then setp^.n[0]^[1] := 270.3;
    IF (stepno = '2' ) then setp^.n[0]^[2] := 315.8;
    IF (stepno = '3' ) then setp^.n[0]^[3] := 378.3;
    IF (stepno = '4' ) then setp^.n[0]^[4] := 207.0;
    IF (stepno = '5' ) then setp^.n[0]^[5] := 206.0;
    IF (stepno = '6' ) then setp^.n[0]^[6] := 206.0;
    IF (stepno = '7' ) then setp^.n[0]^[7] := 206.0;
    IF (stepno = '8' ) then setp^.n[0]^[8] := 900.0;
   END;

  IF (time > 2.1) and (choice = 'O') and (stepd = false) then
   BEGIN
    IF (stepno = '1' ) then setp^.n[0]^[1] := setp^.n[0]^[1]*1.05;
    IF (stepno = '2' ) then setp^.n[0]^[2] := setp^.n[0]^[2]*1.05;
    IF (stepno = '3' ) then setp^.n[0]^[3] := setp^.n[0]^[3]*1.05;
    IF (stepno = '4' ) then setp^.n[0]^[4] := setp^.n[0]^[4]*1.05;
    IF (stepno = '5' ) then setp^.n[0]^[5] := setp^.n[0]^[5]*1.001;
    IF (stepno = '6' ) then setp^.n[0]^[6] := setp^.n[0]^[6]*1.00075;
    IF (stepno = '7' ) then setp^.n[0]^[7] := setp^.n[0]^[7]*1.00075;
    IF (stepno = '8' ) then setp^.n[0]^[8] := setp^.n[0]^[8]*1.0015;
    stepd := true;
   END;

 until (time >= 6.0);


END;




(******************************************************************)
procedure steady;
(******************************************************************)

VAR
 value : real;

BEGIN

 repeat
    calclt;
    value := Chfile^[1].npr - 0.0015
 until (oupt^.n[0]^[1] >= value);
END;
```

```
(*****************************************************************)
procedure finlog;
(*****************************************************************)

BEGIN
 IF (loggd = false) then
  BEGIN
   repeat
    calclt;
   until (loggd = true);
  END;
END;


(*****************************************************************)
procedure produce;
(*****************************************************************)

VAR
   i : integer;

BEGIN
 clrscr;
 REPEAT
  clear_pntr;
  initial;
  dely_fnc;
  startup;

  IF (choice = 'C') then
   BEGIN
    cloop_labl;
    mast_cntrl;
    vect_subtr(setp,oupt,eror);
    mtrx_ip2op(eror,compos,vint);
    mtrx_ip2p2(roll,vint,Chfile,oupt,nstor,m1str,m2str,ymch);
    hght_ip2op(roll,astor,m1str,m2str,oupt,Chfile,ymch);
    FOR i := 4 to 7 do oupt^.n[0]^[i] := 100*SQRT(SQR(oupt^.n[0]^[i]) + SQR(dist/2));
   END;

  IF (choice = 'O') then
   BEGIN
    oloop_labl;
    mtrx_ip2p2(roll,setp,Chfile,oupt,nstor,m1str,m2str,ymch);
    hght_ip2op(roll,astor,m1str,m2str,oupt,Chfile,ymch);
    FOR i := 4 to 7 do oupt^.n[0]^[i] := SQRT(SQR(oupt^.n[0]^[i]) + SQR(dist/2));
   END;

  dis1  := 0.0;
  pts   := 0;
  time  := 0;
  time1 := 0;
```

```
  FOR  n := 1 to 7 do
   BEGIN
    step(n,Chfile);
    standno(roll^[n],roll^[n+1],Chfile^[n].npr,Chfile^[n+1].npr);
    disl := 0;
   END;


  n := 8;
  step(n,Chfile);



  exiting;

 state := false;
 roll^[1] := false;

  FOR n := 1 to 7 do
   BEGIN
    step(n,Chfile);
    disl := dist;
    standno(roll^[n],roll^[n+1],Chfile^[n].npr,Chfile^[n+1].npr);
   END;

  n := 8;
  step(n,Chfile);

  IF (pplot = 'Y') then
   BEGIN
    finlog;
    data_plot (stores,lables,choice);
   END;

 dec(answer);
 until (answer = 0);
END;









(*****************************************************************)
procedure prd_quota;
(*****************************************************************)

VAR
  i : integer;
```

```
BEGIN
 exit4  := false;
 REPEAT
  answer := 1;
  h1inc  := false;h2inc  := false;h3inc  := false;h4inc  := false;
  h1dec  := false;h2dec  := false;h3dec  := false;h4dec  := false;
  stepd   := false;
  clrscr;
  gotoXY(17,07);write('╔════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                            ║');
  gotoXY(17,09);write('║     SELECT NO. OF BILLETS TO PROCESS       ║');
  gotoXY(17,10);write('║        1. :                                ║');
  gotoXY(17,11);write('║                                            ║');
  gotoXY(17,12);write('║        PLOT AFTER EACH BILLET ?            ║');
  gotoXY(17,13);write('║        2. : Yes                            ║');
  gotoXY(17,14);write('║        3. : No                             ║');
  gotoXY(17,15);write('║                                            ║');
  gotoXY(17,16);write('║        x  : Exit                           ║');
  gotoXY(17,17);write('╟────────────────────────────────────────────╢');
  gotoXY(17,18);write('║        OPTION :                            ║');
  gotoXY(17,19);write('╚════════════════════════════════════════════╝');
  gotoXY(30,10);write(answer:2);

  REPEAT
   gotoXY(35,18);write('      ');
   gotoXY(35,18);choice1 := upcase(readkey);
  UNTIL choice1 in valid2;

  CASE choice1 OF
   '1' : BEGIN
          gotoXY(30,10);write('        ');
          gotoXY(31,10);readln(answer);
         END;
   '2' : BEGIN
          pplot := 'Y';
          produce;
         END;
   '3' : BEGIN
          pplot := 'N';
          produce;
         END;
   'X' : exit4 := true;
  END;
 UNTIL exit4;

END;
```

```
(*******************************************************************)
procedure open_loop;
(*******************************************************************)

BEGIN
 exit2 := false;
 REPEAT
  clrscr;
  gotoXY(17,04);write('╔═══════════════════════════════════════════╗');
  gotoXY(17,05);write('║            OPEN LOOP SIMULATION            ║');
  gotoXY(17,06);write('║                                            ║');
  gotoXY(17,07);write('║              SELECT INPUT STEP             ║');
  gotoXY(17,08);write('║                                            ║');
  gotoXY(17,09);write('║       0  : Step none of the Inputs         ║');
  gotoXY(17,10);write('║       1  : Step Speed Input 1              ║');
  gotoXY(17,11);write('║       2  : Step Speed Input 2              ║');
  gotoXY(17,12);write('║       3  : Step Speed Input 3              ║');
  gotoXY(17,13);write('║       4  : Step Speed Input 4              ║');
  gotoXY(17,14);write('║       5  : Step Speed Input 5              ║');
  gotoXY(17,15);write('║       6  : Step Speed Input 6              ║');
  gotoXY(17,16);write('║       7  : Step Speed Input 7              ║');
  gotoXY(17,17);write('║       8  : Step Speed Input 8              ║');
  gotoXY(17,18);write('║       X  : Exit                            ║');
  gotoXY(17,19);write('║                                            ║');
  gotoXY(17,20);write('║            Enter Step no :                 ║');
  gotoXY(17,21);write('╚═══════════════════════════════════════════╝');

  REPEAT
   gotoXY(47,20);write('      ');
   gotoXY(47,20);stepno := upcase(readkey);
  UNTIL stepno in valid1;

  CASE stepno OF
  '0','1','2','3','4','5','6','7','8' : prd_quota;
  'X' : exit2 := true;
  END;

UNTIL exit2;
END;




(*******************************************************************)
procedure clsd_lp;
(*******************************************************************)

BEGIN
 exit3 := false;
 REPEAT
```

```
  clrscr;
  gotoXY(17,04);write('╔═════════════════════════════════════════════╗');
  gotoXY(17,05);write('║          CLOSED LOOP SIMULATION             ║');
  gotoXY(17,06);write('║                                             ║');
  gotoXY(17,07);write('║           SELECT SETPOINT STEP              ║');
  gotoXY(17,08);write('║                                             ║');
  gotoXY(17,09);write('║      0  : Step none of the Setpoints        ║');
  gotoXY(17,10);write('║      1  : Step Current Setpoint 1           ║');
  gotoXY(17,11);write('║      2  : Step Current Setpoint 2           ║');
  gotoXY(17,12);write('║      3  : Step Current Setpoint 3           ║');
  gotoXY(17,13);write('║      4  : Step Loop Height Setpoint 1       ║');
  gotoXY(17,14);write('║      5  : Step Loop Height Setpoint 2       ║');
  gotoXY(17,15);write('║      6  : Step Loop Height Setpoint 3       ║');
  gotoXY(17,16);write('║      7  : Step Loop Height Setpoint 4       ║');
  gotoXY(17,17);write('║      8  : Step Speed Setpoint 1             ║');
  gotoXY(17,18);write('║      x  : Exit                              ║');
  gotoXY(17,19);write('║                                             ║');
  gotoXY(17,20);write('║           Enter Step no :                   ║');
  gotoXY(17,21);write('╚═════════════════════════════════════════════╝');

  REPEAT
   gotoXY(47,20);write('      ');
   gotoXY(47,20);stepno := upcase(readkey);
  UNTIL stepno in valid1;

  CASE stepno OF
  '0','1','2','3','4','5','6','7','8' : prd_quota;
  'X' : exit3 := true;
  END;

UNTIL exit3;
END;



(*******************************************************************)
procedure pi_drv1;
(*******************************************************************)

BEGIN
 exit6  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔═════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                             ║');
  gotoXY(17,09);write('║    SELECT PI CONSTANTS FOR DRIVE NO. 1      ║');
  gotoXY(17,10);write('║                                             ║');
  gotoXY(17,11);write('║        K  :                                 ║');
  gotoXY(17,12);write('║        I  :                                 ║');
  gotoXY(17,13);write('║                                             ║');
  gotoXY(17,14);write('║        x  : Exit                            ║');
  gotoXY(17,15);write('╟─────────────────────────────────────────────╢');
  gotoXY(17,16);write('║        OPTION :                             ║');
  gotoXY(17,17);write('╚═════════════════════════════════════════════╝');
```

```
  gotoXY(30,11);write(k1:6:2);
  gotoXY(30,12);write(I1:6:2);

  REPEAT
   gotoXY(35,16);write('        ');
   gotoXY(35,16);chpi1 := upcase(readkey);
  UNTIL chpi1 in valid3;

  CASE chpi1 OF
   'K' : BEGIN
          gotoXY(30,11);write('          ');
          gotoXY(31,11);readln(k1);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('          ');
          gotoXY(31,12);readln(I1);
         END;
   'X' : exit6 := true;
  END;
 UNTIL exit6;
END;




(*******************************************************************)
procedure pi_drv2;
(*******************************************************************)

BEGIN
 exit7  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔═════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                         ║');
  gotoXY(17,09);write('║   SELECT PI CONSTANTS FOR DRIVE NO. 2   ║');
  gotoXY(17,10);write('║                                         ║');
  gotoXY(17,11);write('║       K  :                              ║');
  gotoXY(17,12);write('║       I  :                              ║');
  gotoXY(17,13);write('║                                         ║');
  gotoXY(17,14);write('║       X  : Exit                         ║');
  gotoXY(17,15);write('╟─────────────────────────────────────────╢');
  gotoXY(17,16);write('║       OPTION :                          ║');
  gotoXY(17,17);write('╚═════════════════════════════════════════╝');
  gotoXY(30,11);write(k2:6:2);
  gotoXY(30,12);write(I2:6:2);

  REPEAT
   gotoXY(35,16);write('        ');
   gotoXY(35,16);chpi2 := upcase(readkey);
  UNTIL chpi2 in valid3;

  CASE chpi2 OF
   'K' : BEGIN
```

```
         gotoXY(30,11);write('         ');
         gotoXY(31,11);readln(k2);
        END;
   'I' : BEGIN
         gotoXY(30,12);write('         ');
         gotoXY(31,12);readln(I2);
        END;
   'X' : exit7 := true;
  END;
 UNTIL exit7;
END;



(*******************************************************************)
procedure pi_drv3;
(*******************************************************************)

BEGIN
 exit8  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔══════════════════════════════════════╗');
  gotoXY(17,08);write('║                                      ║');
  gotoXY(17,09);write('║   SELECT PI CONSTANTS FOR DRIVE NO. 3 ║');
  gotoXY(17,10);write('║                                      ║');
  gotoXY(17,11);write('║       K  :                           ║');
  gotoXY(17,12);write('║       I  :                           ║');
  gotoXY(17,13);write('║                                      ║');
  gotoXY(17,14);write('║       x  : Exit                      ║');
  gotoXY(17,15);write('╟──────────────────────────────────────╢');
  gotoXY(17,16);write('║       OPTION :                       ║');
  gotoXY(17,17);write('╚══════════════════════════════════════╝');
  gotoXY(30,11);write(k3:6:2);
  gotoXY(30,12);write(I3:6:2);

  REPEAT
   gotoXY(35,16);write('       ');
   gotoXY(35,16);chpi3 := upcase(readkey);
  UNTIL chpi3 in valid3;

  CASE chpi3 OF
   'K' : BEGIN
         gotoXY(30,11);write('         ');
         gotoXY(31,11);readln(k3);
        END;
   'I' : BEGIN
         gotoXY(30,12);write('         ');
         gotoXY(31,12);readln(I3);
        END;
   'X' : exit8 := true;
  END;
 UNTIL exit8;
```

```
END;



(*****************************************************************)
procedure pi_drv4;
(*****************************************************************)

BEGIN
 exit9  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔══════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                              ║');
  gotoXY(17,09);write('║    SELECT PI CONSTANTS FOR DRIVE NO. 4       ║');
  gotoXY(17,10);write('║                                              ║');
  gotoXY(17,11);write('║        K  :                                  ║');
  gotoXY(17,12);write('║        I  :                                  ║');
  gotoXY(17,13);write('║                                              ║');
  gotoXY(17,14);write('║        x  : Exit                             ║');
  gotoXY(17,15);write('╟──────────────────────────────────────────────╢');
  gotoXY(17,16);write('║        OPTION :                              ║');
  gotoXY(17,17);write('╚══════════════════════════════════════════════╝');
  gotoXY(30,11);write(k4:6:2);
  gotoXY(30,12);write(I4:6:2);

  REPEAT
   gotoXY(35,16);write('      ');
   gotoXY(35,16);chpi4 := upcase(readkey);
  UNTIL chpi4 in valid3;

  CASE chpi4 OF
   'K' : BEGIN
          gotoXY(30,11);write('         ');
          gotoXY(31,11);readln(k4);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('         ');
          gotoXY(31,12);readln(I4);
         END;
   'X' : exit9 := true;
  END;
 UNTIL exit9;
END;


(*****************************************************************)
procedure pi_drv5;
(*****************************************************************)

BEGIN
 exit10  := false;
 REPEAT
  clrscr;
```

```
  gotoXY(17,07);write('╔═══════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                               ║');
  gotoXY(17,09);write('║     SELECT PI CONSTANTS FOR DRIVE NO. 5       ║');
  gotoXY(17,10);write('║                                               ║');
  gotoXY(17,11);write('║         K  :                                  ║');
  gotoXY(17,12);write('║         I  :                                  ║');
  gotoXY(17,13);write('║                                               ║');
  gotoXY(17,14);write('║         x  : Exit                             ║');
  gotoXY(17,15);write('╟───────────────────────────────────────────────╢');
  gotoXY(17,16);write('║         OPTION :                              ║');
  gotoXY(17,17);write('╚═══════════════════════════════════════════════╝');
  gotoXY(30,11);write(k5:6:2);
  gotoXY(30,12);write(I5:6:2);

  REPEAT
   gotoXY(35,16);write('       ');
   gotoXY(35,16);chpi5 := upcase(readkey);
  UNTIL chpi5 in valid3;

  CASE chpi5 OF
   'K' : BEGIN
          gotoXY(30,11);write('         ');
          gotoXY(31,11);readln(k5);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('         ');
          gotoXY(31,12);readln(I5);
         END;
   'X' : exit10 := true;
  END;
 UNTIL exit10;
END;


(*******************************************************************)
procedure pi_drv6;
(*******************************************************************)

BEGIN
 exit11  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔═══════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                               ║');
  gotoXY(17,09);write('║     SELECT PI CONSTANTS FOR DRIVE NO. 6       ║');
  gotoXY(17,10);write('║                                               ║');
  gotoXY(17,11);write('║         K  :                                  ║');
  gotoXY(17,12);write('║         I  :                                  ║');
  gotoXY(17,13);write('║                                               ║');
  gotoXY(17,14);write('║         x  : Exit                             ║');
  gotoXY(17,15);write('╟───────────────────────────────────────────────╢');
  gotoXY(17,16);write('║         OPTION :                              ║');
  gotoXY(17,17);write('╚═══════════════════════════════════════════════╝');
```

```
  gotoXY(30,11);write(k6:6:2);
  gotoXY(30,12);write(I6:6:2);

  REPEAT
   gotoXY(35,16);write('       ');
   gotoXY(35,16);chpi6 := upcase(readkey);
  UNTIL chpi6 in valid3;

  CASE chpi6 OF
   'K' : BEGIN
          gotoXY(30,11);write('         ');
          gotoXY(31,11);readln(k6);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('         ');
          gotoXY(31,12);readln(I6);
         END;
   'X' : exit11 := true;
  END;
 UNTIL exit11;
END;




(******************************************************************)
procedure pi_drv7;
(******************************************************************)

BEGIN
 exit12  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔══════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                          ║');
  gotoXY(17,09);write('║    SELECT PI CONSTANTS FOR DRIVE NO. 7   ║');
  gotoXY(17,10);write('║                                          ║');
  gotoXY(17,11);write('║        K  :                              ║');
  gotoXY(17,12);write('║        I  :                              ║');
  gotoXY(17,13);write('║                                          ║');
  gotoXY(17,14);write('║        x  : Exit                         ║');
  gotoXY(17,15);write('╟──────────────────────────────────────────╢');
  gotoXY(17,16);write('║        OPTION :                          ║');
  gotoXY(17,17);write('╚══════════════════════════════════════════╝');
  gotoXY(30,11);write(k7:6:2);
  gotoXY(30,12);write(I7:6:2);

  REPEAT
   gotoXY(35,16);write('       ');
   gotoXY(35,16);chpi7 := upcase(readkey);
  UNTIL chpi7 in valid3;
```

```
  CASE chpi7 OF
   'K' : BEGIN
          gotoXY(30,11);write('         ');
          gotoXY(31,11);readln(k7);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('         ');
          gotoXY(31,12);readln(I7);
         END;
   'X' : exit12 := true;
  END;
 UNTIL exit12;
END;




(*******************************************************************)
procedure pi_drv8;
(*******************************************************************)

BEGIN
 exit13  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                            ║');
  gotoXY(17,09);write('║    SELECT PI CONSTANTS FOR DRIVE NO. 8     ║');
  gotoXY(17,10);write('║                                            ║');
  gotoXY(17,11);write('║        K  :                                ║');
  gotoXY(17,12);write('║        I  :                                ║');
  gotoXY(17,13);write('║                                            ║');
  gotoXY(17,14);write('║        x  : Exit                           ║');
  gotoXY(17,15);write('╟────────────────────────────────────────────╢');
  gotoXY(17,16);write('║        OPTION :                            ║');
  gotoXY(17,17);write('╚════════════════════════════════════════════╝');
  gotoXY(30,11);write(k8:6:2);
  gotoXY(30,12);write(I8:6:2);

  REPEAT
   gotoXY(35,16);write('       ');
   gotoXY(35,16);chpi8 := upcase(readkey);
  UNTIL chpi8 in valid3;

  CASE chpi8 OF
   'K' : BEGIN
          gotoXY(30,11);write('         ');
          gotoXY(31,11);readln(k8);
         END;
   'I' : BEGIN
          gotoXY(30,12);write('         ');
          gotoXY(31,12);readln(I8);
```

```
        END;
   'X' : exit13 := true;
  END;
 UNTIL exit13;
END;



(*****************************************************************)
procedure dc_cntr;
(*****************************************************************)

BEGIN
 exit14  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                            ║');
  gotoXY(17,09);write('║      SWITCH IN DECOUPLING CONTROLLER       ║');
  gotoXY(17,10);write('║                                            ║');
  gotoXY(17,11);write('║        1 : NO DECOUPLER                    ║');
  gotoXY(17,12);write('║        2 : SWITCHING DECOUPLER             ║');
  gotoXY(17,13);write('║                                            ║');
  gotoXY(17,14);write('║        x : Exit                            ║');
  gotoXY(17,15);write('╟────────────────────────────────────────────╢');
  gotoXY(17,16);write('║        OPTION :                            ║');
  gotoXY(17,17);write('╚════════════════════════════════════════════╝');
  gotoXY(35,16);write(chdc);

  REPEAT
   gotoXY(35,16);chdcc := upcase(readkey);
  UNTIL chdcc in valid5;

  CASE chdcc OF

   '1' : chdc := '1';
   '2' : chdc := '2';
   'X' : exit14 := true;

  END;
 UNTIL exit14;
END;



(*****************************************************************)
procedure lp_dcrs;
(*****************************************************************)

BEGIN
 exit15  := false;
 REPEAT
```

```
  clrscr;
  gotoXY(17,07);write('╔═══════════════════════════════════════╗');
  gotoXY(17,08);write('║                                       ║');
  gotoXY(17,09);write('║      CHANGE LOOP STEP DECREASE TIME   ║');
  gotoXY(17,10);write('║                                       ║');
  gotoXY(17,11);write('║        LOOP 1 :                       ║');
  gotoXY(17,12);write('║        LOOP 2 :                       ║');
  gotoXY(17,13);write('║        LOOP 3 :                       ║');
  gotoXY(17,14);write('║        LOOP 4 :                       ║');
  gotoXY(17,15);write('║                                       ║');
  gotoXY(17,16);write('║        x : Exit                       ║');
  gotoXY(17,17);write('╟───────────────────────────────────────╢');
  gotoXY(17,18);write('║        OPTION :                       ║');
  gotoXY(17,19);write('╚═══════════════════════════════════════╝');
  gotoXY(35,11);write(ltim2^[4]:6:2);
  gotoXY(35,12);write(ltim2^[5]:6:2);
  gotoXY(35,13);write(ltim2^[6]:6:2);
  gotoXY(35,14);write(ltim2^[7]:6:2);

  REPEAT
   gotoXY(35,18);write('    ');
   gotoXY(35,18);timch := upcase(readkey);
  UNTIL timch in valid4;

  CASE timch OF

   '1' : BEGIN
          gotoXY(37,11);write('         ');
          gotoXY(37,11);readln(ltim2^[4]);
         END;
   '2' : BEGIN
          gotoXY(37,12);write('         ');
          gotoXY(37,12);readln(ltim2^[5]);
         END;
   '3' : BEGIN
          gotoXY(37,13);write('         ');
          gotoXY(37,13);readln(ltim2^[6]);
         END;
   '4' : BEGIN
          gotoXY(37,14);write('         ');
          gotoXY(37,14);readln(ltim2^[7]);
         END;
   'X' : exit15 := true;

  END;
 UNTIL exit15;
END;
```

```
(********************************************************************)
procedure ym_chce;
(********************************************************************)

BEGIN
 exit16  := false;
 REPEAT
  clrscr;
  gotoXY(17,07);write('╔════════════════════════════════════════════╗');
  gotoXY(17,08);write('║                                            ║');
  gotoXY(17,09);write('║         VALUE OF YOUNG`S MODULUS           ║');
  gotoXY(17,10);write('║                                            ║');
  gotoXY(17,11);write('║        Y :                                 ║');
  gotoXY(17,12);write('║                                            ║');
  gotoXY(17,13);write('║        x : Exit                            ║');
  gotoXY(17,14);write('╟────────────────────────────────────────────╢');
  gotoXY(17,15);write('║        CHOICE :                            ║');
  gotoXY(17,16);write('╚════════════════════════════════════════════╝');
  gotoXY(33,11);write(ymch:6:2);

  REPEAT
   gotoXY(35,15);ymchc := upcase(readkey);
  UNTIL ymchc in valid5;

  CASE ymchc OF

   'Y' : BEGIN
          gotoXY(33,11);write('        ');
          gotoXY(33,11);readln(ymch);
         END;
   'X' : exit16 := true;

  END;
 UNTIL exit16;
END;


(********************************************************************)
procedure clsd_loop;
(********************************************************************)

BEGIN
 exit5 := false;
 chdc  := '2';
 ymch  := 400;
 ltim2^[4] := 0.78;
 ltim2^[5] := 0.50;
 ltim2^[6] := 0.38;
 ltim2^[7] := 0.20;
 k1 :=  3.0;I1 :=  0.8;
 k2 :=  3.0;I2 :=  0.85;
 k3 :=  3.0;I3 :=  0.95;
 k4 :=  4.25;I4 := 50.0;
 k5 :=  1.18;I5 := 50.0;
```

```
k6 :=  1.50;I6 := 50.0;
k7 :=  1.60;I7 := 50.0;
k8 :=  2.50;I8 :=  0.4;
REPEAT
 clrscr;
 gotoXY(17,04);write('╔════════════════════════════════════════╗');
 gotoXY(17,05);write('║          CLOSED LOOP SIMULATION        ║');
 gotoXY(17,06);write('║                                        ║');
 gotoXY(17,07);write('║           SELECT PI CONSTANTS          ║');
 gotoXY(17,08);write('║                                        ║');
 gotoXY(17,09);write('║        0  : Simulate                   ║');
 gotoXY(17,10);write('║        1  : Drive No. 1                ║');
 gotoXY(17,11);write('║        2  : Drive No. 2                ║');
 gotoXY(17,12);write('║        3  : Drive No. 3                ║');
 gotoXY(17,13);write('║        4  : Drive No. 4                ║');
 gotoXY(17,14);write('║        5  : Drive No. 5                ║');
 gotoXY(17,15);write('║        6  : Drive No. 6                ║');
 gotoXY(17,16);write('║        7  : Drive No. 7                ║');
 gotoXY(17,17);write('║        8  : Drive No. 8                ║');
 gotoXY(17,18);write('║        D  : Switch in Decoupler        ║');
 gotoXY(17,19);write('║        T  : Loop Removal Times         ║');
 gotoXY(17,20);write('║        Y  : Young's Modulus            ║');
 gotoXY(17,21);write('║        x  : Exit                       ║');
 gotoXY(17,22);write('║                                        ║');
 gotoXY(17,23);write('║        Enter Choice :                  ║');
 gotoXY(17,24);write('╚════════════════════════════════════════╝');

 REPEAT
  gotoXY(41,23);write('      ');
  gotoXY(41,23);drivno := upcase(readkey);
 UNTIL drivno in valid1;

 CASE drivno OF
 '0' : clsd_lp;
 '1' : pi_drv1;
 '2' : pi_drv2;
 '3' : pi_drv3;
 '4' : pi_drv4;
 '5' : pi_drv5;
 '6' : pi_drv6;
 '7' : pi_drv7;
 '8' : pi_drv8;
 'D' : dc_cntr;
 'T' : lp_dcrs;
 'Y' : ym_chce;
 'X' : exit5 := true;
 END;

UNTIL exit5;
END;
```

```
(*****************************************************************)
(*                        MAIN PROGRAM                           *)
(*****************************************************************)


BEGIN
clrscr;
setup_pntr;

clrscr;
gotoXY(17,08);write('╔════════════════════════════════════════════╗');
gotoXY(17,09);write('║                                            ║');
gotoXY(17,10);write('║         ROLLING MILL SIMULATION            ║');
gotoXY(17,11);write('║                                            ║');
gotoXY(17,12);write('║       Eight Drive, Hot Rolling Mill        ║');
gotoXY(17,13);write('║                                            ║');
gotoXY(17,14);write('╚════════════════════════════════════════════╝');
gotoXY(17,17);write('       « Press any key to continue »');
gotoXY(65,25);write('@ GAC (1991)');
gotoXY(53,17);exit := readkey;

exit1 := false;
REPEAT
 clrscr;
 gotoXY(17,08);write('╔════════════════════════════════════════════╗');
 gotoXY(17,09);write('║                                            ║');
 gotoXY(17,10);write('║        SELECT OPEN OR CLOSED LOOP          ║');
 gotoXY(17,11);write('║                                            ║');
 gotoXY(17,12);write('║        o  : Open   loop simulation         ║');
 gotoXY(17,13);write('║        c  : Closed loop simulation         ║');
 gotoXY(17,14);write('║        x  : Exit                           ║');
 gotoXY(17,15);write('║                                            ║');
 gotoXY(17,16);write('║             Enter choice :                 ║');
 gotoXY(17,17);write('╚════════════════════════════════════════════╝');

 REPEAT
  gotoXY(46,16);write('     ');
  gotoXY(46,16);choice := upcase(readkey);
 UNTIL choice in valid0;

 CASE choice OF
  'O' : open_loop;
  'C' : clsd_loop;
  'X' : exit1 := true;
 END;
UNTIL exit1;
clrscr;


END.
→
```

Pascal Units used by the Main Program.

```
UNIT data;

{****************************}
         INTERFACE
{****************************}
USES
    plant;

TYPE
    data_var = record
     u : array [0..max_point] of v_ptr;
     y : array [0..max_point] of v_ptr;
     END;
    data_ptr = ^data_var;

    labl_var = record
     head1 : string;
     head2 : string;
     uname : array [1..max_cntrl] of string;
     yname : array [1..max_cntrl] of string;
     ulabl : array [1..max_cntrl] of string;
     ylabl : array [1..max_cntrl] of string;
     xlabl : string;
     END;
    labl_ptr = ^labl_var;

   PROCEDURE data_init (d:data_ptr);
   PROCEDURE data_logg (z:integer;d:data_ptr;u:v_ptr;y:v_ptr);
   PROCEDURE data_plot (d:data_ptr;l:labl_ptr;ch:char);

{****************************}
        IMPLEMENTATION
{****************************}
USES
    crt,graph;

PROCEDURE data_init (d:data_ptr);
VAR
   i : integer;
BEGIN
 FOR i := 0 to max_point do
  BEGIN
   new (d^.u[i]);
   new (d^.y[i]);
  END;
END;

PROCEDURE data_logg (z:integer;d:data_ptr;u:v_ptr;y:v_ptr);
VAR
   r,c : integer;
BEGIN
```

```
 FOR r := 1 to max_cntrl do
  BEGIN
   d^.u[z]^[r] := u^[r];
   d^.y[z]^[r] := y^[r];
  END;
END;



PROCEDURE data_plot (d:data_ptr;l:labl_ptr;ch:char);
VAR
   y_maxi,y_mini : real;
   y_maxo,y_mino : real;
   x_pos,y_pos   : integer;
   i,row    : integer;
   gd,gm    : integer;
   ch1,ch2 : char;
   exit_1   : boolean;
   s : string;
BEGIN
   gd := detect;
   initgraph (gd,gm,'');
   setviewport (0,0,getmaxx,getmaxy,clipon);clearviewport;

   setviewport  (0,0,719,29,clipon);rectangle (0,0,719,29);
   settextstyle (triplexfont,horizdir,2);
   outtextxy    (30,5,l^.head1);
   setviewport  (0,50,719,339,clipon);rectangle (0,0,719,289);

   row := 1;
   exit_1 := false;
   REPEAT
    { Heading }
    setlinestyle (solidln,1,normwidth);
    setviewport  (100,60,700,90,clipon);clearviewport;rectangle (0,0,600,30);
    settextstyle (triplexfont,horizdir,1);
    outtextxy (30,5,l^.head2+' :    [ '+l^.yname[row]+'  vs  '+l^.uname[row]+' ]');

    { Determine y_maxi,y_mini,y_maxo,y_mino }

    y_maxi := 0;
    y_mini := 1000;
    y_maxo := 0;
    y_mino := 1000;
    for i := 0 to max_point do
     BEGIN
      if d^.u[i]^[row] > y_maxi then y_maxi := d^.u[i]^[row];
      if d^.u[i]^[row] < y_mini then y_mini := d^.u[i]^[row];
      if d^.y[i]^[row] > y_maxo then y_maxo := d^.y[i]^[row];
      if d^.y[i]^[row] < y_mino then y_mino := d^.y[i]^[row];
     END;

    IF (ch = 'C') then
     BEGIN
```

```
    IF (row = 1) or (row = 2) or (row =3) then
     BEGIN
      y_mini := (y_maxi - 55);
      y_maxi := (y_maxo + 5);
      y_maxo := (y_maxo + 5);
      y_mino := (y_mini);
     END;
    IF (row = 4) or (row = 5) or (row =6) or (row = 7) then
     BEGIN
      y_mini := y_mini - 0.25;
      y_mino := y_mini;
      y_maxo := y_maxo + 0.25;
      y_maxi := y_maxo;
     END;
   END;

  IF (ch = '0') then
   BEGIN
    IF (row = 1) then
     BEGIN
      y_mini := (y_mini - 5);
      y_maxi := (y_maxi + 5);
      y_maxo := (y_maxo + 5);
      y_mino := (150);
     END;

    IF (row = 2) then
     BEGIN
      y_mini := (y_mini - 5);
      y_maxi := (y_maxi + 5);
      y_maxo := (y_maxo + 5);
      y_mino := (150 );
     END;

    IF (row = 3) then
     BEGIN
      y_mini := (y_mini - 5);
      y_maxi := (y_maxi + 5);
      y_maxo := (y_maxo + 5);
      y_mino := (320 );
     END;

    IF (row = 4) or (row = 5) or (row =6) or (row = 7) then
     BEGIN
      y_mini := y_mini - 5;
      y_mino := y_mino ;
      y_maxo := y_maxo ;
      y_maxi := y_maxi + 5;
     END;
   END;

  IF (row = 8) then
   BEGIN
    y_mini := (y_mino - 0.25);
```

```
  y_maxi := (y_maxo + 0.25);
  y_maxo := (y_maxo + 0.25);
  y_mino := (y_mino - 0.25);
 END;

{ Plot x-axis }
setviewport  (100,305,718,340,clipoff);
settextstyle (defaultfont,horizdir,1);
line (0,10,600,10);
for i := 0 to 10 do
 BEGIN
  line (i*60,5,i*60,15);
  str (i/10*dx*max_point:3:1,s);outtextxy (i*60-30,17,s+l^.xlabl);
 END;

{ Plot y-axis : input }
setviewport  (1,60,45,320,clipon);clearviewport;
settextstyle (defaultfont,horizdir,1);
outtextxy    (20,15,'U');
outtextxy    (5,25,l^.ulabl[row]);
line (35,40,35,240);
for i := 0 to 4 do
 BEGIN
  line (30,40+i*50,40,40+i*50);
  str  (y_maxi-i*0.25*(y_maxi-y_mini):4:2,s);
  outtextxy ( 0,45+i*50,s);
 END;

{ Plot y-axis : output }
setviewport  (46,60,90,320,clipon);clearviewport;
settextstyle (defaultfont,horizdir,1);
outtextxy    (20,15,'Y');
outtextxy    (5,25,l^.ylabl[row]);
line (35,40,35,240);
for i := 0 to 4 do
 BEGIN
  line (30,40+i*50,40,40+i*50);
  str  (y_maxo-i*0.25*(y_maxo-y_mino):4:2,s);
  outtextxy ( 0,45+i*50,s);
 END;

{ Plot graph }
setviewport  (100,100,700,300,clipon);clearviewport;
rectangle (0,0,600,200);

{ Plot grids }
setlinestyle (dottedln,1,normwidth);
for i := 1 to 19 do line (0,10*i,600,10*i);
for i := 1 to 19 do line (i*30,0,i*30,200);

{ Plot input }
x_pos := 0;
y_pos := 200 - round(200*((d^.u[0]^[row]-y_mini)/(y_maxi-y_mini)));
moveto (x_pos,y_pos);
```

```
    setlinestyle (dashedln,1,thickwidth);
    for i := 1 to max_point do
     BEGIN
      x_pos := i * (600 div max_point);
      y_pos := 200 - round(200*((d^.u[i]^[row]-y_mini)/(y_maxi-y_mini)));
      lineto (x_pos,y_pos);
     END;

    { Plot output }
    x_pos := 0;
    y_pos := 200 - round(200*((d^.y[0]^[row]-y_mino)/(y_maxo-y_mino)));
    moveto (x_pos,y_pos);
    setlinestyle (solidln,1,thickwidth);
    for i := 1 to max_point do
     BEGIN
      x_pos := i * (600 div max_point);
      y_pos := 200 - round(200*((d^.y[i]^[row]-y_mino)/(y_maxo-y_mino)));
      lineto (x_pos,y_pos);
     END;

    REPEAT
     ch1 := upcase (readkey);
     row := ord(ch1) - ord('0');
    UNTIL (row >= 0) AND (row <= max_cntrl);
    CASE ch1 of
     '0' : exit_1 := true;
     END;
   UNTIL exit_1;

   setviewport (0,0,getmaxx,getmaxy,clipon);clearviewport;
   closegraph;
END;

{****************************************}
{***    INITIALIZATION  SECTION      ***}
{****************************************}
BEGIN
END.
```

→

```
UNIT decoup;

(************************)
        INTERFACE
(************************)

USES
    plant;

    PROCEDURE nint_cnt (i: mtrx_ptr);
    PROCEDURE i1_2_cnt (i: mtrx_ptr);
    PROCEDURE i1_3_cnt (i: mtrx_ptr);
    PROCEDURE i1_4_cnt (i: mtrx_ptr);
    PROCEDURE i1_5_cnt (i: mtrx_ptr);
    PROCEDURE i1_6_cnt (i: mtrx_ptr);
    PROCEDURE i1_7_cnt (i: mtrx_ptr);
    PROCEDURE i1_8_cnt (i: mtrx_ptr);
    PROCEDURE i2_8_cnt (i: mtrx_ptr);
    PROCEDURE i3_8_cnt (i: mtrx_ptr);
    PROCEDURE i4_8_cnt (i: mtrx_ptr);
    PROCEDURE i5_8_cnt (i: mtrx_ptr);
    PROCEDURE i6_8_cnt (i: mtrx_ptr);
    PROCEDURE i7_8_cnt (i: mtrx_ptr);
    PROCEDURE i8_8_cnt (i: mtrx_ptr);


(************************)
      IMPLEMENTATION
(************************)



(*****************************************************************)
procedure nint_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[1,1]^.nsz := 0;
 i^[1,1]^.dsz := 0;
 i^[1,1]^.num[0] := 1.0;
 i^[1,1]^.den[0] := 1.0;

 i^[2,2]^.nsz := 0;
 i^[2,2]^.dsz := 0;
 i^[2,2]^.num[0] := 1.0;
 i^[2,2]^.den[0] := 1.0;

 i^[3,3]^.nsz := 0;
 i^[3,3]^.dsz := 0;
 i^[3,3]^.num[0] := 1.0;
 i^[3,3]^.den[0] := 1.0;
```

```
 i^[4,4]^.nsz := 0;
 i^[4,4]^.dsz := 0;
 i^[4,4]^.num[0] := 1.0;
 i^[4,4]^.den[0] := 1.0;

 i^[5,5]^.nsz := 0;
 i^[5,5]^.dsz := 0;
 i^[5,5]^.num[0] := 1.0;
 i^[5,5]^.den[0] := 1.0;

 i^[6,6]^.nsz := 0;
 i^[6,6]^.dsz := 0;
 i^[6,6]^.num[0] := 1.0;
 i^[6,6]^.den[0] := 1.0;

 i^[7,7]^.nsz := 0;
 i^[7,7]^.dsz := 0;
 i^[7,7]^.num[0] := 1.0;
 i^[7,7]^.den[0] := 1.0;

 i^[8,8]^.nsz := 0;
 i^[8,8]^.dsz := 0;
 i^[8,8]^.num[0] := 1.0;
 i^[8,8]^.den[0] := 1.0;

END;



(*******************************************************************)
procedure i1_2_cnt (i: mtrx_ptr);
(*******************************************************************)

BEGIN

 i^[1,2]^.nsz := 0;
 i^[1,2]^.dsz := 0;
 i^[1,2]^.num[0] := 0.0651;
 i^[1,2]^.den[0] := 1.0;

 i^[2,1]^.nsz := 0;
 i^[2,1]^.dsz := 0;
 i^[2,1]^.num[0] := 0.0667;
 i^[2,1]^.den[0] := 1.0;

 i^[2,2]^.nsz := 0;
 i^[2,2]^.dsz := 0;
 i^[2,2]^.num[0] := 1.004342;
 i^[2,2]^.den[0] := 1.0;

END;
```

```
(*******************************************************************)
procedure i1_3_cnt (i: mtrx_ptr);
(*******************************************************************)

BEGIN

 i^[1,2]^.nsz := 0;
 i^[1,2]^.dsz := 0;
 i^[1,2]^.num[0] := 0.1226;
 i^[1,2]^.den[0] := 1.0;

 i^[1,3]^.nsz := 0;
 i^[1,3]^.dsz := 0;
 i^[1,3]^.num[0] := 0.0559;
 i^[1,3]^.den[0] := 1.0;

 i^[2,1]^.nsz := 0;
 i^[2,1]^.dsz := 0;
 i^[2,1]^.num[0] := 0.1452;
 i^[2,1]^.den[0] := 1.0;

 i^[2,2]^.nsz := 0;
 i^[2,2]^.dsz := 0;
 i^[2,2]^.num[0] := 1.017802;
 i^[2,2]^.den[0] := 1.0;

 i^[2,3]^.nsz := 0;
 i^[2,3]^.dsz := 0;
 i^[2,3]^.num[0] := 0.07581668;
 i^[2,3]^.den[0] := 1.0;

 i^[3,1]^.nsz := 0;
 i^[3,1]^.dsz := 0;
 i^[3,1]^.num[0] := 0.09069868;
 i^[3,1]^.den[0] := 1.0;

 i^[3,2]^.nsz := 0;
 i^[3,2]^.dsz := 0;
 i^[3,2]^.num[0] := 0.1020197;
 i^[3,2]^.den[0] := 1.0;

 i^[3,3]^.nsz := 0;
 i^[3,3]^.dsz := 0;
 i^[3,3]^.num[0] := 1.011224;
 i^[3,3]^.den[0] := 1.0;

END;
```

```
(****************************************************************)
procedure il_4_cnt (i: mtrx_ptr);
(****************************************************************)

BEGIN

 i^[1,2]^.nsz := 0;
 i^[1,2]^.dsz := 0;
 i^[1,2]^.num[0] := 0.4301;
 i^[1,2]^.den[0] := 1.0;

 i^[1,3]^.nsz := 0;
 i^[1,3]^.dsz := 0;
 i^[1,3]^.num[0] := 0.3629;
 i^[1,3]^.den[0] := 1.0;

 i^[1,4]^.nsz := 0;
 i^[1,4]^.dsz := 0;
 i^[1,4]^.num[0] := 0.4428;
 i^[1,4]^.den[0] := 1.0;

 i^[2,1]^.nsz := 0;
 i^[2,1]^.dsz := 0;
 i^[2,1]^.num[0] := 0.4693;
 i^[2,1]^.den[0] := 1.0;

 i^[2,2]^.nsz := 0;
 i^[2,2]^.dsz := 0;
 i^[2,2]^.num[0] := 1.201846;
 i^[2,2]^.den[0] := 1.0;

 i^[2,3]^.nsz := 0;
 i^[2,3]^.dsz := 0;
 i^[2,3]^.num[0] := 0.443109;
 i^[2,3]^.den[0] := 1.0;

 i^[2,4]^.nsz := 0;
 i^[2,4]^.dsz := 0;
 i^[2,4]^.num[0] := 0.537406;
 i^[2,4]^.den[0] := 1.0;

 i^[3,1]^.nsz := 0;
 i^[3,1]^.dsz := 0;
 i^[3,1]^.num[0] := 0.5115778;
 i^[3,1]^.den[0] := 1.0;

 i^[3,2]^.nsz := 0;
 i^[3,2]^.dsz := 0;
 i^[3,2]^.num[0] := 0.5660296;
 i^[3,2]^.den[0] := 1.0;

 i^[3,3]^.nsz := 0;
 i^[3,3]^.dsz := 0;
```

```
 i^[3,3]^.num[0] := 1.28004;
 i^[3,3]^.den[0] := 1.0;

 i^[3,4]^.nsz := 0;
 i^[3,4]^.dsz := 0;
 i^[3,4]^.num[0] := 0.6611682;
 i^[3,4]^.den[0] := 1.0;

END;



(******************************************************************)
procedure i1_5_cnt (i: mtrx_ptr);
(******************************************************************)

BEGIN

 i^[1,2]^.nsz := 0;
 i^[1,2]^.dsz := 0;
 i^[1,2]^.num[0] := 0.1222;
 i^[1,2]^.den[0] := 1.0;

 i^[1,3]^.nsz := 0;
 i^[1,3]^.dsz := 0;
 i^[1,3]^.num[0] := 0.0573;
 i^[1,3]^.den[0] := 1.0;

 i^[1,4]^.nsz := 0;
 i^[1,4]^.dsz := 0;
 i^[1,4]^.num[0] := 0.1921;
 i^[1,4]^.den[0] := 1.0;

 i^[1,5]^.nsz := 0;
 i^[1,5]^.dsz := 0;
 i^[1,5]^.num[0] := 0.4333;
 i^[1,5]^.den[0] := 1.0;

 i^[2,1]^.nsz := 0;
 i^[2,1]^.dsz := 0;
 i^[2,1]^.num[0] := 0.141;
 i^[2,1]^.den[0] := 1.0;

 i^[2,2]^.nsz := 0;
 i^[2,2]^.dsz := 0;
 i^[2,2]^.num[0] := 1.01723;
 i^[2,2]^.den[0] := 1.0;

 i^[2,3]^.nsz := 0;
 i^[2,3]^.dsz := 0;
 i^[2,3]^.num[0] := 0.0729793;
 i^[2,3]^.den[0] := 1.0;

 i^[2,4]^.nsz := 0;
```

```
i^[2,4]^.dsz := 0;
i^[2,4]^.num[0] := 0.2331861;
i^[2,4]^.den[0] := 1.0;

i^[2,5]^.nsz := 0;
i^[2,5]^.dsz := 0;
i^[2,5]^.num[0] := 0.5258953;
i^[2,5]^.den[0] := 1.0;

i^[3,1]^.nsz := 0;
i^[3,1]^.dsz := 0;
i^[3,1]^.num[0] := 0.0787596;
i^[3,1]^.den[0] := 1.0;

i^[3,2]^.nsz := 0;
i^[3,2]^.dsz := 0;
i^[3,2]^.num[0] := 0.08522442;
i^[3,2]^.den[0] := 1.0;

i^[3,3]^.nsz := 0;
i^[3,3]^.dsz := 0;
i^[3,3]^.num[0] := 1.009419;
i^[3,3]^.den[0] := 1.0;

i^[3,4]^.nsz := 0;
i^[3,4]^.dsz := 0;
i^[3,4]^.num[0] := 0.2869109;
i^[3,4]^.den[0] := 1.0;

i^[3,5]^.nsz := 0;
i^[3,5]^.dsz := 0;
i^[3,5]^.num[0] := 0.6469654;
i^[3,5]^.den[0] := 1.0;

i^[4,1]^.nsz := 0;
i^[4,1]^.dsz := 0;
i^[4,1]^.num[0] := -0.9958574;
i^[4,1]^.den[0] := 1.0;

i^[4,2]^.nsz := 0;
i^[4,2]^.dsz := 0;
i^[4,2]^.num[0] := -0.9652869;
i^[4,2]^.den[0] := 1.0;

i^[4,3]^.nsz := 0;
i^[4,3]^.dsz := 0;
i^[4,3]^.num[0] := -0.8961118;
i^[4,3]^.den[0] := 1.0;

i^[4,4]^.nsz := 0;
i^[4,4]^.dsz := 0;
i^[4,4]^.num[0] := 0.4338936;
i^[4,4]^.den[0] := 1.0;
```

```
 i^[4,5]^.nsz := 0;
 i^[4,5]^.dsz := 0;
 i^[4,5]^.num[0] := 0.9783028;
 i^[4,5]^.den[0] := 1.0;

END;


(*****************************************************************)
procedure i1_6_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[1,6]^.nsz := 0;
 i^[1,6]^.dsz := 0;
 i^[1,6]^.num[0] := 0.4091;
 i^[1,6]^.den[0] := 1.0;

 i^[2,6]^.nsz := 0;
 i^[2,6]^.dsz := 0;
 i^[2,6]^.num[0] := 0.4964831;
 i^[2,6]^.den[0] := 1.0;

 i^[3,6]^.nsz := 0;
 i^[3,6]^.dsz := 0;
 i^[3,6]^.num[0] := 0.6103938;
 i^[3,6]^.den[0] := 1.0;

 i^[4,6]^.nsz := 0;
 i^[4,6]^.dsz := 0;
 i^[4,6]^.num[0] := 0.9239825;
 i^[4,6]^.den[0] := 1.0;

 i^[5,6]^.nsz := 0;
 i^[5,6]^.dsz := 0;
 i^[5,6]^.num[0] := 0.944;
 i^[5,6]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i1_7_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[1,7]^.nsz := 0;
 i^[1,7]^.dsz := 0;
 i^[1,7]^.num[0] := 0.3601;
 i^[1,7]^.den[0] := 1.0;
```

```
 i^[2,7]^.nsz := 0;
 i^[2,7]^.dsz := 0;
 i^[2,7]^.num[0] := 0.4369741;
 i^[2,7]^.den[0] := 1.0;

 i^[3,7]^.nsz := 0;
 i^[3,7]^.dsz := 0;
 i^[3,7]^.num[0] := 0.537658;
 i^[3,7]^.den[0] := 1.0;

 i^[4,7]^.nsz := 0;
 i^[4,7]^.dsz := 0;
 i^[4,7]^.num[0] := 0.8130536;
 i^[4,7]^.den[0] := 1.0;

 i^[5,7]^.nsz := 0;
 i^[5,7]^.dsz := 0;
 i^[5,7]^.num[0] := 0.8313;
 i^[5,7]^.den[0] := 1.0;

 i^[6,7]^.nsz := 0;
 i^[6,7]^.dsz := 0;
 i^[6,7]^.num[0] := 0.8806;
 i^[6,7]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i1_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[1,8]^.nsz := 0;
 i^[1,8]^.dsz := 0;
 i^[1,8]^.num[0] := 0.3071;
 i^[1,8]^.den[0] := 1.0;

 i^[2,8]^.nsz := 0;
 i^[2,8]^.dsz := 0;
 i^[2,8]^.num[0] := 0.3727011;
 i^[2,8]^.den[0] := 1.0;

 i^[3,8]^.nsz := 0;
 i^[3,8]^.dsz := 0;
 i^[3,8]^.num[0] := 0.4584897;
 i^[3,8]^.den[0] := 1.0;

 i^[4,8]^.nsz := 0;
 i^[4,8]^.dsz := 0;
 i^[4,8]^.num[0] := 0.6932003;
 i^[4,8]^.den[0] := 1.0;
```

```
 i^[5,8]^.nsz := 0;
 i^[5,8]^.dsz := 0;
 i^[5,8]^.num[0] := 0.7088;
 i^[5,8]^.den[0] := 1.0;

 i^[6,8]^.nsz := 0;
 i^[6,8]^.dsz := 0;
 i^[6,8]^.num[0] := 0.7509;
 i^[6,8]^.den[0] := 1.0;

 i^[7,8]^.nsz := 0;
 i^[7,8]^.dsz := 0;
 i^[7,8]^.num[0] := 0.8527;
 i^[7,8]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i2_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[1,2]^.nsz := 0;
 i^[1,2]^.dsz := 0;
 i^[1,2]^.num[0] := 0.0;
 i^[1,2]^.den[0] := 1.0;

 i^[1,3]^.nsz := 0;
 i^[1,3]^.dsz := 0;
 i^[1,3]^.num[0] := 0.0;
 i^[1,3]^.den[0] := 1.0;

 i^[1,4]^.nsz := 0;
 i^[1,4]^.dsz := 0;
 i^[1,4]^.num[0] := 0.0;
 i^[1,4]^.den[0] := 1.0;

 i^[1,5]^.nsz := 0;
 i^[1,5]^.dsz := 0;
 i^[1,5]^.num[0] := 0.0;
 i^[1,5]^.den[0] := 1.0;

 i^[1,6]^.nsz := 0;
 i^[1,6]^.dsz := 0;
 i^[1,6]^.num[0] := 0.0;
 i^[1,6]^.den[0] := 1.0;

 i^[1,7]^.nsz := 0;
 i^[1,7]^.dsz := 0;
 i^[1,7]^.num[0] := 0.0;
```

```
i^[1,7]^.den[0] := 1.0;

i^[1,8]^.nsz := 0;
i^[1,8]^.dsz := 0;
i^[1,8]^.num[0] := 0.0;
i^[1,8]^.den[0] := 1.0;

i^[2,1]^.nsz := 0;
i^[2,1]^.dsz := 0;
i^[2,1]^.num[0] := 0.0;
i^[2,1]^.den[0] := 1.0;

i^[2,2]^.nsz := 0;
i^[2,2]^.dsz := 0;
i^[2,2]^.num[0] := 1.0;
i^[2,2]^.den[0] := 1.0;

i^[2,3]^.nsz := 0;
i^[2,3]^.dsz := 0;
i^[2,3]^.num[0] := 0.0483;
i^[2,3]^.den[0] := 1.0;

i^[2,4]^.nsz := 0;
i^[2,4]^.dsz := 0;
i^[2,4]^.num[0] := 0.2506;
i^[2,4]^.den[0] := 1.0;

i^[2,5]^.nsz := 0;
i^[2,5]^.dsz := 0;
i^[2,5]^.num[0] := 0.5058;
i^[2,5]^.den[0] := 1.0;

i^[2,6]^.nsz := 0;
i^[2,6]^.dsz := 0;
i^[2,6]^.num[0] := 0.4774;
i^[2,6]^.den[0] := 1.0;

i^[2,7]^.nsz := 0;
i^[2,7]^.dsz := 0;
i^[2,7]^.num[0] := 0.4203;
i^[2,7]^.den[0] := 1.0;

i^[2,8]^.nsz := 0;
i^[2,8]^.dsz := 0;
i^[2,8]^.num[0] := 0.3584;
i^[2,8]^.den[0] := 1.0;

i^[3,1]^.nsz := 0;
i^[3,1]^.dsz := 0;
i^[3,1]^.num[0] := 0.0;
i^[3,1]^.den[0] := 1.0;

i^[3,2]^.nsz := 0;
i^[3,2]^.dsz := 0;
```

```
i^[3,2]^.num[0] := 0.0801;
i^[3,2]^.den[0] := 1.0;

i^[3,3]^.nsz := 0;
i^[3,3]^.dsz := 0;
i^[3,3]^.num[0] := 1.003869;
i^[3,3]^.den[0] := 1.0;

i^[3,4]^.nsz := 0;
i^[3,4]^.dsz := 0;
i^[3,4]^.num[0] := 0.3079731;
i^[3,4]^.den[0] := 1.0;

i^[3,5]^.nsz := 0;
i^[3,5]^.dsz := 0;
i^[3,5]^.num[0] := 0.6216146;
i^[3,5]^.den[0] := 1.0;

i^[3,6]^.nsz := 0;
i^[3,6]^.dsz := 0;
i^[3,6]^.num[0] := 0.5867397;
i^[3,6]^.den[0] := 1.0;

i^[3,7]^.nsz := 0;
i^[3,7]^.dsz := 0;
i^[3,7]^.num[0] := 0.516566;
i^[3,7]^.den[0] := 1.0;

i^[3,8]^.nsz := 0;
i^[3,8]^.dsz := 0;
i^[3,8]^.num[0] := 0.4405078;
i^[3,8]^.den[0] := 1.0;

i^[4,1]^.nsz := 0;
i^[4,1]^.dsz := 0;
i^[4,1]^.num[0] := 0.0;
i^[4,1]^.den[0] := 1.0;

i^[4,2]^.nsz := 0;
i^[4,2]^.dsz := 0;
i^[4,2]^.num[0] := -0.934077;
i^[4,2]^.den[0] := 1.0;

i^[4,3]^.nsz := 0;
i^[4,3]^.dsz := 0;
i^[4,3]^.num[0] := -0.9749159;
i^[4,3]^.den[0] := 1.0;

i^[4,4]^.nsz := 0;
i^[4,4]^.dsz := 0;
i^[4,4]^.num[0] := 0.4982309;
i^[4,4]^.den[0] := 1.0;

i^[4,5]^.nsz := 0;
```

```
 i^[4,5]^.dsz := 0;
 i^[4,5]^.num[0] := 1.005237;
 i^[4,5]^.den[0] := 1.0;

 i^[4,6]^.nsz := 0;
 i^[4,6]^.dsz := 0;
 i^[4,6]^.num[0] := 0.9490764;
 i^[4,6]^.den[0] := 1.0;

 i^[4,7]^.nsz := 0;
 i^[4,7]^.dsz := 0;
 i^[4,7]^.num[0] := 0.8354071;
 i^[4,7]^.den[0] := 1.0;

 i^[4,8]^.nsz := 0;
 i^[4,8]^.dsz := 0;
 i^[4,8]^.num[0] := 0.7123351;
 i^[4,8]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i3_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[2,3]^.nsz := 0;
 i^[2,3]^.dsz := 0;
 i^[2,3]^.num[0] := 0.0;
 i^[2,3]^.den[0] := 1.0;

 i^[2,4]^.nsz := 0;
 i^[2,4]^.dsz := 0;
 i^[2,4]^.num[0] := 0.0;
 i^[2,4]^.den[0] := 1.0;

 i^[2,5]^.nsz := 0;
 i^[2,5]^.dsz := 0;
 i^[2,5]^.num[0] := 0.0;
 i^[2,5]^.den[0] := 1.0;

 i^[2,6]^.nsz := 0;
 i^[2,6]^.dsz := 0;
 i^[2,6]^.num[0] := 0.0;
 i^[2,6]^.den[0] := 1.0;

 i^[2,7]^.nsz := 0;
 i^[2,7]^.dsz := 0;
 i^[2,7]^.num[0] := 0.0;
 i^[2,7]^.den[0] := 1.0;
```

```
i^[2,8]^.nsz := 0;
i^[2,8]^.dsz := 0;
i^[2,8]^.num[0] := 0.0;
i^[2,8]^.den[0] := 1.0;

i^[3,2]^.nsz := 0;
i^[3,2]^.dsz := 0;
i^[3,2]^.num[0] := 0.0;
i^[3,2]^.den[0] := 1.0;

i^[3,3]^.nsz := 0;
i^[3,3]^.dsz := 0;
i^[3,3]^.num[0] := 1.0;
i^[3,3]^.den[0] := 1.0;

i^[3,4]^.nsz := 0;
i^[3,4]^.dsz := 0;
i^[3,4]^.num[0] := 0.4922;
i^[3,4]^.den[0] := 1.0;

i^[3,5]^.nsz := 0;
i^[3,5]^.dsz := 0;
i^[3,5]^.num[0] := 0.4528;
i^[3,5]^.den[0] := 1.0;

i^[3,6]^.nsz := 0;
i^[3,6]^.dsz := 0;
i^[3,6]^.num[0] := 0.4275;
i^[3,6]^.den[0] := 1.0;

i^[3,7]^.nsz := 0;
i^[3,7]^.dsz := 0;
i^[3,7]^.num[0] := 0.3764;
i^[3,7]^.den[0] := 1.0;

i^[3,8]^.nsz := 0;
i^[3,8]^.dsz := 0;
i^[3,8]^.num[0] := 0.3209;
i^[3,8]^.den[0] := 1.0;

i^[4,2]^.nsz := 0;
i^[4,2]^.dsz := 0;
i^[4,2]^.num[0] := 0.0;
i^[4,2]^.den[0] := 1.0;

i^[4,3]^.nsz := 0;
i^[4,3]^.dsz := 0;
i^[4,3]^.num[0] := -0.544;
i^[4,3]^.den[0] := 1.0;

i^[4,4]^.nsz := 0;
i^[4,4]^.dsz := 0;
i^[4,4]^.num[0] := 0.7322432;
i^[4,4]^.den[0] := 1.0;
```

```
 i^[4,5]^.nsz := 0;
 i^[4,5]^.dsz := 0;
 i^[4,5]^.num[0] := 0.6736768;
 i^[4,5]^.den[0] := 1.0;

 i^[4,6]^.nsz := 0;
 i^[4,6]^.dsz := 0;
 i^[4,6]^.num[0] := 0.63594;
 i^[4,6]^.den[0] := 1.0;

 i^[4,7]^.nsz := 0;
 i^[4,7]^.dsz := 0;
 i^[4,7]^.num[0] := 0.5600384;
 i^[4,7]^.den[0] := 1.0;

 i^[4,8]^.nsz := 0;
 i^[4,8]^.dsz := 0;
 i^[4,8]^.num[0] := 0.4775304;
 i^[4,8]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i4_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[3,4]^.nsz := 0;
 i^[3,4]^.dsz := 0;
 i^[3,4]^.num[0] := 0.0;
 i^[3,4]^.den[0] := 1.0;

 i^[3,5]^.nsz := 0;
 i^[3,5]^.dsz := 0;
 i^[3,5]^.num[0] := 0.0;
 i^[3,5]^.den[0] := 1.0;

 i^[3,6]^.nsz := 0;
 i^[3,6]^.dsz := 0;
 i^[3,6]^.num[0] := 0.0;
 i^[3,6]^.den[0] := 1.0;

 i^[3,7]^.nsz := 0;
 i^[3,7]^.dsz := 0;
 i^[3,7]^.num[0] := 0.0;
 i^[3,7]^.den[0] := 1.0;

 i^[3,8]^.nsz := 0;
 i^[3,8]^.dsz := 0;
 i^[3,8]^.num[0] := 0.0;
```

```
 i^[3,8]^.den[0] := 1.0;

 i^[4,3]^.nsz := 0;
 i^[4,3]^.dsz := 0;
 i^[4,3]^.num[0] := 0.0;
 i^[4,3]^.den[0] := 1.0;

 i^[4,4]^.nsz := 0;
 i^[4,4]^.dsz := 0;
 i^[4,4]^.num[0] := 1.0;
 i^[4,4]^.den[0] := 1.0;

 i^[4,5]^.nsz := 0;
 i^[4,5]^.dsz := 0;
 i^[4,5]^.num[0] := 0.92;
 i^[4,5]^.den[0] := 1.0;

 i^[4,6]^.nsz := 0;
 i^[4,6]^.dsz := 0;
 i^[4,6]^.num[0] := 0.8685;
 i^[4,6]^.den[0] := 1.0;

 i^[4,7]^.nsz := 0;
 i^[4,7]^.dsz := 0;
 i^[4,7]^.num[0] := 0.7648;
 i^[4,7]^.den[0] := 1.0;

 i^[4,8]^.nsz := 0;
 i^[4,8]^.dsz := 0;
 i^[4,8]^.num[0] := 0.6521;
 i^[4,8]^.den[0] := 1.0;

END;



(*****************************************************************)
procedure i5_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[4,5]^.nsz := 0;
 i^[4,5]^.dsz := 0;
 i^[4,5]^.num[0] := 0.0;
 i^[4,5]^.den[0] := 1.0;

 i^[4,6]^.nsz := 0;
 i^[4,6]^.dsz := 0;
 i^[4,6]^.num[0] := 0.0;
 i^[4,6]^.den[0] := 1.0;

 i^[4,7]^.nsz := 0;
 i^[4,7]^.dsz := 0;
```

```
 i^[4,7]^.num[0] := 0.0;
 i^[4,7]^.den[0] := 1.0;

 i^[4,8]^.nsz := 0;
 i^[4,8]^.dsz := 0;
 i^[4,8]^.num[0] := 0.0;
 i^[4,8]^.den[0] := 1.0;

END;



(******************************************************************)
procedure i6_8_cnt (i: mtrx_ptr);
(******************************************************************)

BEGIN

 i^[5,6]^.nsz := 0;
 i^[5,6]^.dsz := 0;
 i^[5,6]^.num[0] := 0.0;
 i^[5,6]^.den[0] := 1.0;

 i^[5,7]^.nsz := 0;
 i^[5,7]^.dsz := 0;
 i^[5,7]^.num[0] := 0.0;
 i^[5,7]^.den[0] := 1.0;

 i^[5,8]^.nsz := 0;
 i^[5,8]^.dsz := 0;
 i^[5,8]^.num[0] := 0.0;
 i^[5,8]^.den[0] := 1.0;

END;



(******************************************************************)
procedure i7_8_cnt (i: mtrx_ptr);
(******************************************************************)

BEGIN

 i^[6,7]^.nsz := 0;
 i^[6,7]^.dsz := 0;
 i^[6,7]^.num[0] := 0.0;
 i^[6,7]^.den[0] := 1.0;

 i^[6,8]^.nsz := 0;
 i^[6,8]^.dsz := 0;
 i^[6,8]^.num[0] := 0.0;
 i^[6,8]^.den[0] := 1.0;
END;
```

```
(*****************************************************************)
procedure i8_8_cnt (i: mtrx_ptr);
(*****************************************************************)

BEGIN

 i^[7,8]^.nsz := 0;
 i^[7,8]^.dsz := 0;
 i^[7,8]^.num[0] := 0.0;
 i^[7,8]^.den[0] := 1.0;

END;


BEGIN
END.
```

→

```
UNIT matrix;

{*****************************}
          INTERFACE
{*****************************}
USES
    plant;

    PROCEDURE dlay_op2op (p:integer;var j,b1:real;f:crtx_ptr);
    PROCEDURE file_initl (f:crtx_ptr);
    PROCEDURE hght_ip2op (r:stat_ptr;a,m1,m2:temp_ptr;o:vect_ptr;f:crtx_ptr;y:real);
    PROCEDURE mtrx_initl (g:mtrx_ptr);
    PROCEDURE mtrx_ip2op (u:vect_ptr;g:mtrx_ptr;o:vect_ptr);
    PROCEDURE mtrx_ip2p2 (s:stat_ptr;u:vect_ptr;f:crtx_ptr;v:vect_ptr;n,m1,m2:temp_ptr;y:real);
    PROCEDURE mtrx_multp (m1,m2,m3:mtrx_ptr);
    PROCEDURE oupt_clear (o:oupt_ptr);
    PROCEDURE oupt_initl (o:oupt_ptr);
    PROCEDURE stat_initl (s:stat_ptr);
    PROCEDURE temp_initl (t:temp_ptr);
    PROCEDURE tens_intac (r:stat_ptr;c:integer;f:crtx_ptr;m1,m2:temp_ptr;y:real);
    PROCEDURE vect_clear (v:vect_ptr);
    PROCEDURE vect_initl (v:vect_ptr);
    PROCEDURE vect_op2ip (v:vect_ptr;o:oupt_ptr);
    PROCEDURE vect_subtr (v1,v2,v3:vect_ptr);

{*****************************}
        IMPLEMENTATION
{*****************************}

PROCEDURE dlay_op2op (p:integer;var j,b1:real;f:crtx_ptr);

VAR
   yti : integer;
   yto : integer;

BEGIN
 yti := f^[p].yput;
 yto := f^[p].yget;

 f^[p].ytt[yti]  := j;
 j := f^[p].ytt[yto];

 f^[p].B[yti] := b1;
 b1 := f^[p].B[yto];

 inc(yto);
 IF yto > ypnts then yto := 0;
 f^[p].yget := yto;

 inc(yti);
 IF yti > ypnts then yti := 0;
 f^[p].yput := yti;

END;
```

```
PROCEDURE file_initl (f:crtx_ptr);

VAR
   i,j : integer;
   r,c : integer;

BEGIN

 FOR j := 1 to max_cntrl do
  BEGIN
    FOR i := 0 to ypnts do
     BEGIN
      f^[j].ytt[i] := 0;
      f^[j].B[i]   := 0;
     END;
    f^[j].yput   := 0;
    f^[j].yget   := 0;
    f^[j].epr    := 0;
    f^[j].ipr    := 0;
    f^[j].Mpr    := 0;
    f^[j].npr    := 0;
    f^[j].Kpi    := 0;
    f^[j].Ipi    := 0;
    f^[j].Aip    := 0;
    f^[j].Cm0    := 0;
    f^[j].Bv     := 0;
    f^[j].J      := 0;
  END;
END;



PROCEDURE hght_ip2op (r:stat_ptr;a,m1,m2:temp_ptr;o:vect_ptr;f:crtx_ptr;y:real);

VAR
  i  : integer;
  v1 : real;
  v2 : real;

BEGIN
 FOR i := 4 to 7 do
  BEGIN
   v1 := (f^[i].Aip*(f^[i].npr* 0.25* 2* 3.14159)/60);
   v2 := (f^[i+1].Aip*(f^[i+1].npr* 0.25* 2* 3.14159)/60);
   IF (r^[i] = true) and (r^[i+1] = true) then
    BEGIN
     IF (a^[i] > dist/2) or ((v1 - v2 > 0) and (m2^[i] <= 0)) then
      BEGIN
       a^[i] := a^[i] + (((v1 - v2)*(dt))/(2*f^[i].Aip));
       IF (a^[i] < dist/2) then a^[i] := dist/2;
       o^.n[0]^[i] := Sqrt(Sqr(a^[i]) - Sqr(dist/2.0));
```

```
      END
     ELSE
      BEGIN
       m2^[i] := y*(((v1 - v2)*(dt))/(dist))  + m2^[i];
       m1^[i+1] := -y*(((v1 - v2)*(dt))/(dist)) + m1^[i+1];
       o^.n[0]^[i] := 0;
      END;
    END
   ELSE
    BEGIN
     a^[i] := dist/2;
     o^.n[0]^[i] := 0;
     m2^[i]   := 0;
     m1^[i+1] := 0;
    END;
  END;
 FOR i := 4 to 8 do f^[i].Mpr := (m1^[i] + m2^[i])*0.25;
END;




PROCEDURE mtrx_initl (g:mtrx_ptr);

VAR
   i,j : integer;
   r,c : integer;

BEGIN
 FOR r := 1 to max_cntrl do
 FOR c := 1 to max_cntrl do
  BEGIN
   new (g^[r,c]);

   g^[r,c]^.nsz := 0;
   g^[r,c]^.num[0] := 0;
   FOR i := 1 to max_order do
    g^[r,c]^.num[i] := 0;

   g^[r,c]^.dsz := 0;
   g^[r,c]^.den[0] := 1;
   FOR j := 1 to max_order do
    g^[r,c]^.den[j] := 0;
  END;
END;




PROCEDURE mtrx_ip2op (u:vect_ptr;g:mtrx_ptr;o:vect_ptr);

VAR
   r,c : integer;
   dp  : real;
```

```
BEGIN
 FOR r := 1 to max_cntrl do
 BEGIN
 o^.n[0]^[r] := 0;
  FOR c := 1 to max_cntrl do
   BEGIN
    dp := 0;
    dp := g^[r,c]^.num[0]*u^.o[0]^[c]*dt + g^[r,c]^.num[1]*(u^.n[0]^[c] - u^.o[0]^[c]);
     o^.n[0]^[r] := o^.n[0]^[r] + dp;
   END;
 END;
 FOR r := 1 to max_cntrl do
  BEGIN
   o^.n[0]^[r] := o^.o[0]^[r] + o^.n[0]^[r];
   o^.o[0]^[r] := o^.n[0]^[r];
   u^.o[0]^[r] := u^.n[0]^[r];
  END;
END;



PROCEDURE mtrx_ip2p2 (s:stat_ptr;u:vect_ptr;f:crtx_ptr;v:vect_ptr;n,m1,m2:temp_ptr;y:real);

VAR
  c    : integer;
  e,e2 : real;
  di,i : real;
  ir,b : real;
  dn,Ma: real;

BEGIN
 FOR c := 1 to max_cntrl do
  BEGIN
   b  := 0;
   Ma := 0;
   ir := 0;
   e  := 0;
   e2 := 0;
   di := 0;
   dn := 0;
   n^[c]  := 0;

   e  := u^.n[0]^[c] - f^[c].npr;
   e  := (e*0.25*2*3.14159)/(60);
   di := f^[c].Kpi * ((e - f^[c].epr) + (e*dt)/( f^[c].Ipi));
   f^[c].epr := e;

   ir := f^[c].ipr + (di);
   f^[c].ipr := ir;
   b  := f^[c].Bv;

   dlay_op2op(c,ir,b,f);
   IF  (c < 4) then tens_intac(s,c,f,m1,m2,y);
```

```
   Ma := f^[c].Cm0 * ir - f^[c].Mpr;
   e2 := Ma - (b * ((f^[c].npr/60) * (0.25*2*3.14159)));

   dn := (e2*dt) /( f^[c].J);
   n^[c]  := ((f^[c].npr/60)*2*0.25*3.14159) + dn;
   n^[c]  := (n^[c]*60)/(0.25*2*3.14159);
  END;

FOR c := 1 to max_cntrl do
 BEGIN
   f^[c].npr := n^[c];
   ir := f^[c].ipr;
   v^.n[0]^[c] := ir;
  END;
FOR c := 4 to max_cntrl do v^.n[0]^[c] := f^[c].npr;
END;



PROCEDURE mtrx_multp (m1,m2,m3:mtrx_ptr);

VAR
   r,c : integer;
   i,q : integer;
   n,p : integer;

BEGIN
 FOR r := 1 to max_cntrl do
 FOR c := 1 to max_cntrl do
  BEGIN
   m1^[r,c]^.nsz := m2^[r,c]^.nsz + m3^[c,c]^.nsz;
   n := m1^[r,c]^.nsz;

   FOR i := 0 to max_order do
    BEGIN
     m1^[r,c]^.num[i] := 0;
     m1^[r,c]^.den[i] := 0;
    END;

   REPEAT
    p := n;
    q := 0;

    REPEAT
     m1^[r,c]^.num[n] := m1^[r,c]^.num[n] + m2^[r,c]^.num[p] * m3^[c,c]^.num[q];
     dec(p);
     inc(q);
    UNTIL ((p < 0) or (q > m3^[c,c]^.nsz));
    dec(n);
   UNTIL (n < 0);

   m1^[r,c]^.dsz := m2^[r,c]^.dsz + m3^[c,c]^.dsz;
   n := m1^[r,c]^.dsz;
```

```
   REPEAT
    p := n;
    q := 0;

    REPEAT
     m1^[r,c]^.den[n] := m1^[r,c]^.den[n] + m2^[r,c]^.den[p] * m3^[c,c]^.den[q];
     dec(p);
     inc(q);
    UNTIL ((p < 0) or (q > m3^[c,c]^.dsz));
    dec(n);
   UNTIL (n < 0);

  END;
END;



PROCEDURE oupt_clear (o:oupt_ptr);

VAR
   r,c : integer;
   j   : integer;

BEGIN
 FOR j := 0 to max_order do
 FOR r := 1 to max_cntrl do
 FOR c := 1 to max_cntrl do
  BEGIN
   o^.o[j]^[r,c] := 0;
   o^.n[j]^[r,c] := 0;
  END;
END;



PROCEDURE oupt_initl (o:oupt_ptr);

VAR
   r,c : integer;
   j   : integer;

BEGIN
 FOR j := 0 to max_order do
  BEGIN
   new (o^.o[j]);
   new (o^.n[j]);
   FOR r := 1 to max_cntrl do
   FOR c := 1 to max_cntrl do
    BEGIN
     o^.o[j]^[r,c] := 0;
     o^.n[j]^[r,c] := 0;
    END;
  END;
```

```
END;


PROCEDURE temp_initl (t:temp_ptr);

VAR
   c : integer;

BEGIN
 FOR c := 1 to max_cntrl do
  BEGIN
   t^[c] := 0;
  END;
END;



PROCEDURE tens_intac (r:stat_ptr;c:integer;f:crtx_ptr;m1,m2:temp_ptr;y:real);

VAR
  M : real;
  v1,v2: real;

BEGIN
 v1 := f^[c].Aip * ((f^[c].npr* 0.25* 2* 3.14159)/60);
 v2 := f^[c+1].Aip * ((f^[c+1].npr* 0.25* 2* 3.14159)/60);

 IF (r^[c] = true) and (r^[c+1] = true) then
  BEGIN
   m2^[c] := y*(((v1 - v2)*(dt))/(dist))  + m2^[c];
   m1^[c+1] := -y*(((v1 - v2)*(dt))/(dist)) + m1^[c+1];
  END
 ELSE
  BEGIN
   m2^[c]   := 0;
   m1^[c+1] := 0;
  END;

f^[c].Mpr := ((m1^[c] + m2^[c])*0.25);

END;



PROCEDURE vect_clear (v:vect_ptr);

VAR
   c : integer;
   j : integer;

BEGIN
 FOR j := 0 to max_order do
 FOR c := 1 to max_cntrl do
  BEGIN
```

```
   v^.o[j]^[c] := 0;
   v^.n[j]^[c] := 0;
  END;
END;



PROCEDURE vect_initl (v:vect_ptr);

VAR
   c : integer;
   j : integer;

BEGIN
 FOR j := 0 to max_order do
  BEGIN
   new (v^.o[j]);
   new (v^.n[j]);
   FOR c := 1 to max_cntrl do
    BEGIN
     v^.o[j]^[c] := 0;
     v^.n[j]^[c] := 0;
    END;
  END;
END;


PROCEDURE vect_op2ip (v:vect_ptr;o:oupt_ptr);

VAR
   r,c : integer;
   j   : integer;

BEGIN
 FOR j := 0 to max_order do
 FOR r := 1 to max_cntrl do
  BEGIN
   v^.n[j]^[r] := 0;
   FOR c := 1 to max_cntrl do
    BEGIN
     v^.n[j]^[r] := v^.n[j]^[r] + o^.n[j]^[r,c];
    END;
  END;
END;


PROCEDURE vect_subtr (v1,v2,v3:vect_ptr);

VAR
   c : integer;
   j : integer;

BEGIN
 FOR j := 0 to max_order do
```

```
 FOR c := 1 to max_cntrl do
  BEGIN
   v3^.n[j]^[c] := v1^.n[j]^[c] - v2^.n[j]^[c];
  END;
END;


PROCEDURE stat_initl (s:stat_ptr);
VAR
   c : integer;
BEGIN
 FOR c := 1 to max_cntrl do
  BEGIN
   s^[c] := false;
  END;
END;



BEGIN
END.
→
```

```
UNIT plant;

{*****************************}
         INTERFACE
{*****************************}
CONST
     max_state = 64;
     max_cntrl = 8;

     max_order = 2;

     max_point = 600;
     ypnts     = 4;
     dist      = 4;

     di = 0.1;
     dt = 0.005;
     dx = 0.02;

     valid0 : set of '0'..'Z' = ['0','C','X'];
     valid1 : set of '0'..'Z' = ['0','1','2','3','4','5','6','7','8','X'];
     valid2 : set of '0'..'Z' = ['1','2','3','X'];
     valid3 : set of '0'..'Z' = ['K','I','X'];

TYPE
    indv_var = record
     B   : array [0..ypnts] of real;
     ytt : array [0..ypnts] of real;
     yput: integer;
     yget: integer;
     epr : real;
     ipr : real;
     Mpr : real;
     npr : real;
     Kpi : real;
     Ipi : real;
     Aip : real;
     Cm0 : real;
     Bv  : real;
     J   : real;
    END;
    crtx_var = array [1..max_cntrl] of indv_var;
    crtx_ptr = ^crtx_var;


    poly_var = record
     nsz : integer;
     num : array[0..max_order] of real;
     dsz : integer;
     den : array[0..max_order] of real;
     END;
    poly_ptr = ^poly_var;
    mtrx_var = array [1..max_cntrl,1..max_cntrl] of poly_ptr;
    mtrx_ptr = ^mtrx_var;
```

```
    o_var = array [1..max_cntrl,1..max_cntrl] of real;
    o_ptr = ^o_var;
    oupt_var = record
     o : array [0..max_order] of o_ptr;
     n : array [0..max_order] of o_ptr;

     END;
    oupt_ptr = ^oupt_var;

    v_var = array [1..max_cntrl] of real;
    v_ptr = ^v_var;
    vect_var = record
     o : array [0..max_order] of v_ptr;
     n : array [0..max_order] of v_ptr;
     END;
    vect_ptr = ^vect_var;

    stat_var = array [1..max_cntrl] of boolean;
    stat_ptr = ^stat_var;

    temp_var = array [1..max_cntrl] of real;
    temp_ptr = ^temp_var;

(********************************)
      IMPLEMENTATION
(********************************)
procedure dummy;
 BEGIN
 END;

(********************************)
(***   INITIALIZATION        ***)
(********************************)
BEGIN

END.
```

APPENDIX N: PI CONTROLLER CONSTANTS

The transfer function of the proportional-integral controller is of the following form.

$$G_{pi} = \frac{K_{pi}*(1 + I_{pi}*s)}{I_{pi}*s}$$

The constants used in the PI matrices implemented in both control systems are given in table N.1.

		PI Control	MV Control
Tension 1	K	2.0	3.0
	I	1.2	0.8
Tension 2	K	2.0	3.0
	I	1.2	0.9
Tension 3	K	2.0	3.0
	I	1.2	1.0
Loop 1	K	3.0	4.3
	I	50.0	50.0
Loop 2	K	3.0	1.2
	I	50.0	50.0
Loop 3	K	1.5	1.5
	I	50.0	50.0
Loop 4	K	1.6	1.6
	I	50.0	50.0
Speed 1	K	2.5	2.5
	I	0.4	0.4

Table N.1: The table of PI constants.

The values of the constants of the PI controller for the first loop are re-tuned to obtain the best possible performance when the metal model is changed. These re-tuned parameters are given in table N.2.

		PI Control	MV Control
Loop 1	K	2.5	3.0
	I	1000.0	3.0
Young's Modulus		800.0	800.0

		PI Control	MV Control
Loop 1	K	6.0	2.5
	I	50.0	50.0
Young's Modulus		100.0	100.0

Table N.2: The re-tuned PI parameters.

APPENDIX O: THE NYQUIST ARRAY DESIGN THEORY

Figure O.1 shows the position of all the compensating units in the closed loop control of the plant. In the closed loop systems used in this thesis, the post-compensator, L(s), and the feedback element, F(s), are both identity matrices.

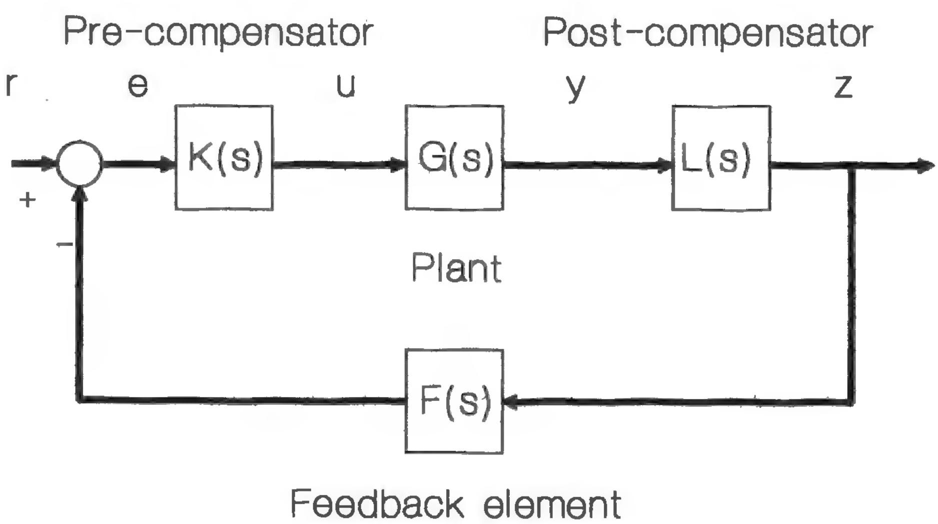


Figure O.1: The closed loop plant diagram.

Using this diagram, the closed loop relation between, r, the setpoint and, y, the output is derived.

$$y = [\ I + Q(s)]^{-1} * Q(s) * r$$

$$= H(s) * r$$

Therefore, the closed loop transfer function, H(s), will be diagonal if the open loop transfer function, Q(s), is diagonal.

The aim of the Nyquist Array design methods is to calculate the pre-compensator unit, K(s), to diagonalize the plant model, G(s), as much as possible[1].

The Direct Nyquist Array uses the open loop transfer function, Q(s). The inverse, $Q(s)^{-1}$, is used in the Inverse Nyquist Array method. This is preferred because the relationship, $H(s)^{-1} = 1 + Q(s)^{-1}$, is simpler than for the direct method, and the accuracy of this method increases as the loop gain increases. The accuracy of the DNA method decreases as the loop gain increases.

The methods are based on designing pre-compensator units for systems so that diagonal dominance is achieved (as described for each array below). The following stability tests can then be used.

<u>In the Direct Nyquist Array.</u>

Row dominance:

$|q_{ii}(s)| > \Sigma_{j<>i}|q_{ij}(s)|$ ...for all rows

Column dominance:

$|q_{ii}(s)| > \Sigma_{i<>j}|q_{ij}(s)|$ ...for all columns

If either of the above dominance tests are passed by the matrix [1 + Q(s)], then the Gershgorin bands for the matrix Q(s), will exclude the critical point, (-1,0) on the Direct Nyquist Array. The closed loop system will then be stable.

If the Gershgorin bands encircle the critical point, then the number of times it does so in a likewise direction to that taken by the Laplace operator along the Nyquist contour, must be equal to the number of unstable open loop poles. Since the number of open loop poles is zero in this plant, the Gershgorin bands must not encircle the critical point on the Direct Nyquist Array.

In the Inverse Nyquist Array.

Since:

$$Det[H(s)^{-1}] = Det[1 + Q(s)^{-1}] / Det[Q(s)^{-1}]$$

If the matrices $[1 + Q(s)^{-1}]$ and $Q(s)^{-1}$ are diagonally dominant for all frequency points, then the Gershgorin bands based on the elements, $q_{ii}(s)$, exclude the critical point, (-1,0), and the origin, (0,0). The closed loop system is only stable iff the number of times that these bands encircle the critical point and the origin in a likewise direction to that taken by the Laplace operator along the Nyquist contour, is equal to the number of unstable open loop poles.

Since this plant is open loop stable the number of unstable open loop poles is zero. Therefore, the Gershgorin bands must not pass between the critical point and the origin on this plant's Inverse Nyquist Array.

The INA design package[2] available at U.C.T. is used to design the pre-compensators required for each control structure. These pre-compensators are designed to achieve diagonal dominance so that the closed loop stability of the system can be determined.

As mentioned in Chapter 6, although the INA method is preferred for it's accuracy, it can not be used to design the pre-compensators for all the restructured binary interaction matrices derived in Chapter 4, because the loop outputs are non-existent at some stages. Therefore, the inverse of Q(s) could not be found. The DNA method is then used.

References.

1.Maciejowski, J.M.: *Multivariable Feedback Design*, Addison -Wesley Publishing, Great Britain, 1989.
2.Venzke, R.H.E.: *Comparison of INA Technique to the Pole Assignment Technique*, MSc Thesis, U.C.T, 1988.