INTELLIGENT ROBOT GRINDING:
PLANNING, OPTIMIZATION, AND CONTROL

by

Matthew Lasché Brown

B.S. Mechanical Engineering, Tulane University (1982)
M.S. Mechanical Engineering, Princeton University (1984)

SUBMITTED TO THE DEPARTMENT OF
MECHANICAL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREEOF
DOCTOR OF SCIENCE IN MECHANICAL ENGINEERING
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1988

Signature of Author

Department of Mechanical Engineering
27 May 1988

Certified by

Daniel E. Whitney
Thesis Supervisor

Certified by

Derek Rowell
Thesis Committee Chairman

Accepted by

Ain A. Sonin
Chairman, Department Committee

DISCLAIMER

This work was supported by the National Science Foundation under grant DMC-8514214. The views and conclusions expressed in this thesis are those of the author and should not be interpreted as necessarily representing the position of the National Science Foundation, The Charles Stark Draper Laboratory, Inc., or the Massachusetts Institute of Technology.

INTELLIGENT ROBOT GRINDING:

PLANNING, OPTIMIZATION, AND CONTROL

by

Matthew Lasché Brown

Submitted to the Departmentof Mechanical Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Science in Mechanical Engineering

27 May, 1988

ABSTRACT

This thesis describes an intelligent manufacturing system that can make decisions about the process in light of the uncertain outcome of these decisions and attempts to minimize the expected economic penalty resulting from those decisions. In particular, it uses weld bead grinding as an example of a previously unmodelled process with significant process variation. The requirements of the weld bead grinding task are to accurately grind off a weld bead within the factory time constraints so that it be indistinguishable from the parent material while avoiding burning the material. Previous work indicated that the grinding process was incapable of removing an entire weld bead, and that the cut depth of seemingly identical passes could vary as much as 20%, although the mean cut depth was predictable. The need for multiple passes, the poor predictability of those passes, the task requirements, and the process constraints conspire to make planning and controlling weld bead grinding a formidable problem. A three tier hierarchical control system has been developed and tested which can plan an optimal sequence of grinding passes, dynamically simulate each pass, execute the planned sequence of controlled grinding passes, and modify the pass sequence as grinding continues. The top tier plans the grinding sequence for each weld bead, and is implemented using Stochastic Dynamic Programming, selecting the volumetric removal and feedspeed for each pass in order to optimize the satisfaction of the task requirements by the entire grinding sequence within the equipment, task, and process constraints. The output of the planner is the optimal grinding policy in a table lookup form given the initial volume of material remaining and the amount of time remaining until the deadline. The resulting optimal policies have quite complex structures, showing foresight, anxiety, indifference, and aggressiveness, depending upon the situation. The second tier is used to plan force trajectories of individual passes that will result in the volumetric removal required by the top tier. It is implemented using a numeric simulation of a controlled grinding process based on a novel dynamic grinding model. The bottom tier is a real time digital force control system which executes the force trajectory planned by the second tier. The force control system has a 3 Hz bandwidth and can control forces up to 90 N. The entire weld bead grinding planning and control system was implemented on a microcomputer controlling a laboratory grinding setup.

Thesis Supervisor: Dr. Daniel E. Whitney, Lecturer in Mechanical Engineering

ACKNOWLEDGEMENTS

I'd like to thank Draper Laboratory for bringing together the people I worked with and creating the environment I worked in which made my learning experience there all the more fruitful. In particular, I'd like to thank the members of the Robotics & Assembly Division; their combination of experience and expertise is rare and I found it to be invaluable. I cannot give enough thanks to my advisor, Dr. Daniel Whitney. His ability to see through to the heart of a problem and identify the key issues is something that they don't teach at engineering school, and is, perhaps, more important than anything else that I could have learned from him. I'd like to thank Alex Edsall for writing software that I could rely on daily, and Don Seltzer, for his assistance with the electronics in my equipment. The technical staff, Bobby Fields, Greg Romano, and Ray Roderick, have each taught me more than some of my professors, and have saved this inept graduate student from much grief and labor. Special thanks go to the other students I have had the good fortune to know, Rob Smith, Max Lui, Alan Tate, and in particular my coworker Tom Kurfess. I cannot thank my PhD committee enough: Stephen Graves, Dave Hardt, and Derek Rowell. They were patient and understanding, and went out of their way to assist me. Special thanks go to Hatch Brown, the sailing master at MIT, whose sailing facilities have made my life enjoyable through these years.

Matthew L. Brown 11 May, 1988

For Seymour, who taught me how to read,

For Dan, who showed me the benefits of perseverance,

For Azby, who taught me how to survive,

For Roger, who taught me what's important,

For my parents, who gave us all life.

This is all your fault.

See what you started.

TABLE OF CONTENTS

ABSTRACT 3

ACKNOWLEDGEMENTS 4

TABLE OF CONTENTS 6

1. INTRODUCTION 11

1.1 SCOPE 12

1.2 PROBLEM STATEMENT 16

1.3 WHAT MAKES THIS PROBLEM HARD 16

1.4 PREVIOUS APPROACHES TO SOLVING THIS PROBLEM 17

1.5 THE APPROACH OF THIS THESIS 19

1.6 TEST CASE 20

1.7 THE AIM OF THIS THESIS 21

2. THE ELEMENTS OF AN INTELLIGENT MANUFACTURING SYSTEM 23

2.1 HIERARCHY 23

2.2 THE TASK GOALS 25

2.3 IMPERFECT PROCESS PERFORMANCE 26

2.4 RATING THE OUTPUT OF A PROCESS VIA PENALTY FUNCTIONS 27

2.5 THE PROCESS MODEL 34

Types of Process Models 34

Modelling Error 35

Propagating Probability Density Functions 36

2.6 THE PROCESS PLANNER 40

2.7 THE OPTIMIZER 42

2.8 CHAPTER SUMMARY 43

3. INTELLIGENT GRINDING SYSTEM OVERVIEW 45

3.1 WELD BEAD GRINDING TASK REQUIREMENTS 45
3.2 IMPLEMENTATION ISSUES 47
3.3 SYSTEM STRUCTURE 50
Grinding Sequence Planner 51
Pass Planner 52
Force Controller 52
3.4 CHAPTER SUMMARY 52
4. THE GRINDING PROCESS MODEL 55
4.1 PREVIOUS GRINDING RESEARCH 55
Static Grinding Models 55
Snagging 59
4.2 THE STATIC GRINDING MODEL 61
Stochastic Model 62
4.3 CHAPTER SUMMARY 65
5. PLANNING WELD BEAD GRINDING SEQUENCES 67
5.1 INTRODUCTION 67
The Grinding Sequence Planner's Task 67
State Transitions 68
Sequential Decision Making 69
5.2 DYNAMIC PROGRAMMING 71
A Simplified Grinding Problem Formulation 71
Penalty Function 72
Dynamic Programming for Planning Grinding Passes 75
Variable Feedspeed 78
5.3 STOCHASTIC DYNAMICS 81
Stochastic Dynamic Programming 81
Implementation 84

5.4 GRINDING POLICIES WITH A FIXED DEADLINE 85
Penalty Function Shape 85
Optimal Cut Depth 87
Reachable Terminal States 89
Already Overground 91
Not Enough Time 92
Cannot Reach Target State 92
Can Reach Target State 93
Second-Order Effects: Optimal Feedspeed 94
Grinding Policies in the Region that Can Reach the Target State 96
Optimal Policies in the Region of Initial States that Cannot Reach the Target State 98
Time-Banded Solution Structure 100
5.5 GRINDING POLICIES WITH A LATENESS PENALTY 107
Lateness Charges and the Quit Now Option 107
Cheap and Expensive Time 108
Insufficient Time 110
Slightly Late 111
The Remainder of the Unreachable States 112
The Reachable Region of the State Space 113
5.6 EFFECTS OF THE NO-BURN CONSTRAINT 116
5.7 A MORE COMPLEX GRINDING PROBLEM 120
5.8 CHAPTER SUMMARY 122
6. PLANNING INDIVIDUAL GRINDING PASSES 125
6.1 COMPUTING THE DISK TIP TRAJECTORY 125
6.2 MOTIVATION FOR THE DYNAMIC MODEL 128
6.3 PREVIOUS DYNAMIC GRINDING MODELS 128

6.4 THE DYNAMIC GRINDING MODEL 130
6.5 VERIFICATION OF THE DYNAMIC GRINDING MODEL 133
Dynamic Grinding Simulation 133
Experimental Verification 136
6.6 PLANNING GRINDING PASSES VIA SIMULATION 140
6.7 RESULTS OF PREPLANNED GRINDING PASSES 141
6.8 CHAPTER SUMMARY 146
7. CONTROLLING THE GRINDING FORCE 147
7.1 THE FORCE CONTROL SYSTEM 147
7.2 THE FORCE CONTROL SYSTEM MODEL 149
The Compliance 149
The Vertical Position Control Loop 151
The Force Sensor 152
7.3 DIGITAL FORCE CONTROL DESIGN 153
8. CONCLUSIONS AND RECOMMENDATIONS 157
8.1 THESIS SUMMARY 157
Controller Hierarchy 157
Top Level Planner 158
Middle Level Planner 159
Bottom Level Controller 159
8.2 CONCLUSIONS 159
Application to Other Processes 160
Material Removal Processes 161
Bending 161
Refining 162
Optimal Grinding Policies 163
8.3 RECOMMENDATIONS FOR FUTURE WORK 163

REFERENCES 167

APPENDIX A: A DYNAMIC PROGRAMMING TUTORIAL 173

A.1 A SIMPLE DYNAMIC PROGRAMMING EXAMPLE 173

A.2 COMPARISON WITH THE PLANNING OF WELD BEAD GRINDING 177

APPENDIX B: MODELLING THE VERTICAL FORCE TABLE 179

B.1 INTRODUCTION 179

B.2 THE MOTOR MODEL 179

B.3 SPEED-CONTROL SYSTEM 181

B.4 POSITION CONTROL 186

1. INTRODUCTION

This thesis describes a method for implementing intelligent automation for manufacturing. Intelligent automation refers to both the automated hardware that performs the manufacturing process, and the software for making the decisions required for manufacturing. As the manufacturing tasks to be automated become more complicated or less predictable, the range of choices becomes broader and more complex than those that are typically faced by modern control systems, and this requires more intelligence of the control system.

The system elements required for automation can be broken down into four areas: **tools** or **actuators** to perform the task, **sensors** to monitor the performance, task **goals** to define what is to be accomplished, and **strategies** to coordinate the actuators and the sensors to bring about those task goals. Some manufacturing tasks have not been automated because they require capabilities or skills in one or more of these elements that are beyond those generally available in conventional control hardware and software. A manufacturing engineer at a large automobile plant stated that "the problem [of implementing advanced manufacturing] is not usually the equipment or the workforce, it's with the controls. We can't get them both smart *and* reliable. [The controls] just can't handle the process variance. I got sources of variance all over the place ... We do Taguchi studies all the time, and I've only had two work right—actually find and reduce the variance. The others just told us that it's not [due to] the hypotheticals [variables]."[1] This thesis is about designing controllers that can handle the process variance.

The necessary levels of capability are displayed by the human workers, who are able to perform such automation-resistant tasks. Their senses are adequate to the task, and are well-integrated. Their muscles, serving as actuators, are perhaps poorly suited to a manufacturing environment, but allow them to use other better suited mechanical tools. However, most of all, humans excel at formulating strategies for achieving loosely-specified goals. They are capable of planning actions, predicting their outcomes, recognizing situations, replanning actions, and dealing with contingencies 'on-line' as the process proceeds. They can recognize and adapt to changes in the

---

[1]Personal communication.

behavior of the process. They know when and how to be careful with a process that can go awry, and can embody this knowledge in their strategies for performing or operating the process. Such advanced capabilities are well-suited to, and indeed required for, the performance of many difficult tasks.

Automation implementation fails when its implementation of one or more of these capabilities is not up to the task. There is currently much research and progress in the fields of actuator and sensor technology. Formulating and understanding the task goals and creating the required strategies needs more development. Creating task goals for automation is a difficult problem, and is not explicitly addressed here. Well-defined goals are, however, crucial to the success of automation implementation.

This thesis addresses the planning and control issues. Planning of strategies is usually done by evaluating the results of predictions of the consequences of putative actions. However, for automation, predictions require models of the process, and such models can be difficult to construct and have poor predictive accuracy for complex tasks. There are two approaches to solving this problem: 1) model the process better, 2) deal better with poor models. This thesis uses primarily the second method to attack the problem of bringing automation to more complex manufacturing tasks.

The remainder of this chapter identifies the scope of the problem that this thesis addresses. Then the type of problem to be solved will be identified, and the reasons why this is a difficult problem will be given. Previous approaches to solving this type of problem will be described, and followed by a description of a new approach to such problems. Then the test case used to demonstrate this approach will be described, and reasons why it is a particularly difficult example of this class of problems will be given. The last section in this chapter is an outline of the remainder of this thesis

1.1 SCOPE

This thesis deals with factory-floor decision making, but only with a specific type of decisions made on the factory floor whose effects are felt over a certain range of time. There are many types of decisions made on the factory floor, but they can be divided roughly into decisions that affect the entire factory and those that affect individual processes. **Factory control** decisions are those that are made either for the factory as a whole, or are inter-process decisions. This thesis is concerned with **process control** decisions, the decisions that are required to operate individual processes.

These decisions can be further sorted by the time span over which they apply. This is illustrated in Table 1-1.

	Factory Control	**Process Control**
Long Term (days to weeks or months)	Inventory Control production rate staffing decisions	Workstation monitoring Change worn tool, shut machine down
Medium Term (minutes to days)	MRP batch size	Task Completion make another pass; try again or quit
Short Term (seconds to hours)	work routing or scheduling material movement	Task Recipe robot trajectory, assembly sequence
Very Short Term (milliseconds to minutes)		Machine Control control motor speed, control reactor temp

Table 1-1 Decision Time Frame for Factory-Wide and Individual Machine Tasks

The first column lists factory control decisions according to their time span. Inventory control is concerned with parts stocking and ordering policies, accounting for long lead times and unknown demands. The long lead times make this a long term decision. MRP is Materials Requirements Planning, which time-phases the production requirements given the delivery schedule for finished goods, the recipe for making the finished goods from raw materials, and the time required at each workstation to build the product. This plans the day-to-day routing of work within the factory, and forms links among the sales department, the raw materials receiving department, and the shipping department. The day-to-day planning aspect of MRP makes it a medium term decision-support system. Work routing or scheduling comes into play when there are identical workstations to which work can be assigned on a minute-by-minute basis. Although some MRP systems can perform this function, it is also often left to a local controller, and can have a strategy as simple as "route the next job to the first available workstation." This is typically the shortest term inter-process decision in a factory.

The second column lists individual process control decisions. The longest term decisions are typically concerned with monitoring and handling slow process variations, such as wear, replacement of raw-material supply, and malfunctions. When the task is iterative, i.e. of the "repeat ... until done" type, there is a medium term

decision regarding whether to repeat the steps or declare the process done. Such decisions occur after each iteration. This lies somewhere between the long term (workstation monitoring) and the short term (task recipe) decisions. The next level of decisions is the task recipe type, which determines the sequence and type of actions to be performed to complete one step of the manufacturing process. Examples of this type are the sequence and trajectories of the robot motions required to perform one step of a process (e.g. those required to install an automobile windshield), or the assembly sequence for a product. These sequences are often prerecorded and replayed unaltered, but real-time alterations are sometimes required, such as when the windshield needs to be guided into its frame. For process controls, the boundary between the medium term and short term decisions is frequently coincident with the boundary between decisions made at discrete times and decisions made more on a continuous basis. A windshield installation robot might be guided by analog controls operating continuously in the short term, but the medium-term decision to make another iteration is inherently discrete. High-speed digital computers have made it possible to effectively implement the continuous control in discrete form. The decision time span can thus be measured by the system cycle frequency, in units of Hertz [Hz].

Systems making such medium term decisions operate roughly over the range from 0.1 Hz to 10 Hz, depending on the process. At this point decision-making begins to make the transition into control; the shortest term process decisions are generally better described as controls rather than individual decisions, even though these too are frequently implemented digitally as decisions made at discrete times. Examples of very short term decision-making are motor speed control by varying the input voltage, and temperature control via a thermostat. These operate over the range of frequencies from roughly 1 Hz to 10,000 Hz. Note that the process control decisions are usually much shorter term than their factory control counterparts. The point here is that decisions made in a factory can be broken into inter- and intra-process decisions, and that these break down further into similar time hierarchies. The level of the hierarchy will be used as an indicator of control complexity, and the control task will be broken down according to the hierarchy.

It should be noted that the selection of the appropriate types of machines and the required sequence of manufacturing steps to build a product or part is also called process planning. This does not appear in Table 1-1 because it is typically done off-line well before production starts as part of planning the implementation of the

manufacturing line. This thesis deals primarily with decisions made during production.

The time spans illustrated in Table 1-1 are by no means definitive nor are all time spans applicable to every process, but most processes can be fit into this framework. In particular, the medium term process control decisions usually occur only for iterative or repetitive tasks, and there are processes for which there are no long term decisions. Note that there are fewer applications for longer-term decisions, probably both because there has been little perceived need for longer term planning and control, and because longer term tasks tend to be too complicated to be worthwhile automating. The shorter term decisions seem to be the most frequently required.

The process control (and to some extent the factory control) can be further classified according to the predictability of the process by the current or best available process model. This is depicted in Figure 1-1 below. The ideal case is labelled "noiseless," and represents the situation in which the model perfectly predicts the behavior of the process. The next level of predictability is labelled "some noise," and represents the case in which the model predicts the process fairly well, but cannot account for some aspect of the process' behavior, other than to classify it as being due to some unpredictable "noise." The worst case considered here is when the model can only roughly predict the behavior of the process. This is labelled "poor model."

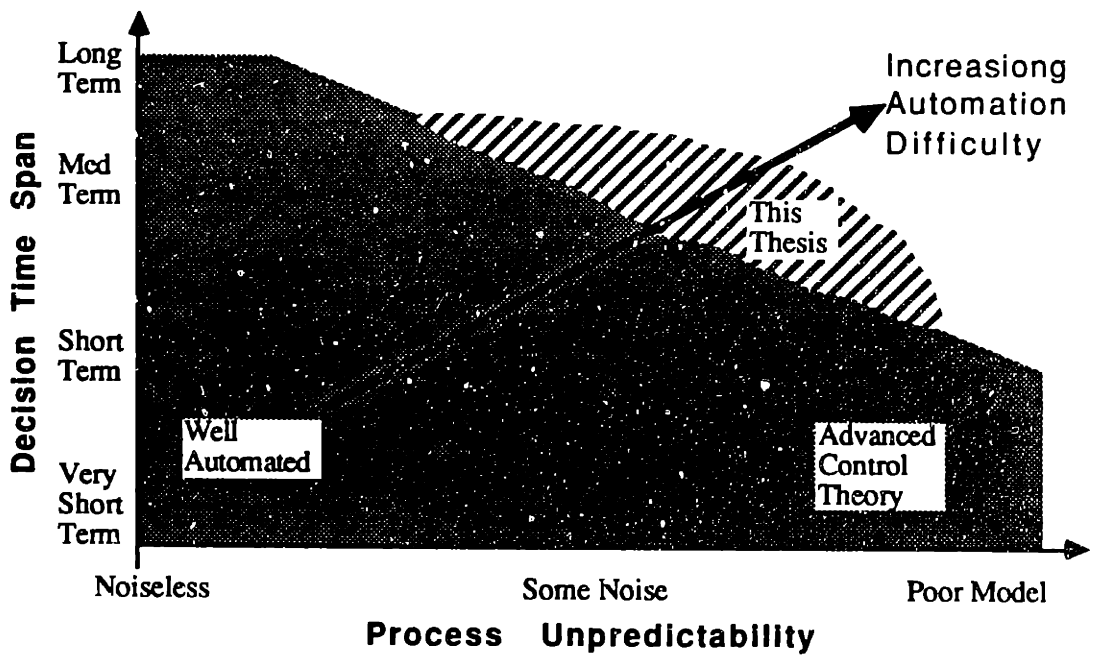


Figure 1-1 **Process Predictability vs. Decision Time Span**

The shaded area illustrates the type of tasks that have been easily automated, or are well-automated to date. The degree of automation difficulty increases from lower left to upper right. In general, very short term decisions tend to be based on very well defined criteria, and have been well-automated via modern control theory. Advanced control theory, in the guise of both robust and adaptive control, has been extending the boundary at the lower right, allowing more accurate control of poorly-modelled or noisy processes under certain specific assumptions and restrictions.

The region that this thesis addresses is hashed. It deals with a longer term, more complex type of decision making than advanced control theory is suited for, yet does not specifically address the complex process modelling problems of the very long term decision level. The goal of this work is to extend the frontier of automation in this area.

1.2 Problem Statement

The problem, then, is to increase the level of manufacturing automation to include some level of automatic on-line decision-making about complex and poorly modelled manufacturing processes. The decision-making studied is restricted to that needed for individual process planning and control. In particular, the automation of a poorly-modelled batch manufacturing process which is iterative in nature will be illustrated. The techniques described herein are adaptable to other processes, such as continuous-flow or non-iterative processes, but these were not studied in detail. The poorly-modelled multi-stage process studied was deemed to be a problem with sufficient difficulty to be a good test case. This test case is robotic weld bead grinding, described in further detail in Section 1.6.

1.3 What Makes This Problem Hard

To make decisions, the consequences of each choice must be evaluated. This requires the ability to predict the behavior of a manufacturing system or process, and so the ability to predict the outcome of a particular choice. But predictions can be inaccurate, especially for complex processes – which are the ones which are most likely not to have been automated already – so planning and control decisions must be made in light of uncertainty. This is not easy, because one must evaluate each outcome of each choice, its probability, and its effects, and then put a value on that choice before the choice with the best value can be selected. It becomes harder when the process has several stages or is iterative. Then the output of one step or stage is the input of the next

stage, and the prediction errors from individual stages propagate throughout the process. Planning must be done on the entire process, not just each step. The available choices multiply, making the decision-making problem harder.

1.4 Previous Approaches to Solving this Problem

Previous approaches to planning for poorly-modelled complex processes have been either to model the process better beforehand, change the process into an easier one to model (particularly by reducing process variation), or to make the planner robust despite the inaccurate model. When you study a process, you have to decide how to go about the study, and how much time and effort to put into the study. Redesigning a process to make it easier to plan for is particularly recommended when this involves reducing process variation. However, this may be impractical or too expensive.

It is easier to try to tweak the process parameters in search of a more predictable process. Genichi Taguchi [1][1] reduces the process variation via the optimization of process parameters. For simple processes he accounts for the effects of the variation of the parameters via the process model, and seeks to reduce their effect on the overall process variation. For complex processes, he demonstrates an efficient experimental method for selecting suboptimal process parameters to reduce the process variation. This is the Taguchi study mentioned in the introduction, and has promise for some processes, and is a recommended first step in most cases, but can only affect the process variation to the degree that it is controllable through the process parameters. This thesis starts where Dr. Taguchi's methods leave off, to enable one to better control unpredictable processes.

Advanced control theory has attacked this problem in two ways. The first technique is robust control, an extension of linear control theory, by which the controller is made more robust against differences between the actual system behavior and that of the process model. The controller is designed to reject a certain class of errors, and steadfastly manipulates the process to attain the desired outputs. Most such controllers can guarantee stability given bounds (in the frequency domain) on the modelling error, and can make limited promises about the quality of the control given other information about the modelling error. Examples are sliding-mode control [2]

---

[1]Numbers in brackets refer to references found at the end of the thesis.

and loop transfer recovery [3]. The second technique is adaptive control theory [4], in which the controller adjusts itself as the process changes. Such control can be more robust than linear robust control, but can make fewer general promises. In fact, a restricted stability guarantee is the most that many such techniques can offer. Adjustment is done in one of two ways: either a set of gains is changed according to the error between the process output and the model output, or the process is monitored and the model is adjusted on-line via system identification theory [5]. These techniques are generally only applicable to the shorter term processes which have simple task specifications, because they require the process to be modelled by simple difference or differential equations.

More complex requirements can be handled by adaptive control for optimization. This comes closest to the techniques described in this thesis, but is good for only single-step processes or one phase of a process, but not multi-step processes. Some examples pertinent to grinding are to be found in [6–8]. The first two describe two approaches for the optimization of the metal removal rate given a steady-state process model. They automatically select optimal metal removal rate subject to burning and surface finish constraints based on a static process r. odel. The system that Amitay implements automatically controls the grinding power at the burning constraint. The burning constraint is computed using the process model and theory described by Malkin. Shibata's work is similar to Amitay's in that the workpiece dimension and tool life are optimized in a heuristic manner by controlling some parameters and monitoring others. These optimize a process using a model, and control part of the process separately, but do not account for the process variation explicitly. That is, they do not use the process model to learn anything about the structure of the variation, and therefore cannot utilize such knowledge to help control the process.

There has been much work in determining optimum process parameters for single stage systems. Ermer [9] derived optimum cutting conditions for a deterministic machining problem with constraints using geometric programming. Ermer and Wu [10] compute the cheapest cutting speed for turning without constraints. Their problem was made stochastic by poorly known coefficients in the tool wear model. Iwata *et al.* [11] consider the optimization of a cutting process when the coefficients in the linear constraint equations were uncertain. They used the technique called "chance-constrained programming" to find the optimum feeds and speeds while insuring that chance of violating the process constraints be less than a specified value. This involved computing the probability that a constraint be violated due to the variation in one of

the constraint coefficients. The resulting deterministic optimization problem was solved by a nonlinear programming technique. Hati and Rao [12] extended this method to show the effects of constraint parameter variations on the machining cost. They later applied the same technique to cold rolling[13]. All these methods connot be applied to multi-stage systems.

There has been some work in optimizing multi-stage systems. Hitomi [14] analytically determined near-optimum cutting speeds for each stage of a multi-stage machining system without constraints. His method was restricted to the optimization of a single control, however. He later considered finding the optimum NC machining sequence under simple deterministic constraints using a branch and bound technique.[15] This was an early attempt at process planning that took only machining time into account when deciding when to drill and when to mill, etc. Rao and Hati [16] considered the deterministic problem of finding the optimum cutting conditions for a multi-stage process involving a specified sequence of operations performed on different machines. The multi-stage problem adds an extra constraint that the bottleneck process must not be kept idle. This, too, is solved via a nonlinear programming technique. In fact, all of the previous work in multi-stage optimization could not be extended to solve stochastic problems.

1.5 The Approach of This Thesis

The approach described in this thesis is to divide the system into hierarchical levels of complexity, usually time-hierarchical, corresponding to those illustrated in Table 1-1. Each level has its own model of the process for which its planner or controller is designed. The planning or control problem of each level is solved independently of the lower or higher levels, which are assumed to have done their job in a manner that can be modelled simply. Each lower level typically operates at a higher temporal frequency than the level above it, is less complex, and has generally better defined goals and process models. Higher levels deal with more complex processes and tasks. Therefore these levels seem more to plan strategies and give advice than to compute control laws and give commands.

And, as the processes get more complex, the differences between the models and reality increase, and it becomes less efficient (in terms of time and money spent) to try to improve the process model. Thus, the control strategy must account for more uncertainty in its ability to predict the process. This is done by making a simple model

of the process variation and its costs and optimizing these costs. Each level requires its own process model for predicting the consequences of choices made at that level, and for predicting the probability of each possible outcome. A penalty is associated with each outcome, according to how well the outcome meets specifications and satisfies constraints. Each choice is evaluated by the expected penalty corresponding to its outcomes. The stochastic planning or control problem then becomes a penalty minimization problem, which is solved by an appropriate analytical or numerical method. In particular, multi-stage processes can be planned using stochastic dynamic programming. This technique can handle arbitrary constraints upon the controls and the state space, and can handle arbitrarily defined cost functions (not just linear).

1.6 TEST CASE

The example considered in this thesis is the grinding of weld beads such as those that remain on the exterior of an automobile after its roof is attached to the door pillars. For cosmetic and structural reasons the resulting weld bead must be ground smooth with the parent material so as to be undetectable after painting. This implies dimensional accuracies of less than 0.1 mm. Weld bead grinding is currently done by humans wielding heavy grinders, and is a tiring and dirty job. It is also unhealthy due to the easily breathable grinding grits. This type of grinding is illustrated in Figure 1-2 below.

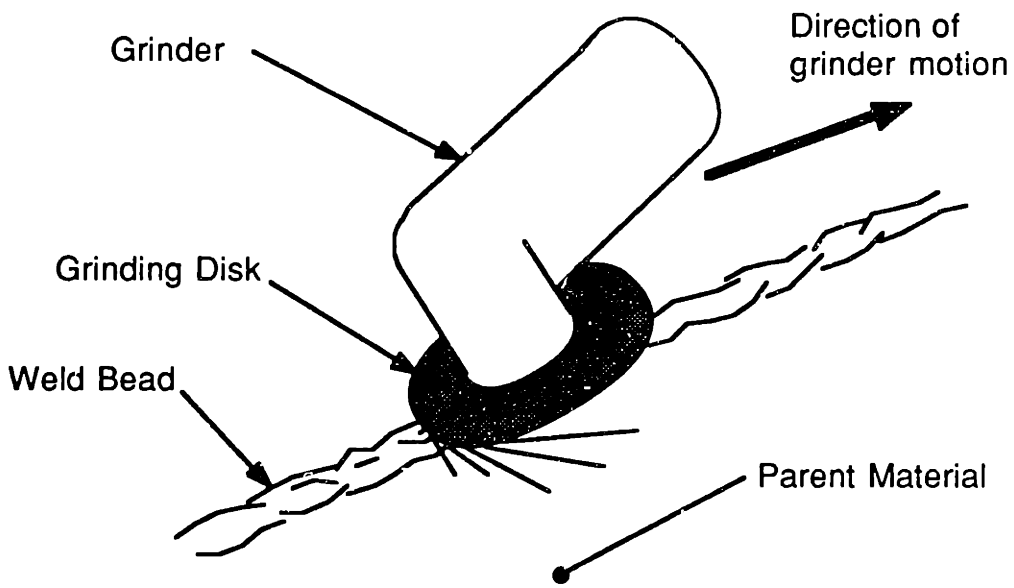


Figure 1-2 Weld Bead Grinding

This is a difficult problem for several reasons which will be outlined here. A detailed description of the weld bead grinding task requirements and issues will be given in Chapter 3. The most common method for controlling ground shapes is to rigidly control the geometry between the workpiece and the grinding disk. However, it is difficult to locate the auto body to the same degree of precision as is required of the final weld contour, so a grinding system which relies on absolute geometry would be impractical. All the relative position sensors studied, both tactile and remote, would have difficulty actually measuring the position to the disk tip to the required accuracy, or could not handle the complex contours of the weld bead and parent material. Therefore geometrical control was unsuitable.

The only option remaining was to control the grinding process. Until now, this type of grinding has not been well studied. There were no models that could predict the resulting shape carved into the weld bead given the initial weld bead shape and the grinding conditions such as the wheel sharpness, grinding feedspeed, wheelspeed, and applied grinding force, nor could a controlled profile be created. When a grinding model with these capabilities was created as part of this thesis, it could not accurately predict the final shape because it relied on estimated empirical coefficients. Controlling the grinding process was going to be difficult.

In addition, there are several constraints on the grinding process that make it a complex process to control. Limitations on the amount of material that can be removed during any one grinding pass make weld bead grinding a multi-pass process. The entire sequence of passes must be planned at one time to insure good results. The results of one pass become the initial conditions of the next, so the unpredictability of any one pass makes planning the entire sequence of passes very difficult.

The final aspect of weld bead grinding that makes it a good test case is the fact that the control issues break down nicely into the hierarchical structure described above. The weld bead grinding hierarchy is divided into planning the entire sequence, planning individual passes, and executing each pass. This, the process unpredictability, and the multi-stage planning problem make weld bead grinding a complex process for control.

1.7 The Aim of this Thesis

This thesis describes an intelligent manufacturing system that can make decisions about the process in light of the uncertain outcome of these decisions. This

system attempts to minimize the expected economic penalty resulting from those decisions. In particular, it uses weld-bead grinding as an example of a previously unmodelled process with significant process variation. A hierarchical control system has been developed and tested which can plan an optimal sequence of grinding passes, dynamically simulate each pass, execute the planned sequence of controlled grinding passes, and modify the pass sequence as grinding continues. The sequence is optimized to insure that the weld-bead is accurately ground off while avoiding burning the material and avoiding overtime, by making overall risk-averse decisions that trade off the risks of violating these requirements among each other.

The remainder of this thesis will describe what is needed for a manufacturing controller with such decision-making capability, and how these are illustrated in the test case grinding system. Chapter 2 describes in general the elements of an intelligent manufacturing system. The corresponding elements that are pertinent to grinding are then discussed in detail in the following chapters. Chapter 3 describes the weld bead grinding problem in detail, presenting the task requirements and the control issues that arise. It then describes the hierarchical control structure used to deal with those requirements and issues. Chapter 4 gives a history of grinding process models and then describes the grinding process model used for planning the grinding pass sequence. Chapter 5 shows how the manufacturing decision-making for a multi-stage process can be implemented in a dynamic program, and how this was done for planning the grinding sequence. Chapter 6 then shows how individual grinding passes are planned using a dynamic grinding model. Chapter 7 shows how these passes were executed using force control, and how the force control system was implemented. Chapter 8 follows with conclusions and recommendations.

2. THE ELEMENTS OF AN INTELLIGENT MANUFACTURING SYSTEM

This chapter will describe, in general, what parts of an intelligent manufacturing system are required, and how they are obtained. The word 'system' here refers to all the hardware, the software, and the concepts needed to create a manufacturing line. The generic ideas and techniques described in this chapter will be demonstrated using the real example of weld bead grinding in the following chapters.

This chapter starts by describing conventional techniques for controlling a complex process, such as defining a process hierarchy and designing controllers for each level of the hierarchy. The elements required for designing each level's controller, such as task goals and process models, will be briefly described, with particular attention paid to the requirements for the more complex levels. Then the causes for imperfect performance will be identified and shown to be unavoidable in many instances, particularly for complex processes. The problem of automating such complex processes thus becomes one of optimizing the process performance within this hierarchical structure, by reducing or ameliorating the effects of imperfect performance. But before this optimization can take place, a means for comparing process performance under different conditions must be developed. So a scheme for comparing process outputs using penalty functions will be presented first, followed by a brief discussion of the formulation of the problem as an optimization problem.

2.1 HIERARCHY

The first step is to break up the process into a hierarchy in which the levels are differentiated chronologically or by complexity. In order to automate each level of a process, one needs to have: 1) task criteria, to define the goal of the level, 2) a process model suitable for that level, 3) control of some variables in the process, 4) ability to measure pertinent variables in the process, 5) a planner with strategies for controlling the process to get the job done, and frequently 6) an optimizer, to find the best way to satisfy the task criteria. These are elements of the process structure discussed in the previous chapter.

Breaking up a task into a hierarchy is a venerable technique for simplifying a job, in this case, the planning or control task. All levels above and below the current

hierarchy level being planned are assumed to have done their job, though perhaps not perfectly. This reduces the range of concerns that each planner or controller has to deal with, and allows each type of decision to be made by a specific controller or planner. For the purposes of planning and control, the levels of the hierarchy correspond to the decision time spans illustrated in Table 1-1.

Each level of the system is made up of a process model, sensors and actuators, a control strategy, and goals for that control strategy. The goals are the input to this level. The outputs of the level are the desired results, which hopefully match the desired goals. The control strategy is either designed with a model of the process in mind, or uses a process model during control. This is depicted below:

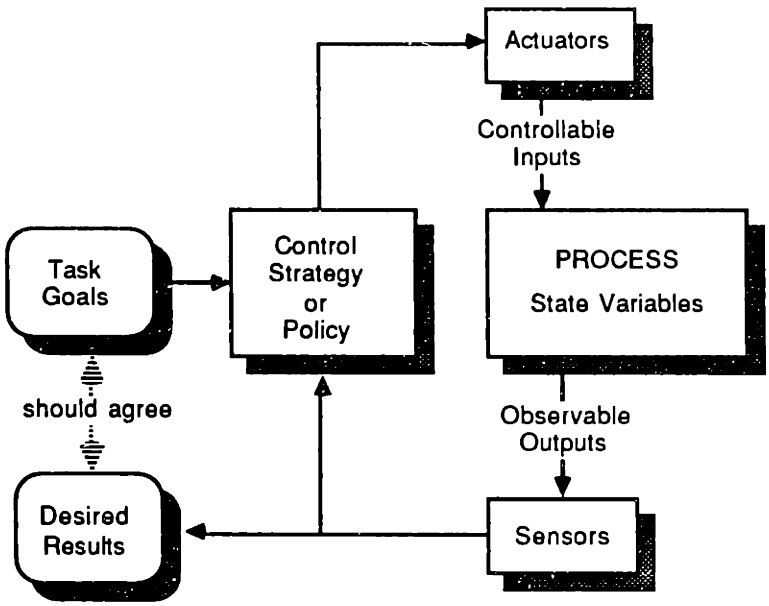


Figure 2-1 A Single Level in a Process

Defining such a hierarchy is a well-trod path, and designing controllers for each of these levels as described above would be routine. However problems of imperfect process performance at the top level and below make the planning problem particularly hard at the top level. In the particular case of the robotic weld-bead grinding system the actuator for the first level – level two – is not very accurate; experimental weld bead grinding results have varied as much as 20% from those desired. So there is a real possibility that a grinding pass won't take off the amount of material that this level's controller had planned. For weld bead grinding this could be worse than overrunning a deadline. The grinder could possibly grind right through the weld bead and into the parent material below, creating costly damage. Or it might not grind enough off. All these possibilities must be weighed by the controller/planner at this level. It must plan

in light of an uncertain process, with the knowledge that these plans may not be executed perfectly. These requirements are beyond the abilities of the planners and controllers of the lower levels. The process and its decision criteria are too complex to be planned or controlled by the techniques that worked for the lower less-complex levels. A new type of planner is required – one that can find the best sequence of decisions given the above criteria and a poorly predictable process.

The next section will discuss, for completeness, what is meant by 'process performance' for a given level of the hierarchy – satisfying task goals. This is followed by a discussion of imperfect process performance, and following that is a section describing how the imperfect performance can be rated.

2.2 The Task Goals

Once the hierarchy of the process has been identified, the planner or controller for each level can be created. The first issue to be settled can be stated as: what is the planner or controller for this level trying to do? One starts with the manufacturing problem for each particular level, typically working from the highest level on down. What is it that this level must do? How is the performance of this task going to be measured? What are the task completion criteria? If the task is cleaning glassware, how is 'clean' going to be measured? What conditions should be avoided? The implementation of the task criteria is called the **task specifications** in this thesis. These must be specified in terms of the available sensors, so that the status of the task can be determined from some combination of the sensor readings. So an important assumption early in the system design process is that of the availability of sensors to measure the desired outputs. Therefore one required element of an intelligent manufacturing system is an algorithm or formula for expressing each of the task specifications in terms of the sensor readings. In the glassware example, the task goal is "clean glassware," and a task specification might be "no visible difference between glassware that is known to be clean and the glassware in question," or "dirt mass less than 0.1 mg." If this formula is designed correctly, satisfying the task specifications will assure that the task goals will be satisfied.

Creating the task specifications from the task goals is one of the most difficult and important tasks when implementing a manufacturing process. The specifications are important because they guide the implementation of the manufacturing system – in particular the selection of the process – and this determines the resulting task

performance. It is easy to go astray here. Having a certain process in mind, it is tempting to create biased specifications and overlook other, possibly better, processes. The two glassware specifications above are biased toward processes that remove visible dirt and processes that remove dirt mass, respectively. Only after the specifications are determined should the actual process be selected. This is an old and well-discussed topic, and out of the scope of this thesis, which assumes that the the task specifications, the process, the actuators, and the sensors have already been selected by the time a planning and control system must be developed. This thesis concentrates upon the the structure of the control system and its strategy for satisfying the task specifications.

2.3 IMPERFECT PROCESS PERFORMANCE

Imperfect process performance refers to a condition in which the output or result of a process does not satisfy the process specifications or does not produce the desired process output. Examples are inaccurate dimensions from a part-forming process such as holes drilled off-center or out of round, incomplete welds, or products that do not themselves produce the desired output, such as a power supply that does not produce the desired voltage.

There are many causes of imperfect process performance, but they can be grouped into four basic causes: 1) equipment failure either in the process or its control system, 2) inadequate inputs, such as substandard materials, 3) the process is incorrect, inadequate, or poorly designed for the task, 4) the process control is poorly designed or inadequate for the task. This thesis will concentrate upon improving the process control, and will assume that the process has been correctly selected for the task and implemented, that there are no breakdowns, and that the inputs to the process are within specifications. In particular, this thesis will describe techniques to ameliorate two causes of controller error: 1) imperfect process models, and 2) imperfect actuators or sensors.

It is clear that a good process model is useful for good planning or control, but process models can never be perfectly accurate. Inaccuracy here refers to the fact that there will always be aspects of the behavior of the process that its model can't explain or predict. This is also called **modelling error**. Note that this can occur whenever there is a difference between the model and the actual process, whether by intentionally ignoring known model details in favor of a simpler, less cumbersome model, or through ignorance of some aspect of process behavior. Overcoming ignorance through more

detailed modelling tends to be both expensive and cumbersome, and can take valuable time. The resulting fiscal and convenience tradeoffs usually cause the model to be inaccurate to some degree. This is especially true for more complex tasks where the modelling is most difficult. A second element of inaccuracy is that there are often external influences on the process that cannot be predicted by any model and cannot be avoided. This is referred to as **external process noise**. For these two reasons, modelling errors will usually be present and must be dealt with. A successful automation implementation must therefore include planners and controllers that can perform their duties despite the inaccuracy of the models for which they were designed. This idea is the basis of this thesis, which will show how to create a planner that accounts for the imprecision of its model's predictions.

The actuators and sensors for one level of the process are often sublevels themselves. Thus the inaccuracy of the sublevel's model is likely to cause the performance of the upper level to be imperfect or unreliable. Thus other levels which rely upon the performance of sublevels should make their plans with this in mind. The planning techniques that this thesis describes can make use of statistical knowledge about the performance of lower levels. Thus planners can be developed that not only can plan with inaccurate process models, but can also anticipate both unreliable information from lower levels and the plan's unreliable execution by lower levels.

2.4 Rating the Output of a Process via Penalty Functions

At this point we now have 1) task specifications, 2) a process to perform the task, 3) sensors to monitor the process, and 4) actuators to manipulate the process. What is needed is a controller or planner with a strategy to use the sensors' outputs to determine how the actuators are to manipulate the process to achieve the task specifications. For complex levels of processes, this will be done by using the process model to predict the behavior of the process when a particular control is applied, and comparing the process output to the task specifications. For the more complex levels of the hierarchy, the comparison scheme must be able to measure different outputs of an unreliable process. Then the strategy can choose the appropriate or best control to attain the task goals.

The planning or control task is necessary at every level of the process, and has been well-implemented for the simpler levels. It is the implementation of the planner at the higher levels that concerns us here. A common technique for doing this is via

weighted penalty functions. Each pertinent task specification is expressed quantitatively in terms of sensor data, and converted by means of a suitable mathematical expression into a value or a penalty which decreases as the condition becomes more desirable. These penalties are then given weights relative to one another so that a single number can be computed as a measure of overall task performance.

Current practice calls for specifying tolerances for process parameters such as specifying that a chemical reaction must be maintained at 100±0.5 °C. This is depicted in Figure 2-3 below.

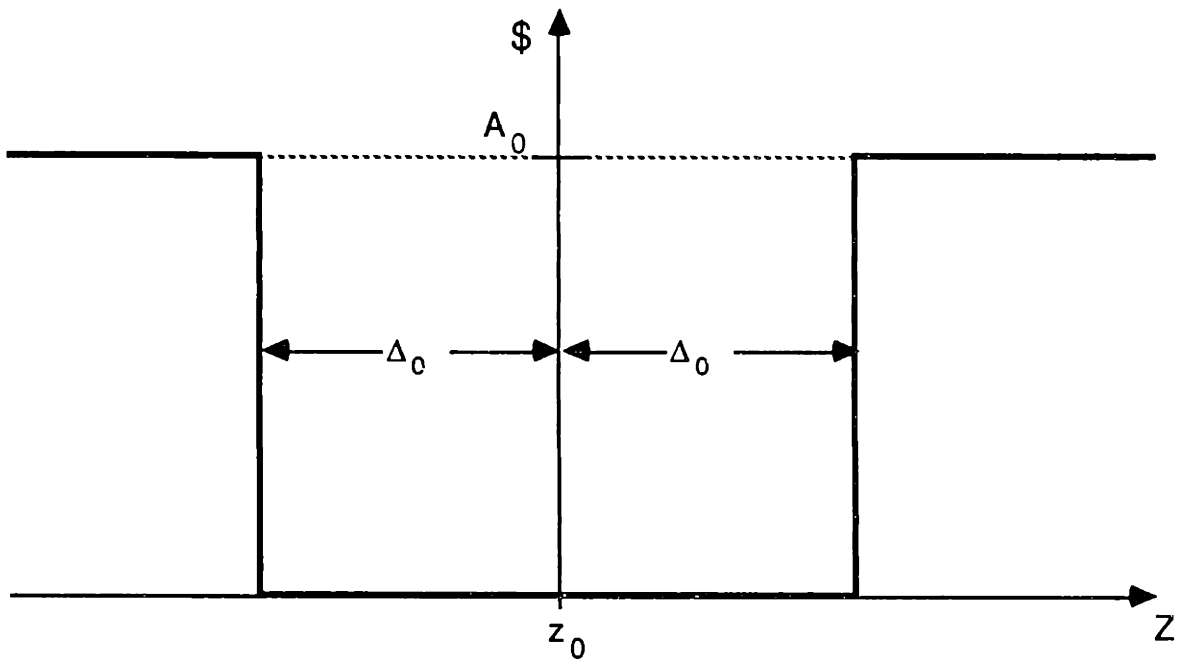


Figure 2-3 Tolerance Penalty Model

Any value outside of this range is deemed out-of-tolerance, and charged a monetary penalty. This is illustrated in Figure 2-3 and is called a **penalty function** or a **loss function**. The horizontal axis describes the output value of a system or process, z, and the vertical axis describes the resulting penalty, $(z), expressed in monetary terms. The desired output value of 100°C is marked by z_0 in the figure. The tolerance range of ±0.5°C is represented by Δ_0 in the figure. If a process consistently (i.e. deterministically) outputs a particular value, the process can be rated by the corresponding penalty. Algebraically, this can be expressed as:

$$\wp = \$(z) \tag{2.1}$$

where: $\wp$ = the penalty assigned to the process

z = the process output

$\$(z)$ = the penalty function

For the tolerance range scheme, this becomes:

$$\wp = \begin{cases} 0 & \text{if } |z-z_0| < \Delta_0 \\ A_0 & \text{if } |z-z_0| \geq \Delta_0 \end{cases} \tag{2.2}$$

where: A_0 = the penalty for an out-of-tolerance output

Prof. Genichi Taguchi[1] has espoused a different technique for measuring process outputs, generating a value which he too expresses in real monetary units. Dr. Taguchi argues that the above tolerance scheme is an inappropriate measure for several reasons. First of all, most processes are stochastic and never produce a single consistent output value. Processes can and do output consistent *mean* values, but are almost never permitted to consistently output an undesired mean value, because the mean can nearly always be adjusted to the desired value. The problem, he argues, is not one of having a process output that is consistently out of tolerance. It is in outputs that stray randomly from the desired value, both beyond and within the tolerance limits.

If the process produces random output, it is common to penalize it only according to the fraction of outputs that exceed the tolerance multiplied by the charge for being out-of-specification:

$$\wp = f A_0 \tag{2.3}$$

where: f = the fraction of the output outside of the tolerances

Or, the process can be measured by statistical techniques, such as the ratio of the tolerance range to the standard deviation of the process output. This number will be large for good processes. Taguchi disagrees with these metrics.

Taguchi's argument goes as follows: 1) A process that yields 100.51°C is only marginally different from one that yields 100.49°C, though processes that yield these two results would score very differently under the tolerance range scheme. 2) Penalty functions should at least be continuous because systems that perform nearly the same should score nearly the same. 3) It is more realistic to choose smooth penalty functions. 4) The simpler the form of the penalty function, the better. It is not efficient to try to model the penalty function accurately. Rather, determine the value of a few points and draw the simplest (i.e. lowest-order polynomial) curve through it. Start by

assigning a zero (0) penalty to a process output occurring right at the desired value. In most cases, one also knows a penalty for an output right at the tolerance limits, being the same as that charged for out-of-tolerance results in the tolerance scheme. This yields two more points (one at the lower tolerance limit, and the other at the upper tolerance limit) for a total of three points. The simplest curve that can generally be passed through three points is a parabola, as illustrated in Figure 2-4 below.

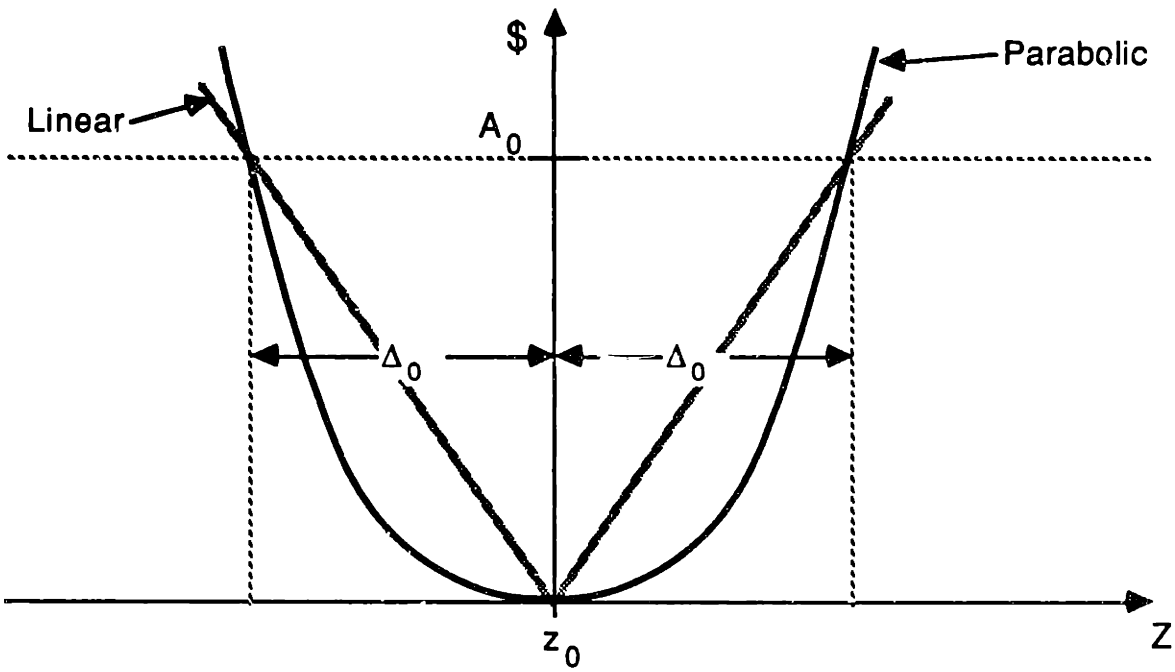


Figure 2-4 Taguchi's Penalty Function

This has the effect that deviations from the desired value within the tolerance range are charged smaller penalties than a linear penalty function would, and deviations outside of the tolerance range are penalized more than a linear charge would. For one-sided situations in which the smallest value is best and a negative value is undefined (e.g. the amount of foreign matter remaining after a cleaning process), a parabola is still used for consistency. An alternative derivation is to consider expressing the penalty function in a Taylor series expanded about the desired value x_0, such as equation 2.4 below.

$$\$(z) = \$(z_0) + \frac{\partial \$}{\partial z}(z_0)\frac{(z-z_0)}{1!} + \frac{\partial^2 \$}{\partial z^2}(z_0)\frac{(z-z_0)^2}{2!} + \ldots \qquad (2.4)$$

Here, $\$(z)$ is the penalty function. By definition, $\$(z_0) = 0$, and since the least penalty is to be at z_0, $\frac{\partial \$}{\partial z}(z_0) = 0$ identically. This leaves a penalty function in the form:

$$\$(z) \propto k(z-z_0)^2 \qquad (2.5)$$

If the penalty is A_0 at a distance Δ_0 from the desired value (see Figure 2-4), then k can be computed by substituting A_0 for $(z) and Δ_0 for $(z-z_0)$ to obtain:

$$k = \frac{A_0}{\Delta_0^2} \qquad (2.6)$$

An important point to remember is that the actual form of the penalty function usually does not affect the resulting value of the penalty enough to merit a study to determine the function to a high degree of accuracy. The parabolic penalty function is good enough in most cases, and even a linear penalty function is arguably a good approximation to the real penalty function.

If the two data points at the tolerance limits are unavailable, alternative data points can be determined efficiently by finding a point at which some action must be taken. For example, consider the manufacture of parts that must later be assembled. If the parts deviate from the specified size, there arises the possibility that they might not fit together. Determine the dimension at which the two parts won't fit together if one of them has that dimension. Evaluate the options then. How much will this situation cost? If a part has to be scrapped, the penalty becomes the scrapping cost. If a part has to be repaired, the penalty should be the repair cost. Include the inspection or assembly line disruption costs. If the firm is a retailer or supplier, the penalty would be the repair-on-site or warranty costs, and should include an estimate of the loss-of-business or loss-of-reputation costs, both difficult to determine precisely, but which can be estimated. Taguchi reports that this usually yields higher penalties for out-of-specification processes than the firm had previously estimated, and he calls this the **cost of quality**, although it is perhaps better termed the 'cost of poor quality'.

If a failure can only be described in probabilistic terms, then use that information to formulate the model. For example, if there is only information that a lathe tool has a 50% chance of chattering at a given feedspeed, then apply 50% of the cost of chattering to that feedspeed. In the above assembly of two parts example, the probability of a misfit can be computed given the dimensional statistics. The expected cost can be computed from these statistics and the cost of an actual misfit [17]. Complex cost analyses are not required; this is a first-order cost estimate.

Multiple process outputs are penalized individually as above, and the sum of these penalties is used as the process metric. Weighting among the outputs is achieved by expressing the penalties in common monetary terms. Process constraints and

conditions to be avoided can be dealt with in a similar manner and summed to the overall penalty function.

Dr. Taguchi recognizes that the output of processes is usually probabilistic, and that it is nearly always possible to adjust the mean of a process to the desired value, so he uses just the variance of the process to compute the penalty. If the process variance is σ^2, the penalty, $\wp$, assigned to that process is computed as:

$$\wp = \frac{A_0}{\Delta_0^2}\,\sigma^2 \qquad (2.7)$$

However Taguchi's method is too restrictive because it assumes that all process outputs can be adjusted so that the mean output is at the desired value. This is not generally the case, and limits this method to single-stage processes in which the process is executed, is expected to have its mean output at the desired value, and then the ensemble output is evaluated. There are, however, multi-stage processes in which a sub-step or partial step is executed and must be evaluated by how closely a desired final goal is reached. This occurs in iterative processes, which must be operated on a "repeat one step until a final goal is reached close enough" basis. Each step is evaluated by the same final goal criterion. But the intermediate steps may not be expected to yield results at the final desired value because to have an intermediate step attempt to satisfy the final goal in one jump may violate other constraints. Dr. Taguchi's metric requires that the mean value of the output of a process be compared with the desired final output, and thus cannot measure the intermediate steps of multi-step processes. Weld bead grinding, described in more detail in Chapter 3, is such a multi-step process.

A technique that avoids this restriction is to compute the **expected penalty**, using the penalty function and the probability distribution for the final output. This can be computed as follows:

$$\wp = \int_{\forall z} p(z)\ \$(z)\ dz \qquad (2.8)$$

where: $p(z)$ = the probability that the output z will occur

$\$(z)$ = the penalty charged for an output z

This is illustrated in Figure 2-5 below, which compares this technique to Taguchi's. The horizontal axis depicts the output value. The vertical axis depicts simultaneously the penalty and the probability that the corresponding output will occur.

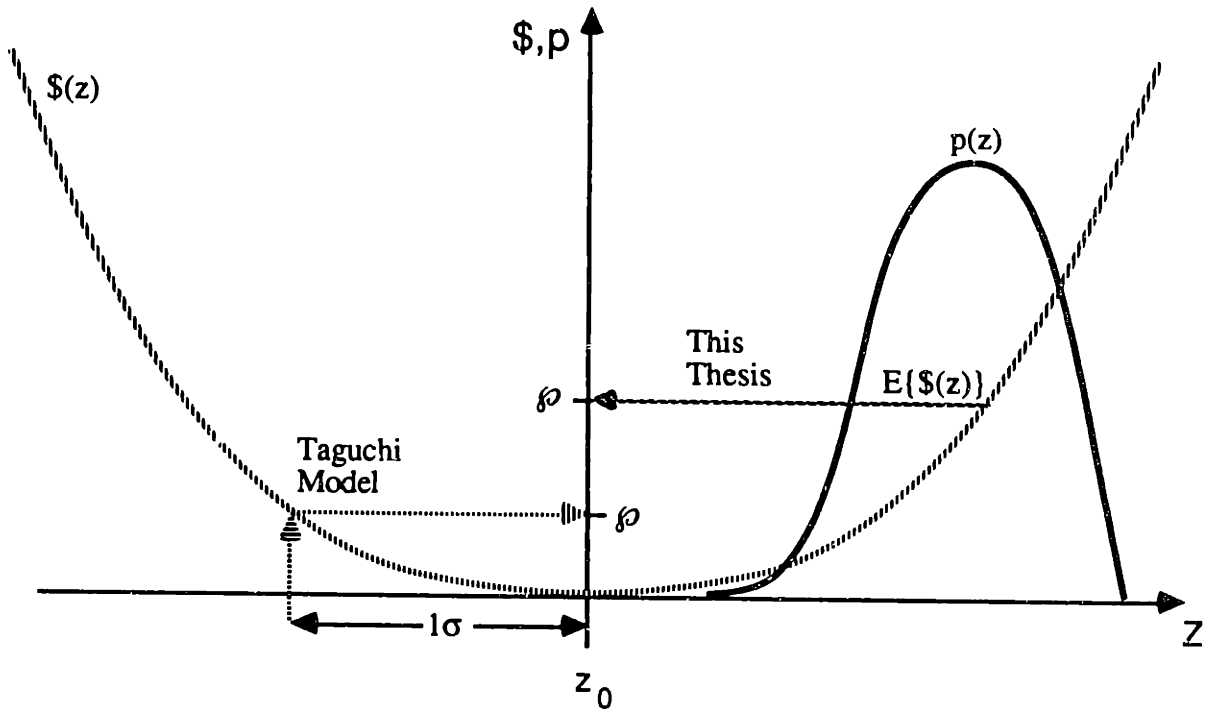


Figure 2-5 Computation of the Expected Penalty vs. the Taguchi Method

This technique is more flexible than Taguchi's, because is uses the entire probability density function (PDF) to evaluate the output, and can rate the output of intermediate steps. It is also conceptually more appealing, since expected penalties are a more appropriate way to compare probabilistic outcomes. The statistics of the process describe what *can* happen to the process, and the resulting expected penalty function can be used to evaluate the consequences of process control decisions. However, this method needs a probability function for the process outcome, and as this subject belongs more in the realm of process modelling than it does in penalty modelling, detailed discussion of this will be postponed until the next section.

Thus three techniques have been presented that will yield a single number, a penalty, to describe the performance of a system or process. These involve constructing a penalty function that maps process outputs to penalties that can be used to compare one process output to another. Such penalties usually increase the further the output varies from that desired. A simple quadratic form for the penalty function has been proposed. If the process yields the same output time after time, it is charged the penalty corresponding to that output. If the process is stochastic, two other techniques have been proposed for evaluating the resulting penalty. The first assumes that the process can be adjusted to have its mean output coincide with the desired output, and evaluates

the penalty at one standard deviation from that output. The second makes no assumptions about the mean output value, and evaluates the expected penalty. This thesis will use this last implementation of the penalty function for evaluating the outputs of higher levels in the hierarchy.

2.5 The Process Model

We now have task goals for each level, task specifications to describe these goals, and the makings of a scheme to measure how well an unreliable process satisfies these goals. The next step is to design a planner or controller to use this measurement to manipulate the process to satisfy the specifications. However the form and structure of such a planner is strongly related to the form and structure of the particular process or level of a process it is to control. And the probability density function required by the penalty function scheme is also strongly dependent upon the nature of the process. So this section will briefly describe types of process models, and will discuss the generation of probability density functions for the outputs of processes.

Types of Process Models

Process models are needed at all levels of the planning/controlling system. These models can vary widely in detail and complexity from level to level within a particular manufacturing system, and from system to system. The basic requirement of a process model for planning or control is that it relates variables which can be externally controlled to variables which are indicative of the desired output. Process models can be classified into two types, **static** and **dynamic**. Dynamic process models use differential or difference equations to describe the behavior of the process as a function of time. Dynamic processes can also be described using a discrete model in the form:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{F}(\mathbf{x}_k, \mathbf{u}_k, k) \\ \mathbf{y}_k &= \mathbf{G}(\mathbf{x}_k, \mathbf{u}_k, t) \end{aligned} \tag{2.9}$$

where: $\mathbf{x}_k$ = a vector of the state variables at time k

$\mathbf{u}_k$ = a vector of the inputs to the process at time k

k = the time index

$\mathbf{y}_k$ = the vector of outputs at time k

$\mathbf{F}(\mathbf{x}_k, \mathbf{u}_k, k)$ = the function describing the behavior of the state variables

$\mathbf{G}(\mathbf{x}_k, \mathbf{u}_k, k)$ = the function describing the outputs

Such a model divides time into discrete increments, with each increment having an index k. Thus the state of the process after one time increment has passed is a function of the current state, the current control, and the current time. Such a model clearly illustrates the concept of a **state transition**, which will be used extensively in Chapter 5. Digital computers must use discrete models to describe dynamic systems, and so such models are becoming increasingly common.

These models can be used to describe continuous processes, but are particularly suited to describing staged processes. Multi-pass cutting operations (like grinding) are one such type of staged process. One can't evaluate the condition of the workpiece as a whole until after an entire pass has been completed, so it is appropriate at one level to think of the process in terms of discrete events or stages, one grinding pass at a time. Of course, the cutting process itself – the microscopic moment-by-moment action of the cutting tool upon the grains of material – can be modelled by a continuous dynamic or static model. Thus the whole process can be modelled on two (or more) different levels. The former model considers the workpiece as a whole from pass to pass. This is useful for planning the sequence of cuts. The latter considers the process on a much finer level of detail, both spatially and chronologically. This type of model would be useful for active feedback control of the position of the cutting head, for example. Other types of staged processes are iterative metal bending, batch chemical processes, and inventory control.

Static models do not use differential equations, but generally use algebraic relationships such as:

$$\mathbf{y} = \mathbf{H}(\mathbf{u},t) \tag{2.10}$$

Such a model would be appropriate if no differential equations (or equivalently, discrete models) are required to adequately model the process. Each type of model is appropriate in certain cases, though static models tend to be simpler to work with.

Modelling Error

As was stated earlier, due to tradeoffs there will always be inaccuracy in the process models. The modelling error and the external noise can combine to make a process poorly predictable. Obviously some processes are easily modelled with accuracy and are not susceptible to external influences and so are highly predictable. These processes are generally easily automated. The problem remains as to what to do with those processes that are poorly modelled or have lots of process noise.

Dr. Taguchi has two recommendations for dealing with noise-sensitive processes. The first involves redesigning the process (via parameter adjustment) to obtain a less noise-sensitive process. The second is to identify the most critical noise sources (typically random variation in material or parts quality) and reduce those variations (by specifying tighter tolerances on those parts). He describes a method in which the first technique can be accomplished cheaply via efficient experimentation, and recommends that this method be applied first because it is cheaper and can be more effective than the second method. The second method is stymied if the noise sources cannot be adjusted or controlled easily.

Sometimes it is possible to model the modelling error and external process noise stochastically as a single process noise (both internal and external noise, internal noise referring to the modelling error). Modelling the noise can be done by hypothesizing a cause for the noise, couching this in a stochastic variable ε. The discrete dynamics might then look like:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{F}(\mathbf{x}_k, \mathbf{u}_k, \varepsilon_k, k) \\ \mathbf{y}_k &= \mathbf{G}(\mathbf{x}_k, \mathbf{u}_k, \varepsilon_k, t) \end{aligned} \qquad (2.11)$$

The presence of the noise can be modelled in any part of the dynamics. Note that this can be done with the other types of process models as well. Such a model makes the output a stochastic variable. The remaining problems are to figure out the statistics of the noise ε_k, and to figure out how to propagate those statistics into those of the output $\mathbf{y}_k$.

Getting the form of the probability function for the output can be difficult, especially if statistics are available only for characteristics imbedded within functions, as considered above. In this case, the probability density function must be propagated through the function that relates the lower-level characteristics to the output [17]. This can be extremely difficult, if not impossible, to do analytically in the general case, and is usually quite time-consuming to do numerically with accuracy by techniques such as a Monte-Carlo analysis. When the PDFs of the imbedded characteristics and the output can be approximated by the Normal PDF, it is possible to propagate PDFs through a limited number of functions, as is described in the sequel.

Propagating Probability Density Functions

First the propagation of general PDFs will be described, and then the specific case when the PDFs are all assumed to be Gaussian. Consider a static monotonic

function $z = f(x)$, where the PDF of x is an arbitrary (i.e. not necessarily Gaussian) function $p(x)$. Let the PDF of z be $q(z)$. Then:

$$q(z) = p(\phi(z)) \left| \frac{d\phi(z)}{dz} \right| \tag{2.12}$$

where: $\phi(z) = f^{-1}(z)$, the inverse of the function $f(x)$

This is the basic form for propagating PDFs through functions. For a proof and extensions to non-monotonic functions (i.e. functions whose inverses are not one-to-one), see any fundamental probability textbook, such as [18].

When the PDFs can be approximated by the Gaussian PDF, only the mean and variance are required for a complete description of the PDF, and these can be propagated through functions by at least three techniques: 1) maximum likelihood estimators, 2) moment generating functions, or 3) partial derivative methods. The latter method will be briefly described in the following. For more details and applications of all three methods, see [17].

Consider the function of two variables, $z = f(x,y)$. Expand a particular value of z, z_i, about its sample mean value m_z[1]:

$$z_i = m_z + \frac{\partial z}{\partial x}\delta x_i + \frac{\partial z}{\partial y}\delta y_i \tag{2.13}$$

where: the derivatives are evaluated at the mean values of x and y, m_x and m_y, respectively.

δx_i = the variation of x from its mean for point i

δy_i = the variation of y from its mean for point i

The variation of z can be written as:

$$\delta z_i = z_i - m_z = \frac{\partial z}{\partial x}\delta x_i + \frac{\partial z}{\partial y}\delta y_i \tag{2.14}$$

---

[1]The true mean and variance will be described by μ_x and σ_x. The sample mean is defined as:

$$m_x = \frac{1}{n}\sum_{i=1}^{n} x_i$$

and the sample variance is defined as:

$$s_x = \frac{1}{n}\sum_{i=1}^{n} (m_x - x_i)^2$$

The mean value of z is the function f(x,y) evaluated at the mean values of x and y:

$$m_Z = f(m_X, m_Y) \tag{2.15}$$

The variance of z is derived as follows:

$$\begin{aligned} s_Z^2 &= \frac{1}{n}\sum_{i=1}^{n}(\delta z_i)^2 \\ &= \frac{1}{n}\left[\left(\frac{\partial z}{dx}\right)^2\sum_{i=1}^{n}\delta x_i^2 + 2\frac{\partial z}{dx}\frac{\partial z}{dy}\sum_{i=1}^{n}\delta x_i\delta y_i + \left(\frac{\partial z}{dy}\right)^2\sum_{i=1}^{n}\delta y_i^2\right] \\ &= \left(\frac{\partial z}{dx}\right)^2 s_x^2 + \left(\frac{\partial z}{dy}\right)^2 s_y^2 \end{aligned} \tag{2.16}$$

The third equality is true since $\sum_{i=1}^{n}\delta x_i\delta y_i \rightarrow 0$ if x and y are independent. This technique can be used to propagate means and variances through a variety of functions. It should be noted that the resulting probability density function, p(z) will be Gaussian, in general, only when the function f(•) is linear. However, the resulting PDF will often be well-approximated by a Gaussian PDF, particularly for small variances. The Gaussian approximation will be good enough for our purposes in most cases, since the penalty function with which it will be used is only an approximation itself. Table 2-1 gives mean and variance propagations for a few simple functions of X and Y independent:

Table 2-1 Selected Propagations of Mean and Variance

Function	Mean, m_Z	Variance, s_Z^2	Correlated Variance[1], s_Z^2
$Z = X + Y$	$m_X + m_Y$	$s_X^2 + s_Y^2$	$s_X^2 + s_Y^2 + 2r\,s_X s_Y$
$Z = X - Y$	$m_X - m_Y$	$s_X^2 + s_Y^2$	$s_X^2 + s_Y^2 - 2r\,s_X s_Y$
$Z = XY$	$m_X\,m_Y$	$m_X^2 s_Y^2 + m_Y^2 s_X^2 + s_X^2 s_Y^2$	$m_X^2 m_Y^2\left(\frac{s_X^2}{m_X^2} + \frac{s_Y^2}{m_Y^2} + \frac{s_X^2}{m_X^2}\frac{s_Y^2}{m_Y^2}\right)(1+r^2)$
$Z = \frac{X}{Y}$	$\frac{m_X}{m_Y}$	$\frac{1}{m_Y^2}\left(\frac{m_X^2 s_Y^2 + m_Y^2 s_X^2}{m_Y^2 + s_Y^2}\right)$	$\frac{m_X^2}{m_Y^2}\left(\frac{s_X^2}{m_X^2} + \frac{s_Y^2}{m_Y^2} - 2r\frac{s_X}{m_X}\frac{s_Y}{m_Y}\right)$

Operations with one variable constant can be propagated by using the expressions above and substituting s=0 for the constant variable. Extensions to functions of more

---

[1] r is defined as the **coefficient of correlation**, which = 0 if the variables are independent, and = 1 if the coefficients are completely linearly dependent upon each other

than two variables are similar, and extensions to other functions are possible through the application of formulas (2.15) and (2.16). Note that the results for $Z = X/Y$ are approximations which ignore higher-order terms of ratios s/m, and that the PDF of Z in this case will not be Gaussian, but approaches Gaussian for small ratios of s/m.

Often the form of the output probability distribution is known, or can be estimated from experimental data, or can be hypothesized. Then a probability density function of the desired form can be constructed which has the mean and variance as propagated above. This technique has the flexibility of using the knowledge of the probability density function if it is available. If there is no such candidate PDF, then a Gaussian PDF can be hypothesized as good as any.

For example, a Gaussian PDF would be inappropriate for modelling stochastic material removal processes, since for these processes material can only be removed and never added. A Gaussian probability density function is defined over the entire set of real numbers, in particular for all negative numbers, and so would be incorrect for modelling the amount of material removed after a cutting process, which cannot be negative. It would be more appropriate to use probability density functions defined only over the set of positive real numbers, such as the Rayleigh or Log-Normal probability density functions for this purpose. This is an approximation, to be sure, but it is an approximation of the same order of accuracy as was that of the penalty function. See Chapter 3 for a more detailed description of the application of this technique to the dynamics of weld bead grinding.

But what characteristics are to be propagated? What fills the roles of X and Y in Table 2-1 above? There are two types of characteristics that can be propagated. The first is a process parameter that is historically known to vary with a given PDF. This is common for material properties such as metal strengths and hardnesses, material compositions, fluid viscosities, and empirical model coefficients. If these can be independently measured over a period of time, their statistics can be computed, and in some cases, their PDFs. The second type of propagatable characteristic is a known poor actuator or sensor performance. For example, consider a chemical reaction process whose input variables are elements like reagent composition, flowrate, and temperature. A reaction planner controls the quality of the output by commanding the second two inputs according to plan. A variable reagent composition is the first type of characteristic. However, the flowrate and temperature are themselves controlled by servo subsystems. An imperfect temperature control subsystem might not be able to control the temperature exactly as the planner specified; the temperature varies

somewhat, apparently stochastically to the planner. It may be possible to collect the statistics of the temperature. They may even be correlated or a function of the process variables, a fact that a Taguchi analysis or other similar series of experiments could reveal, and could be exploited by a planner such as that described in Chapter 5. Details of such a controller vary greatly with the application, and few general statements can be made.

This section described some common types of process models. These were classified into two types, static and dynamic. Dynamic models use differential equations or discrete state transition equations to describe the behavior of processes. Static models generally use algebraic relationships and are generally simpler and easier to use than dynamic models. There are tradeoffs in how complex a model to build, both with its ease of use and with the effort required to build an accurate model. One must stop model development at some point with a somewhat inaccurate model. This makes the model an inaccurate predictor of the process' behavior. There are also uncontrollable and unpredictable external noises to a process that also make it difficult to predict. These can be ameliorated, but only at cost. Knowledge of the statistics of the prediction inaccuracy can help in planning or controlling that process, to guide it towards desirable behavior and to keep it away from undesirable behavior.

2.6 The Process Planner

Given task specifications well-defined in terms of the sensor output and a suitable model for the process, the next step is to determine the strategy for manipulating the actuators to achieve the task goals. This can take on many different forms depending on the model and the task goals. For lower-level tasks such as DC motor speed control, this can be implemented in analog electrical hardware, and would be called a controller. Higher-level tasks may require more sophisticated abilities such as hypothesizing situations, forecasting consequences of decisions, planning with those forecasts, etc. Such requirements can be satisfied by techniques such as lookup tables (for output X choose input Y), system identification, adaptive control, nonlinear computer simulation of dynamics, heuristic rule-based planners, AI, expert systems, etc. There are many such techniques, and each has its own appropriate uses, whether the process is static or dynamic, whether the model accurately describes the process or not, and whether external noise affects the process greatly or not. For each of these cases, different levels of complexity may be required of the planner/controller.

Consider simple processes that are modelled statically. If the model is a good representation of the process, then a simple rule-based controller may suffice. For example, consider the control of the depth of cut in a turning or other common NC process. The depth of cut is controlled by the geometrical position of the cutting tool. If the device is physically rigid, the model for the cutting process is "all material on one side of the cutting edge will be removed", and is simple and accurate. The control scheme can be correspondingly simple. If the model has been simplified, or otherwise cannot accurately predict the process behavior, then a more complex controller may be required, using for example heuristics. Consider, for example, a simple peg-into-hole insertion task which has been geometrically modelled, but does not consider friction or treats frictional effects as uncontrollable external noise affecting the position of the inserted part. A heuristic to overcome this problem might be to experimentally note the conditions under which friction causes a problem and write rules to avoid those conditions. Such a rule might be: "do not insert at an angle greater than X degrees." A more advanced approach would be to accurately model how the friction physically affects the parts during the insertion process, and to design a controller capable of eliminating or rejecting the frictional effects.[1]

As the goals and process become more complex, the solution strategies become more complex, particularly if the process model becomes inaccurate. Examples of more complex static tasks are assembly sequence planning, workpiece routing through a job shop, inventory control, and grinding pass planning. Artificial intelligence and expert systems have been used to solve such problems through their ability to imitate human judgement and sort through a complex set of rules. Artificial intelligence has been applied to process planning for machining operations, applying rules for the selection of appropriate tools and machining parameters to create particular features of a part, and for the selection of the correct machining sequence.[19] Fuzzy logic[20] has been applied to problems with requirements and criteria that can be expressed in varying degrees. Whereas threshold logic states in a binary fashion whether a condition is satisfied or not, fuzzy logic allows a condition to be partially satisfied. Fuzzy logic uses expert-derived curves to define the degree of satisfaction of the condition and uses this to determine the degree of action to take. Optimization theory has been applied to

---

[1]The Remote Center Compliance is a successful example of this approach, developed after a study of the mechanics of part insertion. It is a controller implemented in passive mechanical hardware, and is particularly elegant in its simplicity.

solving work-routing and inventory control problems.[21] Both analytic and numeric solutions have been found, under varying assumptions. Optimization solutions are discussed in more detail in the next section.

The control of simpler (lower-level) dynamic processes has attained a high degree of sophistication, and there is a plethora of knowledge about this topic. When there are good process models, the controller need only be designed for performance and robustness to external noise. When the model is less accurate, more advanced controllers can be applied. Such controllers must exhibit robustness to modelling errors as well as to external noise. More complex (upper-level) dynamic processes may require nonlinear control theory or computer simulation of their dynamics as part of the planning/control process.

Such a planner is good for short and very short term processes, for which the task goals are simple ones like: "make the system follow a certain trajectory as closely as possible." More complex tasks often require the resolution of tradeoffs that can't be handled by the limited techniques of advanced control theory. Such tasks more often need a planner that finds the best plan among alternatives, rather than finding a plan that satisfies simple objectives. This is where the techniques of this thesis are most applicable.

2.7 THE OPTIMIZER

Planning for complex processes can be difficult because of the complexity of choices a planner faces when trying to manipulate the process to attain the task goals. The goals are not always binary; they can be satisfied to varying degrees. Some choices satisfy the task goals better than others. So the planning problem can be formulated as an optimization problem whose solution not only yields a plan that *can* attain the goals, but will attain the goals *as well as possible.* The choices can be made via straightforward optimization (for static models), and optimal control theory (for usually linear, dynamic models). When the models get complex, more elaborate optimization schemes (Linear or Dynamic Programming) can be used.

Optimization requires a criterion to be minimized or maximized. This criterion must be expressible mathematically in terms of process model variables. These criteria express desired goals of the process, either in reference to end results or in reference to the manner in which the process is operated. Examples of such criteria are: minimize total energy expended in the process, or minimize the wear in the process, or minimize

the time required to complete the process, or minimize the variation in the process output. These usually can be expressed as a penalty function in economic terms: the cost of the energy, wear, time, or output variation. This also allows various criteria to be traded-off against each other where several are applicable. Penalty functions were discussed in detail earlier in this chapter. Some criteria are better expressed as constraints which the optimizer operates within. The optimizer then becomes a planner, selecting the controllable parameters to attain an optimal process plan within the constraints of getting the task completed as desired.

The planner for the top level of the weld bead grinding system is such an optimizer. It seeks to minimize the penalties associated with such task goals as: 1) minimize overgrinding, 2) minimize undergrinding, 3a) absolutely no overtime, or 3b) minimize overtime, 4) do not burn the workpiece. Criteria 3a and 4 are constraints within which the optimizer must find a solution. The other criteria are expressed as penalty functions to allow one to be traded off against another. It uses a stochastic static version of the grinding model to generate a plan for grinding, using the mean grinding force and the feedspeed as the process controls. This planner accounts for the fact that the uncertain result of one step of its grinding plan will be the initial condition for the next step of its plan. Details of this planner are given in Chapter 5.

2.8 Chapter Summary

This chapter has presented the elements required for extending the intelligence of manufacturing control systems, and has introduced techniques for implementing these elements. The presentation began with a description of the common technique of breaking up the manufacturing problem into a decision term-based hierarchy, and dividing each level of that hierarchy into functional elements. Then it was shown how this procedure and fundamental modelling tradeoffs will generally lead to a manufacturing system whose output varies from the desired output, and that this variation is generally worse for more complex levels of the hierarchy. Thus such levels require planners or controllers that can plan with uncertain process models. Such planners then will require a scheme for comparing plans that have uncertain outcomes. This chapter presented such a scheme, which uses an expected penalty to compare process outputs in terms of the goal of that level's tasks. This is computed via quadratic penalty functions and the process output's probability density function, propagated through the process model if necessary. This penalty function is used by an

optimizing process planner which can find the best process plan to achieve the level's goals. This optimizer is a form of process planner that is specialized for complex levels of the hierarchy. Less complex levels of the hierarchy can use planners and controllers developed by other workers in fields such as modern control theory.

The techniques introduced here will be demonstrated in a real application (weld bead grinding) in the following chapters. The grinding process will be modelled hierarchically, and specific models for each level of the hierarchy will be developed. Controllers and planners will be developed for each level of the hierarchy, with particular attention paid to the most complex level's controller.

The next chapter describes the weld bead grinding problem and the hierarchical structure of the control system developed for it. The following two chapters describe the most complex level of the hierarchy, first presenting the grinding model, then describing the controller. The two chapters after those each discuss one of the lower levels of the hierarchy.

3. INTELLIGENT GRINDING SYSTEM OVERVIEW

The remainder of this thesis will demonstrate the application of the general techniques discussed in Chapter 2 to the problem of weld bead grinding. This chapter will present an overview of the weld bead grinding problem and its issues. The process hierarchy will be defined, and the goals, issues, and elements of each level of this hierarchy discussed briefly. Subsequent chapters will fill in the details of each level.

A particular example of weld bead grinding considered is the removal of weld beads that remain on the exterior of an automobile after its roof is welded to the door pillars. For cosmetic and structural reasons the resulting weld bead must be ground smooth with the parent material so as to be undetectable after painting. This is currently done by humans wielding heavy grinders, and is a tiring and dirty job. It is also unhealthy due to the airborne grinding grits. This type of grinding is illustrated in Figure 3-1 below.

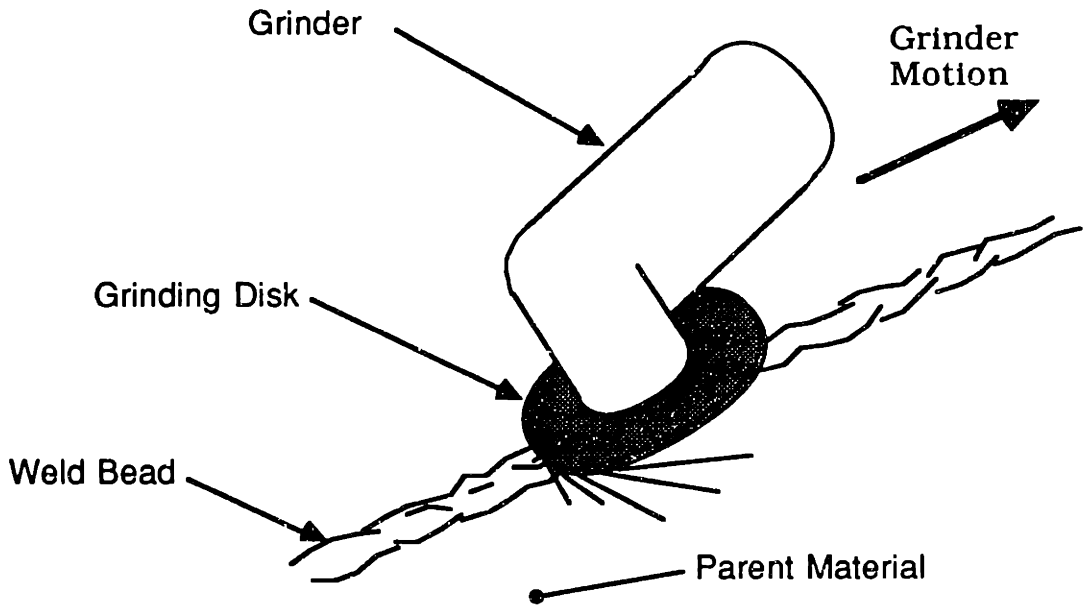


Figure 3-1 Weld Bead Grinding

3.1 WELD BEAD GRINDING TASK REQUIREMENTS

The requirements for the weld bead grinding task are:

1. Remove the entire weld bead accurately This means that when the grinding system is done there should be no weld bead material remaining,

a conditions called **undergrinding**, nor should the grinder have removed too much material, called **overgrinding**. If there is material remaining at the end of grinding, the job is not complete, and some other agent – another grinding system or human grinder – must complete the task. This is costly due to the extra equipment and/or manpower required. Overgrinding can cause costly damage to the parent material beneath the weld. In the case of weld beads on automobile bodies, current practice calls for the resulting damage to be tagged for repair after the entire car has been completed. As a result, overgrinding is generally more expensive than undergrinding[22].

2. Surface contour and finish In some cases (automobiles, for instance), the final goal is to have a surface smooth enough to be undetectable after painting. This requirement has two parts: a **contour smoothness** requirement, and a **surface finish** requirement. The contour smoothness requirement insures that the weld bead blend in with the parent contour so that the transition from weld bead to parent material be undetectable, and that the final shape be aesthetically acceptable. This implies dimensional accuracies of less than 0.1 mm. The surface finish requirement insures that the microscopic finish of the resulting ground surface have a low surface roughness, also for aesthetic purposes. Note that the surface cannot have too fine a surface finish either, since paint will have trouble sticking to it.

3. Weld bead path The grinding system must be able to handle weld beads having complex three-dimensional paths, such as one on a car body or casting.

4. Time considerations The weld bead grinding system must respect factory scheduling constraints. In particular, it is important that the grinding system not take too long to complete its task. Timing requirements can be softened when there are work-in-process buffers both upstream and downstream of the grinding workcell. However the benefits of buffers are offset by the fact that buffer sizes are frequently reduced because they require valuable room on the factory floor, imply extra work-in-process inventory, and allow large amounts of poor quality work to accumulate before discovery. Reduction of inferior quality inventory is a fundamental principle behind Just-In-Time manufacturing (JIT), which is now gaining

popularity worldwide. Where there is no buffer before or after the grinding cell, a time constraint in the form of a **deadline** can be imposed. If there is some buffer space, the deadline constraint can be relaxed and a **lateness penalty** can be imposed.

5. Burning In grinding it is possible to temper the metal if too much heat builds up. This is known as **burning**, and will be described in more detail in the following chapter. Burning results in a structurally weak weld, and must be avoided. Avoiding burning is therefore a constraint on the weld bead grinding process.

The first three requirements are primarily geometrical. The first requirement refers to the total volume of the weld bead remaining, and differs from the second requirement, which refers to the final shape. It is possible to have a smooth contour that still contains too much material. The contour smoothness requirement can be addressed as a practical issue by using a flexible grinding disk. However, there are no grinding models which predict the three-dimensional shape of the weld bead. The grinding model described in this thesis is a two-dimensional model, and was derived for a rigid grinding disk. A model for predicting the surface finish is described in the following chapter, and can be used by the top level controller. The weld bead path requirement calls for the implementation of the grinding system on a 6-axis robot.

3.2 Implementation Issues

Further study of this grinding problem uncovers issues which affect the implementation of the control system:

1. Position control cannot be used It is difficult to locate many workpieces such as automobile bodies to same degree of precision required of the final weld contour, so a grinding system which relies on absolute geometry would be impractical. A grinding system that depends on relative geometries would need to use a rigid disk of known dimensions (and a monitoring scheme to track those dimensions as the disk wears). It would also need to rely on either a relative position sensor between the grinder and the workpiece or bracing strategies. A system based on a relative position sensor (see Figure 3-2) must close the loop through the compliances of all the robot's joints, making it difficult to maintain the required dimensional accuracy. It would also require a sensor capable of

accurately determining the distance between the grinding disk and the parent surface during grinding. A sensor that felt the location of the surface would have difficulty with the complex contours of the auto body sheet metal and with weld beads paths that described complex three-dimensional curves through space. On the other hand, a remote sensor could not 'see' the spot where grinding was taking place directly underneath the grinder. Looking just behind this spot would involve some time delay, and this would degrade control performance. A visual sensor could be blinded by the grinding sparks if it looked too near the spot where grinding was taking place, and moving it far enough away to avoid blinding would mean too much time delay. Another applicable sensor is an acoustic sensor, which would not have the required resolution over the required range. We know of no remote sensors that could do the job. There also remains the problem of determining the relative location *along* the weld bead without relying on absolute geometric information.

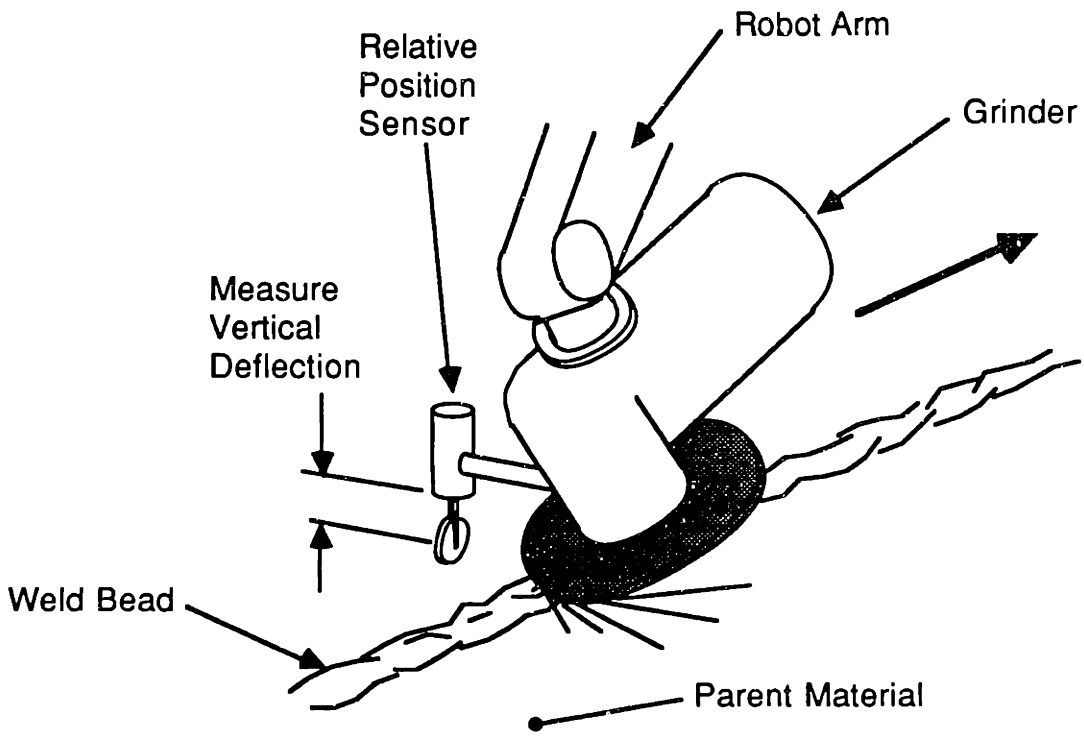
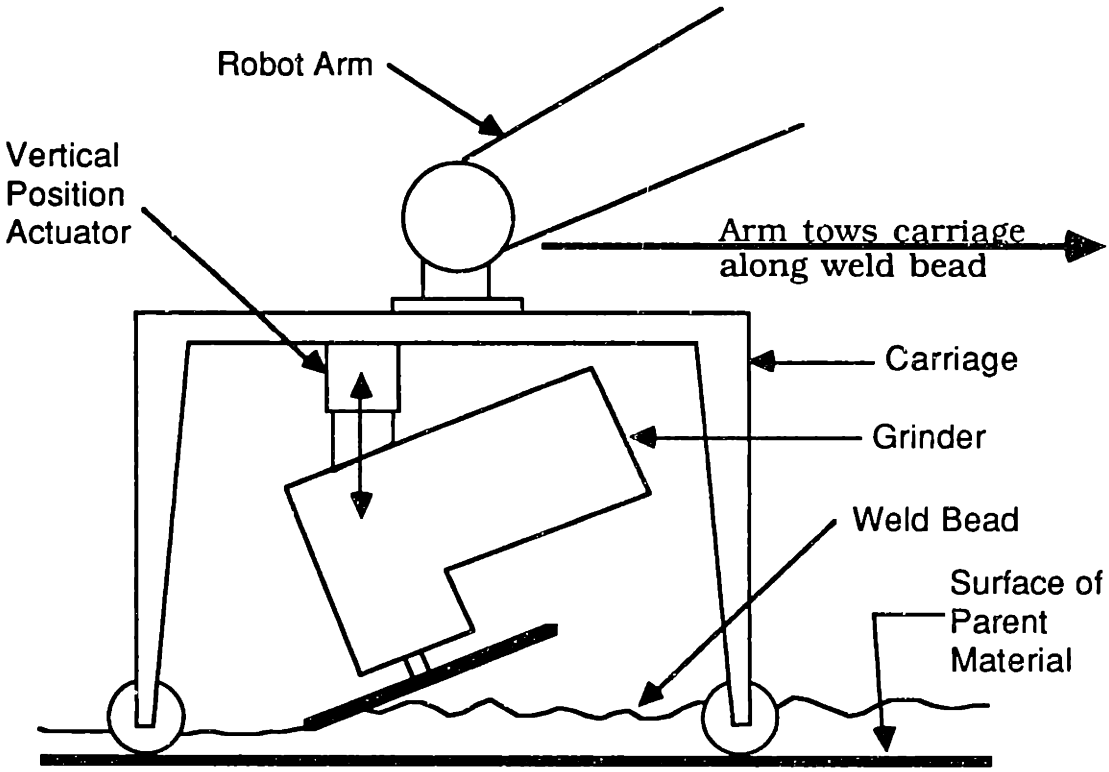


Figure 3-2 Generic Relative Position Sensor

A bracing strategy [23] would have to be designed to avoid damaging the surrounding sheet metal while controlling the substantial grinding forces. See Figure 3-3 for an illustration of an implementation of a bracing

strategy. Bracing strategies would also be vulnerable to dirt getting caught between the brace and the workpiece and ruining the accuracy of the system, and would have the same problems that tactile position sensors have with complex parent material and weld bead contours.



Elevation

Figure 3-3 Bracing Strategy

Preliminary studies of this type of grinding (see the next chapter for details) indicated that it would be possible to control the grinding process via the grinding force. The grinding force could then be controlled via position control and a known compliance. This would require far less dimensional accuracy than either of the pure position control schemes. Our working group had much experience with force sensors and their application in robotics and assembly. So it was decided to try to control the grinding process via the grinding force.

2. Multiple grinding passes are required The early studies of this type of grinding also indicated that there was an inherent physical limit to the material removal rate of the grinding process, and this necessitated

multiple grinding passes to grind a weld bead smooth. This mandated the medium-term type of decisions mentioned earlier, particularly, how many grinding passes to make, and how much material to remove during each pass, and when to declare that the job is done. This is the responsibility of the grinding sequence planner. The time and geometrical requirements can bring about a fundamental tradeoff for this planner near the end of a grinding sequence: what to do if the weld is not completely ground and the deadline is near. The options are: a) quit now with the job unfinished, or b) grind some more and finish late. The grinding sequence planner must be able to weigh the risks and benefits of these two alternatives and make the best decision.

3. Grinding is poorly predictable All grinding models have empirical coefficients that vary with varying workpiece material properties and disk properties. These coefficients can be estimated before a grinding pass only with limited accuracy. Previous research indicated that the cut depth of steady state grinding passes performed under the same nominal conditions varied from 10-20%. Thus the results of a grinding pass are poorly predictable. Any planning system for grinding must take this into account, and plan grinding passes that avoid the risks of violating the task requirements listed above.

3.3 System Structure

The above considerations lead to a hierarchical control system such as that shown in Figure 3-4.

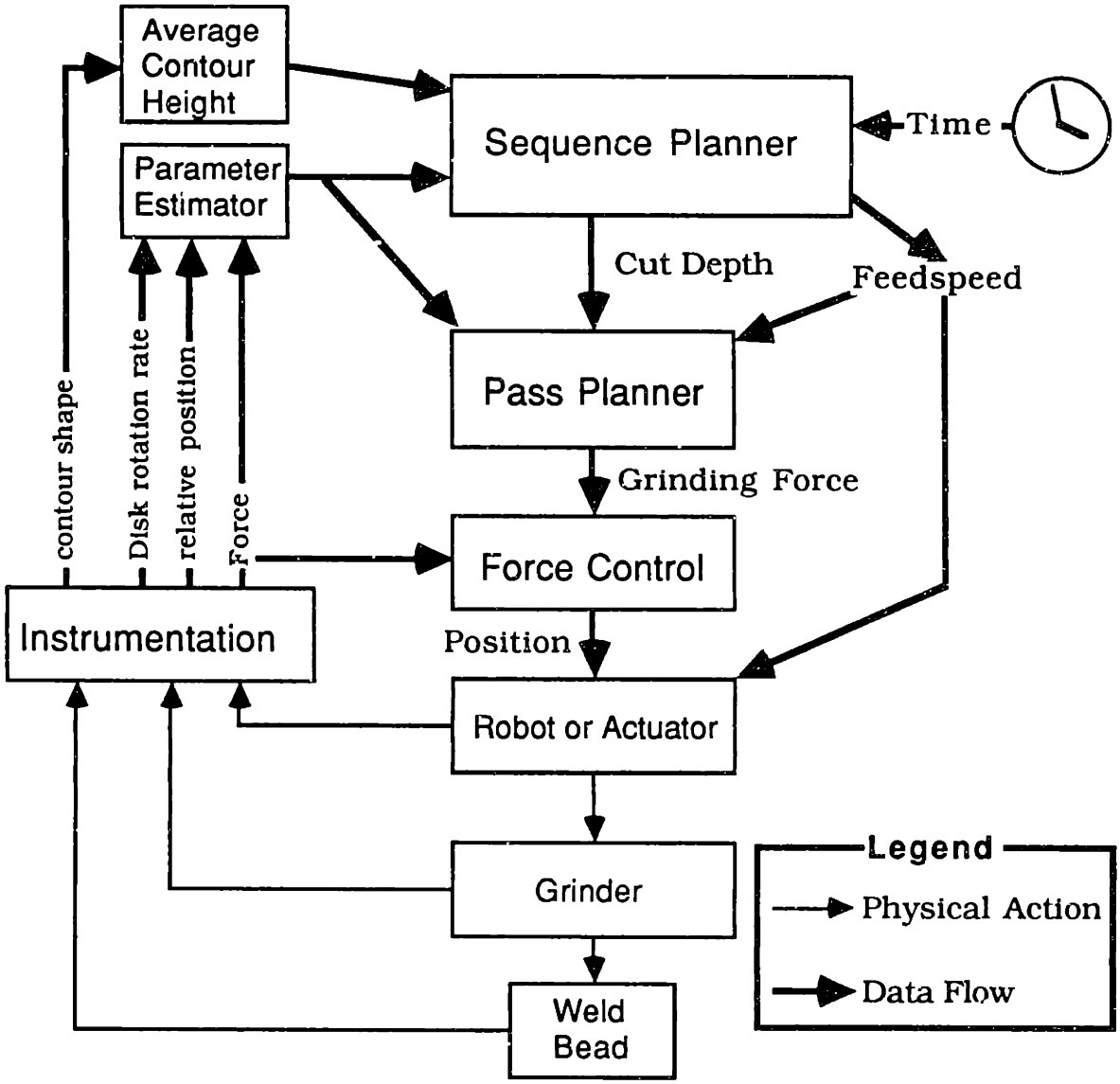


Figure 3-4 Hierarchical Control System

Grinding Sequence Planner

This is the top level of the hierarchical controller in the terminology of the previous chapter, corresponding to a medium term planner, and is responsible for deciding the number and type of grinding passes for a given weld. This is done by specifying the cut depth and feedspeed for each grinding pass in a sequence. This level must create grinding sequences that limit the risk of overgrinding and undergrinding, avoid burning the workpiece, and satisfy the time requirements. Before each grinding pass this level must determine the amount of time and material remaining, decide what cut depth and feedspeed to use, and communicate these requirements to the lower planning and control levels. This level is also responsible for maintaining estimates of the empirical model coefficients. The grinding model used by this planner and its

history are described in the next chapter, and the planner itself is described in detail in the chapter following that.

Pass Planner

This is the second level of the hierarchical control system, corresponding to a short term controller. It computes the desired weld bead contour given the material removal requirements, and uses this, the current model coefficients, and the current contour shape to generate the grinding force trajectory as a function of time. This force trajectory is then commanded to the force controller for execution. This controller is described in detail in Chapter 6.

Force Controller

The force controller commands the robot or force actuator to follow the desired force trajectory, and corresponds to the very short term controller of the previous chapter. This force controller was implemented using a one dimensional actuator in a grinding test stand for this thesis. Real time robot force control for flexible disk grinding was demonstrated by Tate [24]. In both cases force control was achieved using position control and a known compliance. This is described in detail in Chapter 7 of this thesis. The position control loop can be considered as yet a lower level controller, and is analyzed in Appendix B.

3.4 CHAPTER SUMMARY

This chapter presented a control system hierarchy for the weld bead grinding problem. It derived that structure from the nature of the weld bead grinding problem, its requirements, and issues that arose on closer investigation. The task requirements were that a three dimensional weld bead must be entirely removed, leaving a smooth surface contour and finish, and that this must be done within a specified amount of time and without burning the workpiece. Further inspection of the problem indicated that multiple grinding passes would be required, that these could not be predicted accurately, and that it would not be possible to control the grinding process to the desired accuracy via the position of the grinding disk. The resulting control scheme was based upon controlling the grinding force at the lowest level of the hierarchy. The next level up would decide what force trajectory to use to remove a specified amount of material from the weld, and the top level would decide how much material to remove and with how many grinding passes. This top level was responsible for insuring that

most of the task requirements be met, and was designed to make plans that account explicitly for the poor predictability of the grinding process. The resulting plans avoid as well as possible the possibility of violating any of the task requirements.

Given this hierarchy, the remaining chapters describe from the top level down how they were implemented. The next two chapters describe the grinding sequence planner, with an entire chapter devoted to describing the grinding model used by this planner, and the remaining chapter describing the implementation of the sequence planner. The next two chapters describe the implementations of the grinding pass planner and the force controller, respectively.

4. THE GRINDING PROCESS MODEL

This chapter will derive the weld bead grinding process model starting from previous research and first principles. This model is used in three forms in this thesis, corresponding to three levels of the weld bead grinding process. The first section describes previous related work in modelling grinding. The second section derives the static grinding model used by the medium term planner.

4.1 PREVIOUS GRINDING RESEARCH

Grinding, in its various forms, has been studied for many years, coming as an outgrowth of metal cutting theory. One of the earliest of modern treatments of metal cutting is due to Merchant [25–27], who studied the mechanics of metal cutting on a microscopic scale. This work investigated the physics of the chip-tool interface modelling the chip shear plane and the resulting force system. Metal cutting theory proved a fertile topic as other workers extended Merchant's model [28][31,32,34].

Static Grinding Models

Shaw and others extended this work to include grinding. Marshall and Shaw [29] instrumented a grinder and noted the empirical relationships between the grinding forces, disk speed, feedspeed, depth of cut, grit material and size, specimen hardness, and dressing technique. They interpreted their data in terms of the specific energy required to remove the material, and experimentally related that energy to the size of the resulting metal chips, and compared this energy to that required by micromilling and lathing [30]. This work relied heavily upon a geometrical model for the cutting edge path during one type of grinding, and was later extended to other types of grinding in [33]. An adaptation of this work for computer simulation was described in [46]. Merchant's metal cutting theory was later applied in three dimensions at the grit level in [35]. These models are primarily microscopic, attempting to describe grinding at the chip-grit level only.

Hahn [39–43] started at the microscopic level, characterizing the processes that occur a the grit level, and relating this to macroscopic variables, such as the grinding force and the material removal rate. His work concentrated primarily upon precision grinding, in which a cylindrical surface is ground to a precise diameter and finish. This type of grinding is illustrated in Figure 4-1. Both the workpiece and the grinder are held

compliantly, both rotate, and are fed toward each other. The two elements 'mutually machine' one another, with the harder grinding disk wearing more slowly. The contact area between the grinding disk and the workpiece, where the grinding was taking place, is called the **contact patch**. The force perpendicular to the contact patch is called the **normal force**, and the force tangential to the contact patch is called the **tangential force**.

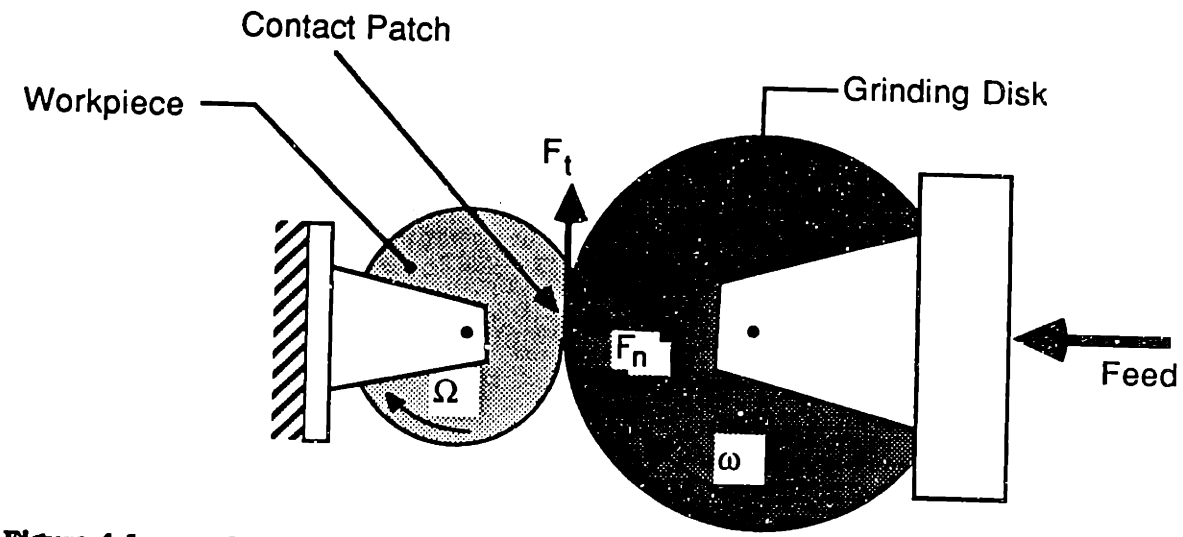


Figure 4-1 **Precision External Grinding**

Hahn described three processes that take place at the microscopic level during grinding: Rubbing, Ploughing, and Cutting. In rubbing, no material removal takes place, but energy is expended in material deformation. Ploughing causes material to be sloughed off on either side of the abrasive grain, and yields low metal removal rates. The cutting process causes a chip to form directly in front of the grain and to peel off, yielding large metal removal rates.

These processes dominate at different force levels between the grinding disk and the workpiece. The normal force correlates well with the material removal rate. At low normal force levels, the rubbing process dominates, and no material removal occurs. At higher force levels, the ploughing process takes over, and generates a small amount of metal removal per energy input. At even higher force levels, the cutting process dominates, and generates still higher metal removal rates. The thresholds for the transitions from rubbing to ploughing to cutting are lower for easy-to-grind (ETG) materials than for hard-to-grind (HTG) materials. Hahn called the ratio of the volume rate of material removal to the normal force the **metal removal parameter**, which is indicative of the efficiency of the grinding process. This parameter decreases as wear flats develop on the abrasive grains and the disk dulls. He called the ratio of the

tangential force to the normal force the **grinding coefficient of friction**, and found it to be roughly constant. Hahn showed empirical relationships between the disk dressing conditions, the composition of the disk, and the metal removal parameter for ETG materials. Hahn also correlated high surface finishes with high grinding forces giving an empirical relationship derived from curve-fitting of experimental data, and showed how this grinding model derived for external grinding of cylindrical pieces could be adapted to internal grinding and flat workpieces via an equivalent diameter.

Malkin [44] approached grinding from an energy point of view, dividing the energy input into sliding (i.e. rubbing), plowing, and chip formation (cutting) specific energies. He experimentally determined the sliding energy by plotting the tangential grinding force vs. the wear flat area, and extrapolating to zero wear flat area. This intercept was the cutting component of the force, and the remaining force was the sliding force component. The specific energies were easily computed by multiplying these tangential forces by the linear disk velocity and dividing by the volumetric removal rate. This specific cutting energy was then correlated experimentally with both the workpiece feedspeed and with the cut depth, decreasing with the increasing values of both feedspeed and cut depth. This was explained by further dividing the cutting energy into plowing energy and chip formation energy. For small cut depths, the grinding grains participated mostly in plowing, yielding relatively little material removal. When the feedspeed was increased, "the maximum undeformed chip thickness also increases, thereby resulting in relatively less plowing and a decrease in specific cutting energy." He then stated that "about 75 percent of the total chip formation energy goes into shearing, with the balance associated with friction." The shearing energy could be approximated by the melting energy of the metal. Some of that shearing energy, and nearly all of the friction energy, heated up the workpiece. His data showed that this results in a total of about 55% of the chip formation energy and 75% of the plowing energy being conducted as heat to the workpiece.

So these two researchers have developed similar macroscopic models for grinding, dividing the energy used into that which goes to metal removal and that which does not. Their models related the grinding power to the material removal rate via experimentally determined coefficients:

$$\Pi = K_1 Q + K_2 \qquad (4.1)$$

where: Q = the volumetric metal removal rate [mm^3/s]

Π = the power delivered to the contact patch [W]

K_1, K_2 are experimentally determined coefficients

Meanwhile, other researchers were evaluating K_1 and K_2 as functions of the feedspeed, disk speed, geometry, disk composition, and disk dressing conditions for a variety of types of grinding [36-38,45]. This generally resulted in empirical relations for steady-state grinding obtained from curve fitting of experimental data. For example, Malkin [44] reported that for the grinding of steel K_1 was very nearly constant at 13.8 J/mm^3, and that K_2 was given by:

$$K_2 = \Pi_{pl} + \Pi_{sl} \tag{4.2a}$$

$$\Pi_{pl} = w\, a_1\, V_t \tag{4.2b}$$

$$\Pi_{sl} = w\left(a_2 + a_3 \frac{V_f}{2V_t R}\right) a_R\, V_t \tag{4.2c}$$

where: Π_{pl} = power due to plowing [kW]

Π_{sl} = power due to sliding [kW]

w = width of grinding patch, [mm]

$a_1 = 9.62\times10^{-7}$ kW/mm^2

V_t = tangential disk speed, = ωR, [mm/s]

ω = grinding disk rotation rate, [rad/s]

R = grinding disk radius, [mm]

$a_2 = 7.55\times10^{-6}$ kW·s/mm^4

$a_3 = 2.1$ kW·s/mm^3

a_R = real contact area per unit width, = $A\sqrt{2Rd}$

A = total disk surface area / total grit wear flat area

Such relations were derived for other variables as well. Disk wear was modelled by several researchers in this manner, as were surface finish conditions. Malkin used his results to model workpiece burn, a condition which is associated with undesirable metallurgical damage resulting from excess heating of the workpiece. The metal becomes discolored with tempering, and austenite is formed, followed by martensite formation as the region cools. The surface of the material shows increased hardness, due to the presence of the martensite. He computed the burning constraint by calculating the sum of local peak surface temperatures associated with individual grains and the temperature of the grinding zone in the vicinity of the contact patch. His burning model was:

$$\Phi = (9.0\times10^{5})d + (4.1\times10^{4})(Dd)^{1/4}\, V_f^{-1/2} \qquad (4.3)$$

where: Φ = energy flux [in.-lb/in^2]

d = cut depth [in]

D = disk diameter [in]

V_f = feedspeed [in/sec]

Snagging

Snagging is a general term for manual 'clean-up' grinding operations. It is used for cleaning castings, smoothing flame cut edges, preparing surfaces for welding, and smoothing weld beads [49]. Snagging is done using the face of rubber-backed paper grinding disks, and using either the edge or the face of rigid grinding disks. Grinding with the face of the disk is easier to control. This differs from the previously described processes which use the edge of the grinding disk solely. Some research on snagging with automation in mind has been recently reported [47–50]. These follow the lines of the previous research for other types of grinding, and obtain similar results.

In particular, Kenwood studied rigid resin-bond disk grinding, also referred to as 'hard' disk grinding. He noted the power vs. volumetric removal rate of equation (4.1). He also noted the phenomenon called glazing, in which the grinding grits get progressively duller, and the disk gradually becomes incapable of removing metal. He discovered that the normal process of grit pullout was not occurring due to insufficient local grinding pressure. The pullout process continually resharpens the wheel as new sharp grits are exposed. Individual grits were not experiencing enough force to extract them because the contact patch area was larger than that which would ordinarily occur in manual grinding. This was caused by the fact that the test apparatus held the grinder too steadily in one orientation, and did not generate the edges on the disk and workpiece that a human grinder would. Grit pullout was found to occur above a critical grinding power, and this coincided with an increase in the grinding coefficient of friction μ and the disk wear rate, and the grinding power. Grinding coefficients of friction varied between 0.25 and 0.4. K_1 was found to be about 10.–16. J/mm^3, a figure which agrees with Malkin.

Ivers substituted a rubber-backed coated abrasive paper disk, also called a 'flexible' disk, for the hard disk used by Kenwood. These disks have one layer of grits glued to a paper disk, and show an initial fast wear regime, followed by slower attritious wear regime, in which wear flats form on the grits and the grits gradually wear down to the paper surface. Ivers found a power vs. volumetric removal rate relationship that

was similar to Kenwood's. His coefficients were in the neighborhood of $K_1 = 8.4$ J/mm^3 and $K_2 = 30$ W for 36 grit disks grinding steel, and $K_1 = 15.$ J/mm^3 and $K_2 \approx 0$ W for 80 grit disks grinding steel. Grinding coefficients of friction were in the neighborhood of 0.8. Ivers also noted that it was possible to achieve very high quality surface finishes using coarse grinding disks. In experiments using 36 grit grinding disks, R_a values[1] less than one micron were obtained. Microscopic inspection of the ground surface revealed that the last grit that cuts largely determines the surface finish as if it were the only grit cutting. The high quality finishes were due to very uniform and slow feedspeeds. He developed a model that partially predicts the surface finish given the individual grit cut depth, the disk rotation rate ω, and the feedspeed V_f.

Both Kenwood and Ivers noted two important aspects of weld bead grinding: 1) restrictions upon the volumetric removal rate imply that several grinding passes must be made to completely remove the weld bead. and 2) the results of apparently identically set-up grinding passes can vary as much as 10-20%. These will be important factors in determining the structure of the medium term controller described in Chapter 5.

For hard disks, one restriction in the removal rate is in rapidly increasing disk wear rates for increasing volumetric removal rates. This leads to an economic tradeoff between the cost of disk replacement and the cost of extra grinding time for slower removal rates. Material removal rates for both hard and flexible disks are limited by the same metal burning modelled by Malkin (equation (4.3)). Ivers derived a similar expression for the maximum temperature, though using cruder assumptions. He found that it was proportional to the grinding power and inversely proportional to the square root of the feedspeed. This appeals to the intuition that the peak temperature rise would increase with friction power and increase with the time that the power source lingers near any one spot.

---

[1] R_a is defined as the average absolute value deviation from the mean:

$$R_a = \frac{1}{L}\int_0^L |f(x)-m|\, dx$$

where: f(x) = the surface profile, m = the mean of f(x) over [0,L], L = the sample length

4.2 THE STATIC GRINDING MODEL

This section will derive the static grinding model using the material removal and coefficient of friction relations. This model is used by the medium term planner to predict the average results of a grinding pass. For convenience, these relations are repeated here:

$$\Pi = K_1 Q + K_2 \tag{4.4a}$$

$$Q = w \delta V_f \tag{4.4b}$$

$$\Pi = \omega R F_t \tag{4.4c}$$

$$F_t = \mu F_n \tag{4.4d}$$

where: Π = power delivered to the contact patch [W]

Q = the volumetric material removal rate [mm^3/s]

K_1 = specific grinding energy [J/mm^3]

K_2 = wasted power [W]

w = weld bead width [mm]

δ = cut depth [mm]

V_f = the feedspeed [mm/s]

ω = the grinding disk rotation rate [rad/s]

R = the grinding disk radius [mm]

F_t = the component of the grinding force tangential to the disk [N]

F_n = the component of the grinding force normal to the disk [N]

μ = the grinding coefficient of friction [dimensionless]

The first equation expresses the empirical power vs. volumetric removal rate relationship. The second equation represents the conservation of volume for rectangular cutting area of dimensions $w \times \delta$ perpendicular to the feed velocity vector. The third equation computes the power delivered to the contact patch. The fourth equation is the empirical grinding coefficient of friction relation. These equations can be combined to yield a single equation giving the steady-state cut depth as a function of the process coefficients and normal force, feedspeed, and disk rotation rate:

$$\delta = \frac{\omega R \mu F_n - K_2}{K_1 w V_f} \tag{4.5}$$

To avoid unnecessary divisions which take more time than multiplications on a digital computer, the cut depth was computed as:

$$\delta = \frac{C_1 C_3 F_n + C_2}{V_f} \tag{4.6}$$

where: $C_1 = \frac{1}{K_1}$

$C_2 = -\frac{K_2}{wK_1}$

$C_3 = \frac{R\mu\omega}{w}$

This is the equation used by the medium term planner to predict the mean cut depth. This is used to predict the amount of material remaining in the weld bead after a grinding pass is completed via the simple expression:

$$X_f = X_i - \delta \tag{4.7}$$

where: X_f = the average initial weld bead height after a pass

X_i = the average initial weld bead height before a pass

Thus the amount of material remaining in the weld bead is equivalent to the average weld bead height for a weld bead with a rectangular cross section of fixed height and width. This average height is referred to as the **state** of the weld bead in the remainder of this thesis. X_i represents the initial state of the grinding pass, and X_f represents the final state of the grinding pass.

A slight modification is required when the grinder motor cannot maintain a constant rotation rate ω. In this case a steady-state motor model must be incorporated into the above equation:

$$\omega = \omega_0 - \beta R F_t \tag{4.8}$$

where: ω_0 = the free-spinning disk rotation rate [rad/s]

β = the motor constant, [rad/s·N·mm]

This expression indicates that the motor rotation rate is a linear function of the shaft torque, and is a good model for the air motor that currently powers the grinder in our laboratory. This expression can be substituted for ω in equation (4.6) using equation (4.4d) to express F_t in terms of F_n. A dynamic motor model is possible, but is not suitable for the planner at this level.

Stochastic Model

To evaluate the expected cost for the grinding pass, a probability density function for the terminal state of that grinding pass is needed. The static grinding model above is used to compute the mean value of the terminal state. This model is extended in the following to describe the probabilistic variation of the terminal state.

As stated above, both Kenwood and Ivers noted a significant variation in the cut depth for identical grinding conditions. The variation of the cut depth can be attributed to randomness in either K_1 or μ. These can be propagated through the dynamics using the relations in Table 2-1 to obtain:

attributing randomness to C_1:

$$\sigma_\delta = \frac{C_3 F_N}{V_f} \sigma_{C_1} \tag{4.9a}$$

or attributing randomness to C_2:

$$\sigma_\delta = \frac{1}{V_f} \sigma_{C_2} \tag{4.9b}$$

or attributing randomness to μ:

$$\sigma_\delta = \frac{C_1 \omega R F_n}{w V_f} \sigma_\mu \tag{4.9c}$$

where: σ_z = the standard deviation of z, where $z = \delta, C_1, \mu$, etc.

and the terminal state standard deviation is given by:

$$\sigma_{X_f} = \sigma_\delta \tag{4.10}$$

The above is strictly true only for probability density functions (PDFs) described completely by their mean and standard deviation and which propagate linearly through linear equations, such as the Normal PDF. However, a Normal PDF for C_1 or μ yields a Normal PDF for the cut depth, and this implies a finite probability for a negative cut depth (i.e. adding material by grinding), which does not make physical sense. We must restrict the PDF for the cut depth to satisfy the constraint that $\delta \geq 0$. Two PDFs come to mind: the log-normal and the Rayleigh. However, these do not propagate easily through the dynamics. An approximate solution is to propagate the mean and variances for Gaussian PDFs and use a Rayleigh or Log-Normal PDF having the resulting mean and variance for the cut depth distribution. See Figure 4-2 for a comparison of these PDFs and Table 4-1 for their formulae and expressions for their means and variances.

All three PDFs were implemented for the medium term planner (see Chapter 5). It was discovered that the planner was insensitive to the PDF shape as long as the mean and variance could be selected independently. As a compromise between the expected probability density function shape and ease of computation, the Rayleigh PDF was

selected to represent the stochastic state transition function. Judging from experimental grinding data, this PDF has about the correct shape.

Within the computer, the Rayleigh PDF can be easily scaled from a template shape to yield a PDF of any desired mean. This is useful from a computational standpoint. However the variance is dependent upon the mean, and this at first seems to violate the criterion that the mean and variance be independent. Within certain limits, the mean and variance can be made independent by first computing a PDF with the desired variance, and then translating the PDF toward higher cut depths to attain the desired mean. Of course, this does not work if the translation must be towards lower cut depths to attain the desired mean, since this would result in negative cut depths, similar to those which made the Normal PDF unsuitable. In this case, the PDF is scaled down to yield the desired mean, with the origin of the PDF coinciding with zero cut depth. This technique results in two different PDF shapes: one in which the mean and variance are independent, and one in which they are dependent. Fortunately, the cut depth variance is dependent upon the mean only for very small cut depths. Where the variance is small, the planner is primarily sensitive to the cut depth variance and relatively insensitive to the cut depth mean. As stated earlier, the planner showed only minor differences between the different PDF shapes when this translation technique was used with the Rayleigh PDF.

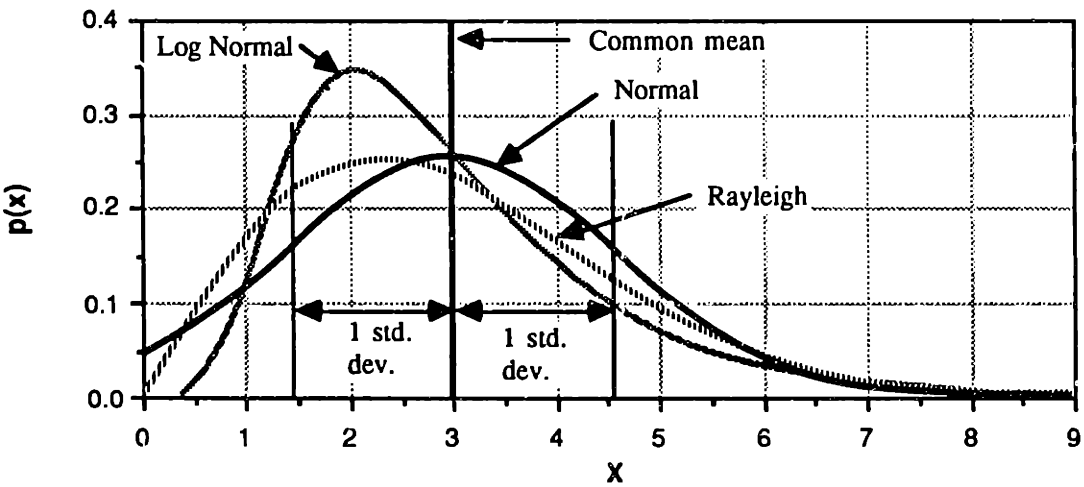


Figure 4-2 Candidate Probability Density Functions

Table 4-1 Formulae for Probability Density Functions

PDF	p(x)	mean	variance
Normal	$\frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-m)^2}{2\sigma^2}}$	m	σ^2
Log-Normal	$e^{\frac{-(\ln x - \mu)^2}{2\sigma^2}}$	$e^{\mu + \frac{1}{2}\sigma^2}$	$e^{2\mu + \sigma^2}\left(e^{\sigma^2}-1\right)$
Rayleigh	$\frac{x}{\sigma^2} e^{\frac{-x^2}{2\sigma^2}}$	$\frac{\sigma\sqrt{2\pi}}{2}$	$\sigma^2\left(2-\frac{\pi}{2}\right)$

4.3 CHAPTER SUMMARY

This chapter derived the grinding process model used by the sequence planner. It was based on cutting process and grinding models that are decades old. This model was static, one dimensional, and stochastic. It relates the mean and variance of the probability density function of the average weld bead height given the grinding force and feedspeed used during the grinding pass and the initial average weld bead height. This model will be used in the grinding sequence planner described in the next chapter.

5. PLANNING WELD BEAD GRINDING SEQUENCES

5.1 INTRODUCTION

This chapter describes the implementation of the grinding sequence planner. It describes the planner's task and shows that this is a sequential decision making problem. Then it describes the solution technique used, stochastic dynamic programming, by first describing deterministic dynamic programming, and showing how this can be used for planning deterministic grinding sequences, then adapting the dynamic programming algorithm for planning stochastic grinding sequences. The chapter continues with an analysis and characterization of the grinding policies produced by the sequence planner. The influence of three different constraints on the grinding process is explored: a fixed deadline, a weld bead burning constraint, and a lateness penalty.

The Grinding Sequence Planner's Task

The problem that this planner addresses is how to plan the number and type of grinding passes required to completely remove a weld bead of initial known height. This planner addresses several of the issues and requirements listed in Chapter 3 such as insuring that the weld bead be accurately ground off, that the time considerations be satisfied, and that the final surface finish requirements be met. It must plan within the constraints of the capabilities of the grinding equipment and process (e.g. grind within the power limits of the grinder and without burning the workpiece).

This planner uses the static grinding model described in the previous chapter. The variables under control during each grinding pass are the feedspeed and the grinding force. These are called the **controls**. Note that the terms **material remaining** and **weld bead height** are used interchangeably in this chapter because they represent the same thing at this level of the control hierarchy. This material is removed in stages during successive grinding passes, with the amount of material removed being a function of the controls applied during each pass. In addition, varying the feedspeed changes the completion time of a grinding pass for a fixed pass starting time. The set of feedspeeds selected for the entire sequence must satisfy the time considerations. Selecting the feedspeeds for every pass is tantamount to selecting the number of

grinding passes. The problem then is to select the feedspeeds and grinding forces for each pass in a sequence of passes so that the entire weld is ground off within the required time.

State Transitions

The dynamic programming technique is based on a concept known as a **state transition**. This concept was introduced briefly in Chapter 2, and will be described here in the weld bead grinding context. The term **state** refers to the condition of the weld at a particular time, i.e., both the volume of material remaining and the amount of time remaining. Consider a weld in a particular state. Call this state the **initial** state. This is illustrated in Figure 5-1 as state A. Application of a particular set of controls during a grinding pass converts the weld to a different state, the **terminal** state, when the grinding pass is done. This grinding pass is called a state transition, and is illustrated in Figure 5-1 as an arrow or directed arc connecting state A to state B, say, for one particular force and feedspeed applied during that grinding pass. If more grinding force is applied, more metal is removed, and the terminal state has less material remaining, such as for state C. If instead, the same grinding force is applied, but a slower feedspeed is used, the terminal state occurs later since the grinding pass takes longer, as in state D. Note that since slower grinding passes remove more material than faster passes, state D has less material remaining than state B, even though the same grinding force is used.

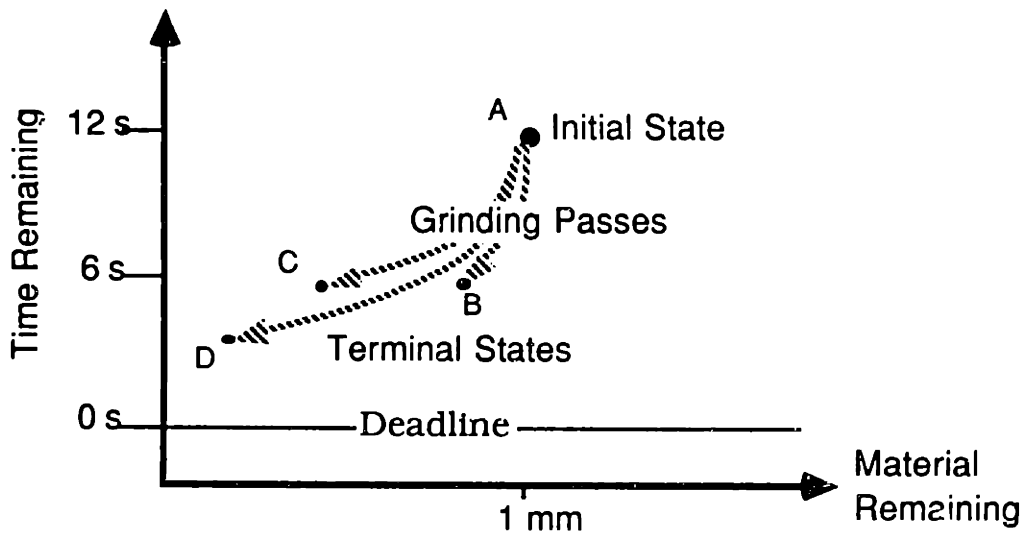


Figure 5-1 State Transitions

A series of passes comprises a **grinding sequence**. The state of the weld bead at the end of the sequence is called the **final state**. Some grinding sequences may take too long, or not remove the right amount of material. A technique for finding a good grinding sequence is to assign penalties for not satisfying the task specifications, and solve for the **best** sequence of grinding passes to satisfy the task requirements and constraints. Figure 5-2 is an illustration of two possible sequences. The requirements for the grinding system and physical limits to the grinding process constrain this selection. It is the nature of these requirements and constraints and how they affect the decision process which make the solution to this problem non-trivial and interesting.

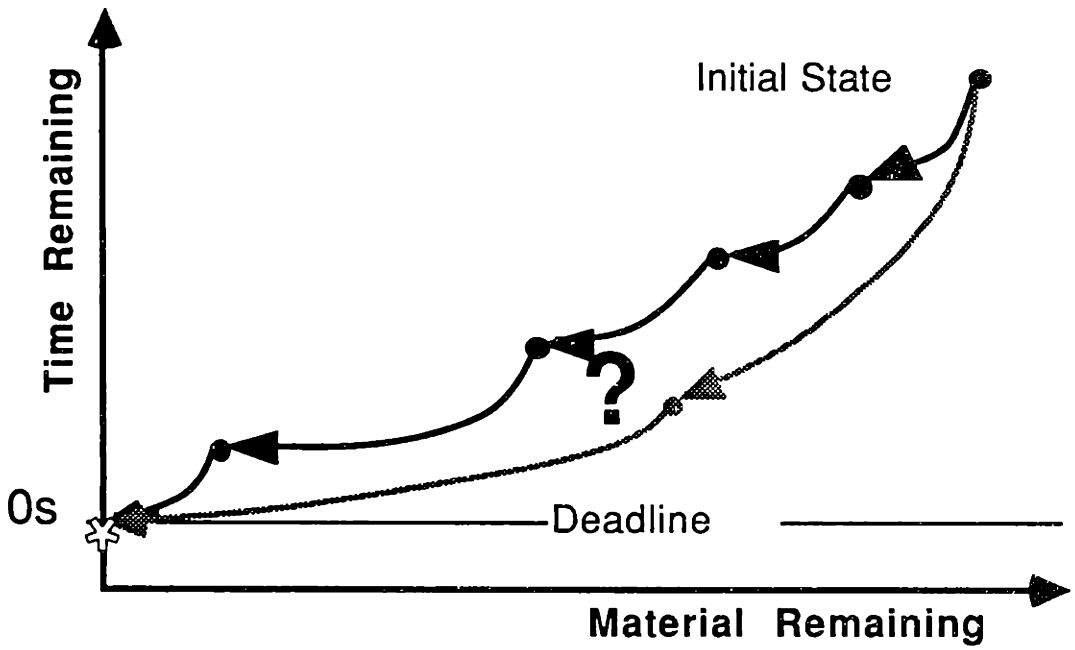


Figure 5-2 Two Grinding Sequences in State Space

Sequential Decision Making

This problem can be thought of as a sequential decision making problem, in which a series of decisions is to be made and the results of previous decisions affect future decisions. The decisions are the choices of state transitions to take given the initial state before each pass. This reduces to choosing the feed speed and normal force to apply during the next pass. Such a problem can be solved either numerically, using discrete approximations to the functions involved, or analytically. Both solution techniques have their advantages and disadvantages. The numerical technique suffers from the inherent problem of not being able to accurately represent a continuous situation with discrete values, and a lack of insight from the solution, but can handle a

wide range of mathematical functions and operations with relative ease. The analytical solution yields both insight and exact answers in many cases, but can fail when faced with a equation that is difficult to solve. This chapter will present both the numerical solution and an analysis of its behavior for further insight.

Due to the multiple passes and multiple decisions available at each pass, a very large number of state transition sequences is possible. An exhaustive search for the optimal sequence would be prohibitively expensive. However, the multiple-stage decision-making problem lends itself well to solution via a technique known as **dynamic programming**, which will be described in the next section.

5.2 Dynamic Programming

This section will show how dynamic programming[1] can be used to plan the grinding sequence. First, a simplified formulation of the grinding problem will be presented in order to demonstrate the basic dynamic programming algorithm. This simplified formulation will gradually be elaborated until all the details of the grinding sequence planner's problem are included.

A Simplified Grinding Problem Formulation

A simplified problem formulation for planning grinding sequences will now be presented to make the explanation of the sequence planner easier. For the simplified formulation, the problem is restricted to one of finding the optimal cut depth for each grinding pass when all the passes take a unit amount of time. Thus we need only select the optimal grinding force, since the feedspeed is a constant. This is equivalent to selecting the mean cut depth, which is linearly related to the normal force for fixed feedspeed. In this simplified problem there will be a maximum *number* of grinding passes, and the actual amount of time each pass takes will not be a concern. The end of the last grinding pass will be referred to as the **deadline**. The independent variable is the index number of the current grinding pass. In dynamic programming, the values of the independent variable are referred to as **stages**. Later the independent variable will be changed to the time remaining until the deadline (or until the planning horizon) and the grinding passes will be allowed to take different amounts of time. Later in this thesis grinding will be allowed *after* the deadline. For now it is simpler to think in terms of grinding passes governed by a single control, and state transitions occurring as discrete, non-overlapping events.

In many dynamic programming problems there are a finite number of states to consider, and a finite number of choices. This is not the case in grinding, for which there is a continuum of allowable states and controls. To make the problem tractable for the dynamic programming solution, the number of allowable states must be restricted to a countable set, located in regions of interest in the state space (typically, though not necessarily, evenly spaced). In this case, one might choose amounts of

---

[1]The reader is referred to Appendix A for a brief tutorial on dynamic programming.

material remaining in a cluster around zero material remaining and zero time remaining, the destination state, and disperse other states to adequately describe amounts of material remaining likely to be encountered during a series of grinding passes. The optimal controls will be selected to minimize the penalty function, and these must be allowed to take on values in the set of real numbers. For computer implementation of the dynamic programming algorithm, the values of the controls will be restricted to the set representable by double precision floating numbers.

Penalty Function

The penalty to be minimized is the sum of a **deadline charge** (according to the state of the weld when the grinding is done) and **in-path charges** accrued during grinding in a particular sequence from that initial state:

$$P(X_i \mid U_{\forall passes}) = C(X_f) + \sum_{\forall\ passes} G(X_{pass}, U_{pass}, pass) \qquad (5.1)$$

where:

$P(X_i)$	= penalty for grinding sequence initial state X_i
$C(X_f)$	= deadline charge for grinding sequence terminal state X_f
$\forall$ passes	refers to every pass in the grinding sequence between X_i and X_f
Xpass	= the state at the beginning of each pass
Upass	= the control applied during each pass
G(X,U,p)	= in-path charge for grinding pass p, with initial state X (before the pass), and control U applied during the pass

The function G represents charges accrued during each grinding pass not directly associated with the state of the weld bead when all grinding is finished. These are termed **in-path** charges. Such charges are imposed, for example, if it is desirable to weight system wear and tear via penalizing certain wear-intensive controls. In the case when grinding after the deadline is permitted, in-path charges will be used to penalize the extra time used after the deadline. In the discussion immediately following , the function G is zero identically; there will be no charges associated with either particular controls or particular intermediate states. Therefore, the charges are entirely a function of the final state of a grinding pass, and this final state is a function of the controls, U, via the state transition function. It is by selecting these controls that the algorithm tries to minimize the penalty function $P(X_i)$.

Deadline charges reflect the fact that undergrinding the weld (material remaining at the deadline), or overgrinding the weld (expressed as a negative amount of

material remaining at the deadline) are both undesirable. The three deadline charge functions used for planning weld bead grinding are illustrated in Figure 5-3. These were obtained using the techniques described in Chapter 2 of this thesis for estimating the costs of undesired results. The first charge function is a simple piecewise constant function, similar to the threshold penalty model described in Chapter 2. For this charge function, any overgrinding is heavily charged, while significant undergrinding is less severely charged. There is no charge in a small region just to the right of zero material remaining. This models a tolerance region, and gives the dynamic program a sufficiently broad low-penalty target state to shoot for. The second charge function is a piecewise linear function, with a steeper charge curve for overgrinding than for undergrinding. The third charge function is a piecewise parabolic function in the style of Taguchi's cost functions. In addition, combinations of these functions were implemented, using the negative material remaining part of one function with the positive material remaining part of another. The differences in results from using any of these functions or their combinations was unremarkable, differing only slightly in degree of controls applied and resulting penalties. This figure illustrates the generic shapes of these functions only; the actual values used were arbitrary, and were not derived from real cost data.

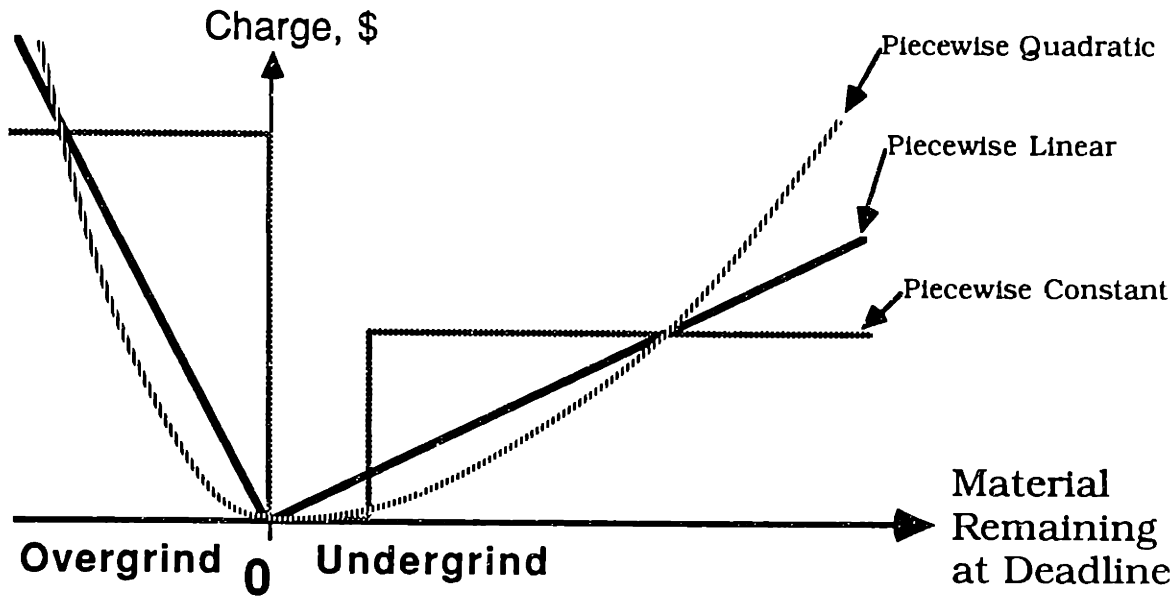


Figure 5-3 The Deadline Charge Functions for Planning Weld Bead Grinding

Since no in-path charges are accrued, the dynamic programming algorithm attempts to select the grinding controls to minimize just this particular deadline state

charge. Thus the dynamic programming algorithm tries to select the grinding sequence to avoid a final overground or underground state. Note that the deadline charge is the same as the deadline penalty; when at the deadline, no further grinding passes can be attempted, so the penalty is merely the deadline charge for that state. The algorithm is primed by loading the deadline charges into the penalty function for each amount of material remaining in the stage corresponding to the deadline.

The algorithm maintains an optimal penalty function J(X,pass stage index) that holds the least penalty $P(X|U_{\forall passes})$ that can be assessed for a grinding sequence starting at that initial state (grinding pass and material remaining). This function is often referred to as the **optimum penalty-to-go**, or simply the **penalty-to-go**. This is similar to the total time to the parking lot associated with each intersection in the skiing example. The basic dynamic programming algorithm can be written as:

For i = final pass to first pass

For all states $X(i) \in$ {legal states for pass i}

The optimal control $u^*(i)$ is the control which minimizes:

$$G(X(i), u(i), i) + J(X_f(i+1), i)$$

where: $$X_f(i+1) = F(X(i), u(i), i)$$

and
- $X(i)$ = the state of the weld at the beginning of pass i
- $u(i)$ = the control applied during pass i
- $X_f(i+1)$ = the state of the weld after the grinding pass i
- $F(X, u, i)$ = the state transition function which yields the state of the weld after pass i given the weld state X before the pass and the control u applied during the pass
- $J(X,i)$ = the optimal penalty for state X at the beginning of pass i

The resulting optimal penalty for each state X(i) is given by:

$$J(X(i)) = G(X(i), u^*(i), i) + J(X_f^*(i+1), i) \qquad (5.2)$$

where: $X_f^*(i+1) = F(X(i), u^*(i))$

Figure 5-4 shows the values of such a penalty function in the state space at a finite set of material-remaining values and time stages. This is the form in which the computer stores the penalty function. Generally the penalty function is treated as a sampled continuous function, and may be referred to elsewhere in this text (especially in figures) as a continuous function. The shape of the penalty function when time is held constant

will be referred to from time to time for explanatory purposes. This is a one-dimensional function: penalty as a function of material remaining. It will also be referred to as a **time-slice** of the penalty function.

Time Remaining [s]	Material Remaining [mm]				
	-0.5	0.0	0.5	1.0	1.5
12.	5.00	0.00	0.09	0.18	0.25
10.	5.00	0.00	0.12	0.20	0.31
8.	5.00	0.00	0.16	0.23	0.36
6.	5.00	0.00	0.21	0.44	0.62
4.	5.00	0.00	0.27	0.56	0.89
2.	5.00	0.00	0.36	0.77	1.30
0.	5.00	0.00	1.00	2.00	3.00

Figure 5-4 The Computer Representation of the Penalty Function

Dynamic Programming for Planning Grinding Passes

For the grinding problem, the dynamic programming algorithm proceeds backwards in time from the deadline to the present, determining (in reverse order) the optimal control sequences starting from progressively earlier states and ending at the deadline. The first step is to consider the final grinding pass, the state transition from the penultimate stage to the final stage. Consider a particular penultimate state (see Figure 5-5). Call the corresponding stage the **decision stage**, since the algorithm at this point only considers decisions whose effects are felt from this stage on.

The algorithm searches over the legal control space to find the control that yields the best state transition from the current state to the deadline. The legal control space consists of all controls that satisfy the constraints at this initial state. In this simple problem formulation, the only control is the cut depth, which must at least satisfy the constraint that the cut depth be positive. The quality of the state transition is measured only by the penalty associated with the final state of the transition. Some sort of numerical minimization scheme is used to find the control that yields the least final state penalty. That is, a candidate control is evaluated by computing the terminal state of the state transition, and then looking up the terminal state penalty from the optimal penalty-to-go function. This continues until the optimal control yielding the

minimum interpolated penalty is obtained. This is how the optimal penalty is computed for that particular state. The optimal control for that state is stored for future reference in a form similar to that of the penalty function.

This procedure is repeated for every amount of material remaining in the countable set of states at that particular stage. The optimal control to be used during the last grinding pass is obtained for each amount of material remaining at that stage. The penalty costs accrued for the optimal passes have also been recorded in the optimal penalty function. The best way to proceed in the last grinding pass from the any amount of material remaining and what penalties will be assessed for starting from those states is now known.

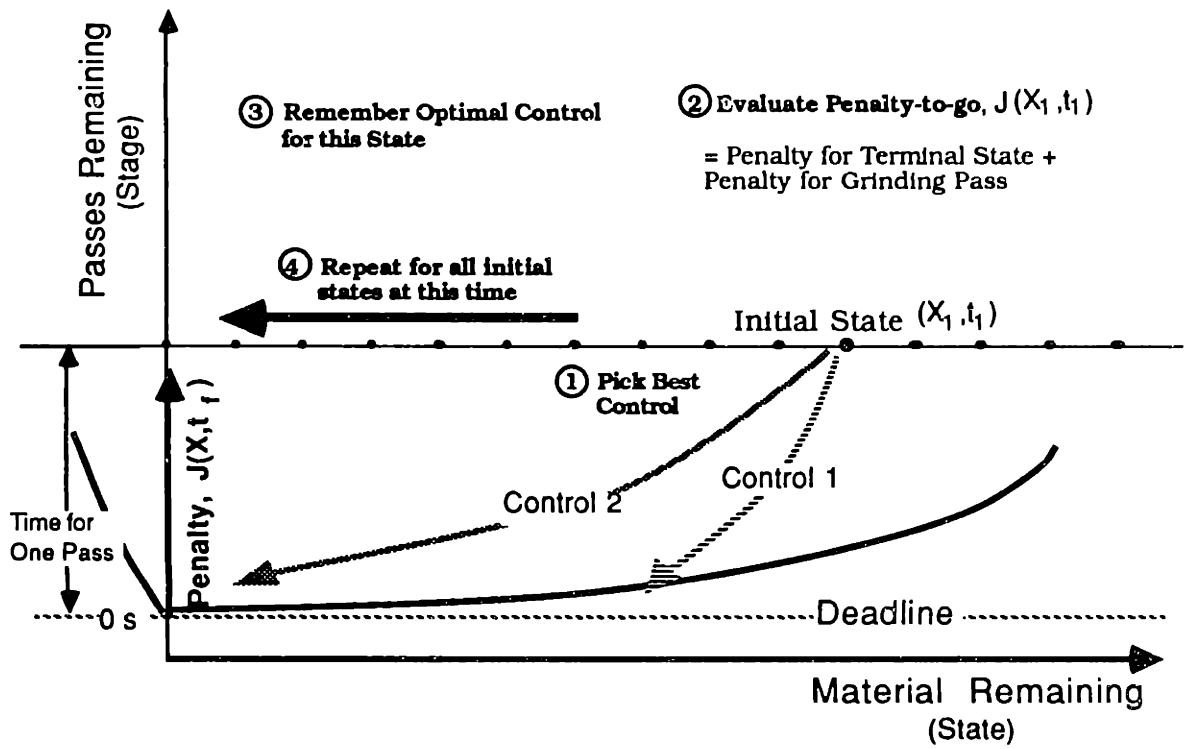


Figure 5-5 Dynamic Programming: The First Steps

Now the algorithm considers the next previous grinding pass (the penultimate pass, starting from the 2nd-to-last stage and terminating at the next-to-last stage). The decision stage is now the one prior to the previous decision stage. See Figure 5-6. All passes starting from here terminate at the penultimate stage, whose optimal penalty function was just computed. So the algorithm can do the same thing it did last time: for every possible amount of material remaining, the control is found which yields the least penalty for the terminal state of the grinding pass, which now happens to occur at

the beginning of the last pass. The penalty function for states at the beginning of the next pass is used as the terminal charge for this pass. This optimal control and the corresponding optimal penalty are recorded. This penalty is now the optimal penalty for proceeding from this state all the way to the deadline. This procedure is continued as far back in time as desired. Working backwards in time insures that the penalty function for a particular stage is evaluated before it is needed for earlier stages that are considered later in the algorithm.

In this formulation, the optimal penalty for state X is identical to the deadline charge of the final state in the optimal sequence passing through X. By selecting the proper controls, the algorithm merely tries to ensure that the optimal final state is reached. Thus in this simplified formulation, all the grinding sequences that can do so will terminate at the optimal final state at the deadline. It is both in the constraints on the grinding process dynamics and in randomness in the actual process dynamics that this problem becomes interesting. The latter limits knowledge of the actual intermediate and final states. This is discussed in more detail in Section 5-3. The former limits the ability of grinding sequences to reach the optimal final deadline state. This is discussed in more detail in Section 5-4.

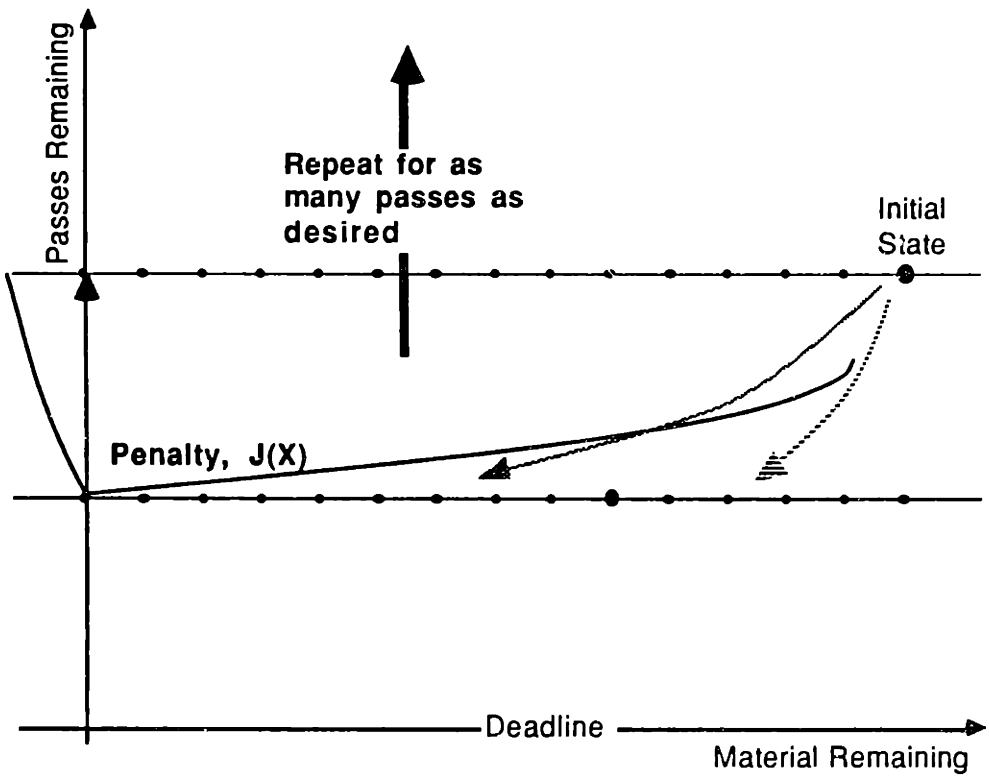


Figure 5-6 Dynamic Programming: The Second Step

The optimum control combinations are stored in a manner similar to the optimal penalties. The result is a table of advice as to what control combination is optimum at any possible state of the weld bead. This table can be referred to before a grinding pass is attempted by noting how much time and material are left, and looking up the corresponding optimal control (some sort of interpolation scheme may be used for times and amounts of material remaining that do not exactly correspond a member of the finite set used in the algorithm). The table becomes relevant and useful in the real world situation where the actual outcomes of grinding passes are different from those intended. As described for the skiing example, when unexpected outcomes occur, the optimal paths need not be recomputed. Rather, the solution from dynamic programming algorithm contains the optimal paths from *every* state to the deadline. No matter what the outcome of the previous pass, we simply refer to the table to see what to do next. Figure 5-7 illustrates this use of the dynamic programming solution for feedback control. The first pass illustrated took off more material than had been planned. The next pass is planned by looking up the optimal plans for passes starting at the first pass' actual termination state. The second pass here took off less material than planned. This process is continued until the last stage or time deadline is reached.

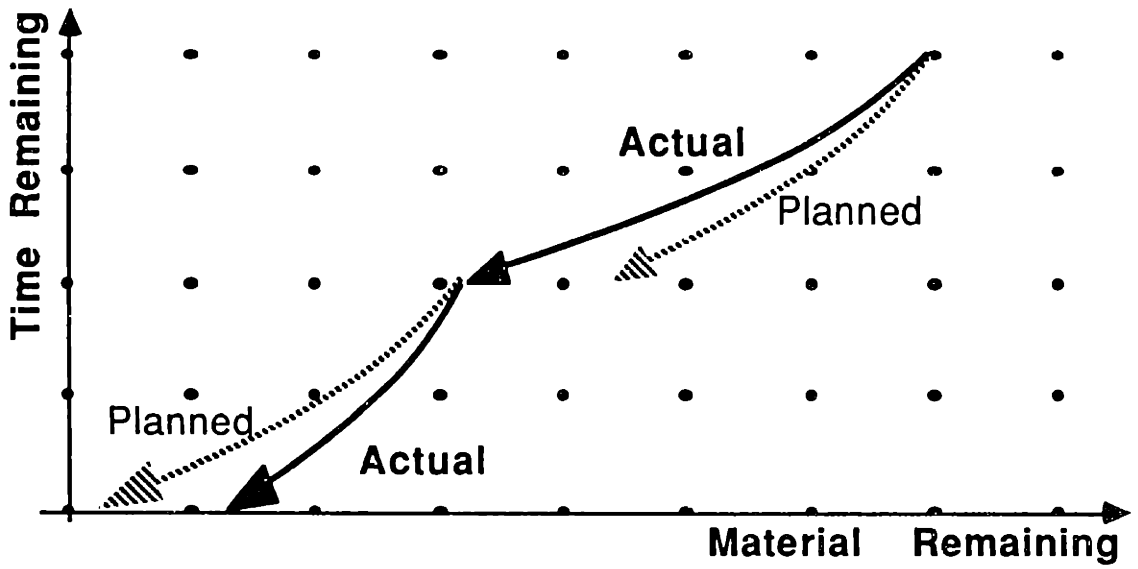


Figure 5-7 Feedback Control from the Dynamic Programming Solution

Variable Feedspeed

The above is a solution for the case in which the grinding pass duration is constant, implicitly requiring a constant feedspeed. This limitation can be removed if the stages represent intervals of time, rather than integral grinding passes. The

independent (stage) variable is now the **time remaining** or the **time-to-go**. See Figure 5-8 for an illustration of the new state space. Then a grinding pass can end anywhere in the set of time stages located after the decision stage. This allows for different passes to have different feedspeeds. Now the planning problem contains a new choice: from the given initial state, should the material be removed in one long grinding pass, or several short grinding passes ? The constraint that all grinding passes must be completed by the deadline must still be satisfied by the grinding plan.

The search for the optimal control is now performed over the two dimensional state space defined by the two controls: the grinding force, and the feedspeed. Different feedspeeds cause the grinding pass to terminate at different final stages. The appropriate penalty function is the one corresponding to the final stage of each grinding pass. A continuously variable feedspeed implies that the grinding passes can end between the discretized time stages. The penalty function can be interpolated at the appropriate pass termination time for the amount of material remaining. This requires a two-dimensional interpolation: one interpolation for the time remaining, and one for the material remaining. In Figure 5-8, the penalty function at t = 3s is used in evaluating the short, fast pass, at the amount of material remaining indicated by x_{sf}. However, the penalty function interpolated at t = 1.4s is used for evaluating the long, slow pass, at the amount of material remaining indicated by x_{lf}.

In the implementation of the sequence planner the feedspeeds were restricted to a finite set. This was done both to avoid having to interpolate in the time dimension within the optimal penalty function and to avoid having to search over a continuous range of feedspeeds to find the optimal feedspeed. The feedspeeds were instead selected to correspond to grinding passes that last an integral number of time stages. To compute the expected penalty, the cost function at the terminal time stage need only be interpolated in the material remaining dimension. The search for the optimal feedspeed is done more easily and quickly over a small set of possible feedspeeds. This is an approximation to the reality of continuous feedspeeds, but, as will be shown in the analysis of results later, it is a good model since the resulting grinding policies are discontinuous in the time domain. The results from this implementation compare well with those obtained when the feedspeed was allowed to take on continuous values.

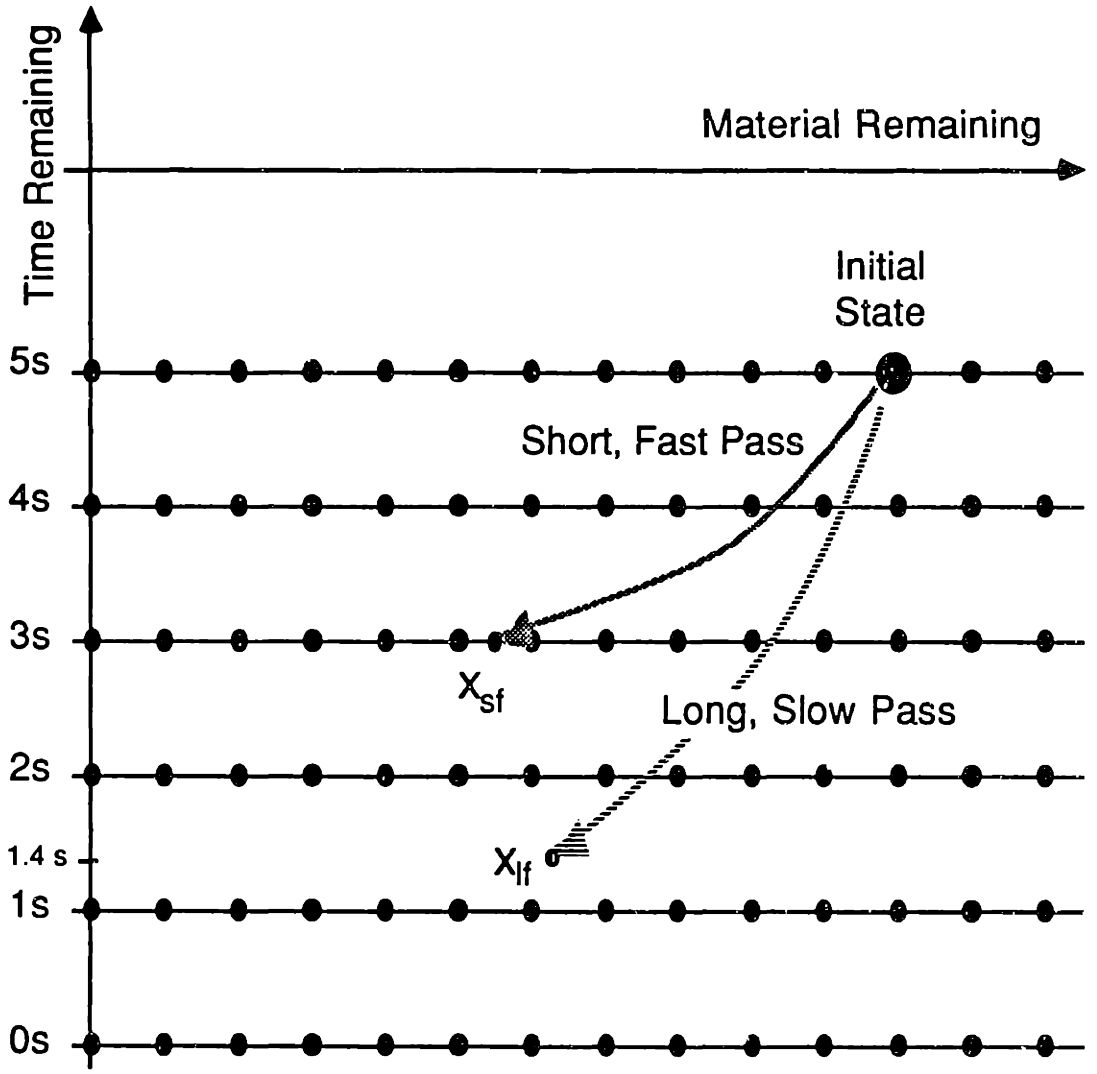


Figure 5-8 **State Space for Variable Feedspeed**

5.3 STOCHASTIC DYNAMICS

The state transitions for grinding are individual grinding passes, and are determined by the cut depth and the pass duration. The cut depth is a function of the controls, but is unpredictable. However, the variations can be described in probabilistic terms, so these state transitions are called **stochastic**. Stochastic grinding models were described in Chapter 4. See Figure 5-9 for an illustration of a stochastic state transition. The probability distribution of the cut depth is also a function of the controls. Now the planner must plan sequences that not only avoid overgrinding, undergrinding, burning, poor surface smoothness, and finishing late, but must avoid the *possibility* of these occurrences. This is done using the same dynamic programming solution scheme by modifying the penalty computation.

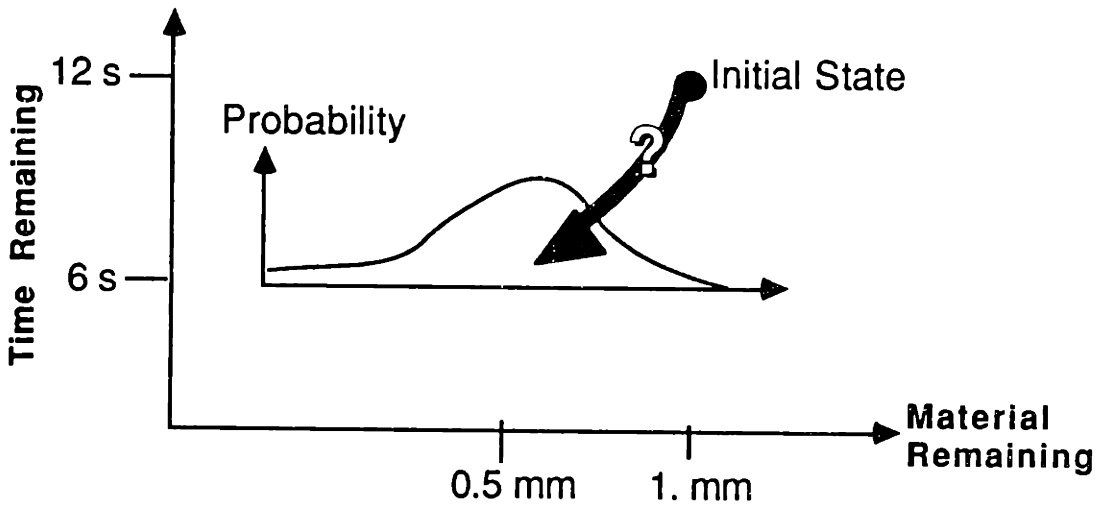


Figure 5-9 Stochastic State Transition

Stochastic Dynamic Programming

In order to obtain the penalty associated with a particular stochastic grinding pass, the result of that pass is evaluated as the expected result, via an integration of the product of the penalty J and the corresponding probability density function. This technique was introduced in Chapter 2 of this thesis. Here, the integrand is the product of the penalty as a function of the terminal state of the grinding pass, $J(X_f$, terminal stage), and the probability density function for the terminal states associated with the initial state and the controls used, $p(X_f \mid X_i$, controls), performed over all possible final weld bead heights for that grinding pass. This is depicted in Figure 5-10.

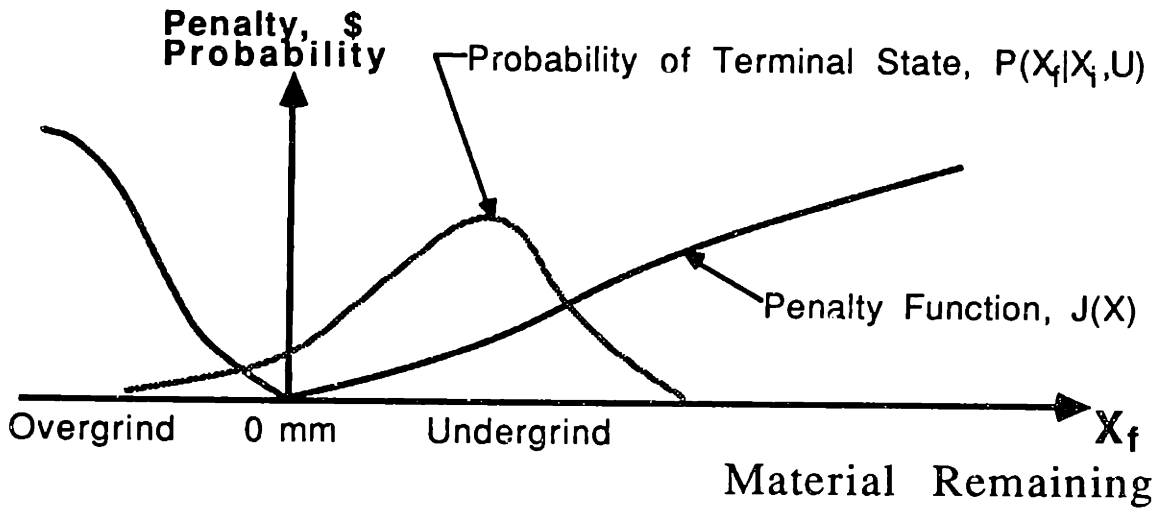


Figure 5-10 Computing the Expected Penalty of a Stochastic State Transition

The evaluation is:

$$E\left\{J(X(t_f),t_f)\right\} = \int_{\forall J} J\, p(J)\, dJ$$

$$= \int_{\forall X(t_f)} J(X(t_f),t_f)\; p\left(X(t_f) \mid X(t_i),\, F_N,\, V_f\right)\, dX(t_f) \qquad (5.3)$$

where:

$X(t_i)$	= the amount of material remaining before the pass
$X(t_f)$	= the amount of material remaining after the pass
t_i	= the time at the beginning of the pass
t_f	= the time at the end of the pass = $t_i + LV_f$
L	= the length of the weld bead
F_N	= the grinding normal force
V_f	= the feedspeed
$J(t,X)$	= the penalty function at time t for material remaining X
$p(X)$	= the probability density function of the pass terminal state X

The rest of the dynamic programming algorithm remains unchanged. The only difference is in the evaluation of the penalty for the candidate grinding pass.

Note that this integration yields the expected penalty for the given initial state and controls used. This is not, in general, the same as the penalty for the expected final

state. When the probability density function is skewed and/or the penalty function is nonlinear, the expected value of the penalty will not, in general, be the same as the penalty of the expected value:

$$E\left\{J(X(t_f),t_f)\right\} \neq J\left(E(X(t_f)),t_f\right) \tag{5.4}$$

In terms of our problem, this means that the expected penalty for a grinding pass given the initial state and controls will not, in general, be the same as the penalty for the expected terminal state. It may seem desirable to plan an expected terminal state with a very low penalty (i.e. very small amounts of material remaining), but this may be too risky. If the grinding variance is large, there is a high probability that overgrinding will occur, making such a plan too dangerous. This integration computes the riskiness of the plan as the expected penalty for that plan. Computing the penalty function via this integration allows this stochastic dynamic programming (SDP) algorithm to trade off high probabilities of low risk (an expected terminal state with a moderate penalty) with low probabilities of high risk (having a small possibility of overgrinding). The SDP algorithm is average risk averse, selecting controls which yield expected final states having moderately high penalties, but low average risk when all possible final states are considered.

The algorithm is listed below:

```
For time = deadline to beginning
{
    For all possible amounts of material remaining
    {
        For all control combinations
        {
            If control is within constraints
            {
                Get probability density function for
                    the material remaining at the end
                    of the grinding pass, P(Xf)
                Get expected penalty of the grinding pass
            }
        }
        Select the control combination with least penalty
```

Store the control in the optimal control function
Store its penalty in the penalty function
}
}

Implementation

The SDP algorithm is a dynamic programming algorithm with a stochastic state transition function. The state space consists of the time remaining until a fixed deadline (the independent variable) and the material remaining in the weld bead. Time remaining is discretized typically in 2 s stages; material remaining is discretized variously, typically in 0.1mm intervals from −1 mm (an overgrind condition) to 2 mm (some material still remaining on the weld, an undergrind condition). Penalties are assigned to the final state of the weld, according to the amount of material remaining when the deadline is reached (to penalize either a final overground or an underground condition). See Figure 5-3 for the shape of such a penalty function. Overground conditions are penalized more heavily than underground conditions because the cost of repairing the damage from an overground condition is assumed to be greater than the cost of taking the time to grind off the rest of the weld in an underground condition.

5.4 GRINDING POLICIES WITH A FIXED DEADLINE

The following discussion explains the results obtained from the stochastic dynamic programming (SDP) algorithm when no grinding is allowed after the deadline. The complementary case when grinding is allowed after the deadline will be considered in a later section. This discussion starts with some basic characteristics of the solution pertaining to the shape of the penalty function, the selection of the cut depth, and the choice of feedspeeds. The results will be described first with a few constraints on the controls, then the effects of other control constraints will be explored. Note that this is a qualitative analysis of the optimal grinding policies, and cannot yield the actual control values except where the optimal controls lie on the constraint boundary in control space. The quantitative policies obtained from the SDP algorithm will be compared with the qualitative policies described in the following.

Penalty Function Shape

The analysis of the sequence planner results begins with a description of the generic shape of the penalty function. A useful concept for analysis is the shape of the penalty function taken in slices at given time stages. The manner in which the shape of the penalty function changes as time progresses determines what state transitions are optimum from different states. The term **penalty function** will often refer to a slice of the penalty function at a particular time stage. This will also be referred to as a **time slice** of the penalty function.

Figure 5-11 illustrates a typical optimal cost function J(X,t). This figure displays three dimensions of information. Two axes give the status of the weld bead—the amount of time and material remaining—when the decision for the next grinding pass is made. Time progresses from lower right to upper left (time remaining increases in the opposite sense). The vertical axis indicates the optimal penalty function for that state. The target state of zero material remaining at the deadline is located at the intersection of the three axes. The shaded areas are time slices of the optimal penalty

function at selected time stages. It can be shown that if the deadline state penalty function is convex[1], then every time slice of the penalty function will be convex. [1]

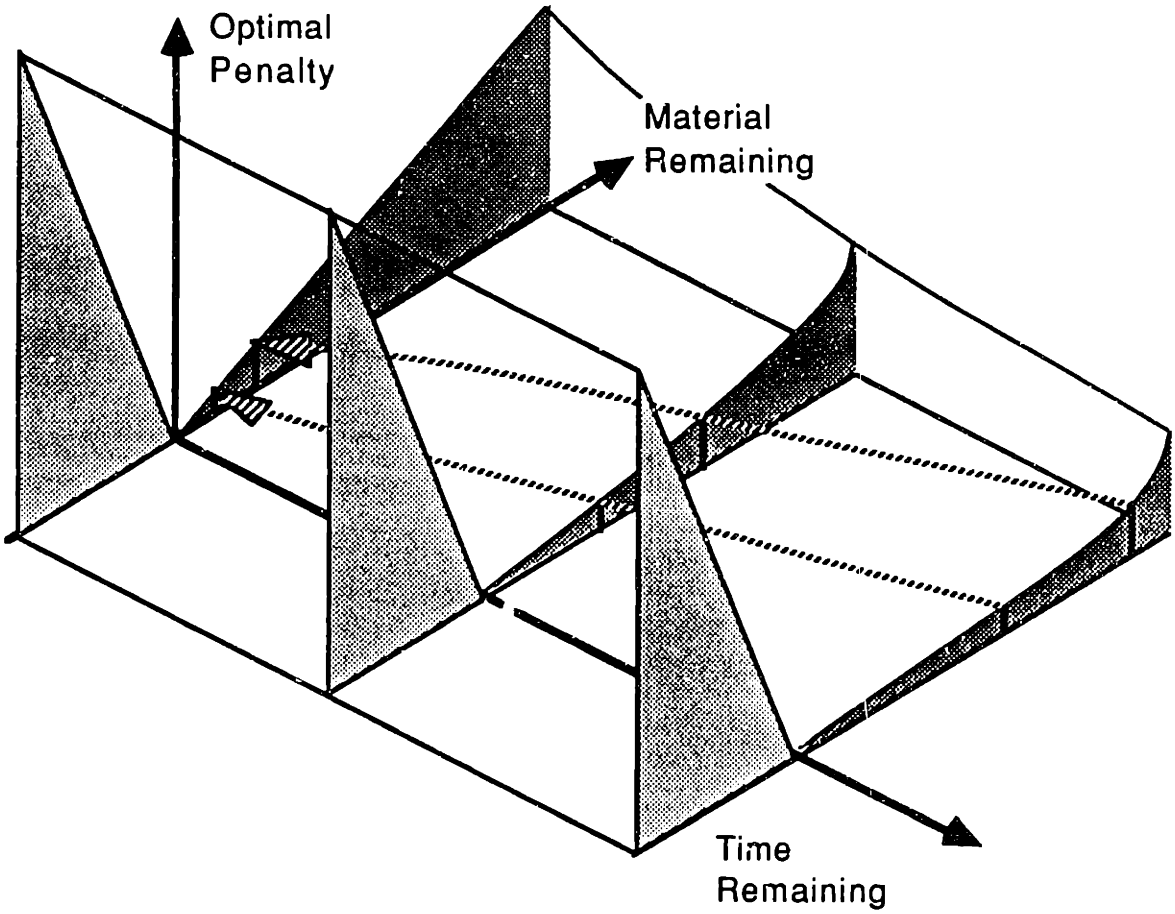


Figure 5-11 The Optimal Penalty Function

The states of a grinding sequence comprise a **state trajectory**. Two optimal trajectories are shown in Figure 5-11 as dashed lines. When there are no in-path charges, states along an optimal trajectory will have the same penalty. Think of the penalty associated with a certain state as the expected score for the final fate of the grinding sequence that passes through that state. The goal of the planner is to optimize this final fate. States along the optimal trajectory will have the same expected optimal fate. Thus, optimal trajectories have constant heights along the optimal penalty function's surface.

---

[1]Convex means that all points on a line segment joining any two points on the cost function curve are above the curve.

For increasing amounts of time remaining (i.e. nearer the beginning), the penalty function flattens out, indicating that, given more time, the system can usually improve the state of the weld. A little grinding will almost always help the situation by yielding a lower amount of material remaining at the deadline which yields a lower deadline charge. Another way of thinking about this is that a weld bead state with a lot of material and time remaining is just as good as one with little material remaining at the deadline, since an optimal grinding sequence can use the remaining time to reduce the amount of material remaining. However, grinding won't help the situation if it must be done with a high probability of overgrinding the weld bead. The penalty at 0.0 mm remaining will always be the minimum for any given time stage (since the algorithm will choose to do no grinding and just wait for the deadline) and if there are no in-path charges, this penalty will be the same for all time. Note that grinding will never help an overground condition, so it is best not to grind at all when the weld bead is already overground. Therefore the time slices of the optimal penalty function do not change shape with time in the overground region of the state space. These are shown as identical triangles in the above figure.

Optimal Cut Depth

Now consider the situation in which the length of the grinding pass is fixed, as in the simple problem formulation of the previous section. The fixed pass length determines the slice of the penalty function that will be used for computing the expected pass penalty. For any given initial state there is a single optimal cut depth that results in the minimum expected penalty. Varying the cut depth changes the position and shape of the PDF for the material remaining at the end of the grinding pass. It can be shown that there is one cut depth that minimizes the integral for the expected penalty. This cut depth locates the mean of the PDF near the minimum of the penalty function. Figure 5-12 is an illustration of the grinding planner's state space. It shows four dimensions of data: the amount of material remaining on the horizontal axis, the amount of time remaining on the vertical axis. Superimposed on the vertical axis are also the optimal penalty function slice for the time stage at the end of the grinding pass and the PDF for the amount of material remaining after the pass. The planner choses controls to move and reshape the PDF to optimize the expected terminal penalty. The actual location of the mean of the PDF is determined by this optimization. For the particular deadline charge functions used in the SDP programs, the mean of the PDF for the material remaining at the end of the grinding pass will lie somewhere in the

underground region. This is typically around 50% of the weld bead height before the start of the pass, depending on the accuracy of the process and the shape of the penalty function; the algorithm prefers to err on the cheaper side of the penalty function, and this occurs on the underground region for all the deadline charge functions used. Note that the fact that the algorithm suggests removing a constant fraction of the material remaining is to be expected since the state transition function (the cut depth) is nearly independent of dimension and the penalty function is very nearly self similar.[2]

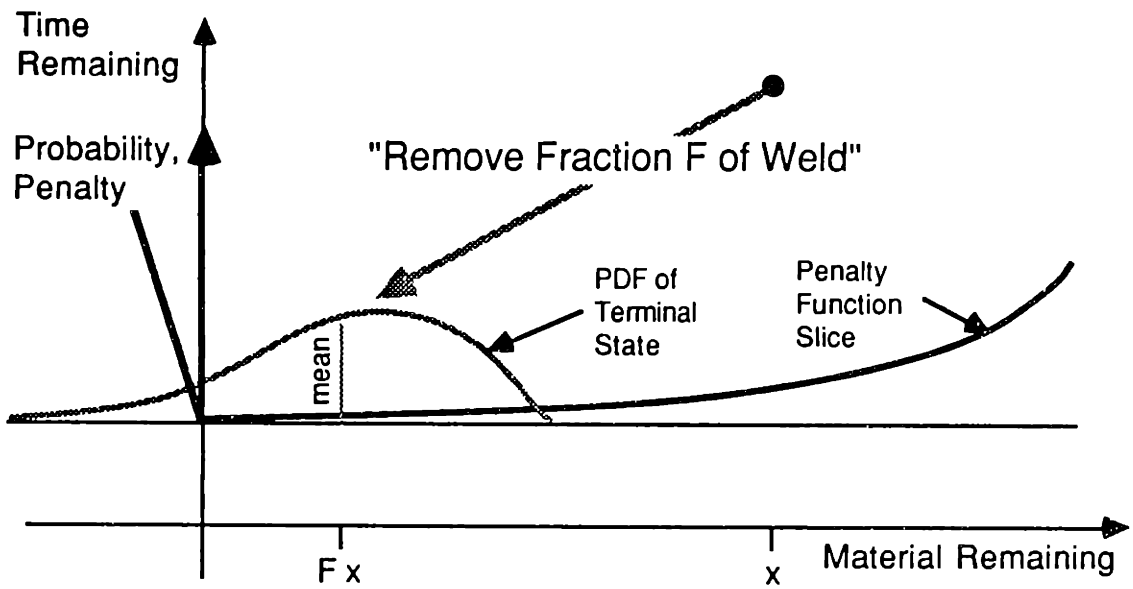


Figure 5-12 **Cut Depth Fraction is Independent of the Amount of Material Remaining**

In general, the algorithm first tries to select controls that yield the optimal cut depth. When it is possible to do this with several different controls, the algorithm selects controls based on second-order effects, such as the shape of the PDF. In particular, if the time-slice of the penalty function is convex, a narrower PDF will yield a lower expected penalty than a wider PDF having the same mean. In addition, if the terminal state is near the zero material remaining axis, a narrower PDF will enable the mean terminal state to get nearer the zero material remaining axis without having the

---

[2]Independence of dimension in this context means that the final state PDF has the same shape irrespective of size. Both the mean cut depth and its standard deviation are scaled by nearly the same factor of the controls. Self-similarity of the penalty function means that it looks about the same regardless of the range it is viewed over.

tails of the PDF penetrate too far into the overground region. That is, more accurate cut depths make it easier to remove more of the weld bead. Therefore, if the desired mean cut depth can be attained by several sets of controls, the algorithm will select the set of controls which yields the narrowest PDF. If the time-slice of the penalty function is linear, the expected penalty will be the penalty of the mean, and there will be no difference in penalties from two PDFs having the same mean but different variances. The algorithm will be indifferent to the choice of controls that yielded the two PDFs. There is another type of second-order effect having to do with the fine structure of the optimal penalty function, but an explanation of this must wait until some of the SDP results are explained. However, it should be reiterated that such second-order effects come into play only after the optimal mean cut depth has been achieved.

Reachable Terminal States

It is instructive to see how constant lines of grinding force and feedspeed appear in the material- and time-remaining state space, for these delimit the region in the state space where the expected terminal state can lie. The steady-state cut depth model derived in the previous chapter was:

$$\delta = \min\left(0, \frac{C_1 C_3 F_N + C_2}{V_f}\right) \tag{5.5}$$

where: δ = the mean or expected value of the cut depth, = $E\{\delta\}$

This means that the constant cut depth curves in the control space are linear, and all pass through the point $[V_f, F_N] = \left[0, -\frac{C_2}{C_1 C_3}\right]$. See Figure 5-13 for a diagram of this control space. Note that the slope of the lines is $\frac{\delta}{C_1 C_3}$, i.e., increasing cut depths correspond to increasing slopes. The higher cut depths are associated with higher normal forces and lower feedspeeds.

The feedspeed is determined by the difference in time between the start and the end of the pass, therefore specifying the time stage at the end of the pass is equivalent to specifying the feedspeed. The termination time for a grinding pass is given by:

$$t_f = t_i + \frac{L}{V_f} + t_R \tag{5.6}$$

where: t_f = the time at the end of the grinding pass

t_i = the time at the start of the grinding pass

L = the length of the weld bead

t_R = the time required for system reset (e.g. to reposition the robot arm back at the beginning of the weld bead, or to perform calculations

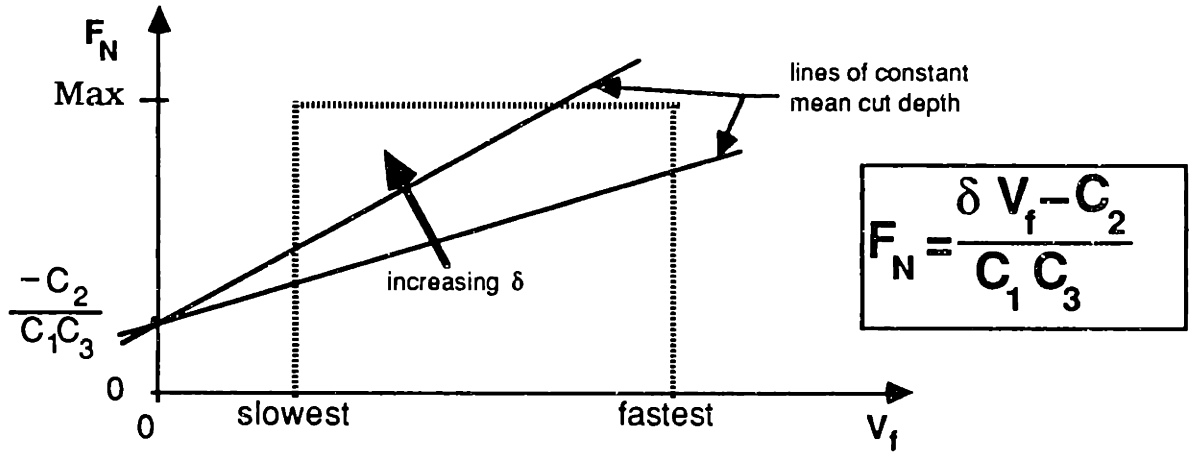


Figure 5-13 Control Space

Therefore for a given initial state, lines of constant feedspeed are equivalent to lines of constant termination time. Solving equation (5.5) for V_f and substituting into equation (5.6) yields an equation for lines of constant grinding force:

$$t_f = \frac{L\delta}{C_1 C_3 F_N + C_2} + t_R \qquad (5.7)$$

These are illustrated in Figure 5-14. The origin of the δ vs. t_f state space is the initial state. The shaded area depicts the region of possible expected terminal states. That is, this is the region of states that could represent the time remaining and material remaining after one grinding pass. The boundaries of this region are due to the maximum and minimum feedspeeds, the maximum normal force, and the fact that grinding cannot add material to the weld bead. This region has a different shape when the burning constraint is applied. See the section on the burning constraint later in this chapter for more details.

The region of states attainable after any grinding pass within a sequence of grinding passes is similar to that illustrated below, with the exception that there is no constraint corresponding to the minimum feedspeed constraint indicated below. This is illustrated in Figure 5-15 below, where the region of expected terminal states for any pass in a grinding sequence is superimposed upon the state space. Note that the extension of the maximum grinding force boundary is stepped, due to the reset time between passes, and that the small, diagonally-filled triangular regions are reachable if

the algorithm is allowed to simply wait. (This option was not given to the SDP algorithm.)

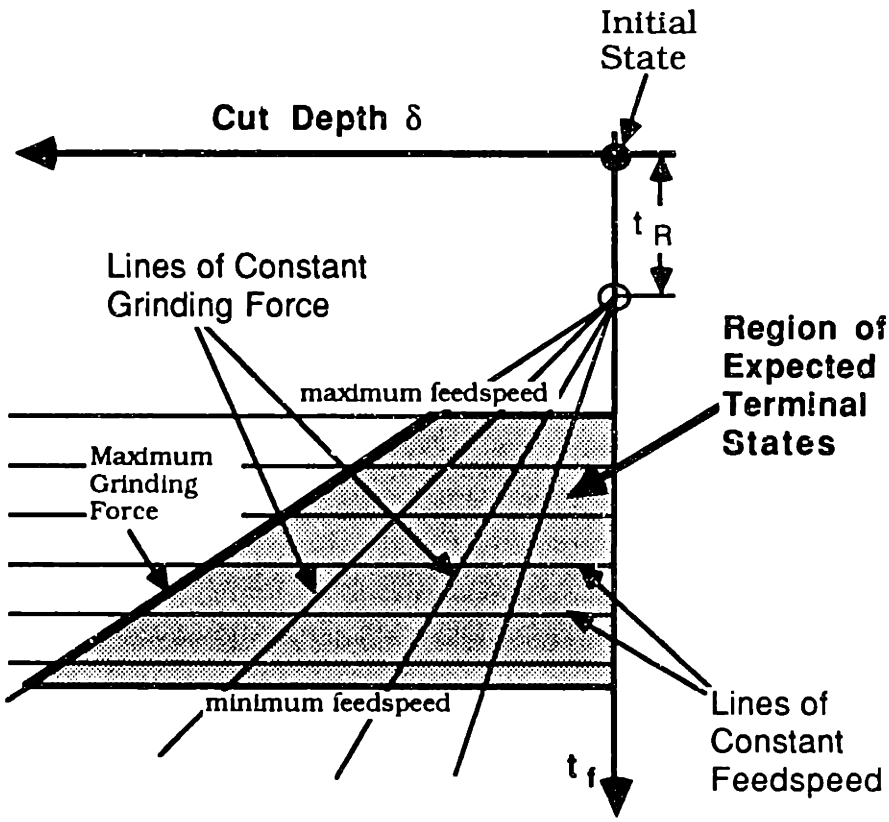


Figure 5-14 Expected Terminal State Region

The region of attainable states can be used to roughly characterize regions in the grinding state space. These are illustrated in Figure 5-16, and described separately below.

Note that the boundary between the regions containing initial states that can't reach the target state and can reach the target state is actually stepped, corresponding to the steps in the maximum normal force boundary below. This is a minor detail, and the boundary between these two regions will be shown as a diagonal line from here on, for simplicity.

Already Overground

When the weld bead is already overground, the planner decides not to grind any more because there is no grinding sequence that can help the situation.

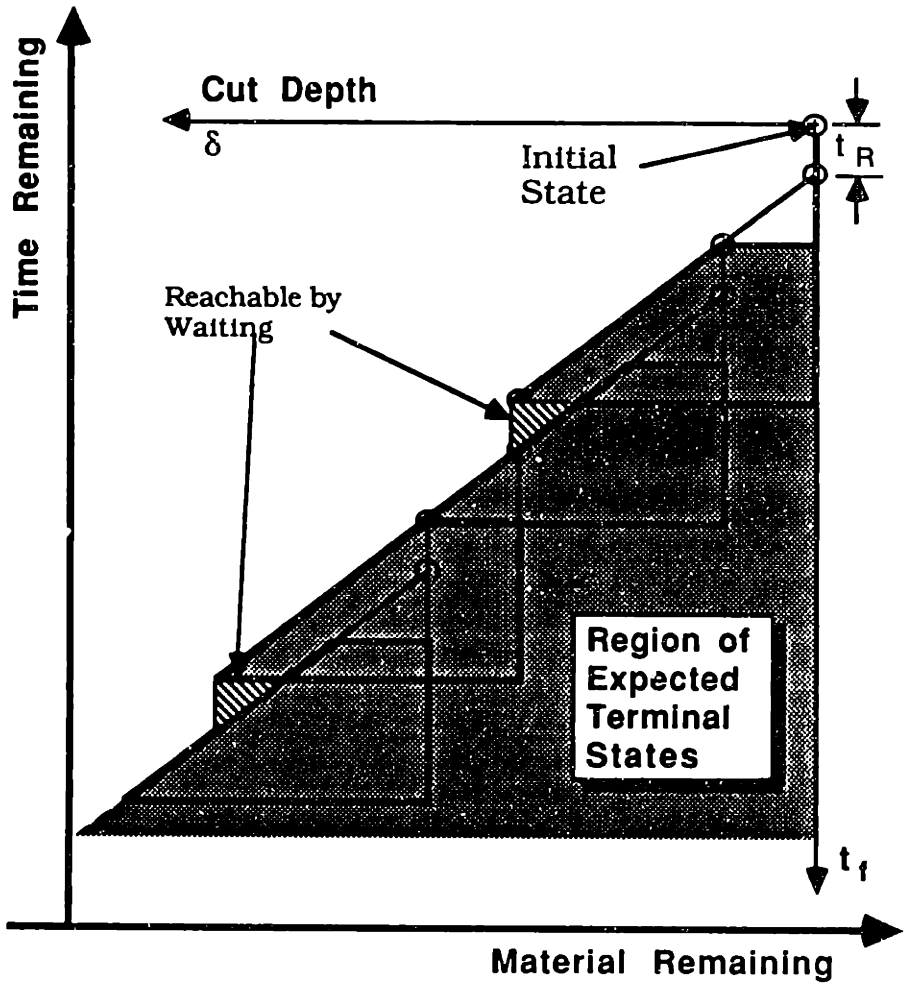


Figure 5-15 Region of Possible Terminal States in a Grinding Sequence

Not Enough Time

Near the deadline there is a region of states for which there is not enough time to complete a single grinding pass. The sequence planner is instructed not to plan to start grinding passes from these states either. Rather, the grinding policy here is to simply wait until the deadline, and be penalized the charge for the amount of material remaining in the initial state.

Cannot Reach Target State

When there is too much material left and not enough time, the system cannot reach the desired target state of no material remaining at the deadline even if it grinds with the maximum grinding force for every pass in the grinding sequence. The planner will choose the grinding sequence that can get the mean mean amount of material remaining of the final state as close to zero as possible. This will generally be a sequence of maximum cut depth grinding passes timed to finish at the deadline. These

states will have a larger penalty-to-go than would have been charged had any grinding sequence been able to reach the target state.

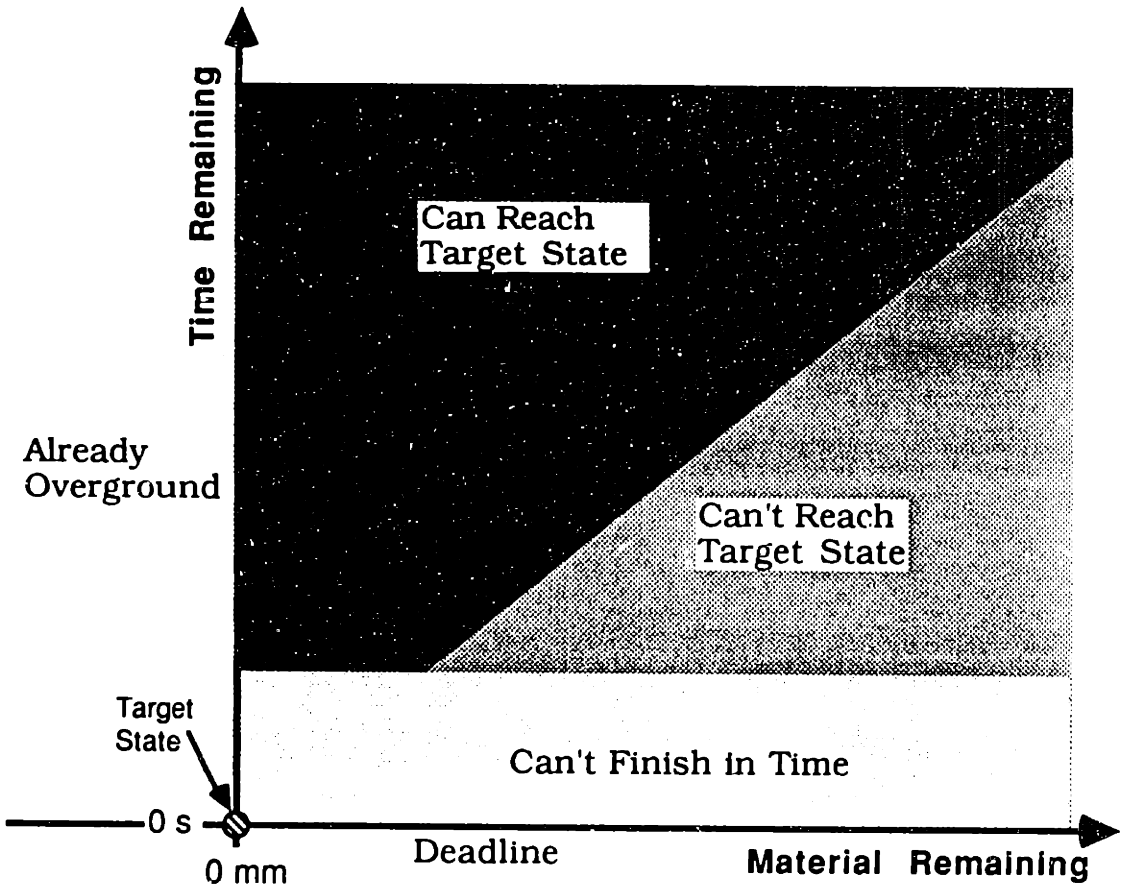


Figure 5-16 Target State Reachability

Can Reach Target State

The remaining region in the figure represents states for which the target state is reachable. These states will have a near-zero expected penalty because there is a good chance that the target state will be reached in grinding sequences starting at these states. For these states the algorithm has some choice as to which set of controls to use, so the second-order effects come into play here. The most common second-order effect is the selection of the feedspeed to yield the narrowest terminal state PDF, and is described in detail in the following section. The optimal grinding policies in each of these regions will be described in the sections following the next section.

Second-Order Effects: Optimal Feedspeed

In Chapter 4 when the variation of the cut depth was attributed to a variation in the coefficient of friction, the grinding process model yielded a standard deviation for the cut depth PDF equal to:

$$\sigma_\delta = \frac{C_3 F_n}{V_f} \sigma_{C_1} \tag{5.8}$$

Thus lines of constant σ_δ radiate away from the origin in the control space, with steeper lines representing greater standard deviations. This is illustrated in Figure 5-17. It can be seen that, for a given cut depth, the cut depth standard deviation decreases with increasing feedspeed. So faster passes are more accurate. The same is true for variations attributed to C_2 and μ; all indicate that faster passes are more accurate.

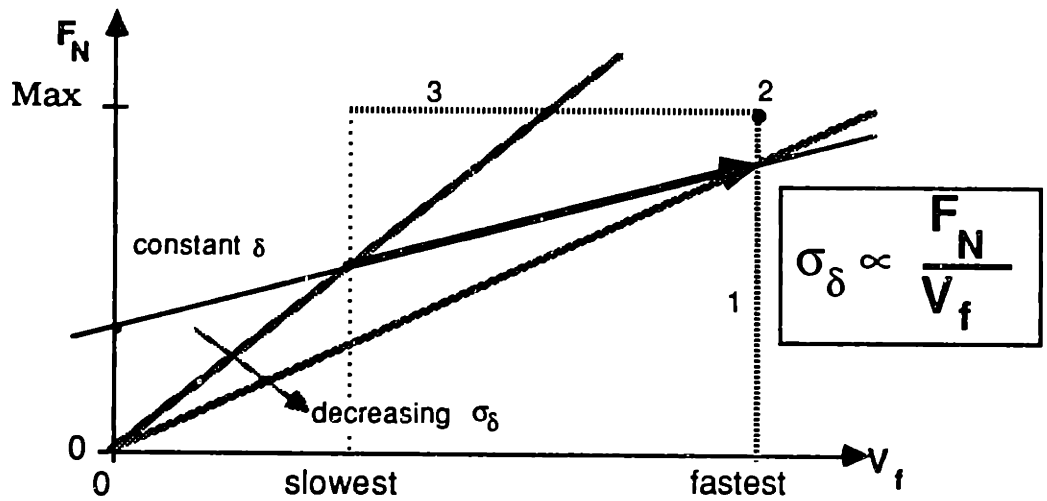


Figure 5-17 Optimum Standard Deviation in Control Space

There is another more subtle effect operating here. The penalty for the faster pass is computed from an earlier time-slice of the optimal penalty function than the slower pass. As has been described, the optimal penalty function generally decreases with increasing amounts of time remaining because when there is more time to grind, the final state of the grinding sequence will be nearer to the target state So even if the faster pass had the same PDF shape, it would have a lower penalty. Both second order effects make faster passes more attractive.

Note that Ivers' surface finish model indicates that the slowest feedspeed yields the highest quality surface finishes, so a low feedspeed may be a better choice for the last pass. It is easy to implement such a consideration as an in-path charge for the final

pass in the SDP algorithm. Passes whose terminal states are those for which no more grinding is recommended would be charged inversely to their feedspeeds, to account for the expected final surface finish. Depending on the structure of the associated charge function, this would tend to bias the passes near the deadline to longer feedspeeds, and effectively widen the region in which no pass is started because a final quick pass would be charged more than ordinarily. However, this was not implemented in the SDP program due to time constraints.

Consider the basic tradeoff between many fast passes and fewer slower passes. This is illustrated in Figure 5-15. Which is better? Consider the example of selecting the optimal controls at a certain initial state. Should the system take one long pass, or two shorter passes, or several even shorter passes? If the normal force that results in the optimal cut depth can be selected without constraint, the algorithm always suggests using the shortest possible grinding pass.

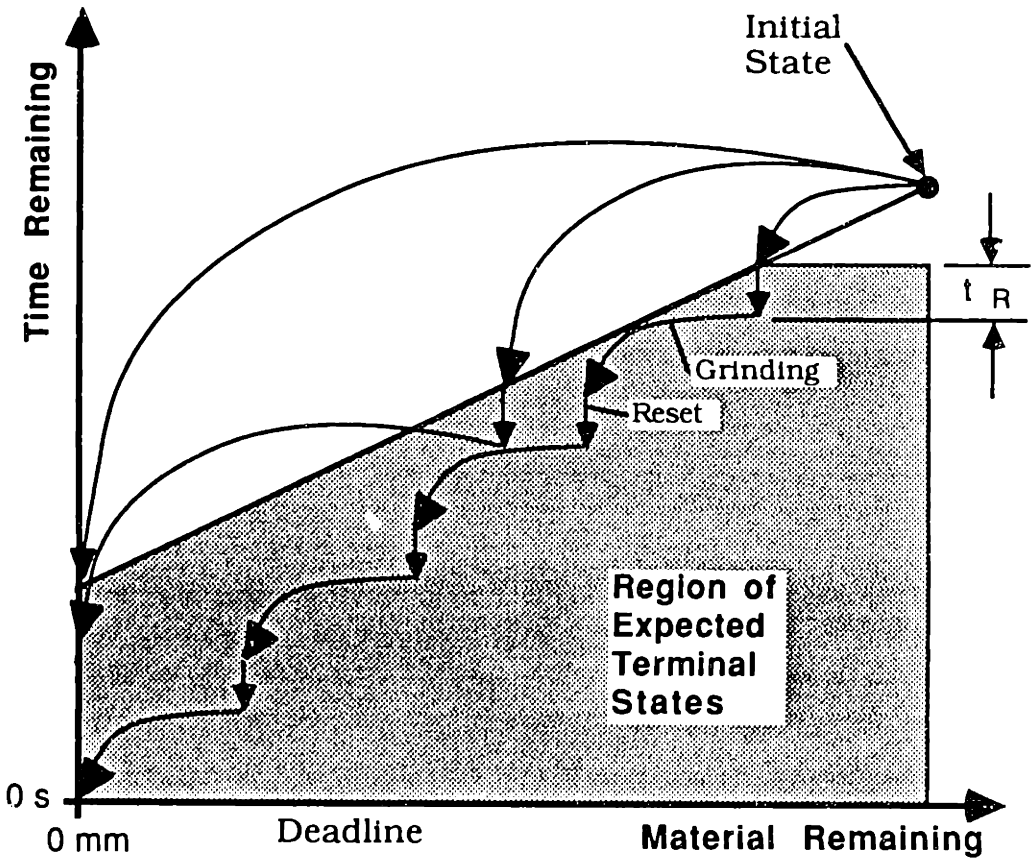


Figure 5-18 Choice of Feedspeed for Stochastic Dynamics

However, if time is required to reset the equipment or perform computations between grinding passes, then a grinding sequence with more frequent grinding passes

will spend proportionally more time resetting than one with fewer passes. But when the amount of time used is not directly penalized via in-path charges, the algorithm sees the deadline only as a constraint, and attempts to plan a grinding sequence that finishes within that deadline. In other words, it will plan the largest number of passes that will fit into the available time, because these will be faster and therefore more accurate. In the figure, all the grinding passes illustrated are operating at the maximum possible force. For simplicity both multiple-pass sequences consist of identical grinding passes. (This is not generally the case, but, for the moment, grinding sequences will be described with identical passes.) The grinding sequences that consist of one and two grinding passes complete the grinding early. The five-pass sequence finishes right on time. Grinding sequences with more passes would not finish on time because too much time would be used in resetting the equipment. When there are no in-path time charges, there is no advantage to finishing early, so the planner opts for the most accurate grinding sequence that finishes by the deadline, and in this case, it is the sequence with the fastest passes.

Grinding Policies in the Region that Can Reach the Target State

The above second-order effects can be used to derive a qualitative description of the optimal grinding policies in the region of the state space containing initial states that can reach the target state. Consider first initial states whose amounts of material remaining are quite close to zero. These states are illustrated in Figure 5-19 as two sub-regions filled with vertical and diagonal lines. Superimposed on the state space in each sub-region is a typical initial state with its region of possible expected terminal states. A grinding pass starting from any initial state in these two sub-regions can reach zero amount of material remaining with a variety of controls; the region of possible terminal states straddles the zero material remaining axis. For these states the second-order effects described above are important. The optimal set of controls is the one that yields the terminal state near the intersection of the maximum feedspeed control boundary and the zero material remaining axis. This is indicated by a black dot in the figure. Note that the possibility of overgrinding requires that the optimal terminal state actually have a small amount of material remaining. The corresponding optimal cut depth throughout this sub-region is a fraction of the initial material remaining (see Figure 5-12 and the discussion immediately preceding it). This fraction is nearly constant throughout these two sub-regions, and approaches unity, but decreases with decreasing process accuracy. The same grinding policy is optimal throughout each sub-

region. Other second order effects slightly affect the policies in each sub-region. These will be discussed later.

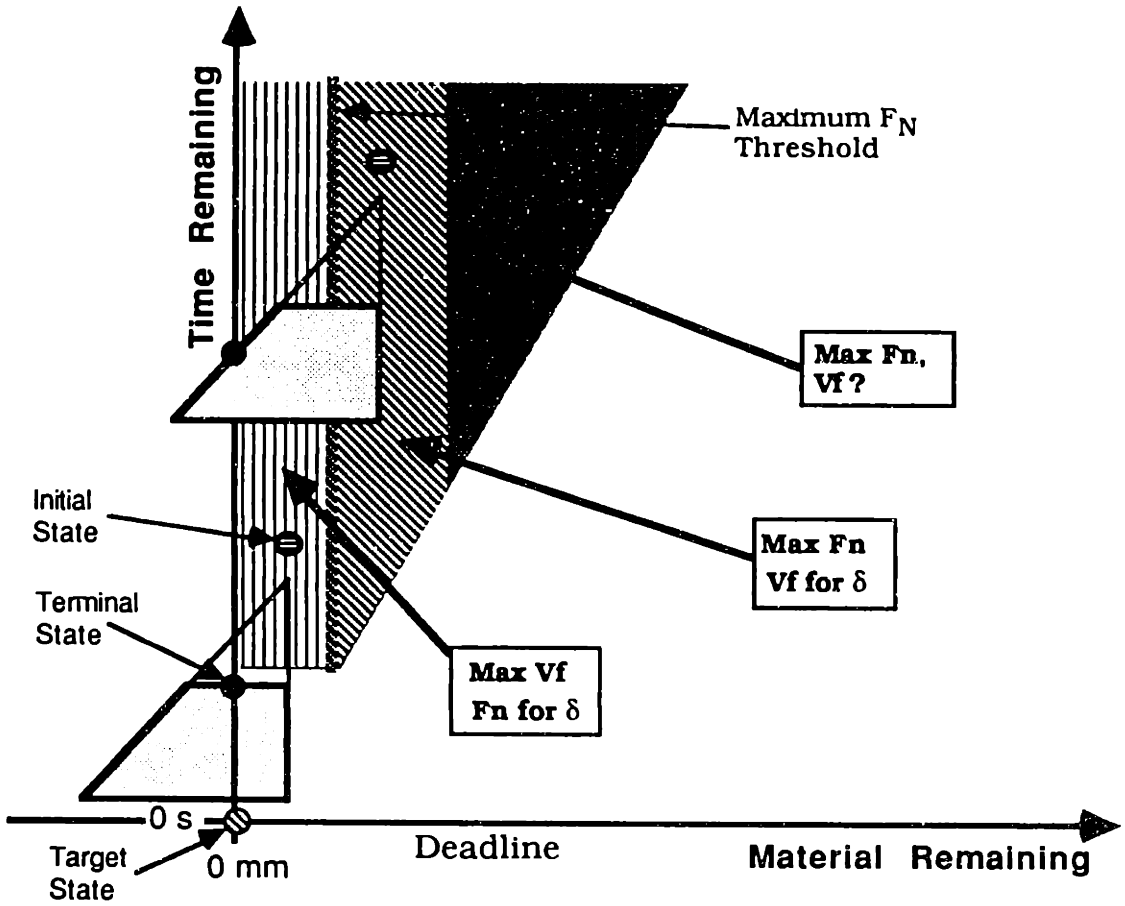


Figure 5-19 Optimal Policies in the Reachable Region

Consider the sub-region of initial states nearest to the zero material remaining axis, indicated by the area filled with vertical lines in Figure 5-19. The optimal policy is to grind with the maximum feedspeed and with a normal force that will yield the optimal cut depth. Now consider the region of initial states having even more material remaining, depicted as a diagonally filled area in the figure. The line of zero material remaining still passes through the region of terminal states, so the optimal cut depth is still attainable. However, the maximum grinding force and a non-maximum feedspeed must be used in order to attain the optimal cut depth The line separating these two sub-regions is called the **maximum normal force threshold**. The same second-order effects apply, so the fastest feedspeed possible is used. The same optimal fraction of the material remaining is suggested as a cut depth, so this sub-region shares the same fractional cut depth as the sub-region nearer the zero material remaining axis. Since

the feedspeed is discrete in the current implementation of the planner, the normal force may vary slightly in the SDP output from its maximum in order to maintain the optimal cut depth, though this is not the true optimal control.

The result is that, when considering increasing amounts of material remaining in this sub-region, the optimum feedspeeds decrease inversely proportional to increasing amounts of material remaining for all time stages. Using x as the amount of material remaining, and F as the fraction of the average weld bead height that is the optimal cut depth, the optimal feedspeed for amounts of material remaining above the normal force threshold can be expressed as:

$$V_f = \frac{C_1 C_3 F_{N\,Max} + C_2}{F\,x} \qquad (5.9)$$

where: x = the amount of initial material remaining

F = the fraction of material to be removed within this time band

The numerator is a constant; the optimal feedspeed is inversely proportional to the amount of material remaining.

For even more amounts of material remaining in the reachable region, no terminal state with zero material remaining is reachable in one grinding pass. Passes starting here will terminate in the above described sub-regions. Here second order effects dominate, and, in general, the optimal grinding policy is to use the maximum possible feedspeed and a grinding force that yields a terminal state within the same reachable region. The planner is quite indifferent to the feedspeed here, as contradicting second-order effects (to be explained later) compete to determine the optimal control.

Optimal Policies in the Region of Initial States that Cannot Reach the Target State

Consider the region of initial states that cannot reach the target state in any legal grinding sequence. Figure 5-20 indicates that the best possible terminal state is one that is reached using the maximum possible grinding force and the slowest legal feedspeed. That is, the best grinding pass is one that is timed to finish right at the deadline and remove as much material as possible. This is only true for the sub-region of initial states (indicated by the shaded region in the figure) which can finish a pass at the deadline. Note that the penalty function slice at the deadline is linear. If the final state is far enough away from the zero material remaining axis so that the tails of its PDF do not extend into the overground region, the expected penalty will be the penalty

of the mean final state. Thus contours of equal penalties, called **iso-penalty contours**, are very nearly straight lines having the same slope as the maximum normal force constraint on the region of terminal states. This fact will be used to explain the optimal policies in the remainder of the unreachable region.

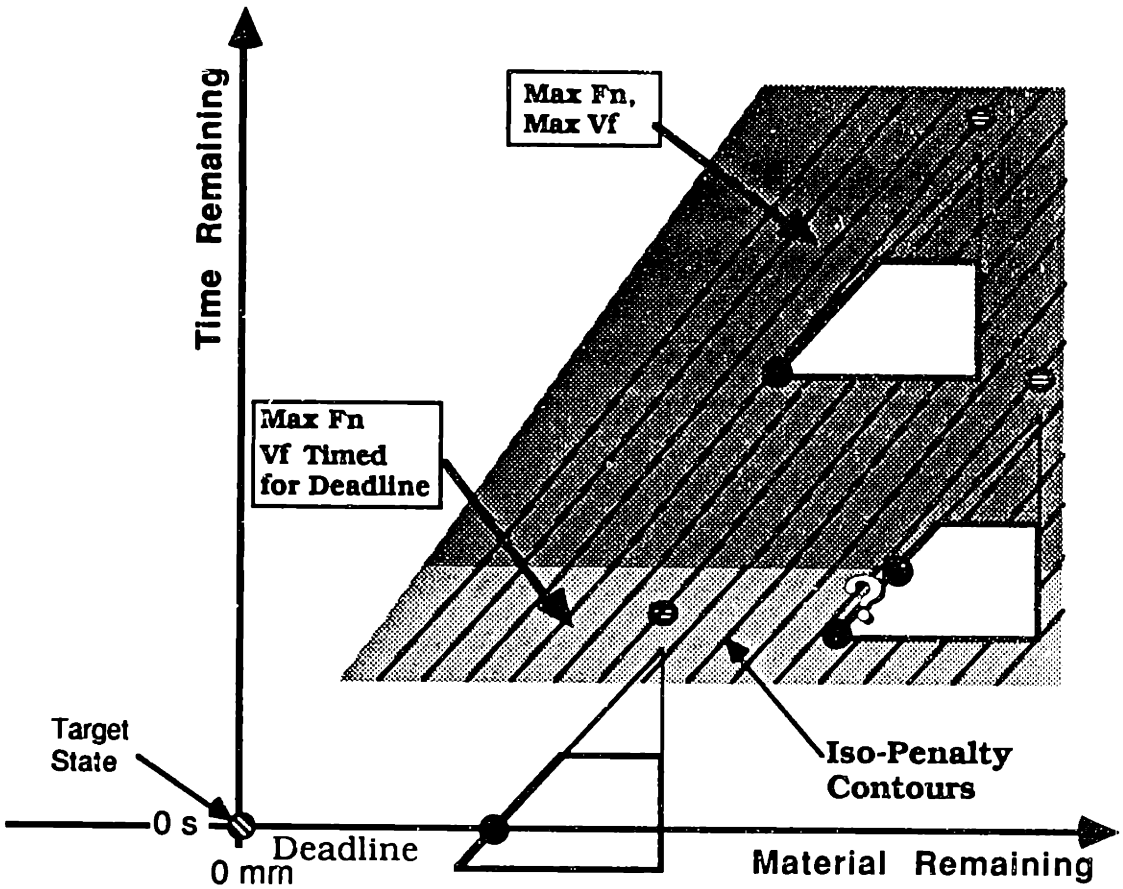


Figure 5-20 Optimal Policies in the Unreachable Region

When there is even more time remaining, a single pass cannot finish right at the deadline, but must finish within the previous sub-region. These initial states are indicated by the darker-filled region in the figure. However, the iso-penalty contours there are nearly parallel to the maximum grinding force constraint, so although it is optimal to use the maximum grinding force, the planner is almost indifferent to the choice of feedspeed because none yields a much better penalty than the other. In theory, for small amounts of time remaining in this sub-region, the best terminal state should be one between the two black dots for the rightmost initial state, marked by the question mark in the figure, because the iso-penalty contours are parallel to the maximum normal force constraint there. Any feedspeed resulting in such a terminal

state will yield about the same penalty. So,in this part of the sub-region, the algorithm *should* be indifferent to the feedspeed. As the initial state has more time remaining in this sub-region, the range of indifference narrows to the point at the lower left corner of the region of expected terminal states, representing a grinding pass with maximum normal force and minimum feedspeed. The local iso-penalty contours begin to take on the slope of a line drawn from the initial state to this terminal state, and this grinding policy becomes the optimal policy for the remainder of the region of initial states that cannot reach the target state. Recall that the lower feedspeed passes waste less time during equipment resets, so a series of long passes will remove more material than a series of fast passes given the same amount of time remaining. So the algorithm should be biased towards long passes in this region. The second-order effects described above do not generally apply because the time-slices of the penalty functions are all very nearly linear, except for the part of the region nearest the zero material remaining axis, where they do apply.

Time-Banded Solution Structure

The minimum time required for a grinding pass generates second-order effects for initial states that can reach the target state. This, in turn, creates an interesting structure in the SDP results. The explanation of this structure starts first with a detailed investigation of the optimal policies described above and the resulting optimal penalty function.

As indicated above, the SDP algorithm does not attempt to start a grinding pass if the pass cannot be completed before the deadline. Hence if the maximum feedspeed requires 6 seconds to complete a pass, no passes are initiated in the last 6 seconds. Call the region in the state space 6 seconds prior to the deadline the **no-grind band**. This is illustrated in Figure 5-21 below. The optimal penalty function is constant with respect to time in those 6 seconds, and is indicated in the figure by thin vertical lines marked with penalty costs. These thin lines represent iso-penalty contours. If there are no in-path time charges, there is no apparent advantage in having a grinding pass end in any time; all time stages are equivalent.

Call the 6 seconds previous to the no-grind band, **band 1**. In this band all time stages are similar; they each are capable of completing a grinding pass that terminates somewhere within the no-grind band, and the penalty functions for the stages in the no-grind band are identical. Recall that faster passes are more accurate, so the algorithm suggests using the highest possible feedspeed for passes terminating in the no-grind

band when the choice is available. Figure 5-21 illustrates band 1 with variously shaded regions in the state space. Consider a grinding pass starting at the state marked A in the figure. This state has less material remaining than the maximum normal force threshold, so a grinding pass starting here can have the optimal cut depth with a choice of feedspeeds. The fastest feedspeed is selected and the grinding pass terminates near the zero material remaining axis within the no-grind band. Because a grinding pass starting here can terminate near the zero material remaining axis, the initial state is charged a very low optimal penalty, say, $0.01. All initial states with the same amount of material remaining as state A will terminate near the zero material remaining axis somewhere in the no-grind band using the fastest feedspeed possible. The SDP algorithm suggests cut depths that are constant with respect to time across this band, and the optimal penalty function is constant with respect to time here as well.

The optimal penalty function is very small and flat for amounts of material remaining in the band below the maximum normal force threshold because these states can reach the target state with equal ease. There is a slight trend towards higher penalties for more amounts of material remaining, since these states require larger cut depths and experience correspondingly larger variances. The result is that there is a discontinuity in the penalty function between these two bands, and that the optimal penalties in band 1 are significantly lower than those of the no-grind band, making band 1 a favorable place to terminate a grinding pass.

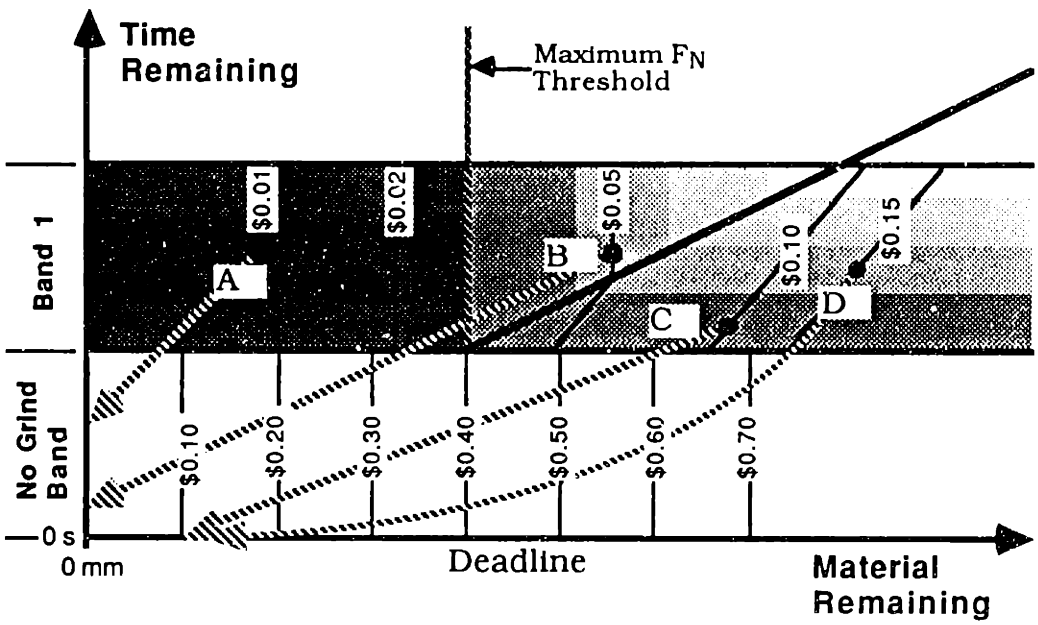


Figure 5-21 Time-Banded Structure of Solution

Now consider the initial state labelled B.in the figure. Grinding passes starting here can have the optimal cut depth, but will have a slightly higher optimal penalty due to the broad PDF associated with a large cut depth. However, grinding passes starting at states C and D are in the unreachable region, and are optimally executed using the maximum grinding force and a feedspeed timed to finish at the deadline. The optimal feedspeeds are indicated by the degree of shading in the figure, with the progression from the darkest shade to the lightest shade representing the faster to slower feedspeeds.

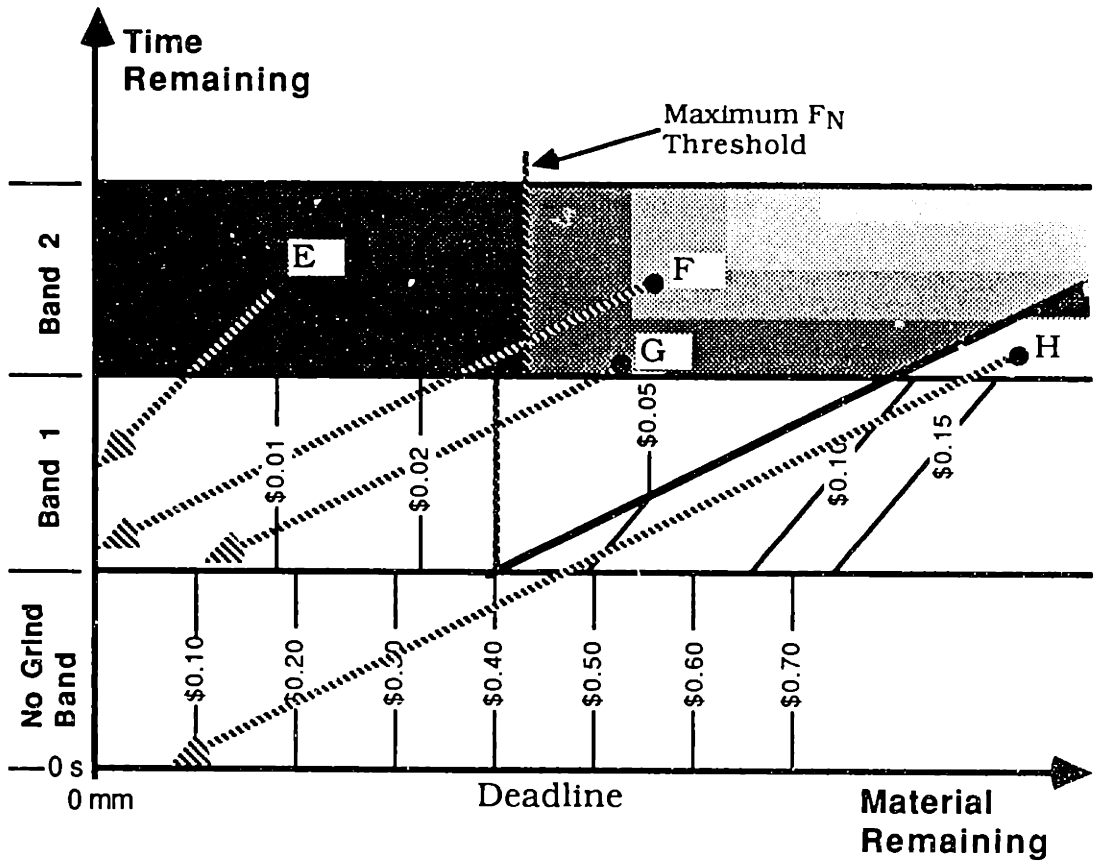


Figure 5-22 Band 2 Details

For the next prior 6 second band (band 2 in Figure 5-22) the pattern repeats itself. Note that if grinding passes terminate in band 1, the optimum cut depth fraction will be smaller because the time-slice of the optimal penalty function is flatter near the zero-material remaining axis. So if the algorithm suggests removing 57% of the material remaining in band 1, it might suggest removing only 49% of the material remaining in this band. Removing a higher percentage of the material remaining in the later bands can be regarded as indicating the algorithm's increasing anxiety at the

approaching deadline. The smaller cut depth fraction also means that the maximum cut depth will be optimal for larger amounts of material remaining than it did in the previous band. Therefore the maximum normal force threshold for this band will fall further to the right in the figure than it did for the previous band.

Grinding passes starting from initial states beneath the maximum normal force threshold such as state E again use the maximum feedspeed. The grinding pass starting at state F is similar to that of state B, with its terminal state falling near the corner formed by the zero material remaining axis and the boundary between the no-grind band and band 1. The high penalties in the no-grind band act like the deadline constraint (indeed many optimization schemes, including the SDP algorithm, implement constraints using very high penalties). The planner is easily able to find better places in the state space to terminate grinding passes. Thus it is better to terminate a grinding pass starting at state G on the boundary between the no-grind band and band 1 rather than having it finish near the zero material remaining axis but inside the no-grind band. State H is within the unreachable region, and the policy here is a continuation of the policy for this region: grind with maximum force and a feedspeed timed to finish at the deadline.

Note that the penalties for any stage in a band are less than the penalties for comparable amounts of material remaining in a later band nearer the deadline since there is time to improve the weld bead via grinding. Recall also that the optimal penalty function becomes shallower with increasing amounts of time remaining (see Figure 5-11).

This banded structure extends through time The bands have a width equal to the time required for the shortest possible grinding pass (the fastest feedspeed). The optimal grinding policy can be summarized with a few rules:

1. If the target state is reachable, attempt to get on a grinding schedule using the maximum grinding force and the maximum feedspeed. Time the first pass so that such a grinding schedule will be possible all the way to the deadline.

2. If the amount of material remaining is under the normal force threshold, attempt to remove a fixed fraction of the material remaining, called the optimal cut depth.

3. The optimal cut depth gets larger with less time remaining until the deadline.

4. If the target state is not reachable, and one grinding pass can terminate at the deadline, grind with the maximum normal force and time the pass to finish at the deadline.

5. In all other cases, grind with the maximum normal force and use the slowest feedspeed.

Figure 5-23 illustrates the results obtained from the SDP algorithm. This figure has four parts, or windows. Each window maps some aspect of the solution onto the state space using different colors[3] to represent different degrees of each aspect of the solution. Thus each window shows three dimensions of information. The time remaining is plotted along the vertical axis, with the deadline at the bottom. The material remaining is plotted along the horizontal axis, with negative amounts of material remaining representing the overground region at the left, and the underground region at the right with positive amounts of material remaining. The discretized nature of the state space is apparent in these figures. The top two windows illustrate the optimal controls. The upper left window shows the optimal feedspeed policy, and the upper right shows the corresponding normal force. The color at each point in the state space indicates the feedspeed of normal force to be used for that initial state. The lower left window illustrates the fraction of the material remaining to be removed. That is, it displays the ratio of the cut depth to the amount of material remaining. The lower right window displays the optimal penalty function.

This figure shows the basic solution structure illustrated in Figure 5-16, and the banded solution structure described above. There is a blank no-grind area both in the overground region and just prior to the deadline. The lower left window shows the large regions beneath the maximum normal force threshold as having a constant fractional cut depth. It also shows how the fractional cut depth decreases with increasing time remaining. The upper right window shows that the normal force increases smoothly in this region up to the normal force threshold, and then remains at the maximum normal force everywhere else. This window also shows how the maximum normal force threshold moves to the right with increasing time remaining.

---

[3]In monochrome reproductions, these colors will reproduce as varying shades of grey.

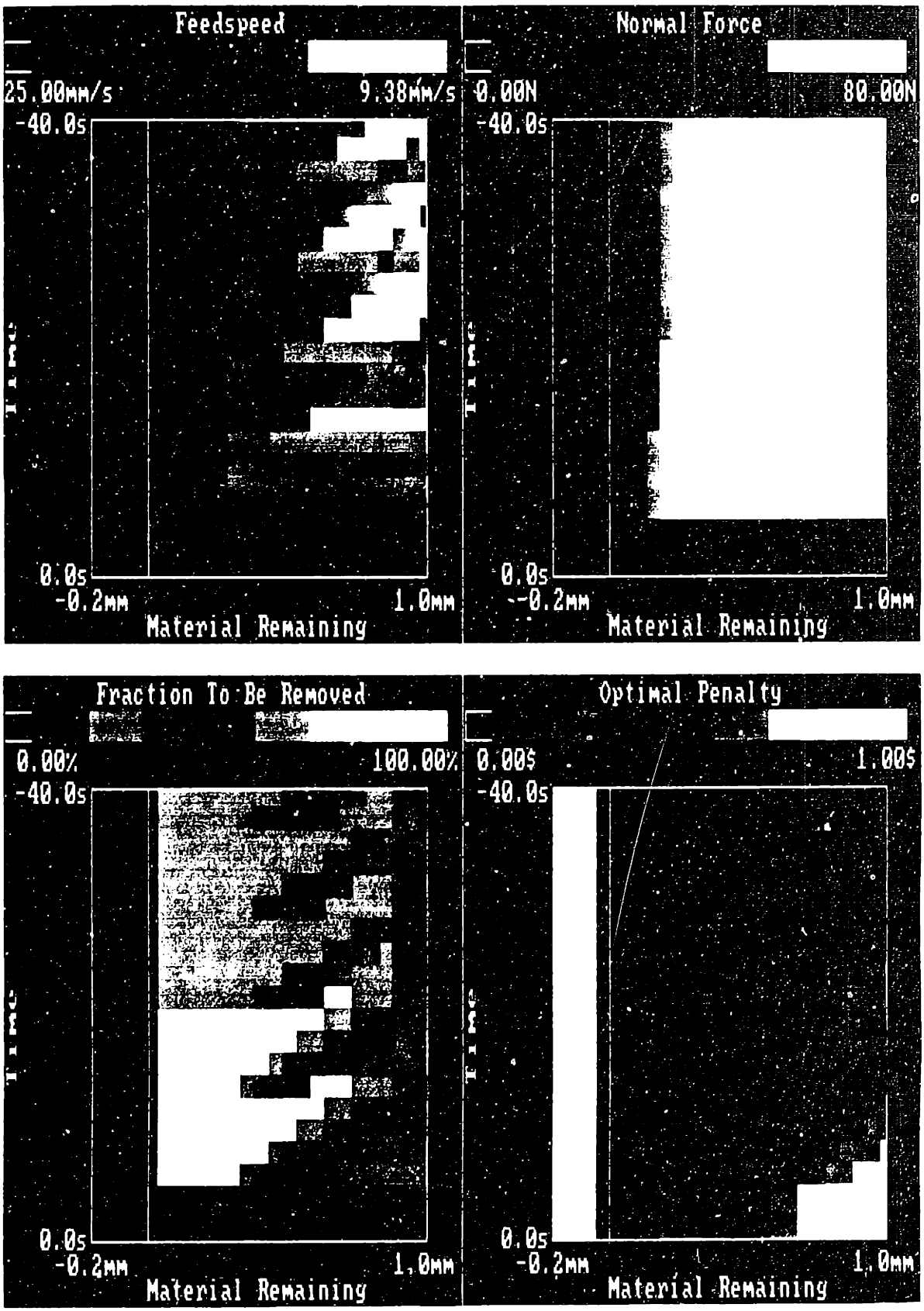


Figure 5-23 **Results from the SDP Program**

The lower right window illustrates the details of the optimal penalty function. It shows that the optimal penalties are constant with respect to time in the no-grind region, and that the reachable region has a very low optimal penalty. The right side of Figure 5-24 shows the same optimal penalty function scaled to show the fine details within the reachable region. It illustrates how the optimal penalty function is constant with respect to time within each time band. The left side of this figure is a repeat of the optimal feedspeeds from the figure above to aid in identifying the banded solution structure.

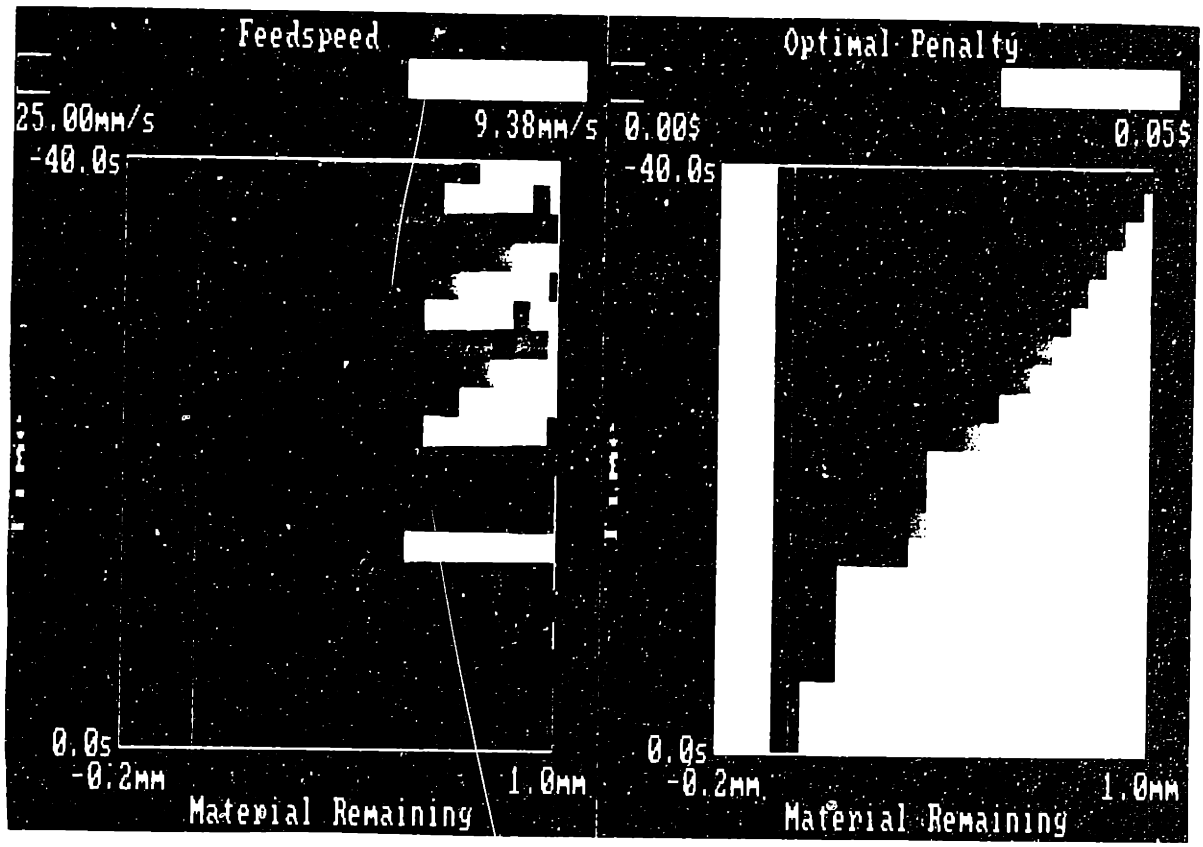


Figure 5-24 Details of the Optimal Penalty Function in the Reachable Region

This section analyzed the weld bead planning problem with a fixed deadline and linear material costs. It derived qualitative optimal grinding policies and verified that the SDP algorithm did yield equivalent quantitative optimal grinding policies.

5.5 GRINDING POLICIES WITH A LATENESS PENALTY

This section explores the effects on the grinding policy if grinding is permitted after the deadline. Theoretically the state space now extends indefinitely in both directions of time. As a practical matter, it is extended to allow grinding only for a limited period of time after the deadline. As will be evident in the sequel, this limitation will affect the grinding policies only in an atypical case. The remainder of this section will describe the modifications that must be made to the problem formulation to implement this new time requirement, and will obtain a qualitative description of the optimal grinding policies.

Lateness Charges and the Quit Now Option

To express the undesirability of violating the deadline, a **lateness charge** will be assessed commensurate with the time overrun. Figure 5-25 illustrates a typical piecewise linear lateness charge.

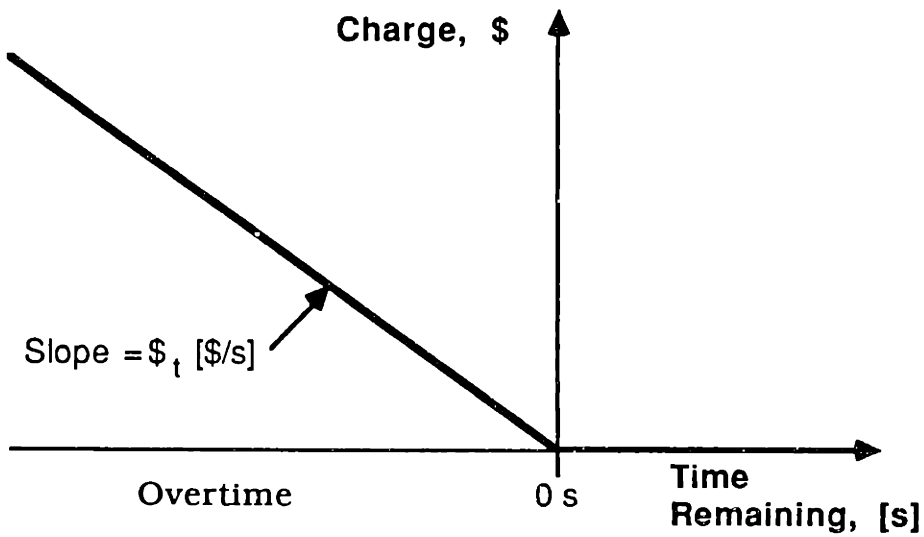


Figure 5-25 Lateness Charge

This type of problem is known as an **optimal stopping problem** [51], in which the algorithm can choose to stop at some point and incur a penalty or continue and attain a lower penalty by stopping later. In this case the SDP algorithm is given an additional choice, called **Quit Now**, to enable it to terminate grinding sequences at specific points in time and thus limit the lateness charge assessed. When there were no charges placed upon the time used, the algorithm could simply chose to grind with zero force and wait for the deadline. Now the algorithm must be able to declare when the grinding sequence

has ended, so that the appropriate charge can be assessed. Thus the lateness charge is only assessed when the Quit Now option is chosen rather than a grinding pass.

Cheap and Expensive Time

When linear charges are assessed for both lateness and the amount of material remaining, the iso-charge curves for the sum of the lateness and material remaining charges are straight lines in the grinding state space. Such iso-charge curves are illustrated in Figure 5-26. Note that these curves become iso-penalty curves only when the Quit Now option is chosen for all states along the curve.

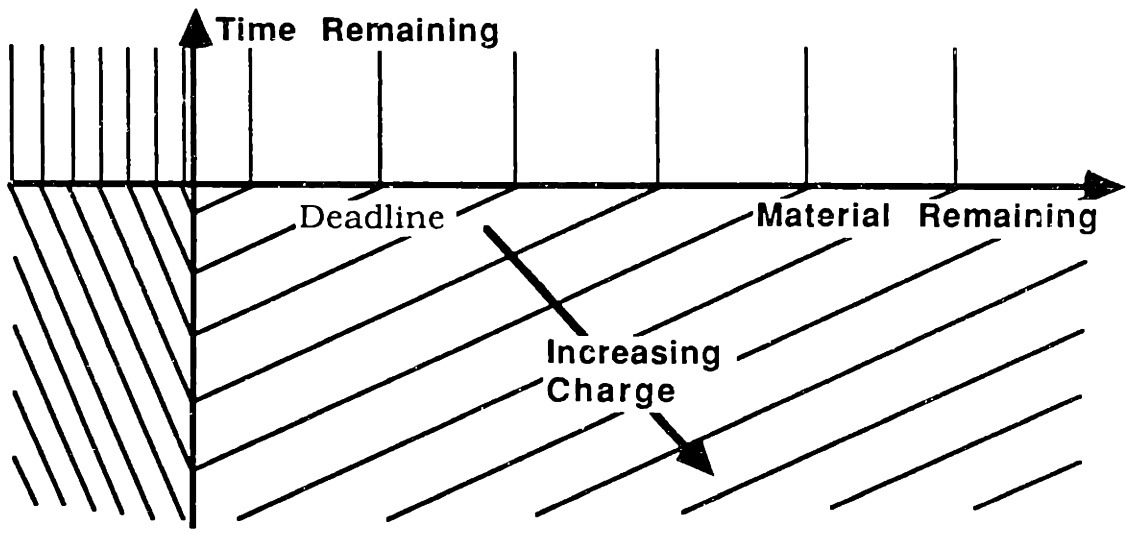
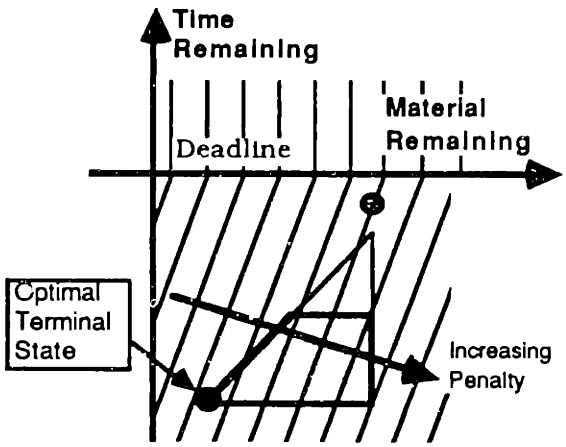


Figure 5-26 Iso-Charge Curves in the Grinding State Space

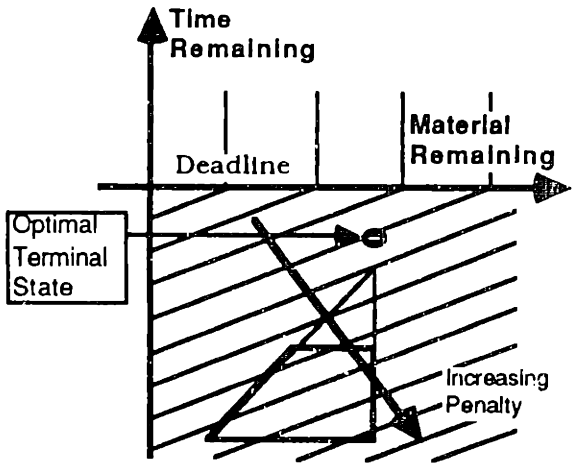
Note that the curves are more closely spaced in the overground region of the state space, indicative of the higher charge for overgrinding than for undergrinding.

Consider a state after the deadline in the underground region of the state space as is illustrated in Figure 5-27a. In this figure, the slope of the iso-charge lines is steeper than the slope of the maximum normal force constraint on the region of expected terminal states. Therefore there will be some terminal state that has a lower charge than the initial state. The terminal state with the least charge is the one associated with the maximum cut depth, and is marked with a solid dot in the figure. Therefore the policy anywhere in this region is to grind with the maximum grinding force and minimum feedspeed until the entire weld bead is removed, because the time used by such a grinding pass will be penalized less than the penalty reduction from removing more material from the weld bead. Time is cheaper than the material, so this Quit Now charge function is called **cheap time**.



Cheap Time

Figure 5-27a



Expensive Time

Figure 5-27b

Figure 5-27b illustrates the complementary situation. The iso-charge curves have a lower slope than the maximum grinding force constraint. Here, none of the terminal states after a grinding pass have lower charges than the initial state. No matter what type of grinding pass is made, none will result in a lower charge than the Quit Now option. The time required by any grinding pass will result in higher charges than can be recouped by the amount of material removed. This Quit Now charge function is called **expensive time**. Quit Now is the optimal policy throughout the underground region after the deadline.

Note that the cheap time charge function results in penalty function contours nearly parallel to the maximum grinding force constraint. This is due to the fact that the penalty assigned to the initial state is the expected charge of the terminal state. Since the grinding policy in uniform everywhere (except near the zero material remaining axis), the optimal penalty contours will have the same slope everywhere. This cannot be reproduced in the finite state space of the SDP program, because there will be end effects near the maximum time limit. The algorithm cannot consider long grinding passes near this time limit because the terminal states of such passes fall outside of the state space. The solution structure given by the SDP program appears similar to the fixed deadline case discussed in the previous section, and is not the correct solution to the problem. This is a limit to the numerical solution technique that is not seen by the analytical solution described above. However, this is not a problem because the more usual case is expensive time, primarily due to the magnitudes of the numbers involved. Equation (5.7) gave the slope of the maximum grinding force constraint as $\frac{L}{C_1C_3F_N + C_2}$. Typical values of the grinding parameters are:

$K_1 = 10\,J/mm^3$	$R = 85\,mm$	$C_1 = 1\times10^{-4} mm^2/N$
$K_2 = 100\,W$	$\omega = 80\,Hz = 500\,rad/s$	$C_2 = -0.8\,mm^2/s$
$w = 12.5\,mm$	$\mu = 0.3$	$C_3 = 1.\times10^{-3}\,s^{-1}$

to yield a slope of 313 s/mm. The slope of the Quit Now charge contours can be computed as:

$$\text{slope} = \frac{\$_x}{\$_t} \tag{5.10}$$

where: $\$_x$ = the charge for material remaining, [$/mm]

$\$_t$ = the charge for overtime, [$/s]

Assumed values are $\$_x$ = $1.00/mm and $\$_t$ = $1.00/s, to yield a slope equal to 1 s/mm, which is far lower than the slope of the maximum grinding force constraint. Note that even though these charges are guesses, it is the ratio of the two that determines the optimal grinding policy, and it is hard to imagine a case in which one second of time would be charged 300 times less than one millimeter of material remaining. The remainder of this discussion analyzes the optimal policies for the expensive time case.

Insufficient Time

Consider now the states just prior to the deadline, illustrated by the darkest region in Figure 5-28. The optimal policy throughout this region is Quit Now. In all these states, there isn't enough time to complete a grinding pass by the deadline, so all the terminal states of grinding passes started from initial states in this region will land after the deadline in regions whose policy is Quit Now. Consider first the expected terminal state region of the middle initial state. All the terminal states have higher charges than the initial state, so the optimal policy here is to Quit Now. Consider next the initial state to the right. Here the charge for Quitting Now is the same as for making one grinding pass at the maximum grinding force and feedspeed. This is the boundary of this region; the algorithm is indifferent to either policy on the boundary. Consider now the leftmost initial state. This has as much time remaining as the rightmost initial state, but the maximum force/feedspeed terminal state lies in the high-charge overground region. No terminal state yields a lower charge than the initial state, so the optimal policy here is Quit Now. The resulting optimal penalty function is identical to the Quit Now charge function, indicating equal penalties for the same amount of material remaining, and no charge for time before the deadline.

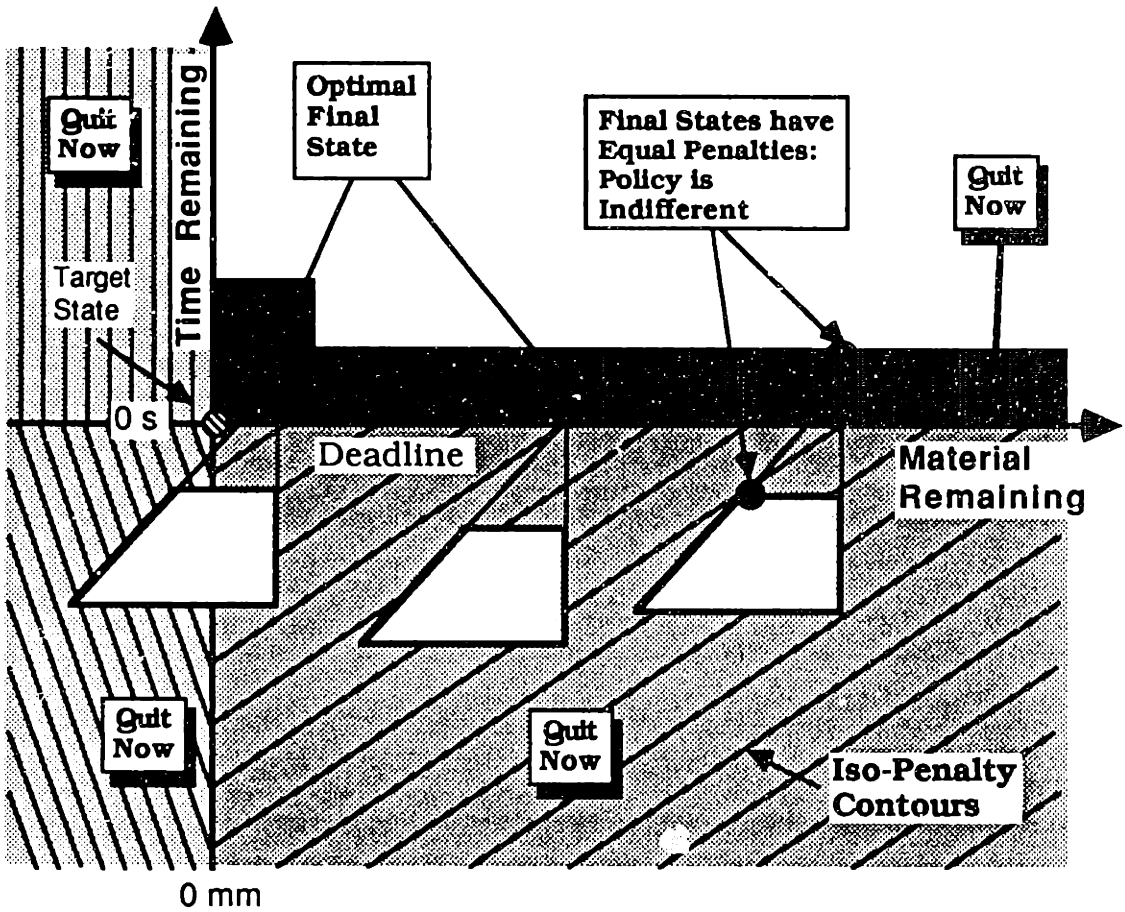


Figure 5-28 Insufficient Time Region

Slightly Late

Now consider the remainder of the initial states for which a grinding pass cannot be completed by the deadline. These are illustrated by the dark region in Figure 5-29. The optimal policy throughout this region is to make one pass with the maximum grinding force and feedspeed, because the resulting terminal state will have a lower charge than any other terminal state or than the charge assessed for quitting at the initial state. These states can take advantage of the ability to finish late. The resulting iso-penalty contours are parallel to the iso-penalty contours for underground states after the deadline.

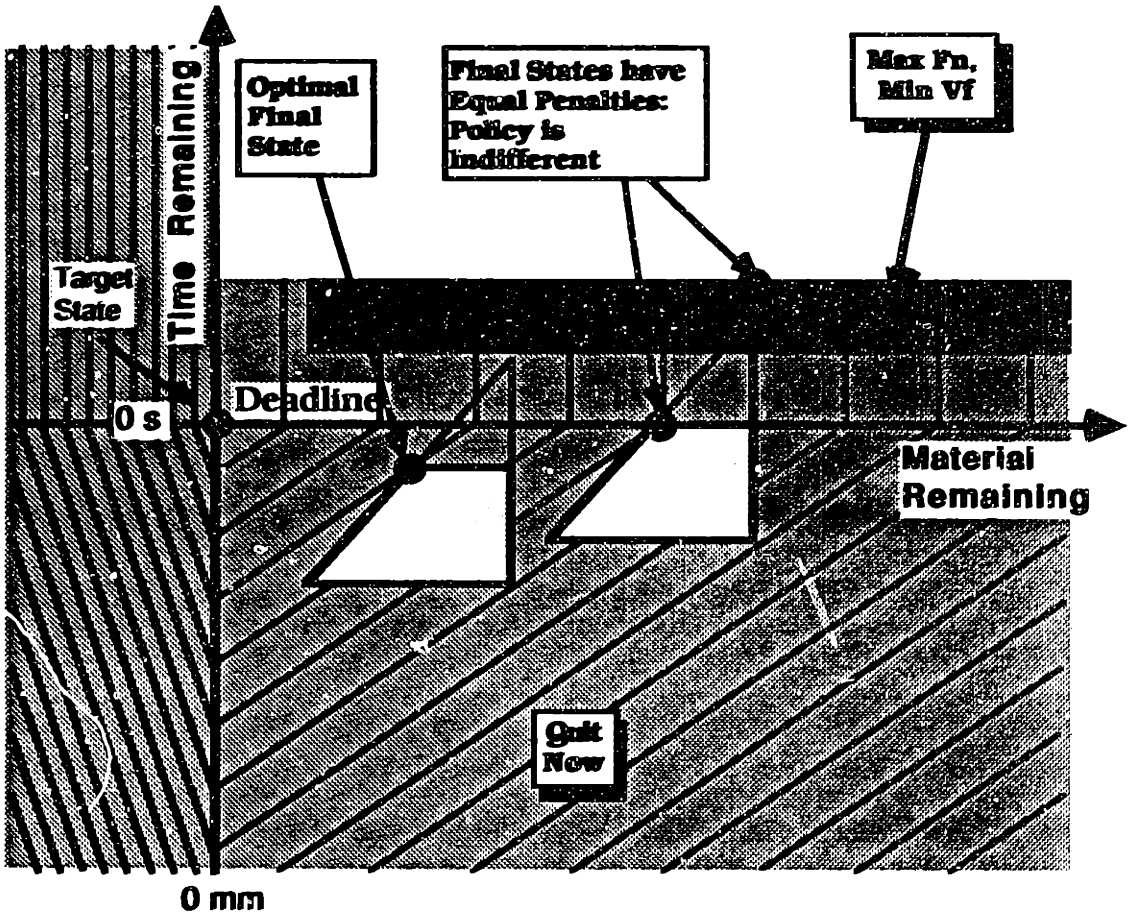


Figure 5-29 Slightly Late Region

The Remainder of the Unreachable States

Now consider the remainder of the initial states which cannot reach the target state via any legal grinding sequence. These are illustrated with two dark regions in Figure 5-30. Consider first those states with the lease amount of time remaining, indicated by the darkest region in the figure. These states can finish a grinding pass before the deadline, and it is optimal for them to do so via a grinding pass using the maximum grinding force and the feedspeed timed to finish the grinding pass right on the deadline. The resulting iso-penalty contours are slightly curved, having a slope which varies from the slope of a line drawn between the initial state and the varying terminal states along the maximum feedspeed constraint.

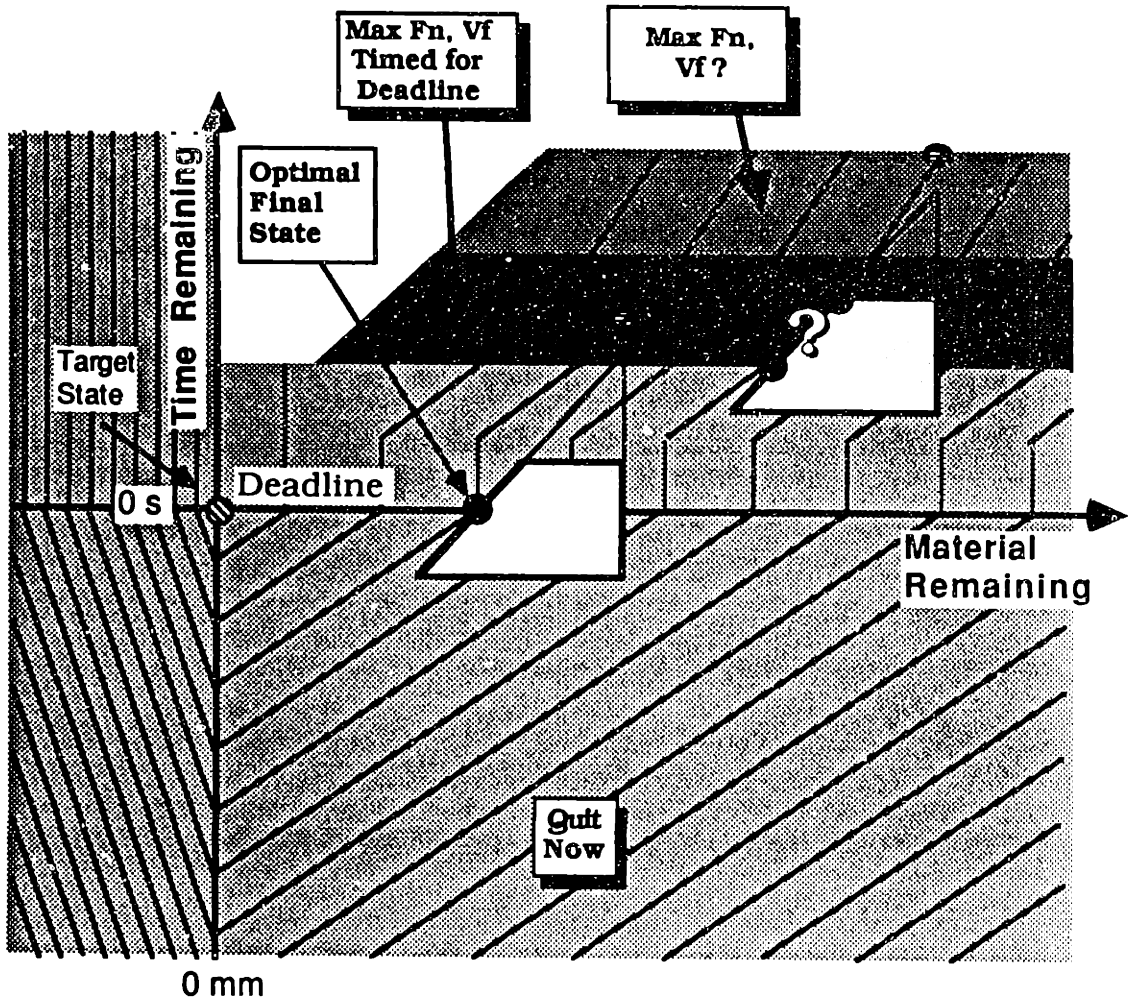


Figure 5-30 The Remainder of the Unreachable Region

For states with even more time remaining, indicated by the slightly lighter region in the figure, no grinding pass can finish at the deadline, so a multiple grinding pass sequence must be executed. The terminal state fo this grinding pass will be along the maximum grinding force constraint. However, the optimal feedspeed is hard to select, because they yield nearly identical terminal state penalties. The algorithm is indifferent to the feedspeed in the remainder of this region because the optimal penalty contours are nearly parallel to the maximum grinding force constraint line.

The Reachable Region of the State Space

The remaining region of the state space can reach the target state, and has no need to use any overtime. Therefore the optimal grinding policies are much the same as those derived for the fixed deadline case in the previous section. The entire solution region is shown in Figure 5-31.

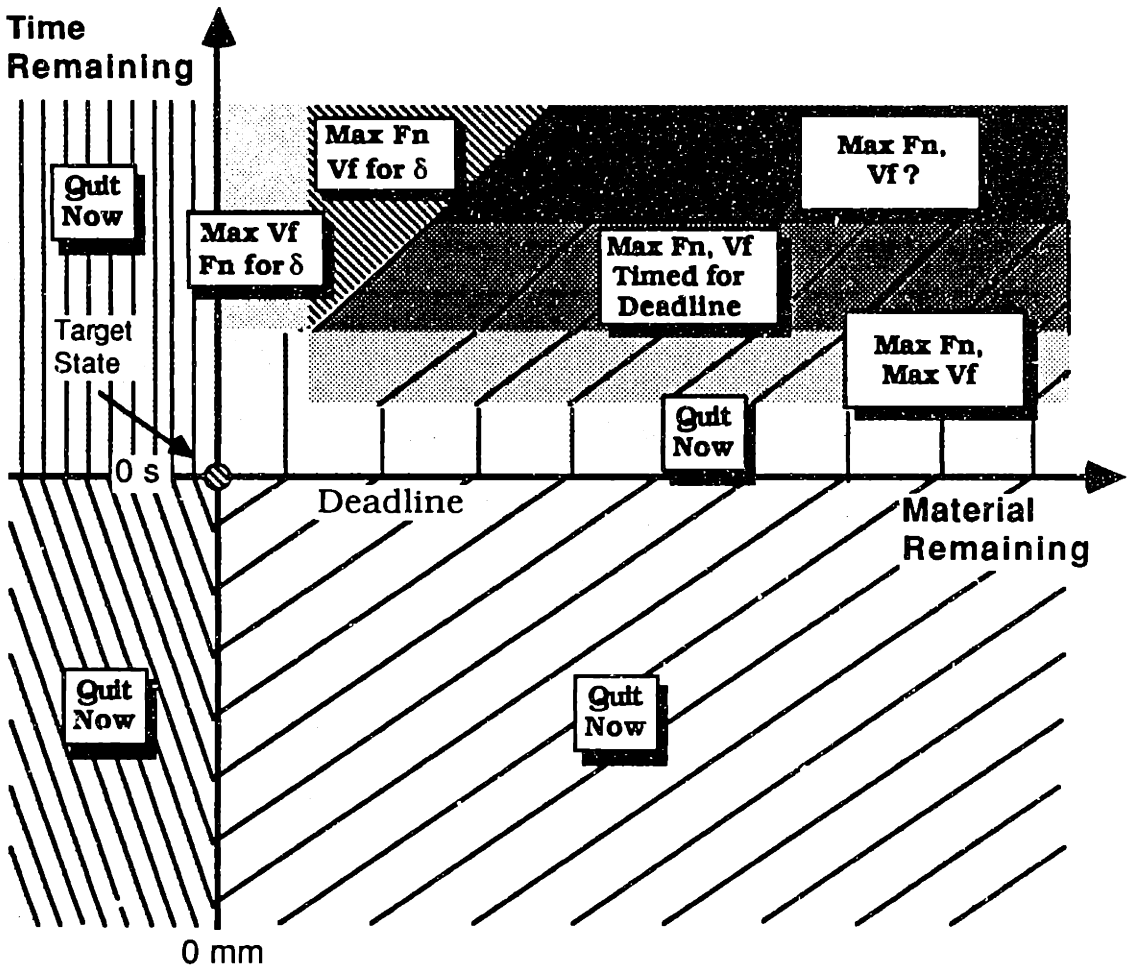


Figure 5-31 Entire Solution Structure for Allowed Overtime

Figure 5-32 shows the corresponding graphical output from the SDP program. Again, the SDP program yields quantitative results equivalent to the qualitative results above. In particular, the program yielded the value of the optimal cut depth which the analysis could not yield. The next section will continue to elaborate the problem given to the planner with another detail from the real problem, the burning constraint.

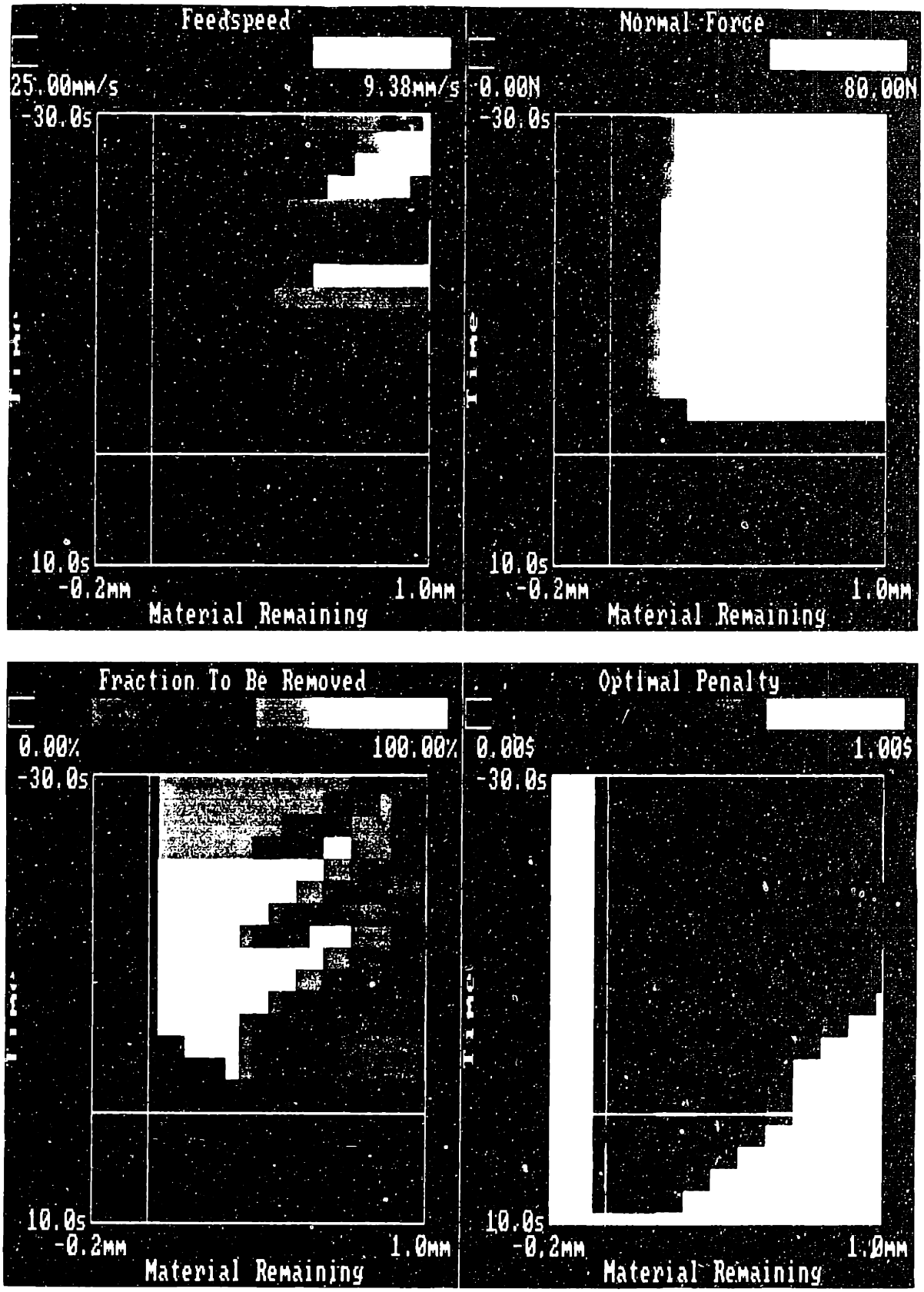


Figure 5-32 **Solution for Allowed Overtime from the SDP Program**

5.6 Effects of the No-Burn Constraint

One of the constraints on the grinding process is that the weld must not be burned as a result of the grinding. Burning occurs during vigorous grinding. Part of the energy of grinding goes to heating the workpiece[44] directly adjacent to the contact patch between the grinding disk. This heating process moves along with the grinding disk as it passes along the weld. Thus the peak temperature of the weld bead is a function of both the instantaneous power supplied to the grinding disk and of the feedspeed. The instantaneous power is proportional to the grinding force, thus the constraint on the peak metal temperature translates into a constraint on grinding force and the feedspeed. This constraint has the form[50]:

$$F_N \le C_B \sqrt{V_f} \tag{5.11}$$

where: V_f = the feedspeed

F_N = the grinding force

C_B = a constant

See Figure 5-33 for the shape of this constraint in the control space. This constraint says that for a given grinding force, there is a minimum feedspeed that avoids metal burning, and the minimum feedspeed increases as the square root of the grinding force.

Now, as the desired optimal cut depth increases, the burning constraint limits the grinding force to smaller and smaller values. The dynamic programming algorithm can no longer choose the maximum grinding force and the associated fastest feedspeed. Now a lower grinding force and slower feedspeed along the burn-constraint curve must be chosen. This yields higher cut depth variances and therefore higher expected penalties. In addition, the maximum possible cut depth is lower when the burn constraint is considered than with just the maximum force constraint.

This new constraint can be mapped into the state space by substituting equations (5.6) and (5.11) into (5.5) to obtain the equation of one boundary delimiting the region of possible terminal states in the state space:

$$\delta = \sqrt{\frac{t_f - t_R}{L}} C_1 C_3 C_B + C_2 \frac{t_f - t_R}{L} \tag{5.12}$$

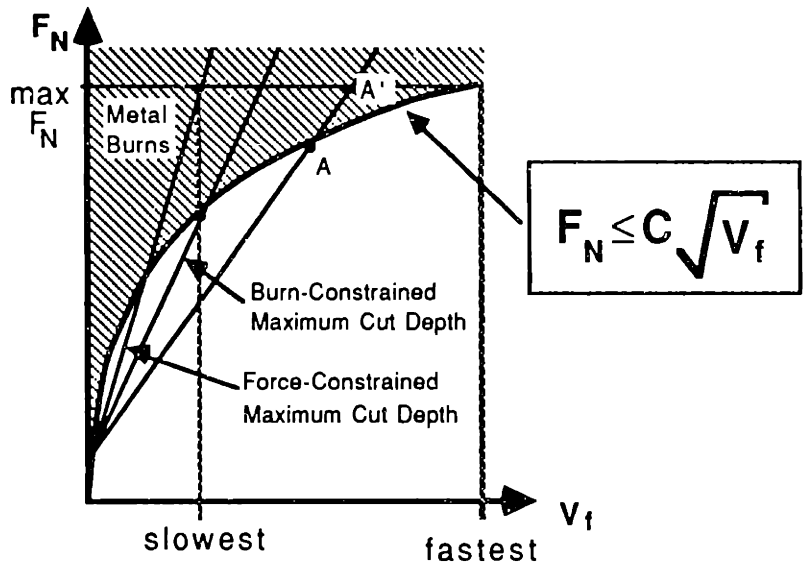


Figure 5-33 Burn Constraint in Control Space

Figure 5-34 illustrates the shape of such a region. The darker shade indicates the region of possible expected terminal states under the burning constraint. The figure illustrates the actual shape using the values used to compute the slope of the iso-penalty contours in the overtime zone in the previous section (see equation 5.10) and $t_R = 0$. This figure illustrates cut depth rather than amount of material remaining, and grinding pass duration, rather than time remaining, because the shape of this region is independent of time- and material remaining. The lighter shade indicates the area that would be included if the no-burn constraint were not applied

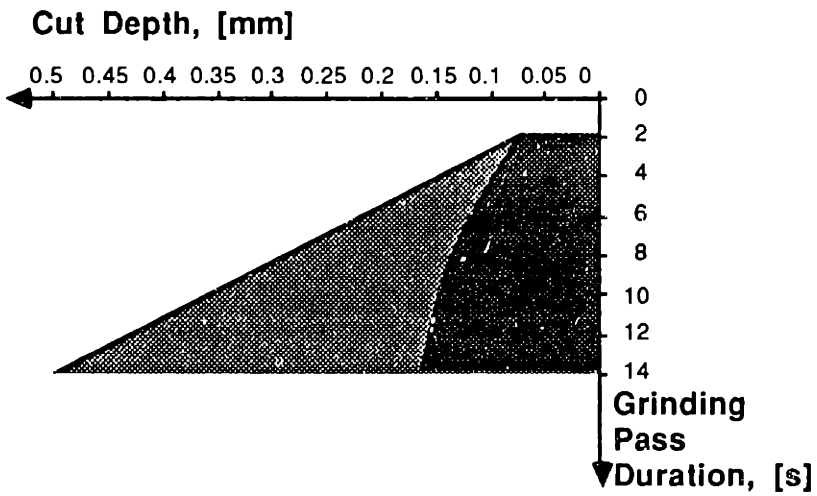


Figure 5-34 Possible Terminal States under the No-Burn Constraint

The analysis techniques of the previous section could be used to determine the qualitative optimal grinding policy, but the analysis becomes difficult with the curved

boundary on the region of possible expected terminal states. This is the type of problem that a numerical solution can solve and an analytical solution has difficulty solving.

Note that because the curved boundary can be approximated by a straight line, an approximate analytical solution similar to that of the previous section is possible. The solution is the same as before in the reachable region of the state space for amounts of material remaining below the maximum normal force threshold. The changes begin when the constant cut depth line intersects the burn constraint curve in the control space. (See Figure 5-33). At this point the previous optimal solution yielded a region in the state space in which the grinding force was held constant at the maximum value and the feedspeed decreased smoothly to give the optimal cut depth. The shape of this region changes subtly from that of the previous solution (since the change in cut depth variance changes with the change in cut depth in a different manner from before). Thus the optimal cut depth no longer scales up to be a fixed fraction of the material remaining above the normal force threshold. Rather, this fraction decreases with increasing material remaining, since this requires cut depths with more variance than before. In addition, the maximum legal cut depth is less for this solution than for the force-constrained solution, hence the suboptimal cut depth region occurs at lower levels of material remaining. In the region of initial states that cannot reach the target state, the time-banded structure observed previously breaks down. The algorithm becomes indifferent to the choice of feedspeed. This is primarily due to the alignment of the optimal penalty contours with the no-burn constraint boundary in the region of expected terminal states. In this case, the difference in penalties resulting from different feedspeeds is of the same order of magnitude as the roundoff error in computing the expected penalty. This is a case in which the numerical solution produces incorrect answers where the analytical solution would not. Figure 5-35 illustrates the results from the SDP algorithm.

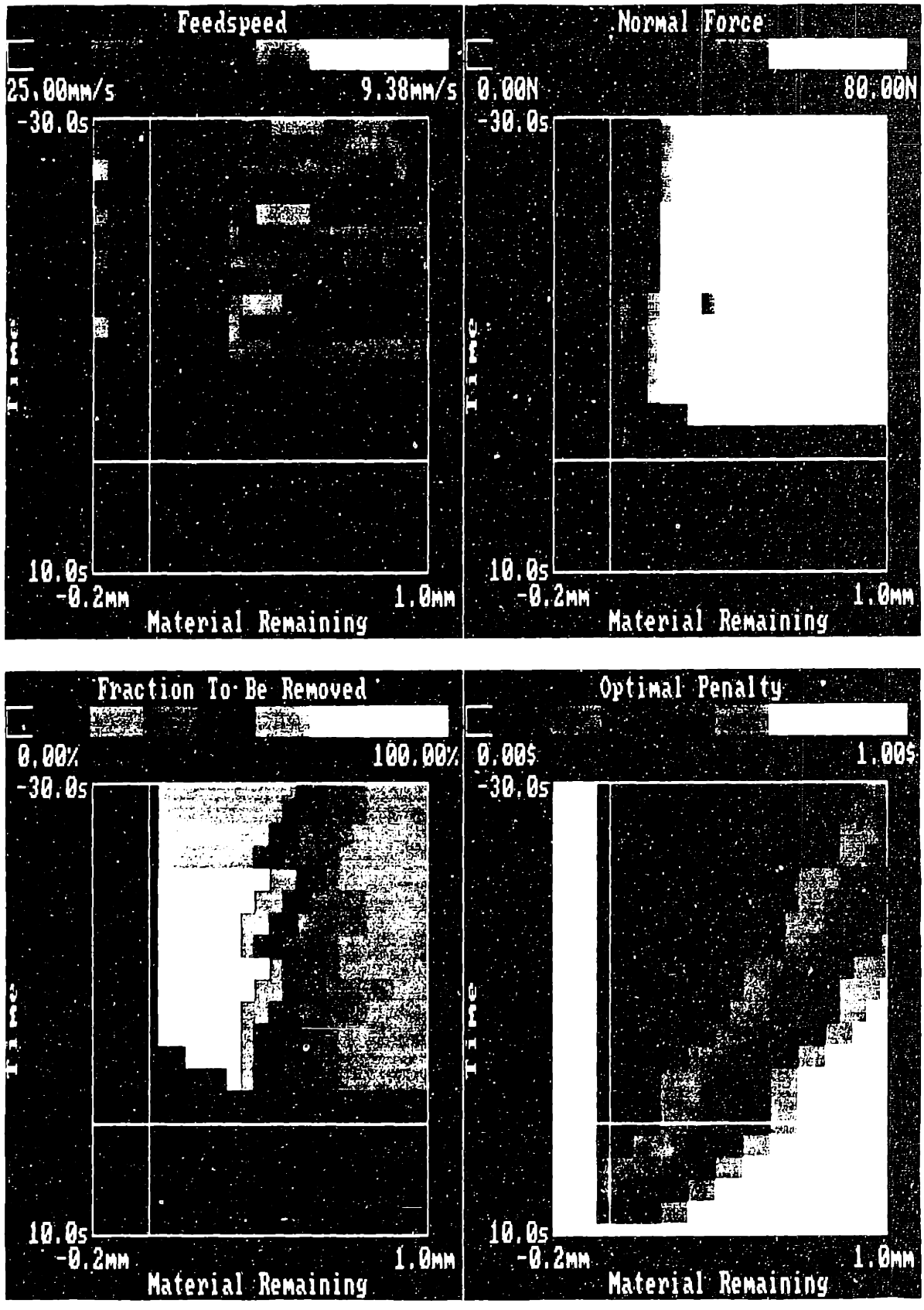


Figure 5-35 SDP Results with the No-Burn Constraint Imposed

5.7 A More Complex Grinding Problem

The grinding problem formulation is now extended to include the quadratic charge functions described in the previous chapter. In particular, the amount of material remaining is charged using the piecewise quadratic function, and the amount of overtime is similarly charged quadratically, rather than linearly. The no-burn constraint remains in force. Analysis of the resulting optimal policies is very difficult, and is not attempted here. Figure 5-36 shows typical results from the SDP algorithm.

Because the reproduction of the color figure below may not show the solution structure clearly, this structure is illustrated below in monochrome in Figure 5-37. This solution is similar to the previous solution, with some small differences. Now the Quit Now region starts after the deadline, since the quadratic lateness charge function doesn't penalize much for being a little late, but penalizes a lot if the sequence finishes more than a little late. So it is optimal to make one last fast grinding pass if there is a lot of material left and no time remaining at the deadline.

In the time band just prior to the deadline with lots of material remaining, the optimal policy is to make one last pass, even though this means finishing a little late. For little material remaining, the optimal policy is to leave well-enough alone and Quit Now. For somewhat more material remaining the optimal policy is to make one grinding pass using the fastest feedspeed, in order to finish as early as possible. The dividing between these two regions within this band is now roughly a diagonal line, suggesting that the grinding system should Quit Now with either very little material remaining or very little time remaining in this band, but not both. For even more material remaining the algorithm gets more aggressive, suggesting that all passes starting in this region of the time band should end at the same late time.

The remainder of the state space displays the same type of policy structure as the previous results, with the same target state reachable/unreachable regions. In the reachable region the optimal policy for initial states near the zero material remaining axis calling for maximum feedspeed passes and normal force tailored to obtain the optimal cut depth. For somewhat more material remaining, the familiar banded structure appears. For initial states in the other bands of the unreachable region, the algorithm is similarly indifferent to the feedspeed, but always suggests using the maximum normal force. Thus the analysis of the optimal policies for linear time and material remaining charges and no no-burn constraint nor motor model is able to yield

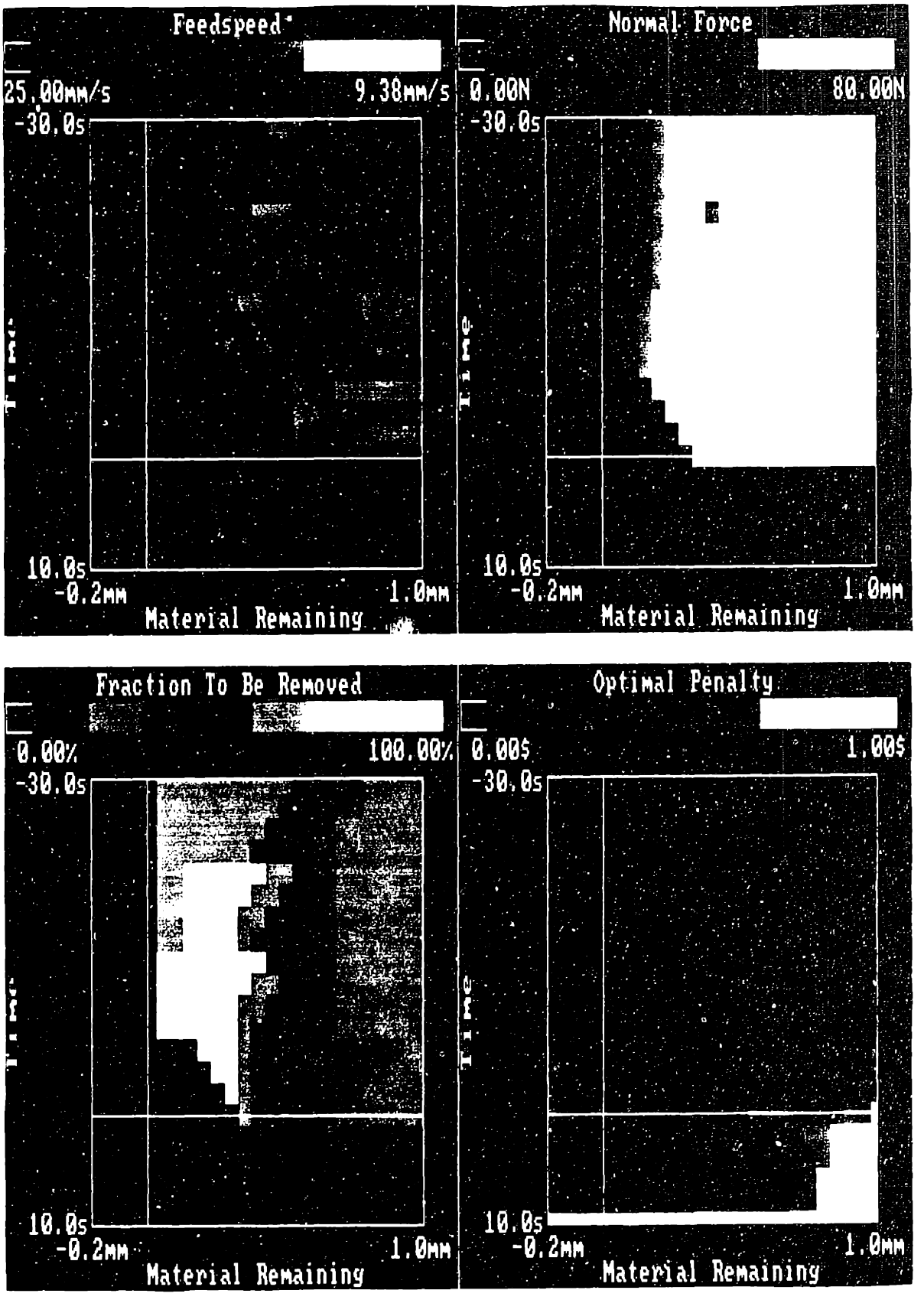


Figure 5-36 SDP Results with Quadratic Time and Material Charges

some insight into the optimal policies for this problem with quadratic charges, no-burn constraint, and motor model. And the analysis yields more believable results where the algorithm becomes indifferent because of numerical problems. However the analysis could not yield the values of the normal force to use near the zero material remaining axis.

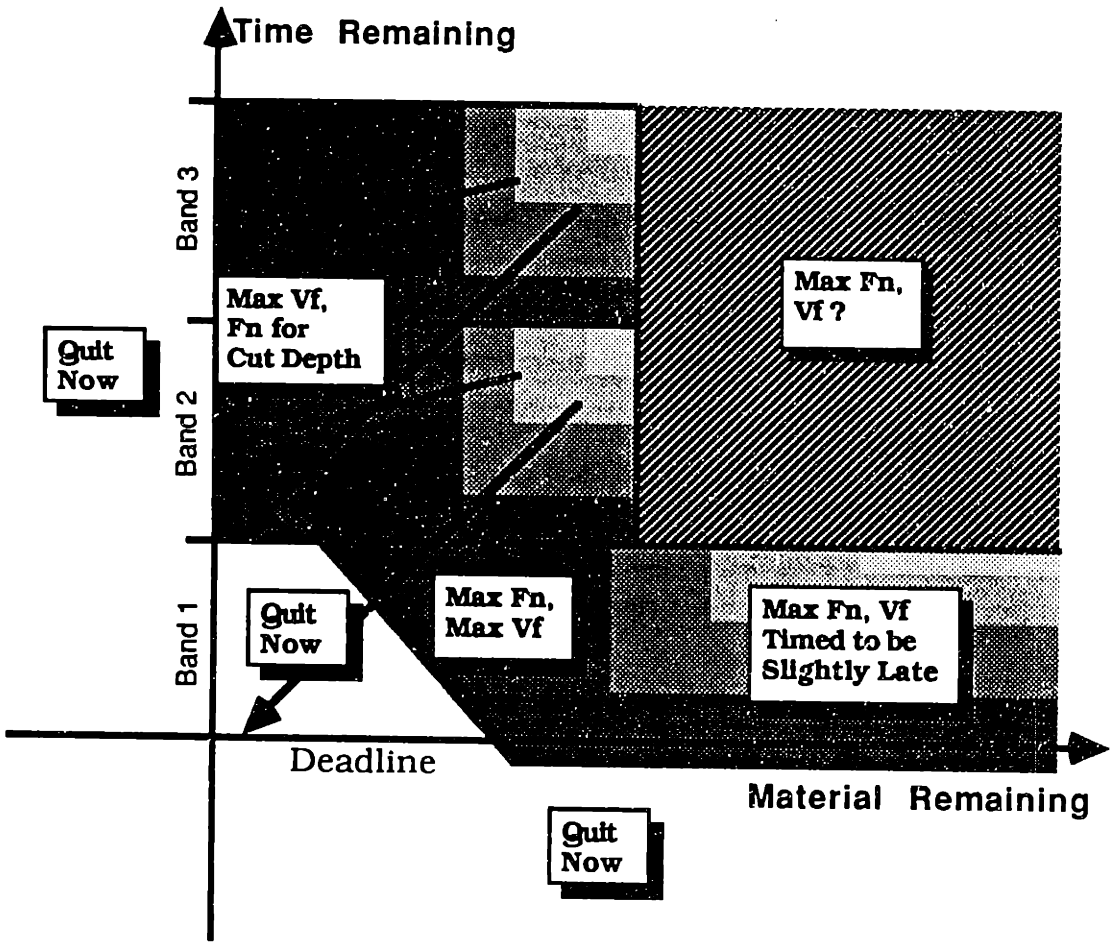


Figure 5-37 **Summary of the Optimal Policies for Quadratic Time and Material Charges**

5.8 CHAPTER SUMMARY

This chapter described the implementation of the long term grinding sequence planner, which planned the sequence of grinding passes required to remove the weld bead safely and accurately. This planner determined the number of grinding passes and how much material should be removed during each pass and at what feedspeed. The solution scheme used was Stochastic Dynamic Programming set up to optimize an expected penalty function by the selection of the normal force and feedspeed given the

initial time remaining and material remaining before each grinding pass. This penalty function measures how well the entire grinding sequence satisfies the weld bead grinding requirements and satisfies the task constraints. The stochastic results of a grinding pass were predicted using a probabilistic model that propagated the probability density function of a grinding model coefficient to the probability density function of the amount.of material remaining after the pass.

A qualitative description of the optimal grinding policies for linear charges was obtained through detailed analysis. The resulting policies were dominated by grinding passes which attempted to reach the target state of no material remaining at the deadline. Where this could be done in different ways, second-order effects determined the optimal policy. These second-order effects included selecting the shortest possible pass because faster passes were more accurate than slower passes, and a time-banded structure due to the minimum time that a grinding pass could take. The SDP algorithm was verified to yield the quantitative equivalent of these policies when the time deadline was rigid and when grinding was allowed after the deadline, except where the difference in penalties charged for different feedspeeds was smaller than the algorithm's numerical accuracy.

With the SDP program verified, more complex grinding requirements and constraints were solved by the algorithm. These involved including a no-burn constraint, a motor model, and quadratic time and material costs. The optimal policies could not easily be analyzed, but were described.

6. PLANNING INDIVIDUAL GRINDING PASSES

The previous chapter described a grinding sequence planner that decides what average cut depth and what feedspeed to use for each grinding pass. These are sent as commands to the second level of the control hierarchy, the pass planner, which converts the average cut depth command into a force trajectory. This chapter describes the individual pass planner.

The conversion of cut depth to force trajectory is done in two stages. First the average cut depth requirement is converted into a total volumetric removal requirement, and this is converted to a disk tip trajectory using the weld bead contour. This is described in the next section. Then a dynamic simulation of a controlled grinding process is run to obtain the required force trajectory. Since the grinding process is simulated, the controller can be omniscient and knows the values of all the process model states at all times. This enables it to control the resulting contour shape very accurately. A by-product of this simulation is the grinding force used during this simulated control. This grinding force is then commanded to the third level of the control hierarchy.

This chapter is organized as follows. The first section describes the conversion from average cut depth to actual cut depth. The next sections describe the development of the controlled grinding simulation, starting with a review of the history of dynamic grinding models, the derivation of the dynamic grinding model used by the planner, its implementation in a computer simulation, the experimental verification of that simulation, and finally the conversion of the simulation to a grinding pass planner. The final section presents results from actual planned grinding passes.

6.1 Computing the Disk Tip Trajectory

The model used by the grinding sequence planner is one-dimensional; it assumes the workpiece to have a constant length and width, and computes the change of height as a function of time. The height really represents the total volume remaining in the weld bead, and is a useful measure for how close the grinding is to being finished. However, the grinding pass planner must deal with a two dimensional contour, so it is necessary to convert that change of volume to a final contour shape.

This is done by converting the cut depth required by the sequence planner into a change of volume via the simple relation:

$$\Delta V = \delta w L \tag{6.1}$$

where: ΔV = the change of volume, [mm^3]

δ = the average cut depth required by the sequence planner, [mm]

w = the average weld bead width, [mm]

L = the weld bead length, [mm]

The next step is to compute a final contour so that the grinding pass will remove the requisite volume from the given initial contour. The final contour shape is chosen to be parallel to the parent material. Refer to Figure 6-1 for an illustration of this computation.

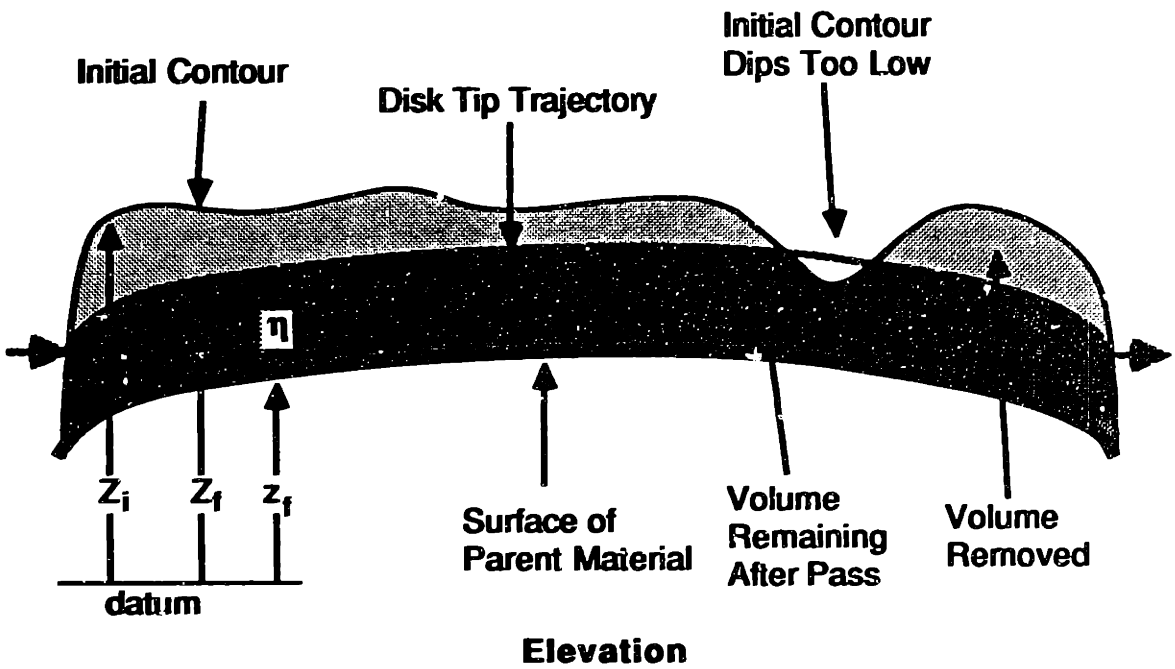


Figure 6-1 Computing the Final Contour

The location of the final contour is adjusted vertically until the computed difference in volume between the initial and final contours matches the desired volume change. The volume difference is computed via a numerical integration. The final contour location is solved for quickly using a binary search scheme. The pertinent relations are:

$$\Delta V(\eta) = \int_0^L g\left(Z_1(x) - z_f(x) + \eta\right)\, dx \tag{6.2a}$$

N is the value of η that solves the equation:

$$\Delta V(\eta) = \Delta V_{des} \tag{6.2b}$$

$$Z_f(x) = \min(z_f(x) + N, Z_1(x)) \tag{6.2c}$$

where: x = the horizontal coordinate

$Z_1(x)$ = the initial contour

$z_f(x)$ = the desired final contour shape

$\Delta V(\eta)$ = the volume difference between the initial and final contours

η = vertical adjustment to the contour height

ΔV_{des} = the desired change in volume, $= \delta w L$

$$g(z) = \begin{cases} z \text{ for } z>0 \\ 0 \text{ for } z\leq 0 \end{cases}$$

N = the vertical adjustment required to attain ΔV_{des}

$Z_1(x)$ = the final contour height at the location x along the weld bead, in the required vertical location

Equation (6.2a) computes the volume change for a grinding pass with the initial contour $Z_1(x)$ and final contour $z_f(x)+\eta$. Equation (6.2b) solves the first equation to obtain the adjustment η needed to yield the volumetric change required by the sequence planner. Equation (6.2c) yields the final contour, accounting for the points in the initial contour that dip below the desired final contour.

In the laboratory simulation, the desired final contour shape was a horizontal line ($z_f(x) = 0$) for use with steel bars simulating weld beads. Having the final contour parallel to the parent material improves the accuracy of the one-dimensional material removal assumption made in the sequence planner's grinding model. It also makes the pass planner's job easier in the future, since the simulation works faster and more accurately with flat contours. However, it is not always possible to have a final contour parallel everywhere to the parent material because the initial contour can dip too low.

The desired final contour $z_f(x)+H$ (N.B. in general, $z_f(x)+H \neq Z_f(x)$) is used as a trajectory for the grinding disk tip in the controlled simulation. If a grinding pass is controlled so that the disk tip follows along this trajectory, the required volume of material ΔV_{des} will be removed.

6.2 Motivation for the Dynamic Model

Grinding is a dynamic process with several levels of detail. The grinding sequence planner can ignore the short- and very short term details of the grinding process behavior because these occur faster than the sequence planner need be bothered with. However, for the grinding pass planner, there are some behavioral details that cannot be ignored. The details involve the time response of the grinder to variations in the initial contour height and to the force applied to the grinder. The output of this level, the shape of the resulting contour, does not react instantaneously to these variations. Such behavior can only be modelled with a differential equation. Therefore a dynamic grinding model is needed for planning individual grinding passes.

6.3 Previous Dynamic Grinding Models

The previous grinding models described steady-state grinding; there was no attempt to describe the grinding process as a function of time. Such a dynamic grinding model is needed for planning the force trajectory. There have been relatively few dynamic grinding models. The following is a summary of a dynamic grinding model that is similar to the model developed for weld bead grinding, and is one of the few dynamic grinding models in the literature.

Hahn considered the time evolution of the shape of the workpiece during precision internal grinding[52]. The goal of such grinding was to create a concave cylindrical surface having a precise center and radius. The mechanism studied was a grinder mounted on a cross-slide of known mass and stiffness. This is shown in Figure 6-2.

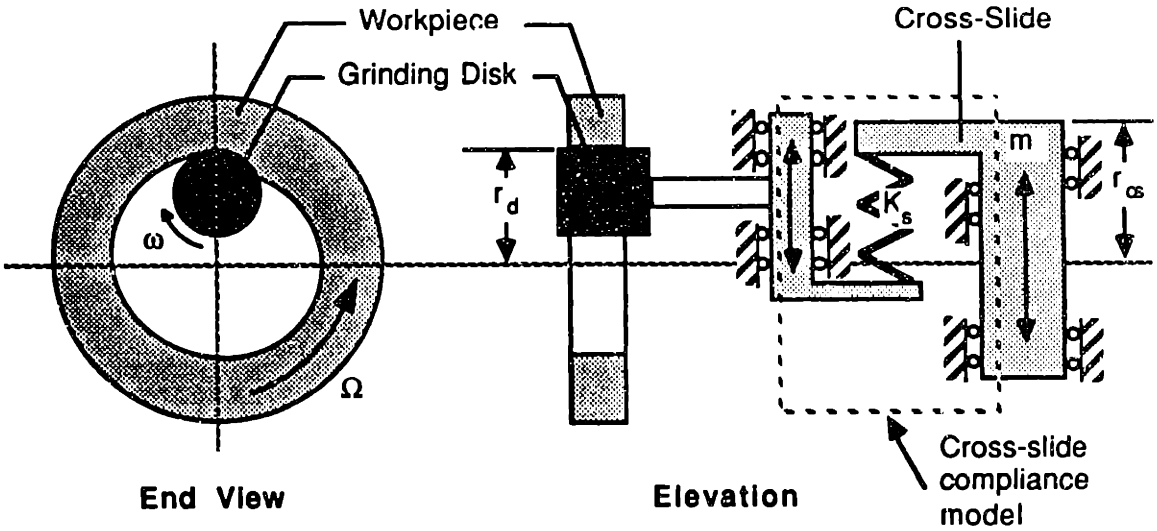


Figure 6-2 Internal Grinding System (from Hahn)

where: F_n = the grinding force

K_S = the stiffness of the cross-slide

r_{cs} = the cross-slide radial position, measured from the center of rotation of the workpiece

r_d = the grinding disk radial position, measured from the center of rotation of the workpiece

μ = the grinding coefficient of friction, = F_t/F_n

F_t = the component of the grinding force tangential to the grinding disk

δ = the cut depth per disk rotation [mm]

Ω = the workpiece rotation rate [rad/s]

$$\Gamma = \frac{K_1 \mu R \omega}{\Omega r_d w}$$

$$n = \frac{\Omega}{2\pi}, \text{ [rev/s]}$$

Assuming that $r_d(t)$ changes slowly, so that its absolute value could be approximated by a constant, r, the equations (6.5) through (6.6) could be combined to obtain a first-order differential equation describing the time evolution of the diameter of the workpiece.

$$\frac{dr_d}{dt} = \Gamma n K_S (r_{cs} - r_d)$$

$$= \frac{K_1 \mu R \omega n K_S}{\Omega r_d w} (r_{cs} - r_d) \qquad (6.7)$$

Hahn solved this equation for a known $r_{cs}(t)$ trajectory (a constant feed rate followed by a 'sparkout' period of no cross-slide movement) and known initial

workpiece radius to obtain a final workpiece diameter. He then derived an expression for the error in the final diameter as a function of the variation of the initial diameter, for fixed grinding process parameters.

In a much later publication [53], Hahn reported no further progress for this research. A literature review failed to uncover any other attempts at dynamic shape control via grinding. There is abundant research in dynamic modelling of the grinding process for chatter [54-56]. However, this work concentrated solely on the chatter problem, and there appears to be no extension towards dynamic shape control.

There is some similarity between multiple-pass weld bead grinding and Hahn's model of precision grinding (equation (6.11)). In Hahn's model, the results of grinding during one rotation were inputs to the next rotation. In weld bead grinding, individual passes correspond to rotations of the workpiece in precision grinding. However, the dynamic excitation of the rotating workpiece is absent in weld bead grinding. In weld bead grinding the excitative forces come from the initial contour and have a continuous frequency distribution. Thus the stability and frequency response analysis techniques used by Hahn are inapplicable for weld bead grinding. The next section describes the dynamic grinding model used by the pass planner.

6.4 The Dynamic Grinding Model

The static grinding model is suitable for the medium term planner, but is not accurate enough for planning individual grinding passes. This section will derive the dynamic grinding model from the static model and first principles.

The dynamic model includes a rotational spring model for the support of a rigid massless grinder, with linear relationships between the grinding normal force and the tangential force, and between the instantaneous material removal rate and the instantaneous power supplied to the contact patch. Figure 6-4 is a schematic of the model and defines the terms. The assumption that the grinder is massless is acceptable because it was found that the rotational accelerations of the grinder were insignificant, and that the resonant frequency of the spring-mass system was in the neighborhood of several hundred Hertz, whereas the natural frequency of the grinding process occurs at much lower frequencies (from 2-10 Hz). At these low frequencies the spring-mass system of the grinder acts as if it were only a spring.

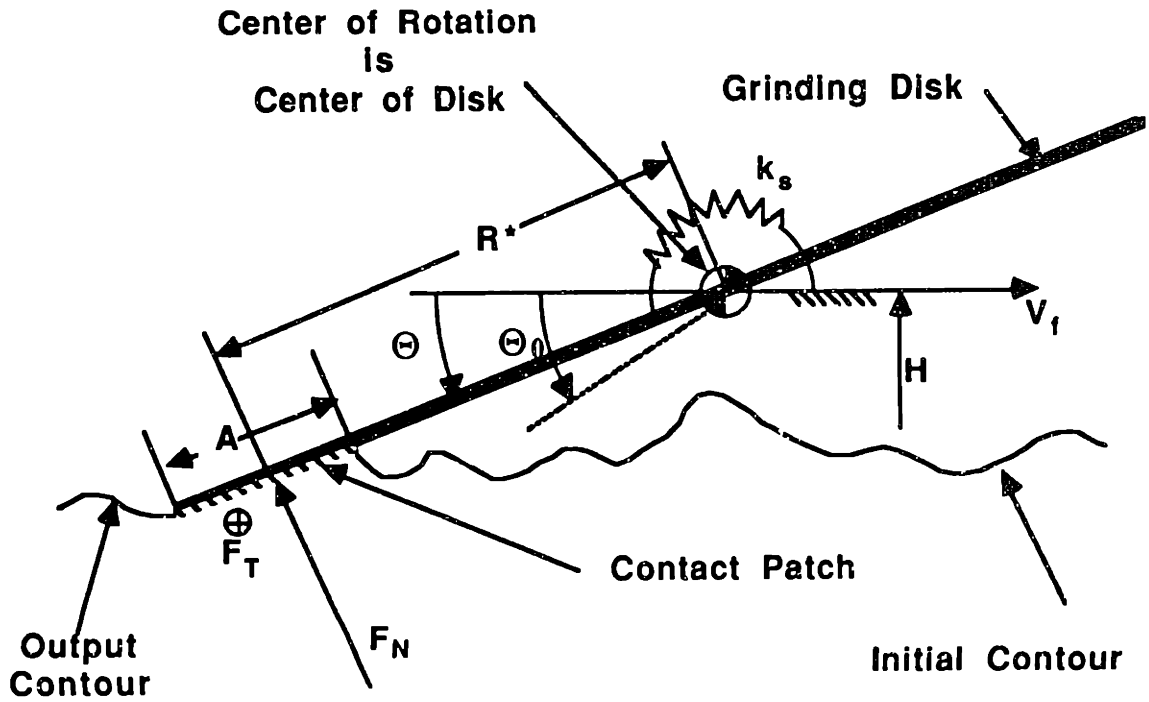


Figure 6-4 Schematic for the Dynamic Grinding Process Model

The derivation is as follows. The angle of the disk is related to the grinding force through the spring model.

$$F_n = \frac{K_S}{R^*}(\theta_0 - \theta) \tag{6.12}$$

where: θ = the deflection of the grinder from its relaxed position θ_0, [rad]

K_S = the spring stiffness [N·mm/rad]

F_n = the grinding force normal to the contact patch [N]

R^* = the average contact patch radius [mm]

The grinding force is one of the factors determining the grinding power. The equations (4.4a-d) can be used to obtain an equation for the material removal rate given the disk rotation rate and disk angle:

$$Q = \frac{\omega \mu K_S(\theta_0 - \theta) - K_2}{K_1} \tag{6.13}$$

Now the material removal rate, Q, is determined by the position and velocity of the disk, and, in particular, by the position and velocity of the contact patch, where all the grinding takes place. Q is derived in a fluid mechanics approach by assuming that all the material that the contact patch runs into is removed through the contact patch 'portal'. The material removal rate is therefore given by the dot product of the contact patch area, A, with the net velocity vector of the contact patch. The velocity vector is a

vector sum of the feedspeed, V_f, the rate of change of the grinder fly height, dH/dt, and the rotational velocity of the grinding disk, dθ/dt. See Figure 6-5 for the vector sum.

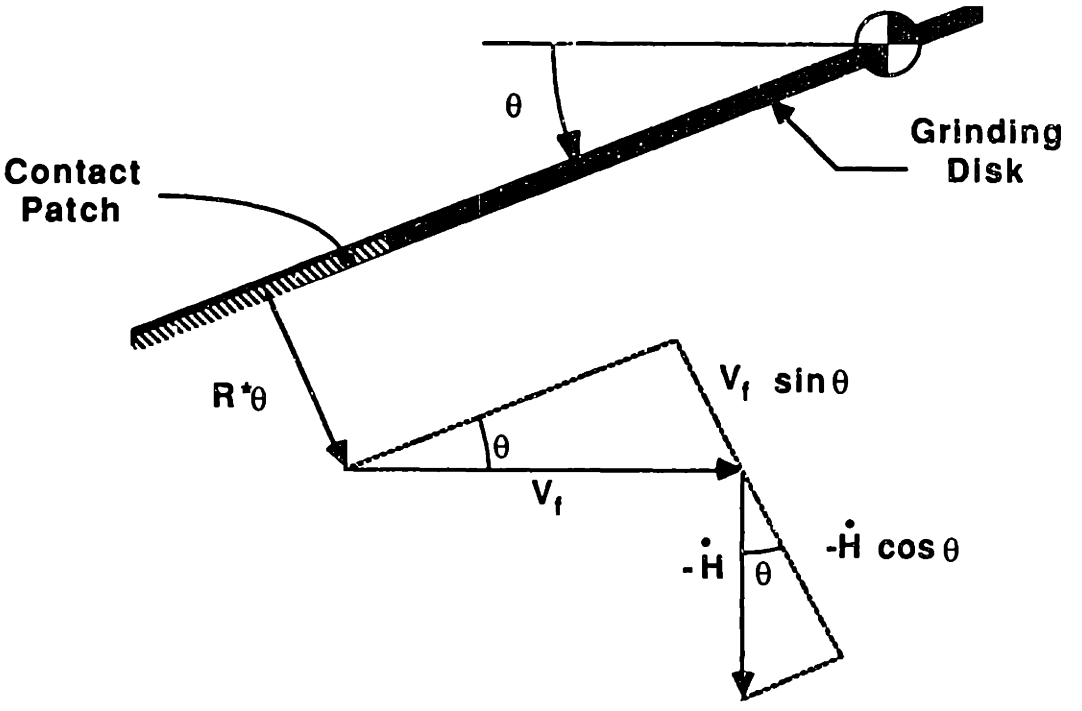


Figure 6-5 Vector Sum for the Contact Patch Velocity

The resulting expression for the material removal rate is:

$$Q = A\left(R^* \frac{d\theta}{dt} + V_f \sin\theta - \frac{dH}{dt}\cos\theta \right) \qquad (6.14)$$

where: H = the height of the center of rotation of the grinder [mm]

Equating this expression with equation (6.13) yields a nonlinear first-order dynamic equation in θ:

$$\frac{d\theta}{dt} = \frac{1}{R^*}\left(\frac{\omega \mu K_S(\theta_0 - \theta) - K_2}{A K_1} - V_f \sin\theta + \frac{dH}{dt}\cos\theta \right) \qquad (6.15)$$

The desired output is the height of the trailing edge of the grinding disk, y:

$$y = H - R\sin\theta \qquad (6.16)$$

Equations (6.15) and (6.16) comprise the dynamic grinding process model. The state variable is the grinder angle θ(t). The controls are the feedspeed $V_f(t)$, the grinder fly height H(t), and the grinder rotation rate ω(t).

6.5 Verification of the Dynamic Grinding Model

The next step was to verify this dynamic grinding model by comparing simulation results to experimental results. This section will describe the simulation first, then the experimental setup, and then will show the comparison of the results.

Dynamic Grinding Simulation

A computer program was created to dynamically simulate the grinding process model in equations (6.15) and (6.16). The heart of the simulation was a numerical integration scheme for solving differential equations. The scheme used was a fourth order Runge-Kutta algorithm with an adaptive stepsize control [57]. The simulation was programmed in C using the Lattice® C compiler to run under the MS-DOS operating system on an IBM PC/XT. It required less than 128K of memory.

The inputs to the simulation were the initial two-dimensional contour, the model parameters (the spring stiffness K_S, the coefficient of friction μ, the grinding specific energy K_1, the wasted power K_2, the relaxed disk angle θ_0, and the disk radius R), the disk rotation rate ω, the feedspeed V_f, the disk initial two-dimensional position, and the finish time. The finish time could also be specified as a horizontal position on the contour beyond which no grinding was allowed; the simulation automatically stopped when the contact patch reached this point. When the electric grinder was replaced with a less-powerful air grinder, a motor model in the form of equation (3.16) was implemented. The motor model coefficients β (torque effect) and ω_0 (free rotation rate) were added to the list of input model parameters. The simulation also required an accuracy parameter which was used to control the stepsize, and needed an initial stepsize guess. Default values for all input parameters were read from an ASCII file, and could be easily edited both in the file with a text editor before the simulation was run and within the simulation program after the default file was read (but before the simulation actually began). The input contour was obtained via a tactile sensor (see the next section describing the experimentation for a description of this sensor) and was read from its own file in ASCII for easy review. The simulation output a similar file representing the contour after the grinding pass was complete. The input and output formats were the same so that several consecutive grinding passes on one sample could be simulated easily. It was also possible to save an intermediate contour file at any point during the simulation. This was useful for capturing the behavior of the grinding disk as it encountered local hills and valleys in the contour.

During the simulation, a graphic display was maintained showing the interaction of the disk with the contour and instantaneous values of interest such as the elapsed time, the position and angle of the disk, the grinding force, the grinding power, the volumetric removal rate, and the contact patch location and area. This was useful for observing the behavior of the grinding system in action and for monitoring the progress of the simulation. An example of this display is shown in Figure 6-6.

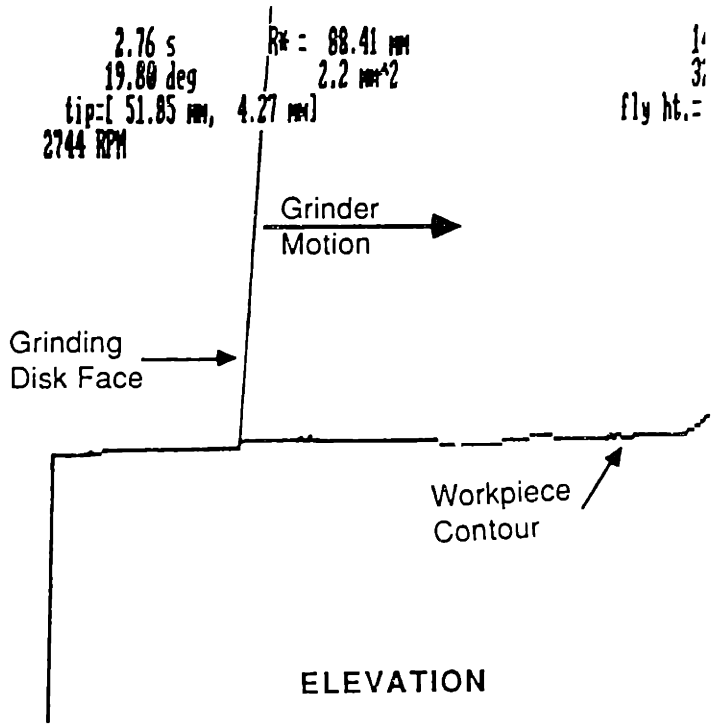


Figure 6-6 Graphic Simulation Display

At the end of the simulation the program wrote an ASCII file containing the time history of some of these variables (the grinding force, power, volumetric removal rate, contact patch area, and the disk angle). Both the contour and output history files were readable by a plot program, with hardcopy output capabilities. The simulation program could also read the filenames for the input contour, the input default parameter, the output contour, and the output history from a list file to facilitate running batches of simulations.

The most difficult aspect of the simulation was in locating the contact patch and computing its area. It is inherently awkward to use a digital computer to simulate the geometrical interaction between the disk face and an arbitrary contour. Contours were represented by discrete points along the surface, with linear interpolation between the data points. The cutting face of the grinding disk was represented in two dimensions by a line segment drawn at the angle θ to the horizontal, of length R. During the course of

the simulation, the grinder was advanced into the contour. This necessitated a recomputation of the contact patch area using the relative position of the disk and the contour. There were four types of disk-contour relative positions: 1) disk and contour not touching, 2) contact patch extending to the tip of the disk, 3) contact patch somewhere along the disk, but not including the tip of the disk, and 4) multiple contact patches. These are illustrated in Figure 6-7.

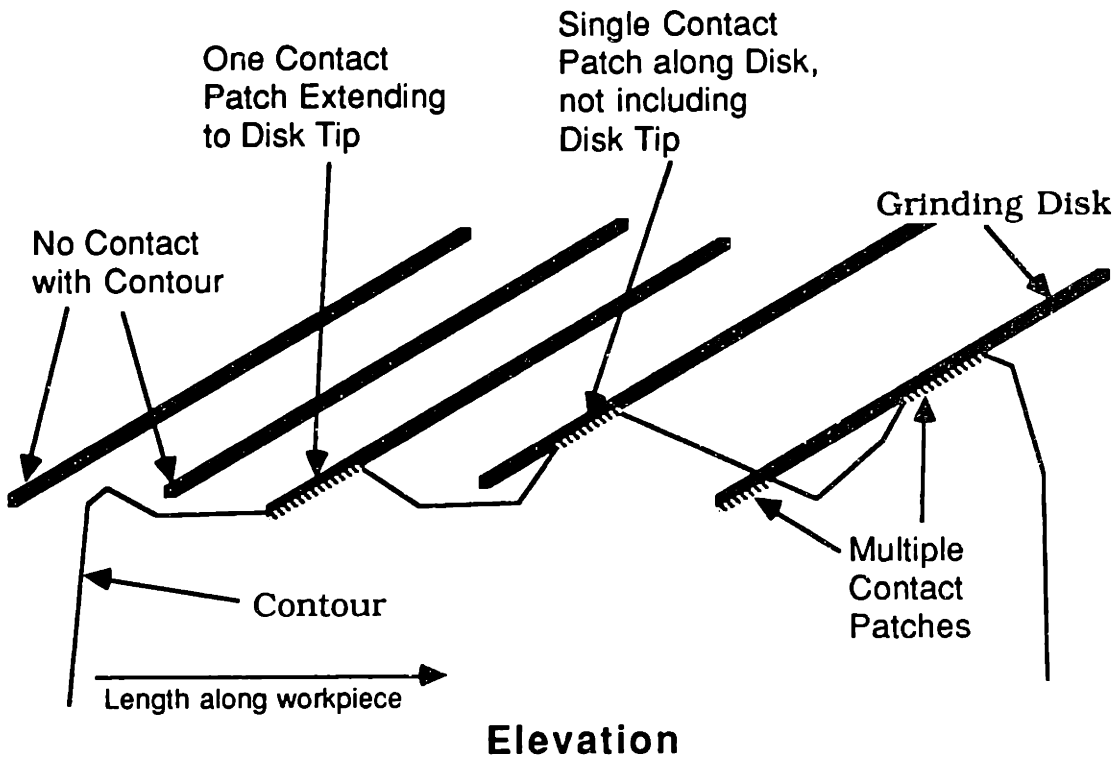


Figure 6-7 Types of Contact Patches

The complexity required to handle multiple contact patches and their relative rarity ruled out implementing such a capability. The differences between types 1, 2, and 3 correspond to the need to locate 0, 1, or 2 intersections of the contour with the disk face in order to locate the contact patch. This required software to identify the type of contact patch, and then to search over the contour for the requisite intersection(s) with the disk face. This had to be done each time the numerical integration scheme needed a dynamics computation, which, for the integration scheme used, is 11 times per integration step. Intelligent search methods were employed to speed up processing time. It was found that techniques such as computing the contact patch area only once per integration step or estimating its value for some of the computations reduced accuracy to unacceptable values and even caused some integration stability problems.

The integration algorithm paused from time to time and updated the contour shape and the graphic screen display. The contour shape was updated by vertically truncating every point on the current contour down to the corresponding point directly beneath it on the disk face. Truncation was also performed upon points between the current location of the disk tip and the disk tip location from the last truncation.

It was found that the simulation would chatter if truncation were performed after each integration step. This was due to the Runge-Kutta integration scheme which computed the dynamics at several (the same 11 dynamics evaluations mentioned above) intermediate positions before actually updating the disk location. When there had been a recent truncation, some of the trial positions were likely to have little or no contact patch area, while others had substantial contact patch area. This would confuse the algorithm, causing it to settle uncertainly upon a lower value of contact patch area than that of the previous integration step. The smaller area then caused the disk to swing faster and farther into the contour in order to maintain a similar volumetric removal rate. This, in turn would result in a larger contact patch area than was computed by the last integration step, so the algorithm would slow the disk swing down. As the algorithm hunted for an equilibrium disk angle in this manner, the disk face would cross back and forth across the discontinuity in the contour formed at the old contact patch location where the contour had been truncated last, and would never settle down. The result was an inaccurate contour. The adaptive stepsize algorithm, meanwhile, would do its best to maintain accuracy, but there is no numerical integration scheme that can handle such discontinuities well.

Experimental Verification

The dynamic model was verified experimentally by Kurfess [58]. A schematic of the experimental apparatus is shown in Figure 6-8. This equipment was slightly modified for demonstrating the hierarchical control system described in this thesis.

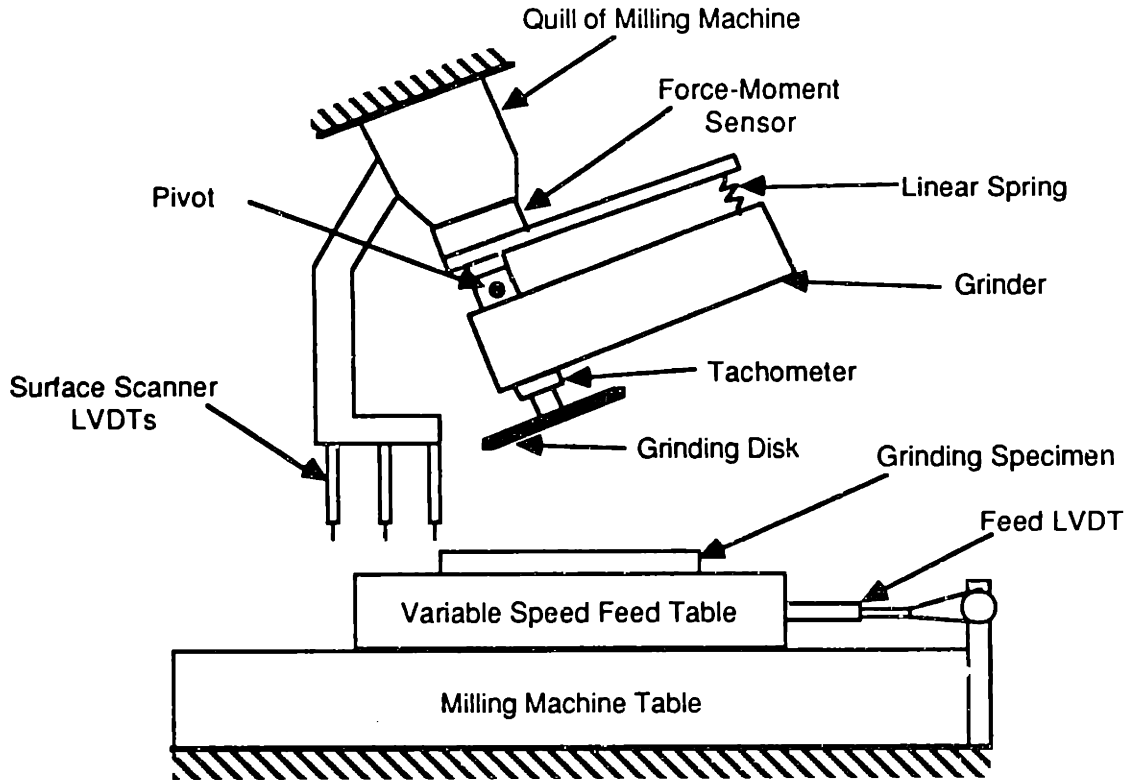


Figure 6-8 Experimental Setup for Grinding Model Verification

The hardware consisted of a grinder mounted in a milling machine-based test stand. An Astek FS6-120A 6-axis force/torque sensor was mounted in the milling machine quill. Attached to that was an engineered rotational compliance in the form of a fixed pivot and a calibrated linear spring mounted on a moment arm. This was the implementation for K_S. The grinder was mounted onto the other side of this compliance. The grinding disks used were 7 inch diameter $\times$ $^1/_4$ inch thick NorZon III QBDA resinoid bond grinding disks manufactured by the Norton company. The disk rotation rate was measured with a magnetic tachometer mounted on the grinding disk shaft. The milling machine quill was rotated at an angle so that θ_0 was 20°. The weld bead was simulated with a bar of cold rolled steel having a 0.5 inch square cross section and a Rockwell B hardness of 93.

The use these bars rather than real weld beads was a convenience, and was not due to any fundamental limitations of the grinding model or hardware. The fixed width of this workpiece eliminated the need to measure the weld bead width and to simulate a variable weld bead width so that a two dimensional model was valid. Todtenkopf described a robot based grinding system capable of measuring and grinding three-dimensional weld beads [59]. The measuring system was a structured-light vision system that could handle non-rectangular cross-sections. However, this system could

only handle planar parent surfaces and roughly linear weld beads. His work was primarily a demonstration of sensor technology and integration, and low frequency robotic force control. His work paralleled that of this thesis, and is compatible with the sequence and pass planners described here. However, for efficiency the pass planner in this work and the control hardware was developed separately, and designed for workpieces with rectangular cross sections. A modification in the pass planner for varying average weld bead width could be easily implemented.

For this thesis, the workpieces were mounted onto a variable speed feed table whose position was measured with a linear variable differential transducer (LVDT). During grinding the grinder remained stationary, tilting slightly about the pivot, while the sample was fed past it. Before and after each grinding pass the surface profile was measured in three dimensions using a tactile surface scanner consisting of three vertically mounted LVDTs. The workpiece was passed beneath the scanner with the LVDTs lightly dragging across the surface, one LVDT passing down the center and the other two along each edge of the workpiece. This scanner was retracted during grinding.

With this equipment it was possible to measure workpiece profiles before and after a grinding pass, and measure the grinding force in three dimensions and power consumed during a grinding pass. The force sensor data yielded the normal and tangential forces F_n and F_t, so an estimate of μ was possible. The grinding power was computed via the tachometer's measurement of ω, the known disk radius R, and the tangential force F_t. The volumetric removal rate was computed using the surface scanner data from before and after the grinding pass. The grinding power required at a particular time was related to the volume of material removed at a particular distance along the contour via the feed table position measurement. Several grinding power vs. volumetric removal rate data points were used to estimate K_1 and K_2 via a least-squares fit to a straight line. Details of these calculations can be found in [58]. It was found that the force data were quite noisy due in part to imperfectly balanced grinding disks, so most data had to be low-pass filtered heavily. Coefficients of friction μ ranged from 0.3 to 0.37. The specific grinding energy K_1 was estimated at 12.8 J/mm^3, and the wasted energy K_2 at 114 W, with a coefficient of correlation of 0.81. The estimates of these coefficients and the input contour were then input to the simulation to see if it could repeat the results of the grinding pass. Typical results are shown in Figure 6-9 below.

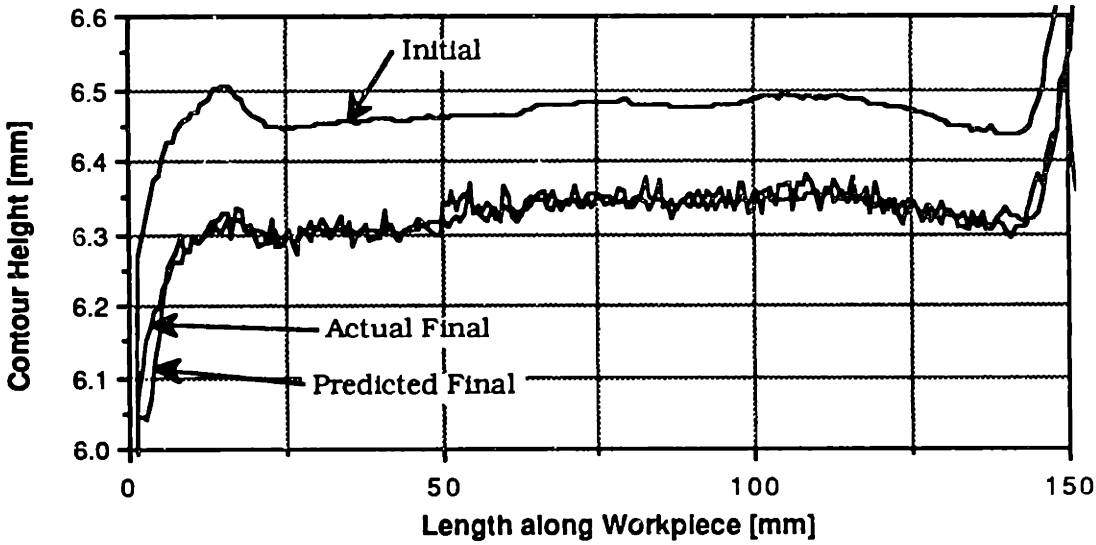


Figure 6-9 Typical Profile Comparison

It was found that the dynamic simulation predicted the final contour to within 10% of its actual value, once an allowance was made for the wear of the grinding disk. Most of the error was due to the implementation of the simulation (integration and roundoff errors, mostly) and not to the grinding model. However coefficients did vary significantly from grinding pass to grinding pass, and this would have caused significant contour shape prediction errors if the data from previous grinding passes were to be used to estimate the grinding coefficients before a grinding pass. Thus the results of a grinding pass are not perfectly predictable, and any controller that uses the simulation for planning individual passes will not achieve the desired volumetric removal rate. This is one of the reasons why the grinding sequence planner needed to plan with the possibility of imperfect plan execution by the pass planner.

This section described the verification of the dynamic grinding model used by the short term controller. This verification took the form of a comparison between results from simulations and experiments. The simulations were implemented on a digital computer using a numerical integration algorithm, and required careful modelling of the geometric interaction between the disk face and the contour. Verification experiments were carried out on an instrumented grinding test stand in the laboratory using steel bars to simulate weld beads. The simulation was able to predict the results of a grinding pass with a 10% error if the grinding coefficients had been computed from data gathered from that pass, and showed errors of as much as 50% if the coefficients were estimated from the data from several previous passes.

6.6 Planning Grinding Passes via Simulation

The next step was to convert the simulation into a pass planner. This was accomplished by simulating a controller along with the grinding dynamics. This controller moved the grinder vertically, simulating a robot attempting to zero the vertical error between the disk tip and the desired disk tip trajectory as computed in Section 6.1. This is illustrated in Figure 6-10.

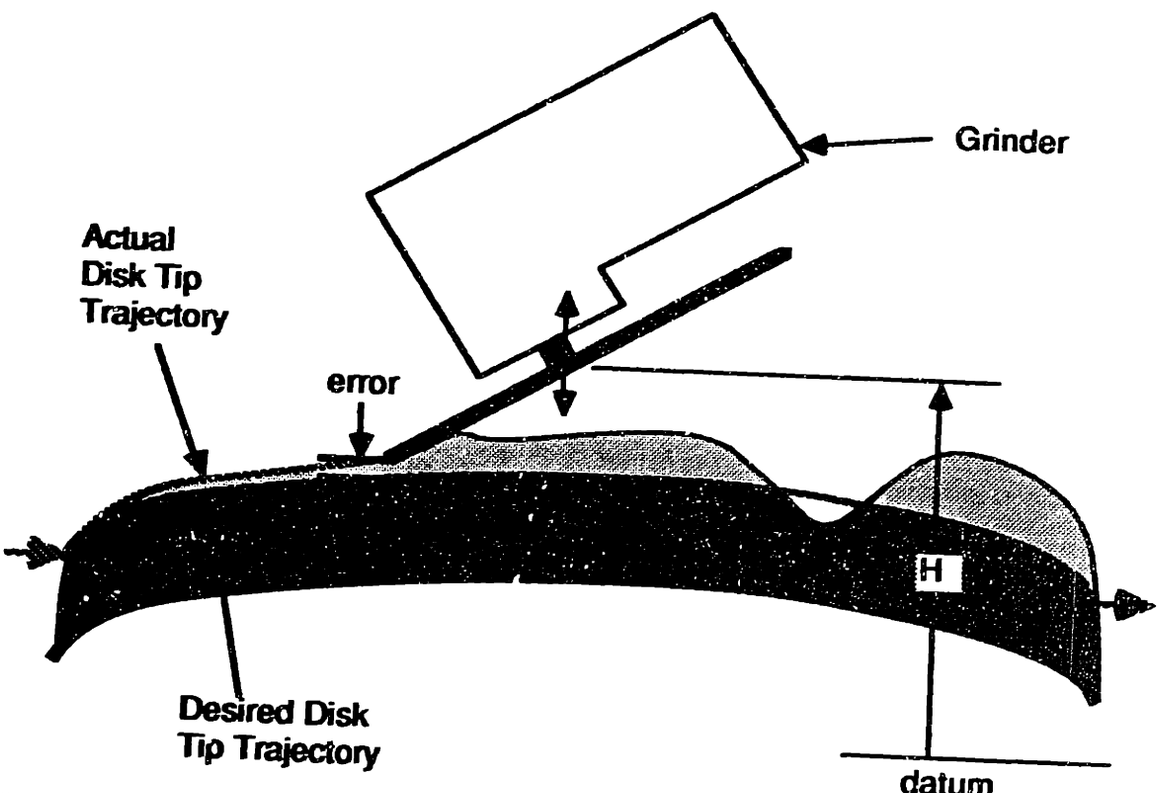


Figure 6-10 Simulation of Controlled Grinding

The disk tip location was computed from the simulation, and its elevation was compared to the elevation of a point on the desired trajectory at the same point along the contour. The resulting error was input to a PI control law moving the grinder vertically as follows:

$$\frac{dH}{dt} = G_1 \frac{de}{dt} + G_2 e \qquad (6.17)$$

where: H = the elevation of the center of the grinding disk

e = the vertical error between the disk tip and its desired trajectory

G_1, G_2 = user-selected gains

The gains were set by trial and error to obtain a very fast deadbeat controller. The resulting final contours were very close to the desired final contours. Actual grinding passes planned with this controller are given in the next section. This system worked fine except when the desired cut depth was too deep and too much power was required from the grinder. In simulations disk tip would not cut deep enough, so the controller would push down harder. This would slow the air motor down, and reduce its cutting ability, further exacerbating the situation. The controller pushed down more, until the grinder stalled. This situation was carefully avoided in experiments by limiting the power required of the grinder to 80% of its maximum power.

A variant on this scheme was tried. Rather than adjust the vertical position of the grinder, the tilt angle of the grinder was varied by changing θ_0, the relaxed disk angle. This varied the grinding force by winding up the rotational compliance. This was only simulated since there wasn't time to build the required hardware. The results were as good as those obtained in simulation by controlling the grinder height. Note that this technique was unsuitable for robotic grinding systems using a simple structured-light vision system mounted on the robot wrist along with the force sensor, such as Todtenkopf's, because these require a fixed camera angle that tilting the robot wrist would ruin.

6.7 Results of Preplanned Grinding Passes

This section will present actual expermental results of preplanned grinding passes. The first step was to verify the grinding simulation since there was an important equipment change since Kurfess' verification. Kurfess used an electric grinder, which was powerful, but heavy enough to seriously limit the effective measurement range of the force sensor. The electric grinder was replaced by a lighter weight and less powerful grinder powered by an air motor. The linear torque-speed curve of the air motor (equation 3.16) was included in the simulation and required verification. In addition, the air hose supplying the new motor made the compliance nonlinear, so it was decided to experimentally determine the compliance curve (torque vs. angle) and implement this in the simulation as a table lookup. This too needed to be verified.

Figure 6-11 illustrates the results from this verification. A grinding sample with an irregular profile was scanned and the resulting profile was input to the simulator along with process model coefficients estimated from previous grinding passes. The simulation was run with the simulated controller gains set to zero so that the grinder would remain at the same height. A comparable grinding pass was run with the force controller turned off. Figure 6-11a compares the actual end of pass contour with the simulated end of pass contour. This is a side elevation view of the workpiece. Grinding begins at the left side of the figure and proceeds to the right. A portion of the contour after the 175 mm position was not ground to provide a fiduciary point for comparing contours accurately. (The software considered the weld bead to start at 0 mm and end at 150 mm.) The simulation and the actual grinding pass shared the same initial contour. The actual and simulated contours match quite well. Figure 6-11b compares the actual grinding force to the simulated grinding force. The actual force is noisy, but is well predicted by the simulation in its low-freqency content. Figure 6-11c compares the actual air motor rotation rate response to the simulated response. Again, the simulation accurately predicts the actual grinding pass.

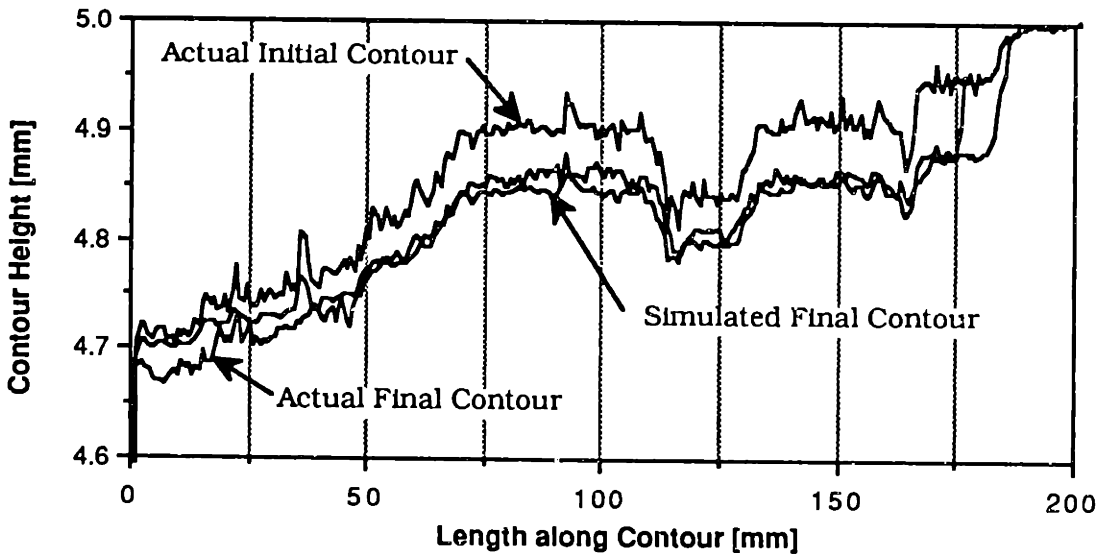


Figure 6-11a Simulation Verification: Workpiece Profile Comparison

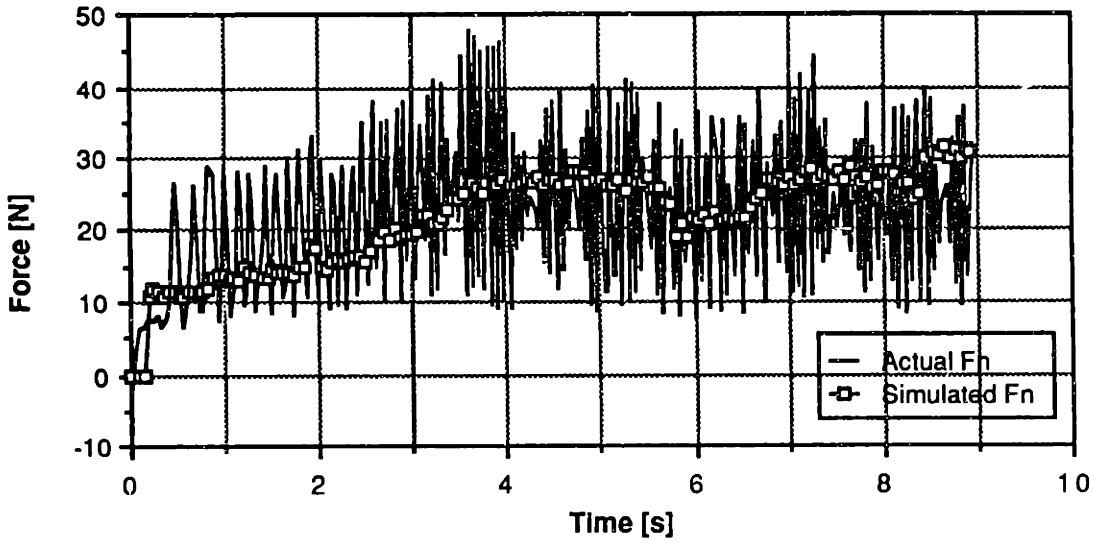


Figure 6-11b Simulation Verification: Grinding Force Comparison

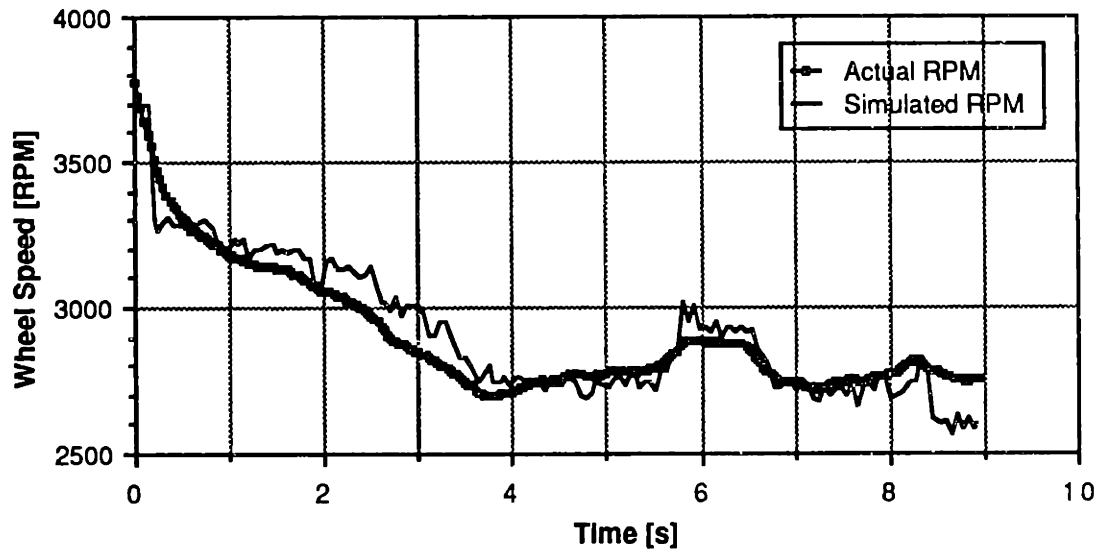


Figure 6-11c Simulation Verification: Air Motor Response Comparison

With the simulation verified, the next step was to execute a preplanned grinding pass. Again, the before-pass contour was measured and input to the simulation. This time, however, the simulation was given the desired final contour and run with non-zero gains, so that the simulated controller would attempt to generate this desired final contour. The desired final contour was generated using the techniques described in Section 6.1 in order to yield the volumetric removal suggested by the SDP program for this particular initial contour. Thus the following results are indicative of how well the

bottom two levels of the control hierarchy follow the advice given by the top level planner. Figure 6-12a is a comparison of the contours before and after the pass. The contour marked simulation is both the desired contour requested of the simulated control system and the simulated final contour; the overlap of the two contours indicate that the individual pass planner did do its job well. Note that these two contours coincide with the contour before the grinding pass from 0 mm to 42 mm along the weld bead; no grinding is intended here.

However, the actual contour after the grinding pass indicated that grinding did occur in this region. Figure 6-12b shows why. This figure illustrates the desired grinding forces as output by the individual pass planner along with the actual forces that occurred during the grinding pass. The force control system did not execute the grinding pass well. Note that the force sensor indicates that the grinding force was oscillatory about zero force for the first two seconds, positive for half of a cycle and*negative* for the other half of a cycle. The negative force indicates that the grinder was pulled (or accelerated) towards the workpiece. This is unexplained. This makes the utility of the force sensor in a vibratory environment questionable. However, the contour indicates that grinding did take place during this period. Note that the force response shows a didtinct phase lag behind the desired force. This is due to the fact that the force controller's bandwidth had to be reduced to reduce the effect of noise upon the control response. This is described in more detail in the next chapter. Figure 6-12c shows that the motor response no longer matched the simulated response. Part of this is due to the fact that the motor model ignores some motor dynamics, and part of it is due to the fact that the actual torque did not match the simulated torque. This demonstrates that the results of grinding passes are unpredictable, in this case, due more to poor execution rather than to poor planning.

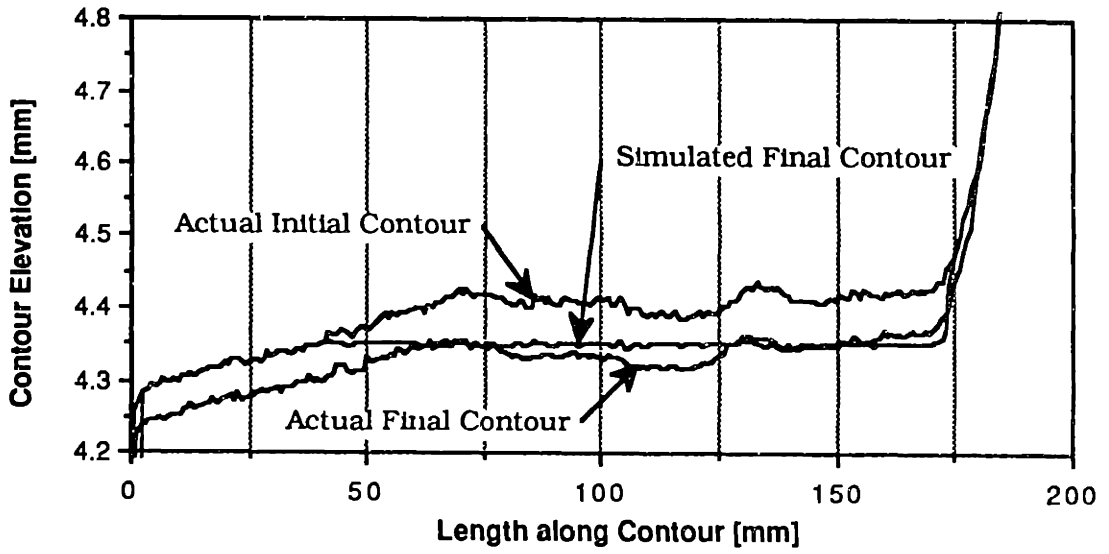


Figure 6-11a **Preplanned Grinding: Workpiece Profile Comparison**

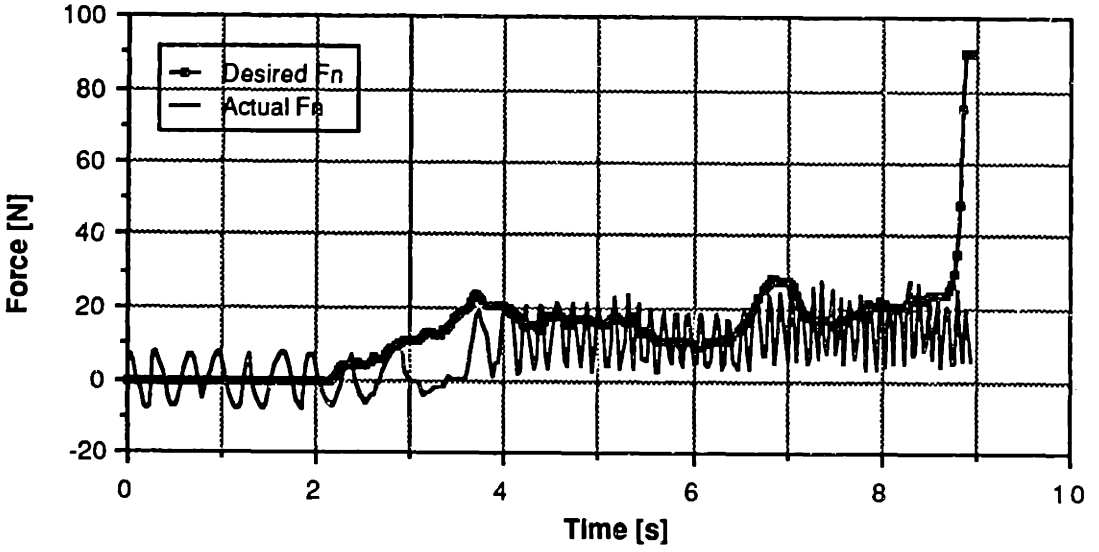


Figure 6-12b **Preplanned Grinding: Grinding Force Comparison**

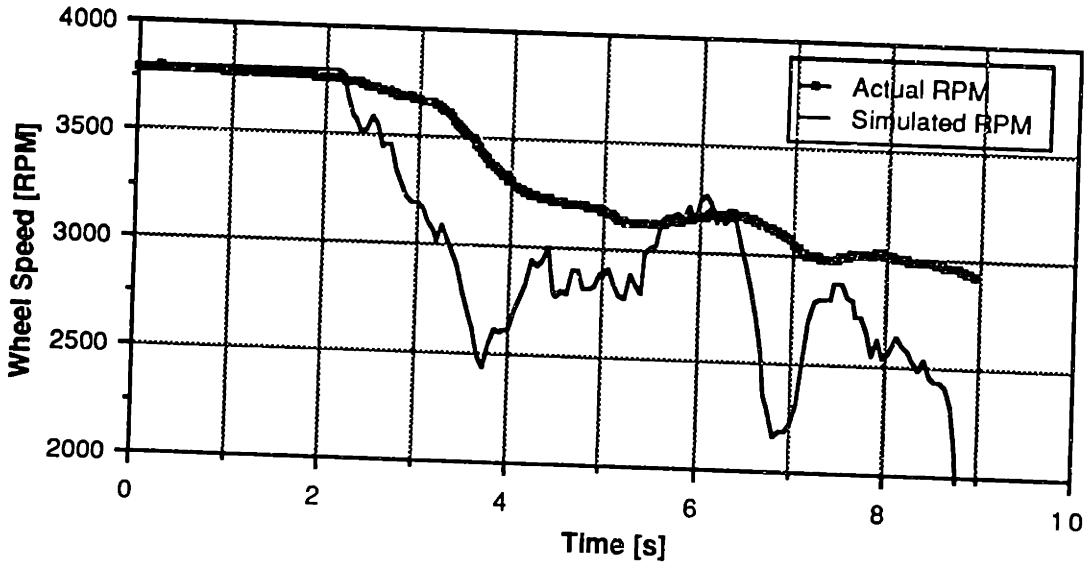


Figure 6-12c **Preplanned Grinding: Air Motor Response Comparison**

6.8 CHAPTER SUMMARY

This chapter described the development, implementation, and results from the individual pass planner. This planner was based upon a simulation of a dynamic grinding model. This dynamic model was an extension of a static model, modified to include the effects of the two-dimensional geometrical interaction of the rigid grinding disk with the weld bead. The first step to planning an individual grinding pass was to determine the shape of the final contour to yield the volumetric removal rate suggested by the top level planner. The second step was to use a simulation of a controlled grinding process to generate the corresponding grinding force trajectory. The simulation of the controlled grinder was necessary because it could determine the shape of the weld bead during 'grinding' and adjust the grinding force to obtain the desired shape. It was not possible to measure the shape of the weld bead quickly enough during actual grinding to enable effective control. The results indicated that the individual pass planner worked as planned, but that the execution of the desired force trajectory did not go as planned. The next chapter will show why this occurred.

7. CONTROLLING THE GRINDING FORCE

This chapter will describe the implementation of the very short term force-control system, the lowest level of the control hierarchy. The description is in four parts: a description of the elements of the force control system, a description of the dynamic model of the force control system, the development of the controller, and the performance of the system.

7.1 THE FORCE CONTROL SYSTEM

This section describes the components the force controller. These were a vertical position actuator, a compliance through which the position actuator generates the force, the force sensor, and the computer in which the controller was implemented.

The actuator for this controller was a box mounted between the variable speed feed table and the workpiece. The top of this box was a platform whose vertical position was precisely controlled. This is shown in Figure 7-1 below. See also Figure 6-8 for a diagram of the entire experimental apparatus.

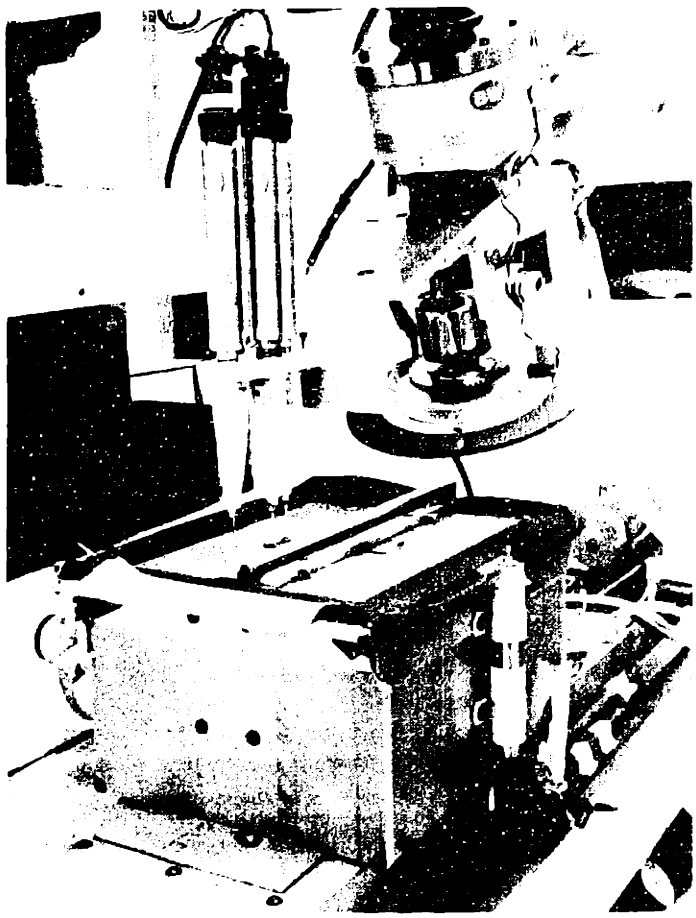


Figure 7-1 The Force-Controlled Grinding System

The platform was mechanically constrained to move only in the vertical direction by precision linear slides, and was quite stiff to any other sort of motion, either side-to-side translations or rotations. This platform was driven by a low-backlash ball-screw arrangement that was, in turn, driven by a low inertia printed-circuit moving-coil DC motor (Motocraft MCM-1040). This motor appears partially obscured on the left side of the box in the photo. The DC motor was controlled closed-loop by an analog controller (ElectroCraft LA5600) using feedback from both a tachometer mounted on the motor shaft and a position sensor (LVDT) measuring the vertical position of the platform. The position LVDT is the vertical cylinder mounted on the right side of the box. The motor controller itself was controlled from an IBM PC/XT computer (not shown) via a digital-to-analog (D/A) output. The computer received force feedback via an RS-232 serial communications line from the 6-axis force sensor (Astek FS6-120A-600) mounted between the milling machine quill and the

rotational compliance. The force sensor is at the top of the photo, with the letters "BW" visible.

The platform pushed the workpiece against the air-powered grinder, visible in the background of the photo. The resulting grinder rotation was opposed by the rotational compliance, mounted between the grinder and the force sensor, generating the force required for grinding. Also visible in the photo are the horizontal feed table, upon which the box is mounted, and the surface profile scanner, consisting of the three vertical LVDTs in the foreground mounted on the L-shaped bar. The vertical position actuator was designed and constructed for this thesis; the remainder was constructed by Ivers[50] and Kurfess [58]. The force control system comprises the vertical table, its drive motor, the motor controller, the rotational compliance, the force sensor, and the computer. The system block diagram is shown in Figure 7-2 below.

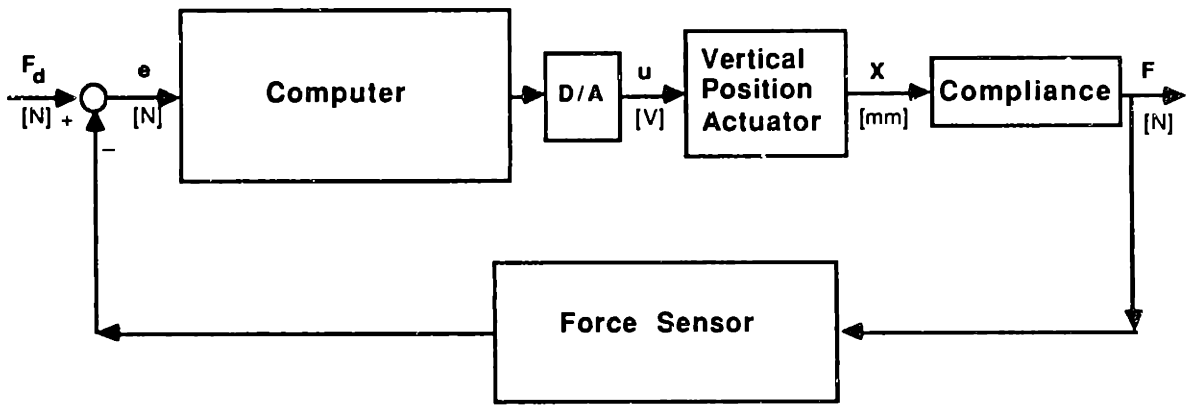


Figure 7-2 Force Control System Block Diagram

7.2 THE FORCE CONTROL SYSTEM MODEL

Modelling the force control system consisted of modelling each of the subelements individually, and then modelling how they interact. The plant controlled by the computer consisted of the vertical position control loop, the compliance, and the force sensor. These are described separately in this section.

The Compliance

The rotational compliance was well-modelled with a linear rotational spring model. This was experimentally verified by accurately rotating the grinder and measuring the torque. The data from this experiment was fit to a straight line via a least-squares analysis, and is shown in Figure 7-3 below. The compliance could be

adjusted by moving the linear spring to a position with a greater moment arm, but was typically left at 6×10^5 N·mm/rad.

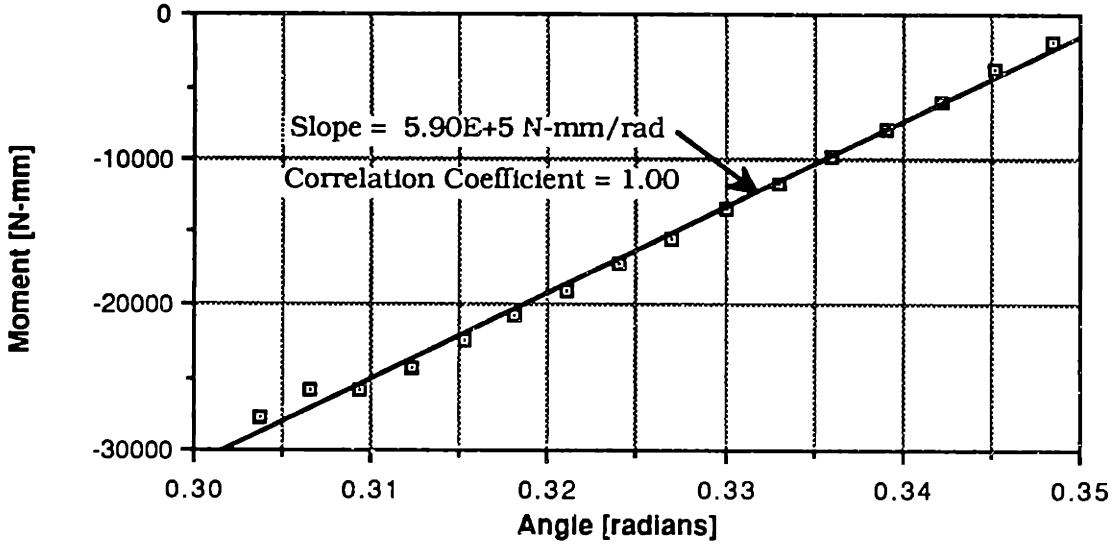


Figure 7-3 Rotational Compliance Model Curve Fit

Since the rotational motion of the rotational compliance was in fact quite small, it was possible to linearly relate the grinding force to the platform position (assuming constant contact between the workpiece and the grinder). The geometry is shown in Figure 7-4.

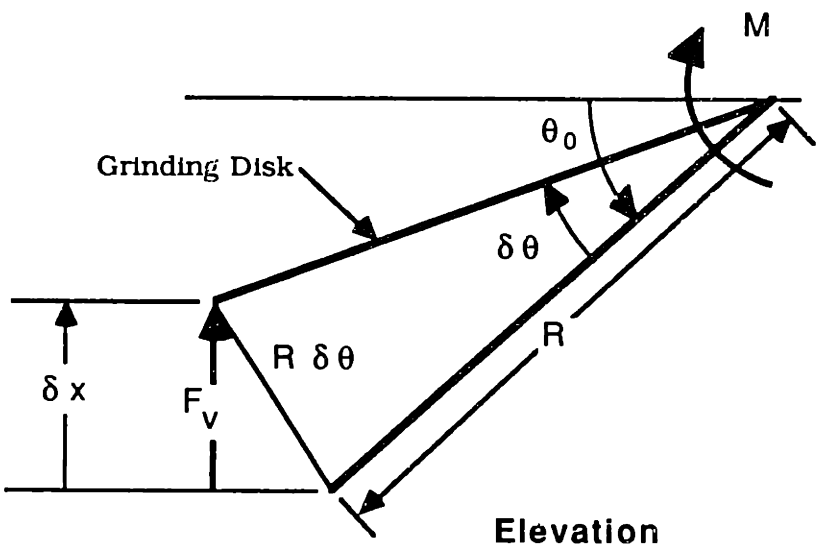


Figure 7-4 Geometry Relating Small Motions of Grinder and Platform

In the figure, θ_0 represents the disk angle for which the rotational compliance is undeflected . The corresponding platform position is X_0. The platform moves up by an amount δX, and the grinder rotates a corresponding amount $\delta\theta$. For small rotations $\delta\theta$, the platform deflection δX is given by:

$$\delta X = R \cos\theta_0 \, \delta\theta \tag{7.1a}$$

and the linear stiffness of the grinder seen by the table is given by:

$$K_X = \frac{K_\theta}{R^2} \tag{7.1b}$$

where: K_θ = the rotational stiffness, [N·mm/rad]

K_X = the linear stiffness, [N/mm]

R = the disk radius, [mm]

The Vertical Position Control Loop

The vertical position control loop was modelled using models of the motor and its analog controller obtained from published data and experimentation. See Appendix A for details. The resulting model was:

$$\frac{X}{X_c} = \frac{K_G(s+z)}{s^3 + \alpha_1 s^2 + \alpha_2 s + \alpha_3} \tag{7.2}$$

where: X = the actual platform position [mm]

X_c = the commanded platform position[mm]

and the coefficients K_G, z, α_1, α_2, and α_3 were constants, and functions of the published data and experimental results. This in turn was well-approximated by a system with unit steady-state gain and two poles, one at 9 Hz, and one at 40 Hz:

$$\frac{X}{X_c} = \frac{\beta_1\beta_2}{(s+\beta_1)(s+\beta_2)} \tag{7.3}$$

where: $\beta_1 = 2\pi(9\ \text{Hz})$

$\beta_2 = 2\pi(40\ \text{Hz})$

This model ignores high frequency details of both the model described by equation (7.2) and of unmodelled high frequency dynamics such as the platform resonances in various tilting modes which occurred in the vicinity of 80 Hz and 400 Hz and a 220 Hz vibration due to backlash in the right-angle gearbox connecting the motor shaft to the ball-screw shaft. In addition, it was discovered that there was significant stiction in the DC motor. This stiction caused the motor to hesitate slightly during direction

reversals, and was verified experimentally (via a spin-down test) and by simulation to be equivalent to about 4.8 N in grinding normal force. The motor friction model is shown in Figure 7-5 below.

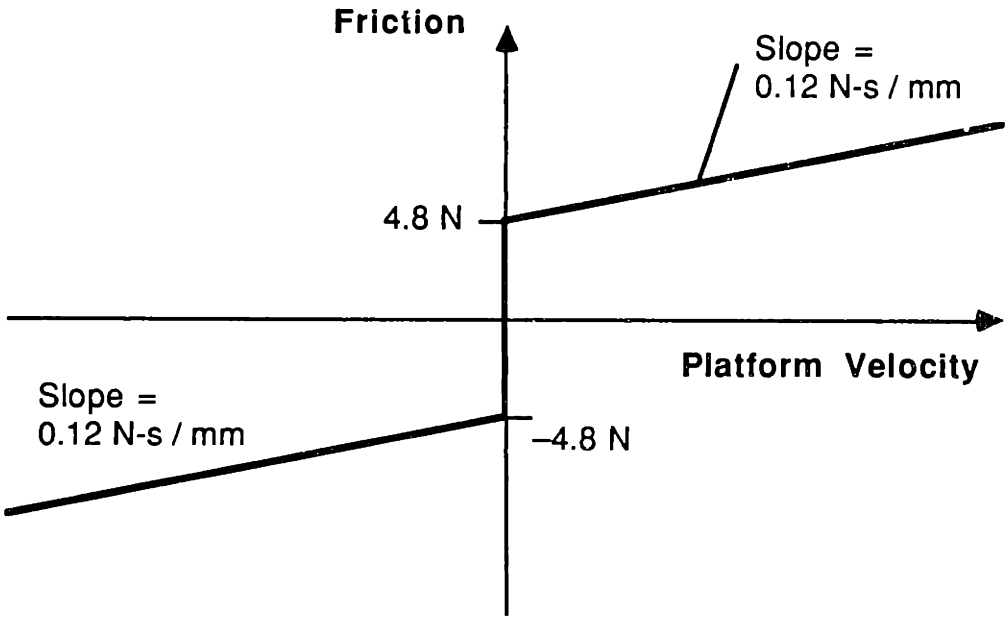


Figure 7-5 DC Motor Friction Reflected to the Position Actuator Output

The Force Sensor

The force sensor dynamics were determined by its construction. The heart of the force sensor was an X-shaped piece of metal. The center of the X was connected to the grinder via the rotational compliance. The four arms were connected to the milling machine via a circumferential ring. Strain gauges mounted along the arms measured the stress in each arm. The signals from the strain gauges passed through a 120 Hz 3-pole Butterworth filter, and then were sampled at 480 Hz by a microprocessor inside the force sensor. This microprocessor further filtered the signals with a programmable digital 2-pole low-pass filter, then multiplies the signals by a calibration matrix to obtain the three force and three moment components. These are then output to the RS -232 port at a user-selectable rate. The bandpass of the digital filter is automatically set to satisfy the Nyquist criterion for the data output through the RS-232 port.

Little is known of the overall force sensor dynamics. For example, the manufacturer knows of no frequency response tests of the 6-axis force sensor used in this thesis. The force control algorithm was operated at a 60 Hz sample rate; this was the sample rate selected for the force sensor as well, so the programmable digital filter within the force sensor was automatically set to have a 30 Hz bandpass. Therefore the

dominant dynamics of the force sensor were assumed to be those of the digital low-pass filter, i.e., that of a 2-pole 30 Hz low-pass filter.

7.3 DIGITAL FORCE CONTROL DESIGN

The force controller was implemented digitally in the computer. The design methods and implementation were straightforward and conventional. The input to the controller was the grinding normal force measured by the force sensor. This was compared to the desired force to yield a force error, and this error was input to a discrete implementation of a proportional-plus-integral (PI) controller. The output of the controller was a command voltage that was sent to the motor controller via the D/A port.

The PI controller was selected for several reasons: 1) it could be easily implemented in a discrete form, 2) it was simple, and therefore could be quickly executed in the computer, 3) it would have zero steady-state error, 4) it could partially overcome the effects of the stiction. The transfer function description of the PI controller is:

$$U(s) = \left[K_p + \frac{K_I}{s}\right] E(s) \tag{7.4}$$

where: $U(s)$ = the frequency domain representation of the input to the motor controller

$E(s)$ = the frequency domain representation of the error between the desired and actual force

K_p = the proportional gain

K_I = the integral gain

The Heaviside operator s used in the description of the continuous controller's dynamics was replaced by its Tustin approximation in discrete dynamics [60]:

$$s = \frac{2}{T}\frac{z-1}{z+1} \tag{7.5}$$

where: T = the sample time period, = (1/60) second

z = the time delay operator

The resulting control algorithm was easily implemented in the digital computer as:

$$\xi_{k+1} = \xi_k + e_k$$
$$u_k = (K_I T)\,\xi_k + \left[K_p + \frac{K_I T}{2}\right] e_k \tag{7.6}$$

where: u_k = the input to the motor controller at sample k

e_k = the error signal at sample k

ξ_k = an auxiliary variable

The resulting force control system block diagram is shown in Figure 7-6 below.

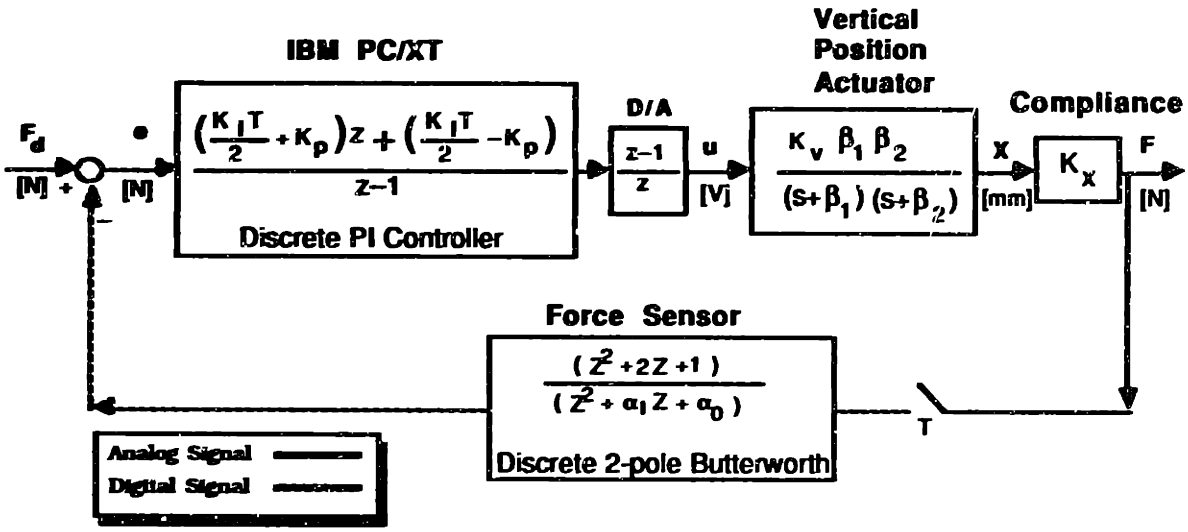


Figure 7-6 Force Control System Block Diagram

The controller gains were initially selected via analysis and simulation of the linear dynamics above, neglecting the effects of the right-angle gear box backlash and the motor stiction. However, when this was implemented, the stiction effects produced an unsatisfactory response. As a result, the gains had to be selected by trial and error. A typical step response of the system is shown in Figure 7-7. Note that this data was obtained with the grinder motor off, and a lateral bearing placed between the grinding disk and the platform to allow the grinder to tilt freely.

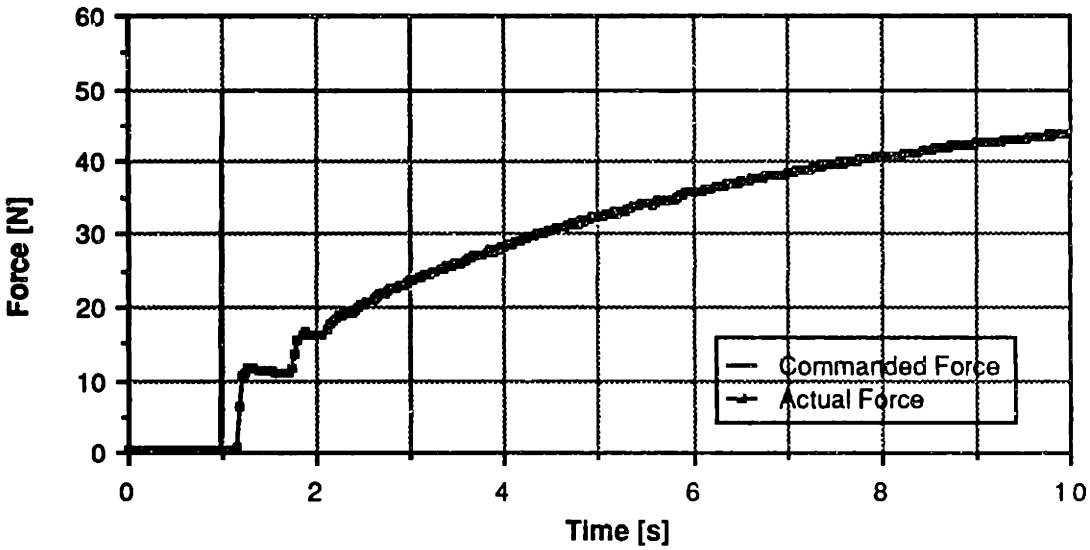


Figure 7-7 Force Control Step Response (No Grinding)

When operating with a rigid disk, the grinder added a lot of noise to the force signal. The dominant frequency component of this noise occurred at the rotational frequency of the disk and was due primarily to the disk surface not being perpendicular to the shaft The magnitude of the noise component corresponded very well to the magnitude of force that would be generated if the platform were displaced a distance equal to the disk face runout. It was very difficult to mount the rigid disks square to the shaft. After much effort, it was only possible to reduce the magnitude of this component of the noise by half. (This noise did disappear when the rigid disk was replaced by a flexible disk, but the planner was designed specifically for the rigid disk.)

When the air motor was used, its rotation rate varied from 50 Hz down to as low as 20 Hz when grinding. Thus the bandwidth of the force controller had to be reduced to as low as 0.2 Hz in order to eliminate the effect of this noise on the controller. If the bandwidth of the force controller were higher, its response to the oscillatory noise input would be 180 degrees out of phase, leading to instability as the delayed response would add to the noise force.. Therefore the bandwidth of the force controller had to be kept quite low. This, in turn, led to poor execution of grinding plans. Figure 7-8 illustrates the step response of the force control system.

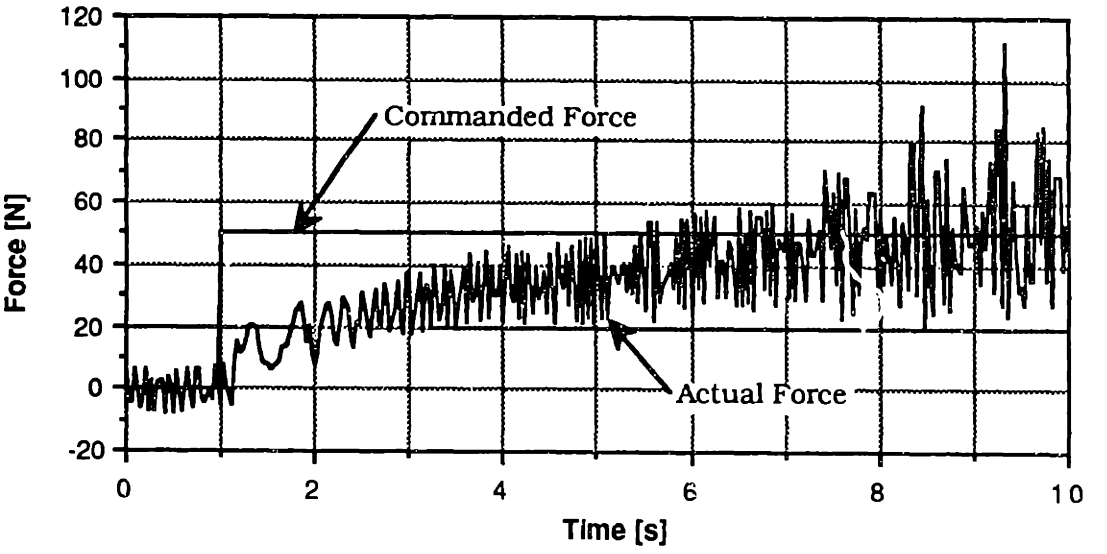


Figure 7-8 **Force Control Step Response (During Grinding)**

8. CONCLUSIONS AND RECOMMENDATIONS

8.1 THESIS SUMMARY

This thesis describes an intelligent system for planning weld bead grinding. The intelligence was embodied in the planning and control system. The requirements for the weld bead task are that an arbitrary three-dimensional weld bead must be ground off so that it be undetectable from the parent material within the time constraints of the factory and without burning the weld or the parent material. Being undetectable implies that the final weld bead contour blend in with the contour of the parent material and that the final surface finish be not too coarse yet not too smooth. The three-dimensional shape requirement refers to both the path that the weld bead traces across the parent material and the shape of the weld bead itself. The former implies that a robotic grinding system be employed. The latter makes demands upon the sensor technology and the grinding process model. A vision system capable of measuring the three-dimensional contour of a weld bead has been demonstrated. The limitations on the force and feedspeed capabilities of the robot and the requirements that the robot reset its position and perform computations between passes are also constraints on the grinding system. The grinding sample and process model have been restricted to two dimensions for simplicity. The grinding process is incapable of removing typical weld beads in one pass, due in part to the no-burn constraint, and is poorly predictable , due in part to inadequate models and in part to uncontrolled random variation of process parameters. The need for multiple grinding passes, the poor predictability of those passes, the limitations of the robot, and the task requirements make planning and controlling weld bead grinding a formidable problem.

Controller Hierarchy

The nature of the weld bead grinding task made the planning and control job suitable for a three tier hierarchical control system, with the longest term planning function at the top of the hierarchy and shortest term feedback control system at the bottom. The top level determines the number and length of each grinding pass and the volume of material to be removed during each pass given the time remaining until a deadline and the initial weld bead volume. The middle level planner plans the details of each grinding pass, determining the grinding force trajectory required to satisfy the

top level planner's volumetric removal requirements for each pass. The bottom level then executes the plans made by the upper levels, grinding with the feedspeed commanded by the top level planner and executing the normal force trajectory planned by the medium level planner.

Top Level Planner

The problem faced by top level planner was formulated as an optimization problem, with penalties assigned to grinding sequences which do not satisfy the task requirements or which violate process constraints. This planner takes into consideration the random behavior of the weld bead grinding process via a stochastic grinding model that yields the probability density function of the volume of the weld bead after a grinding pass as a function of the controls applied during the grinding pass. This model was based on a static, deterministic grinding model. The poor predictability of grinding passes requires the planner to think in terms of possibilities of satisfying the task requirements and risks of violating the constraints, and must be able to replan grinding sequences that are likely to turn out other than planned. This planner plans sequences of grinding passes that balance the task requirements against the risks in a table-lookup form that does not need to be replanned when grinding passes go awry. A system capable of making such a decision can be called intelligent.

The stochastic results of grinding passes were handled by computing the expected penalty of the entire grinding sequence. The multi-pass nature of the weld bead grinding task made the planning problem faced by the top level ideal for implementation using Stochastic Dynamic Programming to optimize the expected results of the grinding sequence. The solution is generated in a table-lookup form, giving the optimal grinding policy as a function of the initial time and amount of material remaining. This table lookup allows the planner to operate in a feedback control mode, giving advice for the next pass only after the results from the previous pass are known, rather than planning out an entire grinding sequence in advance. Of course, whenever the grinding process changes significantly, the entire solution must be regenerated. However, this is not expected to occur frequently for grinding, because the most common changes in the grinding process—the disk sharpness and the material grindability—vary slowly with time.

The stochastic dynamic programming algorithm was implemented on an IBM PC/XT microcomputer and programmed in the C language. Its results were verified for simple problem formulations by comparing them with the results obtained from

detailed analysis of the simplified problem. The full problem formulation (minus the surface finish constraint) was solved.

Middle Level Planner

The middle level planner first computes the contour after the grinding pass that will result in the removal of the volume of material required by the top level planner for each pass, then computes the force trajectory needed to do this. This level uses a more detailed grinding model, capable of predicting the grinding system's behavior during one pass over a short enough time period to accurately predict the resulting shape of the weld bead given the grinding force trajectory used during the grinding pass. This dynamic model is an extension of the static grinding model, and is based on fluid mechanics principles. It was experimentally verified to yield the contour shape after the grinding pass. The middle level planner uses a simulation of this model plus an omniscient controller to generate the force trajectory required to yield the required final contour. The simulation was also implemented in C on the same IBM PC/XT computer, and was complex due to the complexity of digitally modelling the geometric interface of the grinding disk face with the weld bead.

Bottom Level Controller

The bottom level controlled the grinding force via a known compliance by controlling position. In the experimental setup, the grinder was mounted on a rotational compliance and the workpiece was pushed up against the grinder. The position of the workpiece was controlled with a DC motor-based actuator built for this purpose, and the actuator was controlled from the computer through a D/A port and an analog motor controller. The grinding force was measured by a 6-axis force sensor, and the force control was implemented in the computer using conventional digital control techniques. It turned out that rigid disk grinding is a very noisy process and is quite difficult to control. The control problem was made more difficult by the fact that there was significant stiction in the actuator and that no force was generated when contact between the grinder and the workpiece was lost.

8.2 CONCLUSIONS

The planning and control scheme described above was tailored to the specific structure and needs of the weld bead grinding task. However, many of the techniques used are applicable to the planning and control jobs for other processes. In particular,

the quadratic nature of the penalty function, and its evaluation as an expected penalty for stochastic processes should be applicable to most real processes whose output can be measured more or less continuously (i.e. not discretely, such as in an assembly process where parts can be described as either assembled or unassembled) and is random. The control hierarchy is novel to grinding, and is a useful technique for adding more intelligence to the control of any process. However, in order to implement the type of intelligent hierarchical control system described in this thesis, each planner must have an appropriate model for the process under its control. This means that the model must describe the behavior of the process accurately over a fine enough time scale. Just how accurate and how fine a time scale is a function of the process, and there will always be tradeoffs between the model accuracy and its utility. If the process has significant random variation, the model must be able to encompass this, and be able to describe how this variation is affected by the controls, if at all. This gives the planner control over the process variation, which might be used to advantage. It was possible to do this for the weld bead grinding problem because the variation of the grinding process' behavior was predictable via the grinding model. This thesis has shown how to use process models for planning, and how to use stochastic process models for planning unpredictable processes.

Application to Other Processes

The techniques described in this thesis can be used on a large class of processes. The basic criteria for applicability are:

1. The output of the process must be measurable, or determined from a combination of measurable quantities. Although applicable to processes with discrete outputs (such as assembly, the results of which can be specified as either "assembled" or "unassembled"), the penalty function technique is better suited for continuous outputs.

2. The process results must be somewhat unpredictable, yet modellable stochastically. The cause for the unpredictability can either be due to inaccurate models or imperfect execution. In particular, the probability density function of the output of the process must be modellable. Better models allow for better-informed decisions.

3. The dynamic programming technique is applicable to multi-stage processes where each stage can be described as a state transition. It is also suited for optimal stopping problems, in which a process can be continued

in an attempt to satisfy a certain condition, and the decision of when to stop must be made.

The following are examples of other processes where the techniques of this thesis might be applied.

Material Removal Processes

Most material removal processes can be planned using the techniques described above for grinding, although most do not seem to be as unpredictable as grinding is because they rely on the control of the position of a rigidly held cutter relative to a rigidly held workpiece. Turning and milling fall into this category. Inaccuracy results when either the tool or the workpiece deflect due to high interaction forces. Such forces can arise from high static loading, for example, when too great a cut depth is attempted, or from dynamic forces due to tool chatter or workpiece runout. Additional considerations are those of surface finish and tool wear, both of which are functions of the feeds and speeds chosen for each cutting pass. There are many models for machining. Representative examples can be found in [61-64]. Most machining models for cutting accuracy, finish, and tool wear are based upon poorly known or unpredictable metal properties or empirical constants. Predictions made using these models will be inaccurate, and their accuracy dependent upon the controls selected. Machining is generally a multi-pass operation, so it is quite similar to grinding, and the techniques used above would be directly applicable.

Bending

When shaping metals by bending, a moment is applied to the workpiece and it deflects past the elastic limit, and deforms plastically. When the moment is released, the metal springs part of the way back to the original shape in an elastic manner. The final shape is a function of the maximum applied moment and the properties of the material, including the previous strain history via strain hardening, etc. If the final shape is not the desired shape, it can either be bent again, or left as is. Therefore bending is an optimal stopping problem. Hardt [65] has modelled the roll-bending process, and has put it under a closed-loop control that is independent of the material properties with some success. He reports errors of constant radius bends of under 3% of the desired radius for the first bend.

This process can be planned using the stochastic dynamic programming technique. Each bend is a state transition, with the final shape (e.g. the final radius) being the state variable. The state transition is unpredictable either due to varying

material properties when the process is controlled open-loop, or due to the inaccurate performance of the closed-loop control scheme. When controlled open-loop, the variation of the material properties can be propagated through the bending process model to obtain the output PDF. Some modelling would have to be done to determine the output PDF under closed-loop control to see if it is correlated with the commanded shape, if anything. The validity of this propagation would have to be verified. A related problem is sheet metal die forming[66] in which a three-dimensional sheet metal die is formed by pushing an array of pins against an initially flat sheet of metal. This, too, is an iterative bending process. The bending problem fits nicely into the framework of the stochastic dynamic programming technique for process planning.

Refining

In refining the desired output is the purity of the product. In petrochemical refining the quality of the products is a function of the type of crude entering the refinery, the processing parameters such as processing times, temperatures, pressures, and flowrates, and other parameters pertaining to the processing equipment and whatever chemical catalysts may be required. The process models are quite complex, and are based on chemical engineering theory and empirical relationships. These models are typically implemented on computers which are consulted to decide the process settings required to produce a given mix of products. The process models are good, but not entirely accurate. The computer is instructed to output conservative settings, to avoid the risk of producing poor quality products. The algorithm which decides these conservative settings is in general proprietary, but is frequently based upon arbitrary safety factors. There are three options when a substandard product is produced: 1) recycle the product back through the refining process to improve the quality, 2) blend the product with higher quality product to bring the average properties up to par, and 3) demote the product to a lower standard of quality. Further decisions that depend upon the operating parameters are when to recharge the catalyst and when to service equipment[67]. In addition, the refining process is often a sequence of several different processes, each of which must be planned. Because the allowable states and decisions can vary with the stage, the dynamic programming technique is capable of planning such processes.

The above three examples show that this dynamic programming technique is quite powerful and capable of planning a variety of multi-stage stochastic processes. The next section will describe how the resulting optimal policies can be described as intelligent.

Optimal Grinding Policies

The derivation of the optimal grinding policies through detailed analysis would be very difficult if the penalties were not linear, or nonlinear constraints were imposed, and becomes increasingly inaccurate as the accuracy of the process decreases. However, the numerical Stochastic Dynamic Programming solution can easily handle such penalties, constraints, and inaccuracies, and yield quantitative results as well, at the expense of less insight and restricted by quantization of the state space inherent in Dynamic Programming (and many other numeric) solution schemes. It is interesting to note that the numerical solution yielded quantitative optimal grinding policies which could be described using the insight gained from the analysis of the simple problem. Such policies could be described using heuristic rules, such as "do not attempt to grind if it is too late after the deadline or the weld has been overground already," and "it's ok to be a little late if there's still a lot of material left," and "if there is very little material left and plenty of time, try to take most of it off in one pass," and "if there is more material left and enough time, plan a sequence of passes that are the fastest possible and finish right at the deadline with no material left," and "if there is a lot of material left and not enough time, grind as hard as possible, in order to waste as little time as possible during equipment resets." Such policies are particular to the formulation of the weld bead grinding problem given to the planner, couched in the equations, coefficients, constraints, and charge functions programmed into the algorithm. These policies may not be extendable to other formulations. It is interesting to note that theses grinding policies were not obtained from a human grinding expert nor from any type of reasoning or inference engine. However, these policies are more advanced than those typically obtained by modern control theory, and the planner that generated them can be called intelligent.

8.3 Recommendations for Future Work

1. The SDP algorithm was implemented for easy modification for research purposes, and not for speed or ease of use. There are many improvements that could be made to the algorithm to make it faster and more accurate. Noting that most of the grinding policy indicates that the controls should be set to values on the boundary of the control space, a modification of the algorithm to search only or search first along the boundary of the control space.

The search scheme was limited to discrete feedspeeds for speed and simplicity. There is no reason why a two dimensional search could not be implemented to yield continuous feedspeeds. In fact, a variety of two dimensional search schemes were implemented early in this research and abandoned. These attempts included golden section searches, Powell's method, and a Hooke&Jeeves pattern search. It was discovered later that this was due to the time-banded nature of the optimal grinding policies which made such a search difficult because few optimization algorithms work both well and robustly when the function to be optimized is discontinuous.

2. The force control with the rigid disk was quite difficult, and did not work as well as hoped for. Much of this was due to the noise generated by the rigid grinding disks which were difficult to mount perpendicularly to the grinder shaft. The resulting noise was transmitted directly to the force sensor via the pivot, and this evidently played havoc with the internal sensor computations. A better hardware arrangement would have the positions of the linear spring and the pivot exchanged, or the grinder mounted on a vertically oriented linear slide with a linear compliance. This would put a spring between the grinder and the force sensor to isolate the force sensor from much of the grinder's vibrations. A second way to improve the force control would be to increase the high frequency rolloff of the controller to allow for higher bandwidth control while limiting the effects of the grinding noise. Eliminating the stiction in the actuator motor would also help. These enhancements were not attempted due to time considerations.

3. It should be noted that quite good force control results were obtained using a flexible grinding disk, because it absorbed much of the grinding noise. However, the grinding model used in the simulation for planning the grinding passes was specifically designed for the rigid disk. It should be possible to construct a similar simulation for the flexible disk grinding. Note that because the geometry of the flexible disk is much more variable, it will be difficult to model this successfully. In particular, the position of the grinder before actual grinding could be reproduced in the simulation because it was possible to accurately locate the grinder with respect to the workpiece by touching the disk to a fiduciary point on the weld bead. This

would be difficult to do with the flexible disk because the paper disk warps, and cannot be easily modelled in two or even three dimensions, because this warping is affected by humidity and temperature and is uncontrollable and unpredictable.

However, a static grinding model may be serviceable for a pass planner. This would involve computing the desired final contour shape as described in Chapter 6, then computing the cross-sectional area of the weld bead as a function of the length along the weld bead. Multiplying this by the feedspeed would yield the desired material removal rate as a function of the position along the weld bead. This could be converted into a desired grinding force trajectory via the power vs. material removal rate equations and the rotational compliance table lookup. A table lookup would be required for the compliance because the flexible disk does not have a linear compliance, and would have to be used as the force control compliance because a softer spring would not generate enough grinding force in a reasonable angular or linear displacement. Todtenkopf used a simplified form of such a planner with some success.

4. The stochastic grinding model used in the sequence planner was not scientifically verified. The probabilistic relationship between the variation of the process model parameters and the actual cut depth was never experimentally determined, nor was the shape of any PDF determined. The relationships can be determined from experimental data using correlation and analysis of variance (ANOVA) methods. The shape of PDFs can be estimated from experimental data using either of two techniques. The first technique involves expansion by kernel PDFs. An arbitrary (usually Normal) kernel PDF is constructed for each data point so that the mean of each kernel lies on a data point. The overall PDF is a normalized sum of all the kernel data points. The second technique is to sort the data and construct a cumulative probability function by fitting a smooth curve (e.g. a Bezier or cubic spline function) through the points. The PDF is obtained by differentiating the cumulative probability function. The expansion by kernel technique is less accurate, but can be used to update the PDF as the data arrive, so could be used to generate PDFs on-line. The second technique is more accurate, but must be done at one time.

REFERENCES

1. G. Taguchi, Introduction to Quality Engineering, Asian Productivity Organization, Tokyo, 1986

2. J.J.Slotine, "Robust Control of Robot Manipulators", Int. J. Robotics Research, 4(2), 1985

3. G. Stein, M. Athans, "The LQG/LTR Procedure for Multivariable Feedback Control Design", IEEE Trans. Auto. Control, Vol. AC-32, No. 2, Feb. 1987

4. V.V. Chalam, Adaptive Control Systems, Marcel Dekker Inc., New York, 1987

5. L. Ljung, System Identification: Theory for the User, Prentice Hall, 1987, London

6. Mayne, Malkin, "Parameter Optimization of the Steel Grinding Process", Trans. ASME J. Engineering for Industry, Aug. 1976, pp. 1048-1050

7. Amitay, Malkin, Koren, "Adaptive Control Optimization of Grinding", Trans. ASME J. Engineering for Industry, Aug. 1981, pp. 103-108

8. J. Shibata, "Adaptive Control for Conveyor-Type Belt Grinder", Annals of the CIRP Vol. 29/1/1980, pp. 217-220

9. D. S. Ermer, "Optimization of the Constrained Machining Economics Problem by Geometric Programming," Trans. ASME J. Engineering for Industry, Vol. 93, 1971, pp.1067–1072

10. D. S. Ermer and S. M. Wu, "The Effect of Experimental Error on the Determination of the Optimum Metal-Cutting Conditions," Trans. ASME J. Engineering for Industry, Vol. 89, May 1967, pp..315–322

11. K. Iwata, Y. Morutsu, T. Iwatsubo, S. Fujii, "A Probabilistic Approach to the Determination of the Optimum Cutting Conditions," Trans. ASME J. Engineering for Industry, Vol. 94, Nov., 1972, pp.1099–1107

12. S. K. Hati, S. S. Rao, "Determination of Optimum Machining Conditions–Deterministic and Probabilistic Approaches," Trans. ASME J. Engineering for Industry, Vol. 98, Feb., 1976, pp.354–359

13. S. S. Rao, "Optimization of Cold Rolling by nonlinear Programming," Trans. ASME J. Engineering for Industry, Vol. 100, May, 1978, pg.186

14. K. Hitomi, "Optimization of Multistage Machining Systems: Analysis of Optimal Machining Conditions for the Flow-type Machining System," Trans. ASME J. Engineering for Industry, Vol. 93, Feb., 1971, pp.498–506

15. K. Hitomi, "Optimum Process Design for a Single Stage Multifunctional Production System," Trans. ASME J. Engineering for Industry, Vol. 103, Feb., 1981, pg.218

16. S. S. Rao, S. K. Hati, "Computerized Selection of Optimum Machining Conditions for a Job Requiring Multiple Operations," Trans. ASME J. Engineering for Industry, Vol. 100, Aug., 1978, pp.356–361

17. Edward Haugen, Probabilistic Approaches to Design, John Wiley & Sons, New York, 1968

18. Alvin Drake, Fundamentals of Applied Probability, Theory, McGraw-Hill, New York, 1967

19. Process Planning Program/GARI and TOM Tutorials, R-83-PPP-01, Computer Aided Manufacturing - International, Inc., 611 Ryan Plaza Dr., Suite 1107, Arlington ,TX 76011

20. Fuzzy Logic, IEEE Spectrum, 1984, p.30

21. Hanssmann, F. "Optimal Inventory Location and Control in Production and Distribution Network," Operations Research, 1959, Vol. 7, No. 9, pp. 483–498

22. C. Klien, Engineer General Motors Corp., personal communication.

23. H. West, Kinematic Analysis for the Design and Control of Braced Manipulators, MIT Dept. Mechanical Engineering, PhD Thesis, June 1986

24. A. Tate, Closed Loop Force Control for a Robotic Grinding System, MIT Dept. Mechanical Engineering S.M. Thesis, June 1986

25. H. Ernst, M. Merchant, "Chip Formation, Friction, and High Quality Machined Surfaces," Surface Treatment of Metals, Trans. ASM, 1942, pp. 299-378

26. M. Merchant, "Mechanics of the Metal Cutting Process. I. Orthogonal Cutting and a Type 2 Chip", J. Applied Physics, Vol. 16, No. 5, May, 1945, pp. 267-275

27. M. Merchant, "Mechanics of the Metal Cutting Process. II. Plasticity Conditions in Orthogonal Cutting", J. Applied Physics, Vol. 16, No. 6, June, 1945, pp. 318-324

28. V. Piispanen, "Theory of Formation of Metal Chips," J. Applied Physics, vol.19, 1948, pp. 876-881

29. Marshall, Shaw, "Forces in Dry Surface Grinding", Trans. ASME, Jan. 1952 pp. 51-59

30. Backer, Marshall, Shaw, "The Size Effect in Metal Cutting", Trans. ASME, Jan. 1952 pp. 61-72

31. Leone, Saibel, "The Friction Terms in Metal Cutting", Trans. ASME Feb. 1954, pp.195-198

32. Shaw, Finnie, "The Shear Stress in Metal Cutting", Trans. ASME, Feb. 1955, pp. 115-125

33. Reichenbach, Mayer, Kalpalcioglu, Shaw, "The Role of Chip Thickness in Grinding", Trans. ASME, May 1956, pp. 847-859

34. Finnie, Shaw, "The Friction Process in Metal Cutting", Trans. ASME, Nov. 1956, pp. 1649-1657

35. A. J. Sedriks, T. O. Mulhearn, "Mechanics of Cutting and Rubbing in Simulated Abrasive Processes," Wear, 6 (1963), pp. 457-466

36. L. P. Tarasov, "An Introduction to Abrasive Machining," American Machinist/ Metalworking Manufacturing, April 1962, pp. 125-136

37. C. Pollock, "The Effect of Some Operating Variables in Vertical Spindle Abrasive Machining of Mild Steel," ASME paper no.66-WA/Prod.-16

38. R. P. Lindsay, "The Effect of Conformity, Wheelspeed, Dressing, and Wheel Composition on the Grinding Performance of Conventional and High Temperature Alloys," SME paper no. MR70-803, 1970

39. Hahn, Lindsay, "Principles of Grinding...Part I Basic Relationships in Precision Grinding", Machinery, July 1971, pp. 55-62

40. Hahn, Lindsay, "Principles of Grinding...Part II The Metal Removal Parameter", Machinery, August 1971, pp. 33-39

41. Hahn, Lindsay, "Principles of Grinding...Part III The Wheel Removal Parameter", Machinery, September 1971, p.33-39

42. Hahn, Lindsay, "Principles of Grinding...Part IV Surface Finish, Geometry, Integrity", Machinery, October 1971, pp.57-67

43. Hahn, Lindsay, "Principles of Grinding...Part V Grinding Chatter", Machinery, November 1971, pp. 48-53

44. Malkin, Anderson, "Thermal Aspects of Grinding, Parts 1 & 2", Trans. ASME J. Eng. for Industry, Nov. 1974, pp. 1177-1191

45. J. Shibata, I. Inasaki, S. Yonetsu, "The Relation between the Wear of Grain Cutting Edges and their Metal Removal Ability in Coated Abrasive Belt Grinding," Wear, 55(1979) 331-344

46. K. Steffens, W. König, "Closed Loop Simulation of Grinding," Annals of the CIRP, Vol. 32/1/1983, pp. 255-259

47. T. Matsuo, K. Nakasako, "Selection of Grinding Wheels for the Snagging of Steels and Cast Iron," Milton C. Shaw Grinding Symposium of the Winter Annual Meeting of the ASME, Nov. 1985, pp. 273-287

48. C. Guangi *et al.*, "An Experimental Investigation of the Effects of Head Force on Steel Conditioning," J. of Northeast Institute of Technology, No.4, 1982, pp.63-77

49. G. Kenwood, Process Modelling of Weld Bead Grinding as Part of a Robotic Solution, S.M. Thesis, MIT Dept. Mechanical Eng., 1984

50. D. Ivers, Process Modelling of Coated Abrasive Disk Grinding as Part of a Robotic Solution, S.M. Thesis, MIT Dept. Mechanical Eng., 1985

51. Dimitri P. Bertsekas, A Course in Dynamic Programming and Stochastic Control, Chapter 2, Academic Press, 1987

52. R. S. Hahn, "Controlled-Force Grinding—A New Technique for Precision Internal Grinding," Trans. ASME J. Eng. for Industry, Aug. 1964, pp.287-293

53. R. King, R. Hahn, Handbook of Modern Grinding Technology, Chapman and Hall, New York, 1986, TJ1280.K53

54. R. Snoeys, "Grinding Chatter," Milton C. Shaw Grinding Symposium of the Winter Annual Meeting of the ASME, Nov. 1985, pp. 415-420

55. M. Weck, K. Schiefer, "Interaction of the Dynamic Behaviour between Machine, Tool, and Cutting Process for Grinding," Annals of the CIRP, 28 (1979) 1, pp. 218-285.

56. N. Goldfine, Process Analysis and Design for Grinding Robot Tool Holders, S. M. Thesis, MIT Dept. Mechanical Engineering , June 1985

57. Press *et al.*, Numerical Recipes; The Art of Scientific Computing, Cambridge University Press, Cambridge UK, 1986, §15.2

58. T. Kurfess, Verification of a Dynamic Grinding Model, S.M. Thesis, MIT Dept. Mechanical Engineering, May 1987

59. A. Todtenkopf, Vision and Force Controlled Robot Weld Bead Grinding System, S. M. Thesis, MIT Dept. Mechanical Engineering, January 1988

60. Franklin, Powell, Digital Control of Dynamic Systems, Addison-Wesley, Reading, Mass., 1980

61. Burney, Pandit, Wu, "A Stochastic Approach to Characterization of Machine Tool Dynamics under Actual Working Conditions", Trans. ASME J. Engineering for Industry, May, 1976.

62. Klamecki, "Process Models for Anticipatory Control of Turning Operations," Modeling, Sensing, and Control of Manufacturing Processes, ASME, 1986, p.221-230

63. Danai, Ulsoy, "A Dynamic State Model for On-Line Tool Wear Estimation in Turning, " Trans. ASME J. Engineering for Industry, Nov. 1987, pp.396-399

64. Masonry, "Real Time Estimation of Cutting Process Parameters in Turning," ASME J. Engineering for Industry, vol. 106, August 1984

65. D. E. Hardt, M. A. Roberts, K. A. Stelson, "Closed-Loop Shape Control of a Roll-Bending Process", Trans. ASME J. Dynamic Systems, Measurement, and Control, Dec. 1982, Vol. 104, pp. 317-322.

66. D. E. Hardt, R. D. Webb, "Sheet Metal Die Forming using Closed-Loop Shape Control," Annals of the CIRP, Vol.31/1/1982, pp.165-169

67. Prof. John Longwell, MIT Dept. Chemical Engineering, personal communication.

68. Bellman, Richard Dynamic Programming, Princeton University Press, Princeton

APPENDIX A: A DYNAMIC PROGRAMMING TUTORIAL

Dynamic programming is a powerful optimization technique first developed by Richard Bellman at the Rand corporation in the 1950s. [68] It has found widespread use in diverse fields such as operations research and optimal control. However, it is not generally taught in required engineering curricula, so a tutorial on the subject, based on a related example, is presented here. The second sectionwill relate this simple example to the problem of planning weld bead grinding.

A.1 A SIMPLE DYNAMIC PROGRAMMING EXAMPLE

Consider the problem of finding the fastest route down a mountain at a ski resort from a given point on the mountain. A map of the ski trails is shown in Figure A-1. The destination is one of the three entrances to the parking lot at the bottom, indicated by the three heavy circles. For speed, one must remain on the ski trails and must only ski downward (climbing is rather slow). The trails meet at intersections where one must decide the next trail to proceed on. These are indicated by circles in Figure A-1, with letters identifying each intersection. Different trails have different slopes and difficulties, and so take different amounts of time to ski down. From past experience, it is known how long it takes to traverse each trail. These times are marked along each trail in Figure A-1. The problem is to select the route down the mountain that minimizes the total time required to get to the parking lot.

The solution is to first consider being at each of the intersections nearest the destination, such as those marked A, B, and C in Figure A-2. From *each* of these intersections consider all the routes to the destination, and pick the fastest route from that intersection to the destination. Then note down the best route to take from each of these intersections, and how much time it would take if you took each of those routes. In Figure A-2 the optimal routes are shown as heavy lines with arrows, and the total time to the destination is written adjacent to each intersection.

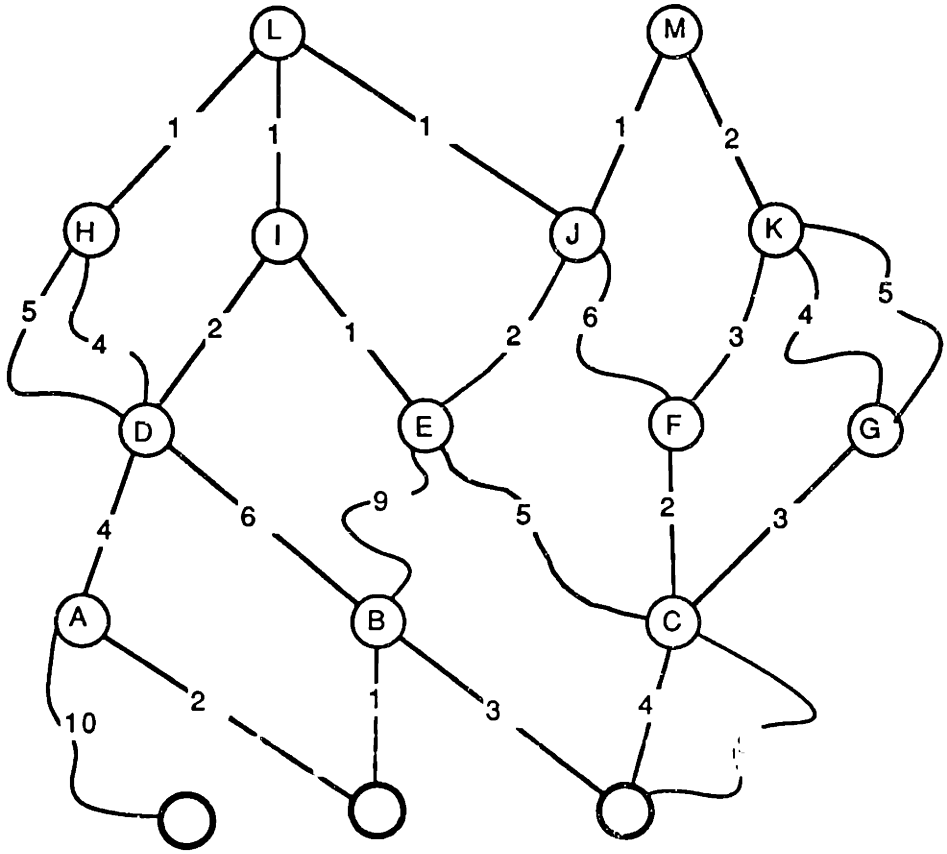


Figure A-1 A Ski Trail Map

Now consider each of the nearby intersections higher up. These are labelled D through G in Figure A-2. From each of these intersections, find the fastest route *all the way to the destination.* Note that if the optimal route down had previously been found from each of the intersections A–C beneath this set of intersections, then the fastest route to the destination from any one of the higher intersections, D–G, must include the fastest route from one of the lower intersections A, B, or C, to the destination. Thus, in this step one only needs to compute the optimal routes from the higher intersections to the lower intersections The rest of the optimal route downward has already been determined. This is called the **principle of optimality**. Here, for example, there are two routes down from intersection D: head to either intersections A or B. The total travel time via A would be 4 minutes along the trail to A plus 2 minutes from there on down, for a total of 6 minutes. Similarly the total travel time for the other route is 6 minutes plus 1 minute, or 7 minutes. The optimal overall route is to head initially towards intersection A, and then to proceed along the optimal route from there to the parking lot. This is repeated for each of the intersections at this level before continuing on to the higher intersections.

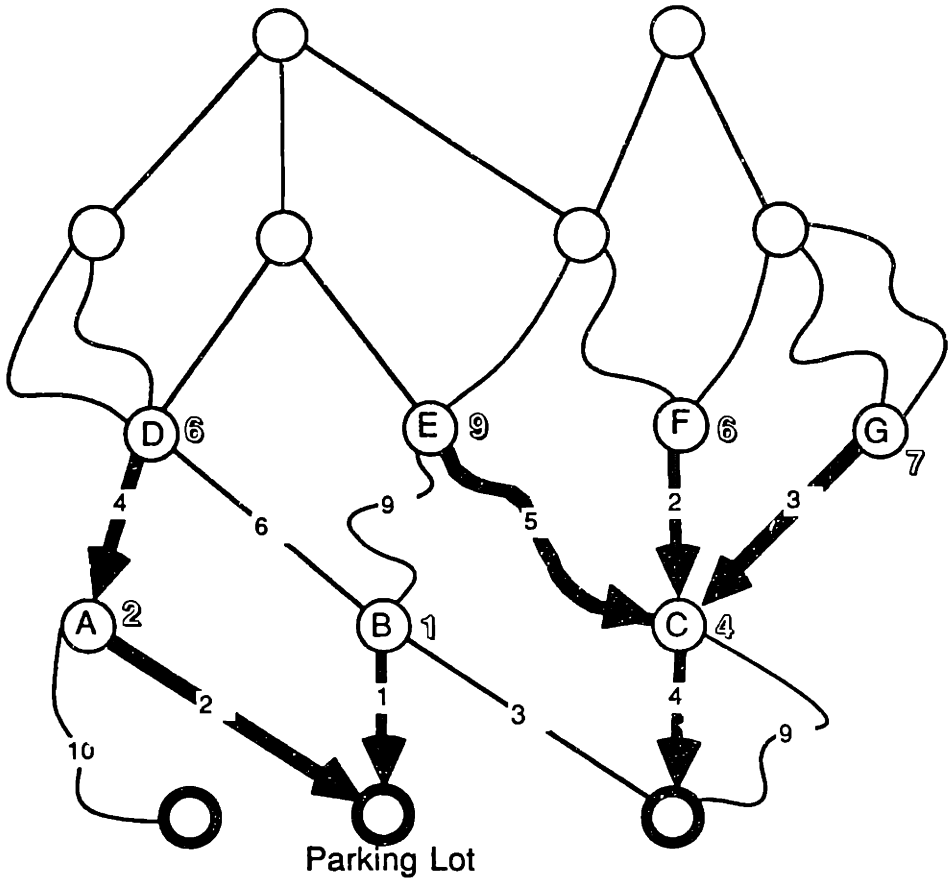


Figure A-2 Partial Optimal Route Solution

Now consider the set of intersections one level higher, H–K, in particular intersection I. See Figure A-3. There are two routes down, towards intersections D and E. The time to intersection D is 2 minutes, and the optimal time down from there is 6 minutes, so the total travel time along that route will be 8 minutes. Similarly, the total time along the route towards intersection E would be 10 minutes, so the best route is towards D, even though it takes longer to get to D than to E. Note that the principle of optimality allows us to ignore the details of the optimal routes down from the next lower set of intersections.

So, the algorithm is to start at the destination and work upwards one level at a time, selecting the route with the shortest total time from each intersection all the way to the destination. The principle of optimality makes the job easier, because the optimal routes and times from all the lower intersections have already been identified. The complete solution is shown in Figure A-3 below.

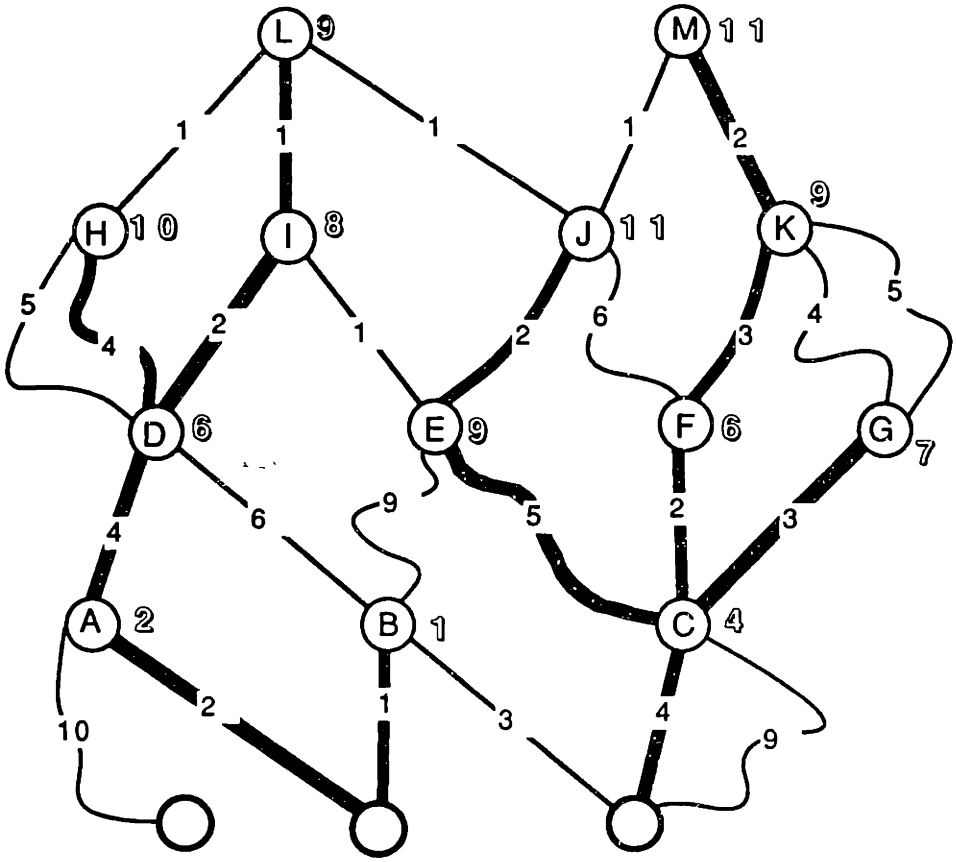


Figure A-3 Complete Optimal Route Solution

Now, for every intersection on the map, the local directions have been noted for the fastest route from that intersection to the destination. When one skis down to an intersection, one only have to look up the directions for that intersection in his notes to know what trail to take next. For example, starting from intersection I, the optimal route is to ski down the trail to intersection D, and then proceed optimally from there. The optimal route from D was worked out in the previous steps in the algorithm. Your notes indicate that the total time from I to the bottom will be 8 minutes.

If one makes a wrong turn, his notes will contain the optimal directions from whatever intersection he comes to, though this new route may no longer be along *the* optimal route from your original location. For example, suppose one went from I down the trail to intersection E rather than to intersection A. The best route from there is now towards intersection C, thence to the bottom. The total time to the bottom, however, will be 10 minutes rather than 8 minutes.

This is the dynamic programming algorithm. The important thing to note is that it yields the optimal route from *every* intersection to the bottom in the form of a

table lookup of local directions from each intersection. This table lookup solution also allows for optimal error recovery in case the original optimal path is strayed from. In effect, this is an instantaneous optimal replan, and is a very useful characteristic for planning real processes.

A.2 Comparison with the Planning of Weld Bead Grinding

Note that the weld bead grinding problem is similar to the skiing example. Taking a grinding pass is similar to skiing along a trail; both are state transitions. The grinding deadline is similar to the parking lot at the bottom. The state of the weld bead (grinding pass and volume of material remaining) is similar to the intersection (height and location left to right along the mountain). The choice of cut depth is similar to the choice of trails. The criterion to be minimized was the total skiing time in the skiing example. For grinding, it will be a penalty function which measures how well the task specifications have been satisfied. Finding the best grinding sequence is similar to finding the best route down the mountain.

APPENDIX B: MODELLING THE VERTICAL FORCE TABLE

B.1 INTRODUCTION

The purpose of this work is to find a suitable dynamic model of the vertical force table so that a force controller can be designed for it. This requires knowledge of both how the drive motor works and how its associated (position) controller works. A schematic of the entire system is shown below. A DC motor drives a table up and down through a ball-screw. This motor is controlled by an analog amplifier which uses both velocity and position feedback to control the position of the table according to voltage commands sent from a computer. The table pushes up against a compliance (in this case, a flexible grinding disk), so the force generated is a function of the position of the table. The force is measured by a force sensor, and is read in digitally by the computer. The computer uses this measured force to send the appropriate position commands to the amplifier to correct small force errors. The goal of this work is to model the behavior of the table as a force actuator so that the appropriate force control algorithm can be implemented in the computer.

B.2 THE MOTOR MODEL

The motor used in the vertical table was the Electro-Craft 1040 MCM DC moving coil motor. It features a high torque to inertia ratio which makes it ideal for servomechanism applications, and comes with a tachometer. The motor model is typical for DC motors, having a LR electrical circuit driving a mass-damper mechanical system. A back EMF generated by the spinning rotor is fed back from the mechanical system to the electrical circuit. This is illustrated in Figure B-1.

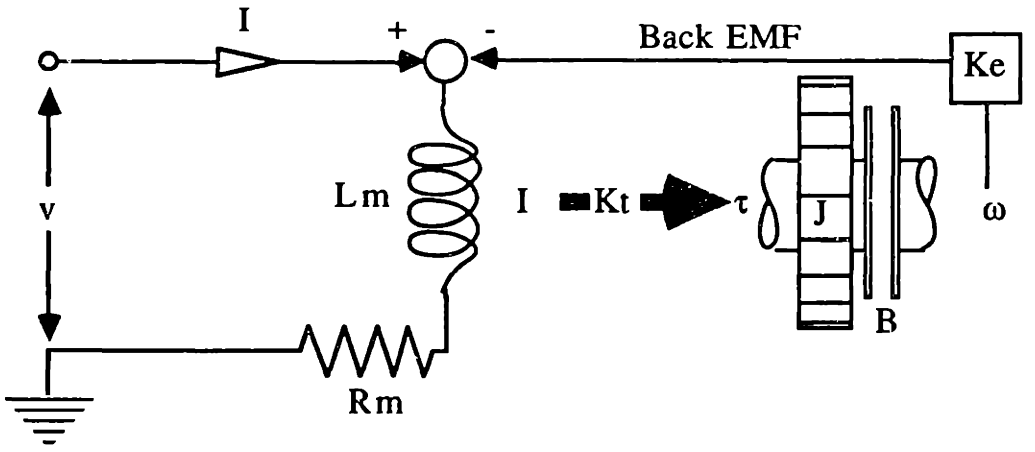


Figure B-1 Motor Model Schematic

The model block diagram is shown in Figure B-2 below:

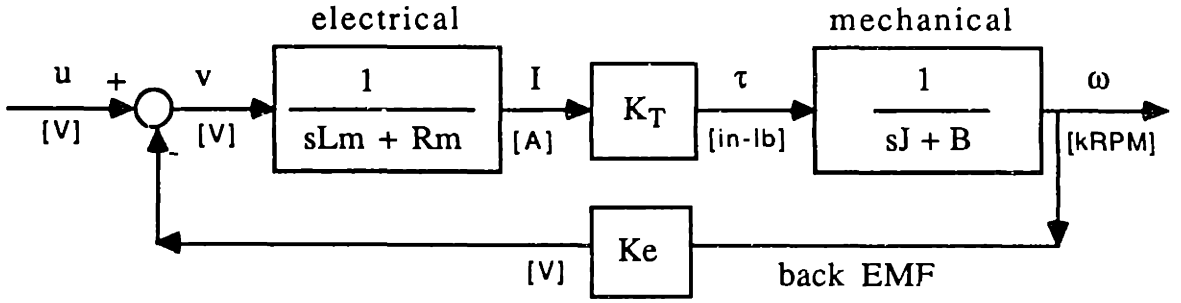


Figure B-2 1040 MCM Motor Model Block Diagram

The constants and variables are as follows:

J	Rotational Inertia[1]	4.06×10-5	in·lb·s2
B	Viscous Friction	0.063	in·lb/kRPM
Lm	Armature Inductance	0.09	mH
Rm	Armature Resistance	0.7	W
K_T	Torque Constant	0.363	in·lb/A
Ke	Back EMF Constant	4.3	V/kRPM

---

[1] This value is consistent with a system in which torque is measured in in·lb and rotational acceleration is measured in rad/s2. To be consistent with the units of the other constants, acceleration should be measured in kRPM/s and inertia in in·lb·s/kRPM. To make the conversion, the value in the table should be multiplied by (1min/60s)(2π rad/rev)(1000 rev/krev) to yield J = 4.25×10-3 in·lb·s/kRPM.

u	Command Voltage	V
v	Armature Voltage	V
I	Armature Current	A
t	Rotor Torque	in·lb
ω	Shaft Angular Velocity	kRPM

A little block diagram algebra yields:

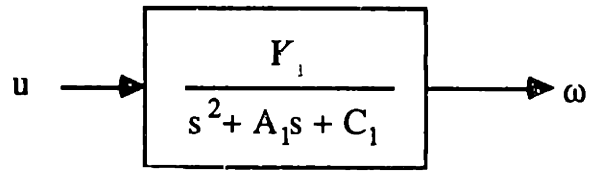


Figure B-3 1040 MCM Motor Transfer Function

where:

$K_1 = K_T/(L_m J) = 9.49\times10^5$ kRPM/V·s^2

$A_1 = (R_m/L_m) + (B/J)$

$= 7778\ s^{-1} + 14.82\ s^{-1} = 7793.\ s^{-1}$

$C_1 = (R_m B + K_e K_T)/(L_m J)$

$=(4.41\times10^{-2} + 1.56)/3.825\times10^{-7}$

$= 4.195\times10^6\ s^{-2}$

The poles of this system are at -582. rad/s and -7211. rad/s, which can be ascribed to the mechanical and electrical systems, respectively. The DC gain is 0.226 kRPM/V. Note that the terms containing the viscous friction term B are dominated by the terms which with they are summed. Thus A_1 can be approximated by (Rm/Lm) and C_1 by (Ke K_T)/(Lm J). Note also that the motor can be adequately described by the mecanical system alone, with the transfer function:

$$\frac{1/K_e}{s\,T_m + 1}$$

where: $T_m = (R_a J)/(K_e K_T) = 1.91$ ms. $1/T_m = 524. \cong 582.$

$1/K_e = 0.233 \cong 0.226$

B.3 SPEED-CONTROL SYSTEM

The controller for the motor is the Electro-Craft LA5600 linear amplifier. It has a notch filter at 4kHz to avoid exitation of the torsional resonance of the rotor-shaft

combination. This notch filter is at frequencies well above those we are interested in, so it will be omitted from the model. The controller uses feedback from the tachometer to control the speed of the motor. The simplified circuit diagram for the amplifier connected to the motor is shown in Figure B-4 below:

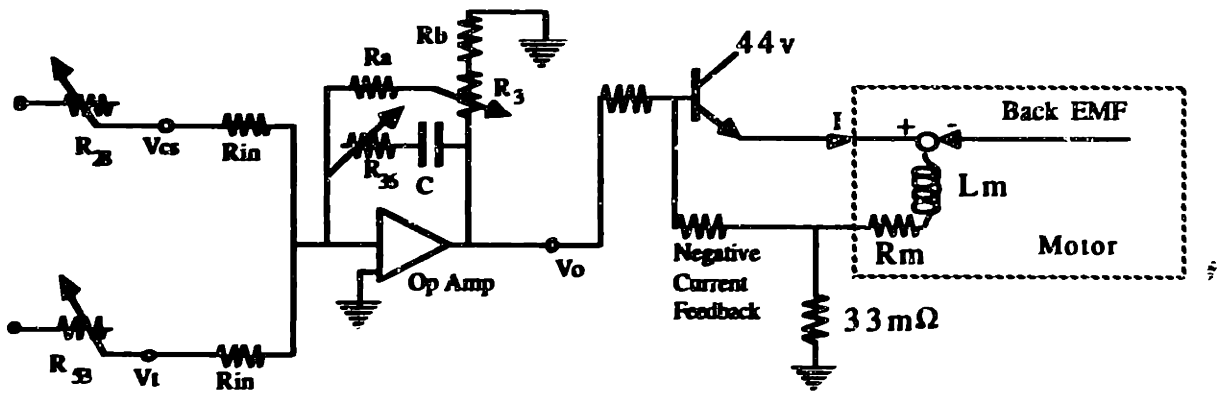


Figure B-4. Amplifier Circuit Diagram

Analysis of the left side of the circuit indicates that the transfer function from Vcs to Vo is given by:

$$\frac{Vo}{Vcs} = K_2\left[\frac{\frac{s}{z_a}+1}{\frac{s}{p_a}+1}\right]$$

where: $K_2 = -\gamma G R_a R_3 / R_{in}$

$z_a = 1/C R_{36}$

$p_a = 1/C(\gamma G R_a R_3 + R_{36}) \quad \leq z_a$

$\gamma = 1/[R_b + (1-G)R_3] + 1/R_a + 1/GR_3$

G = the fraction that R_3 is adjusted towards Rb.

The right hand side of the circuit is a current control loop. The voltage Vo controls the power transistor. The current I is controlled proportionally to this voltage as long as rotor back EMF and voltage across the motor inductance and resistances

don't overwhelm the 44v power supply[2]. The motor current is grounded through a small resistor (33mΩ), and the resulting voltage drop is summed with Vo to give negative voltage feedback. Thus the motor current is controlled proportional to Vo.

The block diagram is shown in Figure B-5 below. Note the extra controller pole.

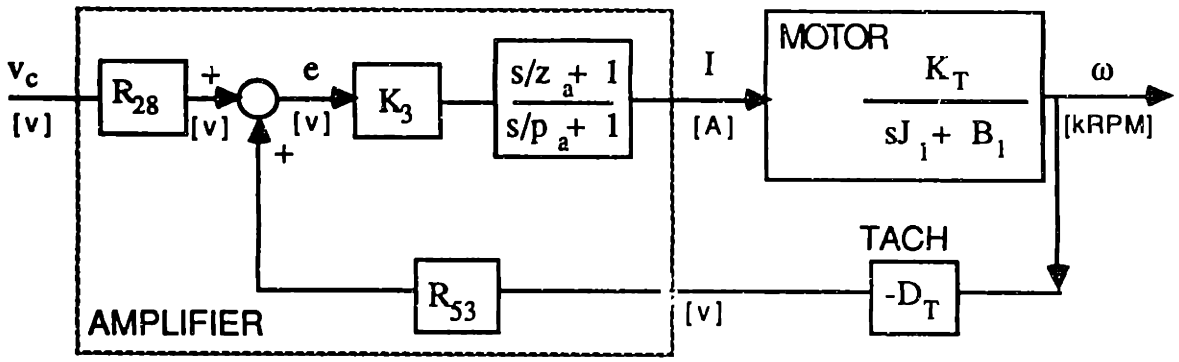


Figure B-5 Motor plus Amplifier Speed Control Block Diagram

v_c is the command voltage into the amplifier. D_T is the tachometer constant, which was calibrated using a calibrated stroboscope and a precision voltmeter, and found to be 3.09 V/kRPM. K_3 is the combination of the gain of the Vo to I circuit and the gain K_2 (of the Vcs to Vo circuit). R_{36}, R_{28}, and R_{53} are potentiometers that can be set (in a limited range) in the amplifier. These were set to give a fast, deadbeat response, with a DC (steady-state) gain of 0.4 kRPM/V.

The values of many of these constants are difficult to determine *a priori*, due to the complexity of the circuit board, but the important constants, K_3, p_a, and z_a, can be obtained through careful experimentation. Now R_{28}, R_{53}, and K_3 can be measured directly when the motor is running at a constant speed. This is facilitated by the existence of test points on the LA5600 circuit board, where we can measure the voltage signals just beyond R_{28} and R_{53}, and the voltage drop across the 33mΩ resistor to get the motor current. Thus we find that R_{28} = 1.00, and R_{53} = 0.806. Subtracting the signal just after R_{28} from the signal just after R_{53} gives the error signal e. Dividing the

---

[2] This can occur at high rotor speeds, when the back EMF is high, or for high motor torques, when the voltage drop across the motor resistance is high due to the high current level, or when the current changes magnitude quickly, causing a high voltage across the motor coil inductance. If this occurs, the circuit becomes nonlinear.

steady state motor current signal I by the steady state error signal gives the amplifier DC gain, K_3. This was found to be ≅ 36.7 A/V.

Examining Figure B-5, we see that the transfer function from v_c to ω is given by:

$$\frac{\omega}{V_C} = \frac{\frac{p_a}{J_1}(R_{28}K_3K_T)\left(\frac{s}{z_a}+1\right)}{s^2 + \left(p_a + \frac{B_1}{J_1} + \frac{\alpha\, p_a}{J_1 z_a}\right)s + \frac{p_a}{J_1}(B_1+\alpha)}$$

where: $\alpha = D_T R_{53} K_3 K_T$

$= (3.09\ V/kRPM)(0.806)(36.7\ A/V)(0.363\ in{\cdot}lb/A)$

$= 33.2\ in{\cdot}lb/kRPM.$

J_1 and B_1 are constants supplied with the motor.[3] The problem is how to determine p_a and z_a. Note that we can write the transfer function as:

$$\frac{\omega}{V_C} = \frac{K_V\Omega^2\left(\frac{s}{z_a}+1\right)}{s^2 + 2\zeta\Omega s + \Omega^2}$$

where: ζ is the damping ratio

Ω is the natural frequency, in rad/s

K_V is the system DC Gain, = 0.4 kRPM/V

Then p_a is given by

$$p_a = \frac{J_1\,\Omega^2}{B_1 + \alpha}$$

---

[3] B_1 has been experimentally verified by measuring the motor current at a range of steady motor speeds. It was found that there is also a small amount of stiction, =0.168 in·lb/kRPM.

and z_a is given by:

$$z_a = \frac{p_a \alpha}{J_1(2\zeta\Omega - p_a - B_1/J_1)}$$

The frequency response of this speed control system was obtained by connecting the amplifier input to a voltage signal generator and measuring the speed via the tachometer on an oscilloscope. The tachometer output was observed on an oscilloscope, and the amplitude of the output signal was measured. The resulting frequency response is shown in Figure B-6 below:

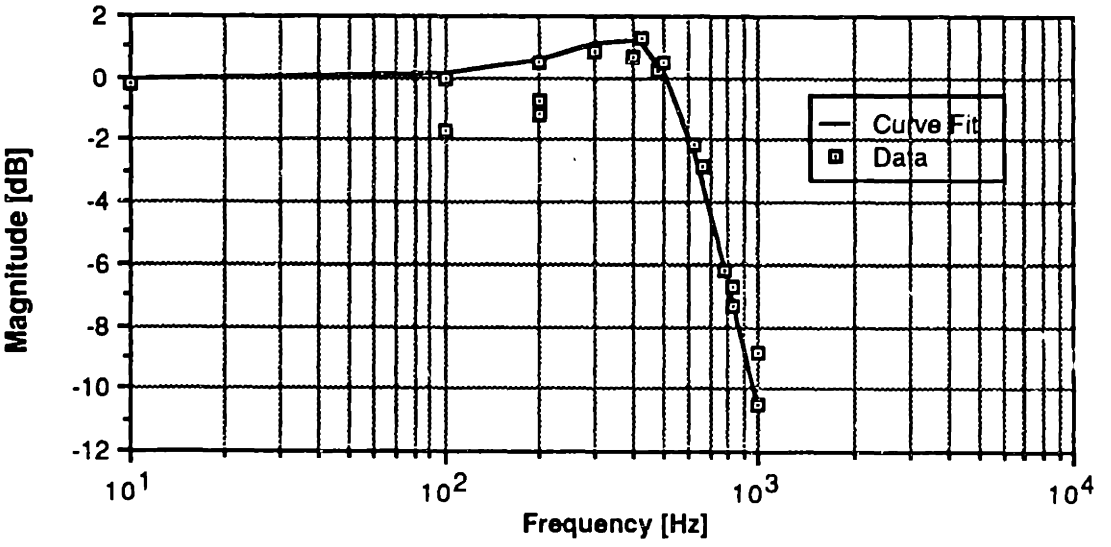


Figure B-6a **Speed Control Frequency Response Magnitude**

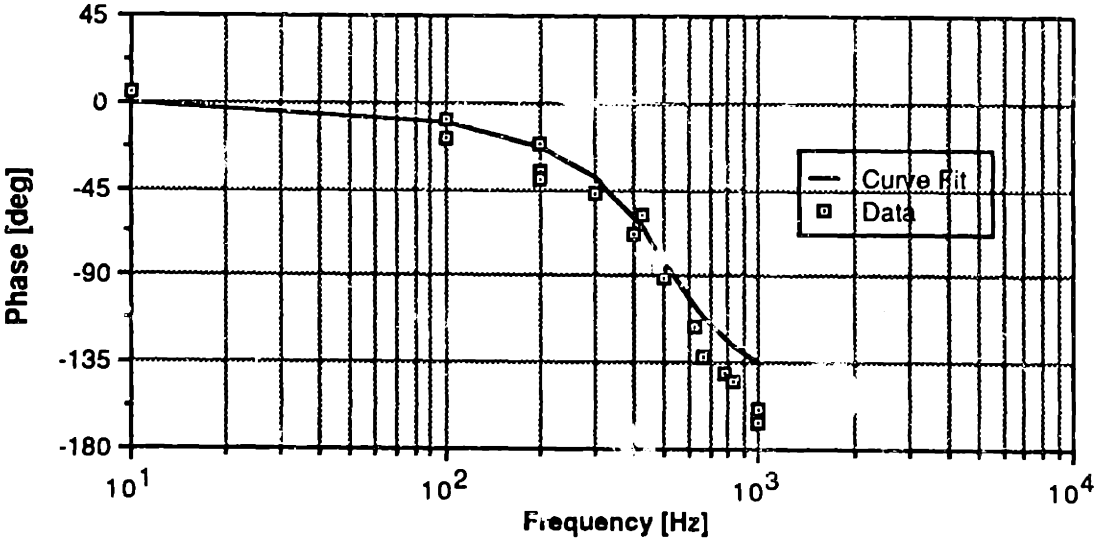


Figure B-6b Speed Control Frequency Response Phase

The model was obtained by trial-and-error, by varying the position of the model poles until a good curve fit was obtained. Thus, within the range of frequencies examined, the system is experimentally described by a well-damped pole (ζ=0.5) at 510 Hz, with the controller zero z_a well above 1000 Hz. By the above calculations, $p_a \equiv 1{,}312.$ rad/s = 209 Hz. Using the above calculations, the value of K_V is verified to be 0.403 kRPM/V, but z_a is computed to be 869. rad/s. In modelling the controller circuitry, it was noted that $p_a \leq z_a$. This calculation is very sensitive to values in both p_a and α, both of which are sensitive to error-prone measurements. It makes sense that $p_a \leq z_a$, since the purpose of these singularities is to function as a lag filter. The value of R_{36} was set to give the highest frequencies for these singularities. For now, z_a will be assumed to be above the range of frequencies of interest, and effectively neglected.

B.4 Position Control

Next, the motor was installed into the vertical position table. The motor drove a right-angle gear box, which drove a ball-screw. The ball-screw, in turn, converted the rotary motion into the linear vertical motion of the table. Now the inertias of the right-angle drive, the ball-screw, and the table added to the inertia of the motor, thus requiring recalculation of the J_2 and B_2 constants. The position of the table was also fed back to the controller via a LVDT, and this signal was summed with the tachometer signal. The block diagram of the position control system is shown in Figure B-7 below:

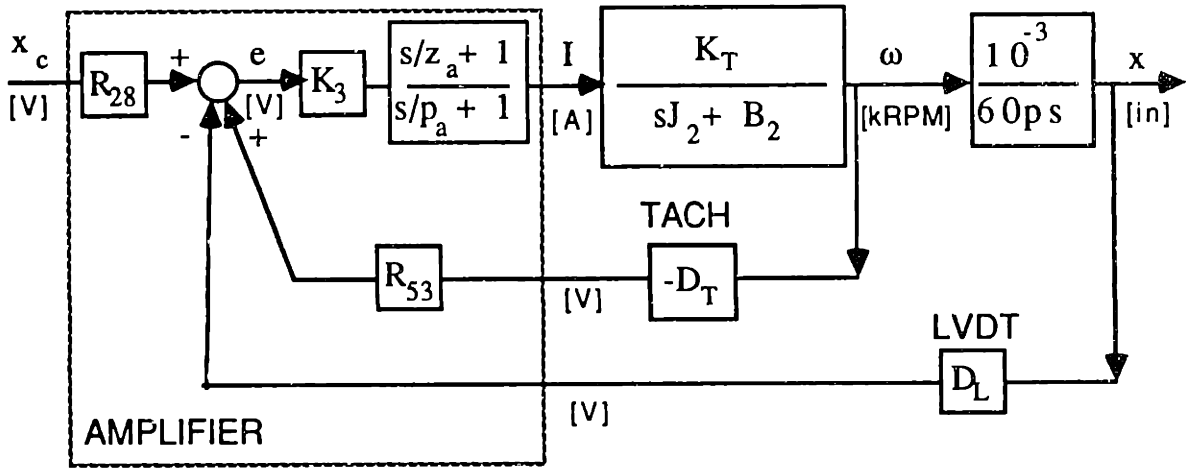


Figure B-7 Vertical Table Position Control Block Diagram

Here, p is the pitch of the lead screw, 5 rev/in., x_C is the command voltage, and x is the vertical distance moved, in inches. D_L is the calibration constant for the LVDT, which was calibrated using a dial gauge and a precision voltmeter to be 41.878 V/in. J_2 is the equivalent inertia seen by the motor. This includes the rotary inertias of the motor, the coupling, the right-angle gear box, and the equivalent inertia of the tabletop and all linearly-moving parts. The resulting inertia is increased to $J_2 = 9.313\times10^{-3}$ in·lb·s/kRPM.

The transfer function from x_C to x is given by:

$$e = R_{28}x_C - R_{53}D_T\omega - D_L x$$

$$\omega = \left(\frac{10^{-3}\text{kRPM}}{\text{rev/min}}\right)\left(60\frac{\text{sec}}{\text{min}}\right)(\text{p[rev/in]})\, s\, x$$

$$x = \left[\frac{10^3}{60ps}\right]\left[\frac{K_T}{sJ_2 + B_2}\right]\left[\frac{s/z_a + 1}{s/p_a + 1}\right] K_3\, e$$

$$\frac{X}{X_C} = \frac{R_{28}\beta\left(\frac{s}{z_a} + 1\right)}{s^3 + \left(B_2 + \frac{B_2}{J_2 p_a} + \frac{B_2\alpha}{J_2 z_a}\right)s^2 + \left(\frac{B_2^2}{J_2} + \frac{\alpha B_2}{J_2} + \frac{D_L\beta}{z_a}\right)s + D_L\beta}$$

where:

$$\beta = \frac{10^{-3} K_T K_3 B_2}{(60\ \text{s/min})\ p\ J_2}$$

The values of p_a and z_a can be better determined when comparing the results of an actual frequency response test with the transfer function above. Values of pa and za were estimated and tried in the above transfer function, until a reasonable curve fit was obtained. In the Figure B-8 below, the results of the frequency response test are shown as 'actual' data points, and the transfer function above, with pa = 1100 rad/s, and za = 5000 rad/s, is shown as the 'theory' curve.

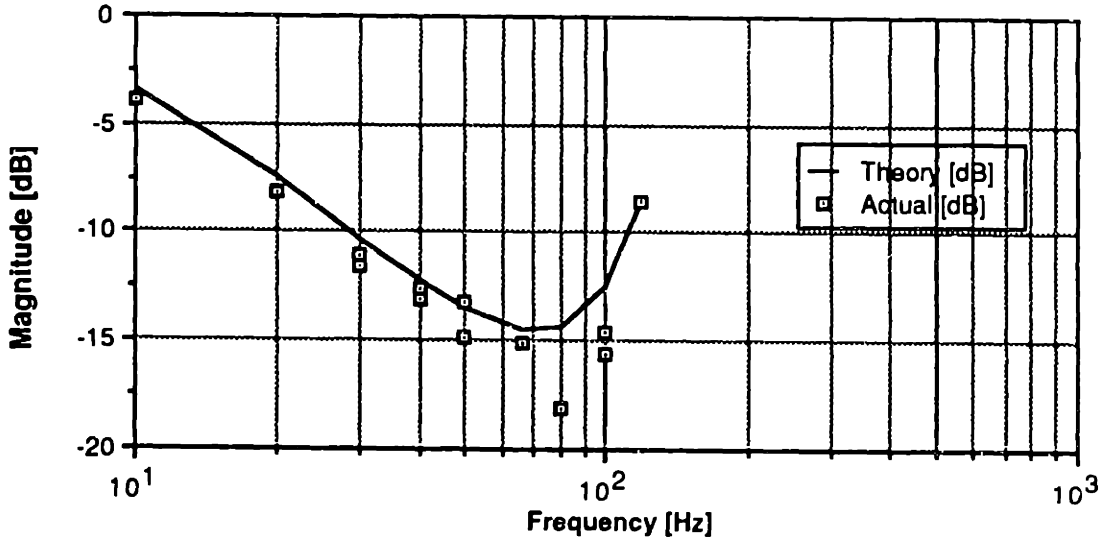


Figure B-8a Frequency Response Magnitude Comparison

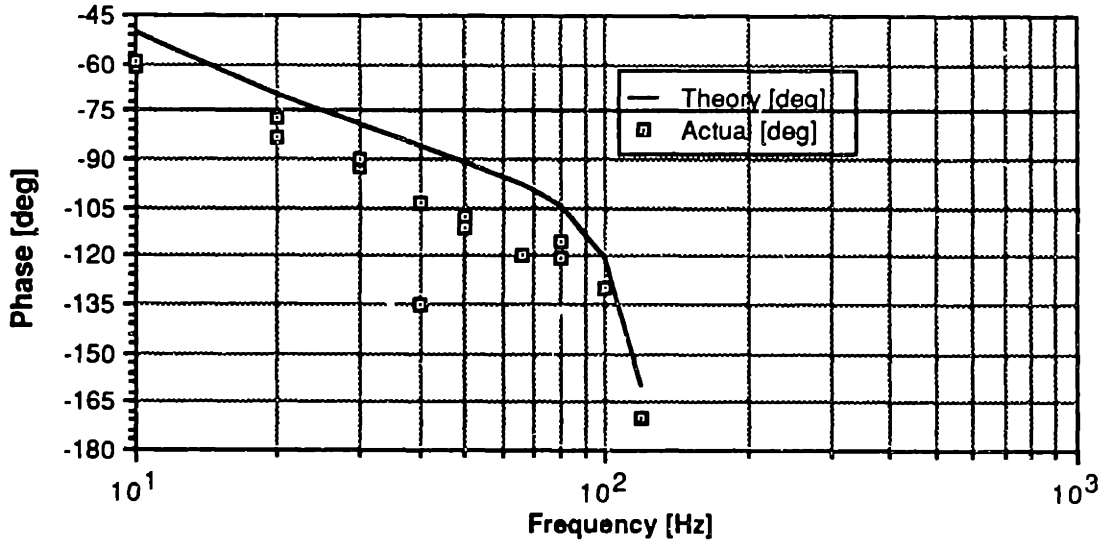


Figure B-8b Frequency Response Phase Comparison