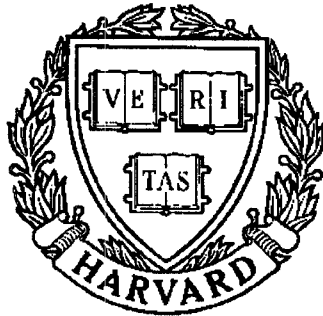


THESIS REPORT
Master's Degree



S Y S T E M S
R E S E A R C H
C E N T E R



*Supported by the
National Science Foundation
Engineering Research Center
Program (NSFD CD 8803012),
the University of Maryland,
Harvard University,
and Industry*

**A Neural Network Approach to On-line
Monitoring of Machining Processes**

*by R.G. Khanchustambham
Advisor: G.M. Zhang*

Abstract

Title of Thesis: A Neural Network Approach to On-line
 Monitoring of Machining Processes

Name of degree candidate: Raju G. Khanchustambham

Degree and Year: Master of Science, 1992

Thesis directed by: Dr. Guangming Zhang, Assistant Professor,
 Department of Mechanical Engineering, and
 Systems Research Center

In recent years, production automation has been the focus of this research endeavor to improve product quality and to increase productivity. Implementation of computer-based untended machining has attracted great attention in the manufacturing community. For a successful implementation of untended machining, a better understanding of the machining processes and the functions they perform is required. This necessitates the development of sensors and intelligent decision-making systems.

In this thesis work, a framework for sensor-based intelligent decision-making systems to perform on-line monitoring is proposed. Such a monitoring system interprets the detected signals from the sensors, extracts the relevant information, and decide on the appropriate control action. In this thesis, emphasis is given to applying neural networks to perform information processing, and to recognize the process abnormalities in a machining operation. For signal detection, an instrumented force transducer is designed and implemented in real time turning operation. A neural network program based on feedforward back-propagation algorithm is developed. The program is tested by the simulation data and verified by the experimental data. The superior learning and noise suppression abilities of the developed program enable high success rates for monitoring the tool wear and surface roughness under machining of advanced ceramic materials.

It is evident that the development of hardware neural networks provides the monitoring system with fast computational capabilities. The advances in sensor technology enables inexpensive sensors to be easily mounted for monitoring the machining process. All these evidences justify that neural networks will become an attractive modeling tool for use in on-line monitoring of machining processes in near future.

A NEURAL NETWORK APPROACH TO ON-LINE MONITORING OF MACHINING PROCESSES

by

Raju G. Khanchustambham

Thesis submitted to the Faculty of the Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1992

Advisory Committee:

Assistant Professor Guangming Zhang, Chairman/Advisor
Professor Andre Tits
Assistant Professor Emmanuel Ssemakula
Assistant Professor Mohammed Abdelhamid

Dedication

To The Memory of My Father

Acknowledgments

Foremost, I wish to express my deep gratitude to Professor Guangming Zhang, who has been my guide academically and personally and who has encouraged and supported me throughout my graduate program to its fruitful conclusion. By his personal example, he sets a unique standard in research and presentation which I can only try to emulate in my professional carrier.

I would also like to thank Professors M. Abdelhamid, E. Ssemakula, and A. Tits not only serving on my committee, but also for their valuable guidance in finishing this thesis. Also I would like thank Dr. Jahanmir, Tsuwei Hwang and H. Dave for their valuable support and suggestions in performing experiments. Last but not the least, I would like to thank various friends and fellow students. In particular, I wish to thank Srinivas Nagulapalli and Ajay Arora for their valuable support during the preparation of this thesis. Finally, my thanks and appreciation to the members of Department of Mechanical Engineering, University of Maryland, for their support.

Contents

1	Introduction	1
1.1	Background	1
1.2	Outline of Thesis	3
2	Literature Review	5
2.1	History of Neural Networks	5
2.2	Motivations for ANN Research	7
2.3	Introduction to Neural Networks	9
2.4	Existing Methods to Monitor the Machining Processes	13
2.5	Significance of Neural Networks in On-line Monitoring	16
2.6	Applications of Neural Networks	18
3	On-Line Monitoring Methodology	23
3.1	Description of an On-line Monitoring System	24
3.2	Identification of Relationship Between Force and Cutting Parameters	28
3.2.1	Introduction to Adaline Network	29
3.2.2	Adaline Model	32
3.3	Monitoring of Tool Wear and Surface Roughness	39
3.3.1	Basic Concepts of Tool Wear	39
3.3.2	Basic Concepts of Surface Roughness	43
3.3.3	Feedforward Neural Networks	47
3.3.4	Factors Related to Tool Wear and Surface Roughness	59

3.3.5	On-line Monitor Model	61
3.4	ANN Applications to Machining of Ceramics	65
3.4.1	Overview	65
3.4.2	Effect of Cutting Parameters on Tangential Force and Sur- face Roughness	76
3.4.3	Conclusions	78
4	Experimental Verification	82
4.1	Experimental Arrangement	82
4.1.1	Description of the Experiments	82
4.1.2	Test Procedure	86
4.1.3	Tool Wear Measurement	87
4.1.4	Cutting Force Measurement	90
4.2	Design of an Instrumented Transducer	92
4.3	Case Study: Machining of Ceramics	96
4.3.1	Experimentation	96
4.4	Data Analysis	99
4.4.1	Derivation of Taylor's Equation	99
5	Framework of an On-line Monitoring System	108
5.1	Basic Requirements of an On-line Monitoring System	109
5.1.1	Sensor Signal Detection System	109
5.1.2	On-line Monitoring System	112
5.1.3	Expert System	113
5.2	Implementation of an On-line Monitoring System	114
5.2.1	Instrumented Transducer Vs Commercial Dynamometers .	116
5.2.2	Tool Wear Measurement	116
5.2.3	On-Line Monitoring of a Turning Process	117
5.3	Advantages and Disadvantages of Feedforward Neural Networks .	120
5.3.1	Advantages	120

5.3.2	Disadvantages	120
5.4	Control System	121
5.5	Description of Code	122
5.6	Comparision With Other's Work	134
5.7	Limitations and Generalization of the Neural Network Monitor . .	135
5.8	Conclusions	135
6	Conclusions and Recommendations	137
6.1	Conclusions	137
6.2	Future Work and Recommendations	139
7	Bibliography	141
8	Appendix	147

List of Figures

2.1	The General Aspects of Neural Networks	9
3.1	Block Diagram of an On-Line Monitoring of a Machining Process	26
3.2	The Meso and Micro Structure of an Adaline and Madaline	31
3.3	The Architecture of Adaline Network	34
3.4	Adaline Network's Output	37
3.5	The Plot Between Actual and Desired Output of an Adaline . . .	38
3.6	Tool Wear Criteria and Failure Mechanisms	41
3.7	Tool Wear Classification	43
3.8	A Three Dimensional Surface Profile	45
3.9	The Important Quantities of a Surface Profile	46
3.10	The Meso and Micro Structures of a BPN	50
3.11	Sigmoid Function and It's Derivative	52
3.12	The Plot Between Error and Weight Vector	58
3.13	A Schematic View of an ANN	63
3.14	The Architecture of an On-Line Model	66
3.15	The Architecture of a Force and Surface Roughness Model	67
3.16	Results of Tool Wear Network	70
3.17	Results of Surface Roughness Network	71
3.18	Results of Tool Wear and Surface Roughness Network	72
3.19	Plots of Tool Wear Model	73
3.20	Plots of Surface Roughness Model	74

3.21	Plots of Tool Wear and Surface Roughness Model	75
3.22	Architecture of Neural Network Used for Ceramics	77
3.23	Results of Neural Networks for Ceramics (Tangential Force) . . .	79
3.24	Results of Neural Networks for Ceramics (Surface Roughness) . .	80
4.1	Experimental Arrangement for Conducting the Machining Tests .	84
4.2	Setup of Data Acquisition System	85
4.3	Structure of an Optical Microscope	88
4.4	The Wheatstone Bridge Circuit	94
4.5	Strain Gage Calibration	95
4.6	Experimental Setup For Ceramic Tests	98
4.7	Factorial Design of Tangential Cutting Force	100
4.8	Factorial Design of Surface Roughness	101
4.9	Tool Wear of Diamond Insert	102
4.10	Tool Wear Curves Obtained from Experimental Results	104
4.11	Log-Log Plot of Cutting Speed and Tool Life	105
5.1	Conventional and Ideal On-Line Monitoring Turning Process . . .	110
5.2	Methodology of an On-line Monitoring of a Turning Process . . .	111
5.3	Implementation of an On-Line Monitoring in a Turning Process .	115
5.4	On-line Monitoring Charts of Ceramics Machining	118
5.5	Basic Working Principle of a BPN Network	126
5.6	Basic Principle of Weight Changes in Output Layer	129
5.7	Basic Principle of Weight Changes in Hidden Layer	130

List of Tables

3.1	Cutting Force Data	33
3.2	Experimental Data	68
3.3	Experimental Data	69
3.4	Effect of Cutting Parameters on Tangential Force	78
3.5	Effect of Cutting Parameters on Surface Roughness	81
4.1	Tool Wear Land Values at Two Different Cutting Speeds	89
4.2	Measured Tangential and Feed Forces (feed=0.14 mm/rev, depth of cut=1.00, speed=625)	91
4.3	Variation of Tangential and Feed Forces With Tool Wear (feed=0.14 mm/rev, depth of cut= 1.00 mm, speed=470 rpm)	91
4.4	Calibration in Feed Direction	96
4.5	Calibration in Tangential Direction	97
4.6	Experimental Results for Type 1 Fluid	99
4.7	Experimental Results for Type 3 Fluid	103

Chapter 1

Introduction

1.1 Background

Computer Integrated Manufacturing (CIM), systems have emerged in response to the requirements for greater flexibility, productivity, high precision and quality of the product. The need to improve quality and decrease scrap rate while increasing the production rate is forcing industry to consider untended machining as a viable alternative. But this leads the operator, who attempts to sense the effect of process variables and adjust the conditions accordingly, out of loop in many cases. Also sometimes the operator is incapable of responding fast to alter the conditions of operation accordingly. The former leads to high scrap rate and higher cost to the need for rework. The later leads to reduced productivity. Therefore appropriate sensors and associated monitors are, therefore, the key to the successful implementations of an untended machining process.

On-line monitoring of a machining process is the key success of an untended machining process. The monitoring systems should be highly reliable, in order to leave the intelligent human operator out of manufacturing loop. The methodology to develop a monitoring system involves several key issues, like tool wear model, on-line signal detection, digital signal processing, and model-based process controller. The model-based controller serves as a link between the machin-

ing process stage and the detected signals from sensors. For successful on-line monitoring, various sensors have been evaluated. These include, among others, sensors based on force, torque, power, vibration, deflection, acoustic emission, vision and radioactivity. Though these sensors are successful in manufacturing shop floor, the need is felt more than ever if we were to be successful in implementing an untended manufacturing efficiently. In some machining applications, it is the lack of system's process monitoring that is preventing total automation. The assurance of product quality and the minimization of manufacturing cost call for the use of nondestructive, in-process sensing techniques to characterize, not only geometrical dimensions, shape and size, but also the microstructure, internal defects, and material properties of a part. The availability of product quality information on-line enables us to control a manufacturing process in real time, realizing the objective of building quality into a product by minimizing variability in the product's characteristics.

Successful implementation of an untended machining process relies, to a great extent, on the ability to recognize process irregularities and initiate corrective actions. In an untended machining process (where intelligent human operator is missing), this function has to be performed with sensors and associated decision-making systems which are able to interpret incoming information related to the machining process and control the process approximately. An integrated system consisting of sensing elements, signal conditioning devices, signal processing algorithms, and signal interpretation and decision-making procedures constitutes an intelligent sensor. Such a sensor system is a necessary requirement for successful implementation of an untended machining process characterized by noisy and unpredictable environments. The major concern in an automated machining process, where normal cutting condition is desired without the human intervenes, is the correct identification of the tool state as the machining process is going on. On-line tool wear information is indispensable in precision machining, since it has direct influence on part dimensions and also it assure safe cutting operation.

On-line wear monitoring is very important, in order to achieve a full automation of machine tools and to avoid a very conservative tool changing. Prediction of tool wear, which effects the surface quality of the machined part, vibrational level in the machine tool, can be done by analyzing the signals from different sensors. By studying the sensor outputs one can change the tool to achieve the best quality of the machined product.

Intelligent sensor systems are expected to replace the knowledge, experience, and sensory and pattern recognition abilities of human operators. For a successful implementation of this task, the quality of information generated by the monitoring sensors should be of high quality and the learning and decision-making procedures used to analyze this information in the context of the process state should be of high reliability. The quality of information depends on the type and number of sensors used, and the signal to noise ratio of the information generated by these sensors. Sensing strategies for unmanned machining should aim at integrating the above factors, there by allowing for a sensor system design which possesses the ability to successfully implement the sensory abilities and pattern recognition skills of the intelligent human operators.

The main objective of this thesis work is to develop an intelligent on-line monitor, to recognize the process irregularities and to estimate the corrective actions in an untended machining operation. By exploring the advantages of neural networks, an Artificial Neural Network monitor is developed. For successful implementation of the developed intelligent monitor in real time, a force transducer is designed and implemented in real time machining process.

1.2 Outline of Thesis

This thesis is organized into six chapters. The contents of each chapter are summarized below.

Chapter 2 describes an overview of pertinent literature. An introduction to

neural networks and their significance in on-line monitoring is discussed. Existing on-line monitoring methods and applications of neural networks are also discussed.

Chapter 3 describes the basic methodology used in on-line monitoring system. Here we explored the feedforward neural networks. Three types of neural networks applications are discussed. In first, a relationship between the cutting force and the cutting parameters is obtained. In second application, tool wear and surface roughness are predicted. Finally applications of neural networks in machining of advanced ceramics are discussed.

Chapter 4 deals with the experiments and analysis of experimental data. It illustrates the experimental setup to perform the machining tests. Measurement of cutting force and tool wear were discussed. Design and implementation of an instrumented transducer is explained. Taylor's speed-tool life curves are drawn using the experimental data.

Chapter 5 describes about the framework of an on-line monitoring system. Merits and demerits of instrumented transducer over commercial dynamometer are discussed. The advantages of feedforward neural networks and their significance in on-line monitoring are also discussed. The basic control strategy of an on-line monitoring system is analyzed. A brief description of code developed for a Back-Propagation Network is given.

Chapter 6 provides conclusions of this thesis and recommendations for future work. Here we briefly summarize the results of this thesis work. The contributions of neural network in the field of machining processes are presented. The success rate and difficulties of various architectures of neural networks in the applications of machining processes are also discussed.

Chapter 2

Literature Review

Overview

This chapter discusses briefly about the relevant literature of this thesis. It is mainly divided into six sections. The first section describes briefly about the history of neural networks. The second section describes the motivation for Artificial Neural Networks (ANN) research. The third section describes the basic concepts of neural networks. The fourth section discusses the existing monitoring systems in machining processes. The fifth section discusses about the significance of neural networks in on-line monitoring. A few practical applications of neural networks are discussed in last section.

2.1 History of Neural Networks

A technique for intelligent tool condition monitoring which employs information from the sensors is integrated through a neural network. The artificial neural networks consist of many massive interconnections of rather simple neurons which simulates the biological nervous system. These networks are also referred to as parallel distributed processing. It consists of a set of processing units, known as neurons. These neurons have a pattern of connectivity among them and the

knowledge can be represented by the strength of the connections, which is not fixed. Instead, the weights can be modified based on the experience, so the network can learn from its past experience, so as to simulate the human brain. So we can say these type of networks are intelligent.

Neurons are living nerve cells and neural networks are networks of these cells. The cerebral cortex of the brain is an example of a neural network. Somehow, such a network of neurons thinks, feels, learns and remembers. In the past, many investigators attempted to build models to study neural networks. These models fall into two categories. First one is biological modeling, where the goal is to study the structure and function of real brains in order to explain biological data on aspects such as behavior. In technological modeling, the goal is to study the brains in order to extract concepts to be used in new computational methodologies. There are points of contention on which of these two branches should constitute the true focus of research in neural network modeling. If the goal is to advance our understanding of biological intelligence, then the validity of the models should be corroborated with experimental evidence. However, many scientists and engineers are usually content with models inspired by brain function. Although this controversy continues, the latter viewpoint is taken by several investigators working in the area of artificial neural networks (ANN) and neuro computers.

The two objectives of research in ANN's may be paraphrased as follows.

1. To understand how the brain imparts abilities like perceptual interpretation, associative recall, common sense reasoning and learning to humans. Toward this goal it is necessary to understand how computations are organized and carried out in the brain. These computations are of different kind than the formal manipulation of symbolic expressions.
2. To understand the subclass of neural network models that emphasize computational power rather than their biological fidelity. To achieve this object,

it is admissible to incorporate features in a model even if those features are not neurobiologically possible.

The restricted view taken by technological modelers also has a turbulent history. In the early days of computers, two philosophically opposing views of what computers could be emerged and struggled for recognition. One school believed that both minds and digital computers are symbol-manipulating systems. Symbolic logic and programming became the tools of their trade. The opposing school felt that the ultimate goal of computation is better by modeling the brain itself rather than manipulating the mind symbolic representation of the external world. Stated differently, the symbol-manipulating school believed that the problem-solving process is essentially algorithmic. Although initial demonstrations proved the viability of both these approaches, the brain modelers lost some ground when digital computers were successfully used in 1956 by Newell and Simon to solve puzzles and prove theorems. By this time Rosenblatt also succeeded in building a device, called the perceptron, and demonstrated the viability of the opposing school. In the decade that followed, this school received a severe blow when in 1965 Minsky and Papert claimed that the perceptron approach is fundamentally flawed.

2.2 Motivations for ANN Research

The recent renewal of interest in ANN's is prompted by advances in computation technology as well as a deeper understanding of how the human brain works. One motivation is a desire to build a new breed of powerful computers to solve a variety of problems that are proving to be very difficult with conventional digital computers. Cognitive tasks such as recognizing a familiar face, learning to speak and understand a natural language, retrieving contextually appropriate from memory, and guiding a mechanical hand to grasp objects of different shapes and consistencies are some examples that come to mind quickly. Problems of

this kind typically involve pattern recognition under real world conditions, fuzzy pattern matching, nonlinear discrimination, or combinatorial optimization. That is, these tasks are analogous to those typically performed by our brain, and are beyond the reach of conventionally programmed computers as well as rule based expert systems.

Another motivation behind the spurt of activity in this area is a desire to develop cognitive models that can serve as a foundation for artificial intelligence. It is well known that the brain is not as good at performing arithmetic operations as a digital computers. But it is good in association, categorization, generalization, classification, feature extraction, and optimization. These capabilities fall under three main categories, i.e., searching, representation, and learning. These aspects are closely related to the associative property and self organizing capability of the brain. By associative property, we mean the capability of recalling an entire complex of information by using a small part of it as a key to a search process. The brain does this remarkably well. By self organizing, we mean the ability to acquire knowledge through a trial-and-error learning process involving organizing and reorganizing in response to external stimuli.

A device that is familiar to electrical engineers and that behaves somewhat like a neuron is the electronic analog operational amplifier, configured as an integrator. If one can visualize hundreds of these integrators interconnected together through potentiometers, then what we had a crude model of an ANN. Aside from technological challenges, there are number of problems of theoretical nature that need to be addressed and solved if ANNs have been built, each with a different architecture and each aimed at solving at different problem, there seems to be a lack of commonality in their theoretical basis. Similarly much remains to be learned about learning algorithms. In spite of these uncertainties, a forceful motive behind the current enthusiasm for artificial neural networks is in the promise they hold in solving the diversity of hard problems.

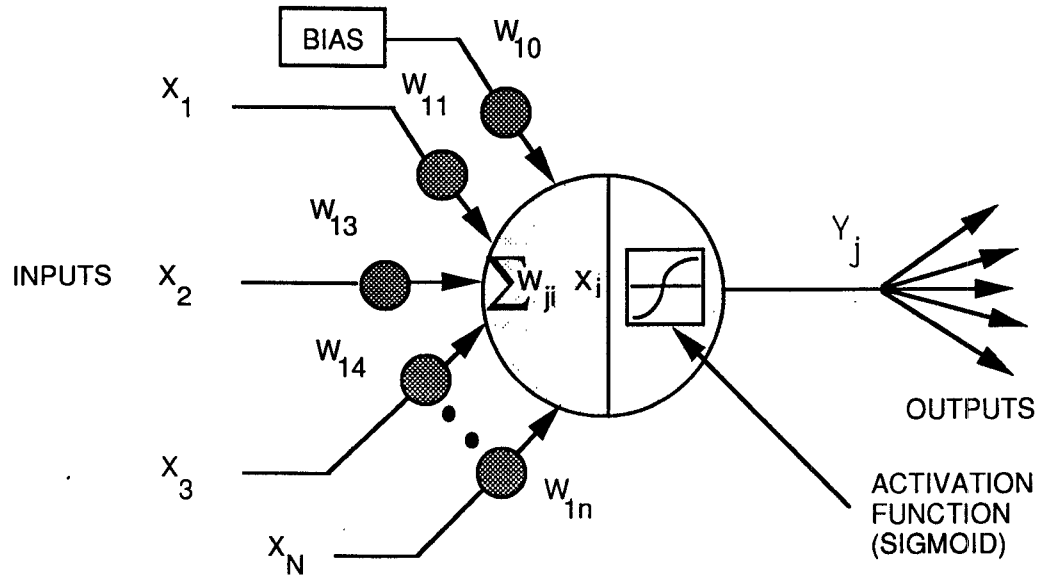


Figure 2.1: The General Aspects of Neural Networks

2.3 Introduction to Neural Networks

Kohonen defined neural networks in 1988 as "Artificial neural networks are massively parallel interconnected networks of simple, usually adaptive, elements and their hierarchical organizations which are intended to interact with the objects of the real world in the same way as biological nervous system do".

The general aspects of neural network is illustrated in Figure 2.1 There are a set of processing units (neurons) indicated by circles in the figure. At each point in time, each unit has an activation value $a(t)$. This activation value is passed through a function to produce an output value $Y(t)$, which will pass through a set of unidirectional connections (shown by arrows in Fig. 2.1) to other units in the system, so that

$$Y_i = F_i[a_i]$$

There is associated with each connection a real number called weight or strength $W(t)$ of the connection, which determines the amount of affect that the first unit on the second unit. Now all inputs with their weights are combined by some operator, usually additor, to yield

$$net_i = \sum W_{ij}(t) * Y_j(t)$$

The new activation value is now determined through a function “f”, which is a function of the current activation value and the combined inputs, so that

$$a_i(t) = f_i(a_i, net_i)$$

These neural networks are viewed as plastic, as the patterns of interconnection are not fixed all of the time. A neural network learns by undergoing weights modification as a function of experience. In this way (we can assume) a neural network system is evolved.

McClelland and Rumelhart proposed eight aspects of a neural network or parallel distributed processing model [McRu 86]. Each one of them is explained briefly in the following paragraphs.

1. Set of Processing Units

The whole processing of a neural net is done by these units. Each unit’s job is to receive inputs from its neighbor’s to compute an output value based on a function of the input it receives and to send the output to its neighbors. The system is inherently parallel i.e., all units carry out their computations at the same time. These units are divided into input, hidden and output units. Input units receive inputs from sources external to the system under study. These inputs may be either sensory input or inputs from other parts of the processing system in which the model is embedded. The hidden units are those whose only inputs and outputs are within the system. They are not visible to outside systems. The output units send signal out of the system.

2. The State of Activation

The state of the system at any time 't' is specified by a vector of n real numbers $a(t)$, which represents the pattern of activation over the set of processing units. Each element of the vector represents the activation of a unit at time 't'. Different models make different assumptions about the activation of a unit. Activation values may be continuous or discrete.

3. Output of the Units

Associated with each unit, u_i , there is an output function $F_i(a_i(t))$, which maps the current state of activation, a_i , to an output signal, $Y_i(t)$. So that

$$Y_i(t) = F_i(a_i(t))$$

In some models 'F' is a threshold function, so that a unit has no affect on another unit unless its activation exceeds a certain value. In some other models 'F' is assumed to be a stochastic function in which the output of the unit depends in a probabilistic fashion on its activation values.

4. The Pattern of Connectivity

The pattern of connectivity of the system constitutes what the system knows and determines how it will respond to any arbitrary input. In many cases, the total input to the unit is simply the weighted sum of the separated inputs from each of the individual units. Thus the pattern of connectivity can be represented by merely specifying the weights for each of the connection in the system.

5. The Rule of Propagation

This rule combines the output vector $Y(t)$, the output values of the units, with the connectivity matrices 'W' to produce a net input for each unit i.e.,

$$net = W * Y(t)$$

6. Activation Rule

The net inputs of each type impinging on a particular unit are combined with one another and with the current state of the unit to produce a new state of activation in accordance with a rule 'F'. For example, assuming 'F' is an identity function and all connections are of the same type. Then,

$$a(t + 1) = W * Y(t) = net(t)$$

We can assume 'F' to be a sigmoid function, when $a(t)$ is assumed to take on continuous values. In this case, an individual unit can saturate and reach a minimum or maximum value of activation.

7. Learning Rule

A learning rule is used to change the weights of interconnections. In principle there are three kinds of modifications: They are

- The development of new connections
- The loss of existing connections
- The modification of the strengths of connections that already exist

The first two can be considered a special case of last one. This is due to the fact that when we change the strength of connection away from zero to some positive or negative value, it has the same effect as growing a new connection. When we change the strength connection to zero, it has the same effect as losing an existing connection. There are many kinds of learning rules. Virtually all of them can be considered a variant of the Hebbian learning rule suggested by Hebb, in his classic book "Organization of Behavior (1949)". His basic idea is: If a unit u_j receives a input from another unit u_i , then if both are highly active, the weights W_{ij} from u_i to u_j should be strengthened.

8. Representation of the Environment

This refers to a model of the environment in which this model is to exist.

We often represent the environment as a time-varying stochastic function over the space of input patterns.

2.4 Existing Methods to Monitor the Machining Processes

In an untended machining, the knowledge of tool state plays a central role in the process diagnostic system. In most of the machining processes, the primary method of producing form and dimensions on a workpiece is the removal of material from the work piece using edged cutting tools. During this process the tool also loses some amount of material, which is referred as tool wear. Tool wear is undesirable as it gives poor quality of the machined surfaces and causes unpredictable changes in work piece geometry. A cutting tool may be removed from service as a result of either gradual wear on the faces of the tool or catastrophic failure resulting from sudden fracture or high temperature weakening of the tool. The monitoring of gradual wear requires the development of sensitive, accurate, and reliable devices.

On-line monitoring and compensation of the tool wear would be of a great help to avoid the increase in cutting force, loss of accuracy, deterioration in surface finish, increase in cutting temperature and increase in vibration due to tool wear. Also it would result in the improved quality of the product. One of the major problems arising in the development of on-line monitoring of a machining process, is the correct choice and availability of the tool wear sensors that should have accuracy, reliability and be economical at the same time. The tool wear can be monitored in various ways. One method to monitor the tool wear is off-line tool wear monitoring. Off-line monitoring is preferred by a person (using a microscope) or automatically by sensors that measure the tool geometry or the amount of tool particles in chips. Since it involves a time delay in tool wear measurement, they are not suitable for on-line applications. Tool life algebraic equations have

been widely used for more than eighty years and one of that equation is Taylor's tool life equation [Ta 06]. This equation is based only on the final state of tool wear and gives the period of time during which the cutting tool, under certain cutting conditions, will develop a predefined amount of wear. This relation does not describe the state of the tool wear during cutting. This limitation prevents their usage in on-line applications such as dimensional compensation, tool breakage prediction and adaptive control and they are useful only in the pre-process calculation of the tool changing times. Tool wear can also be determined by differential equations that are derived by off-line measurements [Ko 78, Ru 76], which uses the whole cutting process for tool wear measurement. But the accuracy of these estimation depends purely on completeness and correctness of the whole cutting process formulation which is very hard to obtain.

The other ways of monitoring the tool wear are direct and indirect methods. Although both methods use sensors to monitor the tool wear, but they differ in some aspects. In the direct method, sensors directly measure the tool wear, such as optical scanning technique, electrical resistance, radioactive technique, measurement of tool geometry, change in work piece size, and analysis of tool wear particles in the chips. The optical scanning technique is limited because of the uncertainty of the wear zone and the production environment (i.e., coolant, chips etc.). The electrical resistance involves special preparation of the tools which is costly and inconvenient and radioactive technique has a potential danger of radioactivity. The measurements about the change in workpiece dimensions may be prone to error because of the effect of temperature rise, deflection due to cutting forces and inaccuracies in the machine tool structure.

The above difficulties lead to the indirect measuring techniques to measure accessible process variables (such as, cutting forces, machine tool vibrations, acoustic emission etc.), which are related to tool wear and correlating the changes in these parameters to the change in the tool wear. Changes in the cutting force because of gradual wear of the tool is usually very small to be used for any

accurate and reliable correlation between the two. Indirect methods fall into two categories: static methods and dynamic methods. The static methods use some static characteristics of the monitoring signal such as the mean, the rms etc. Unfortunately, static methods are often too sensitive to the variation of the cutting conditions. Therefore, the dynamic methods are developed, which use the dynamic characteristics of the monitoring signal to identify the tool condition. However the dynamic methods have two major problems. First, the estimation of the process model requires heavy computation; the second, the calculations usually involve a non-linear search, which may not give accurate results.

The majority of the techniques discussed above for tool wear sensing is limited due to inadaptability to on-line monitoring, influence of external disturbances unrelated to tool wear on the measured signal, and inability of instrumentation to operate reliably in the immediate vicinity of the cutting process.

Investigations have been carried out by several investigators on various tool wear sensors based on radioactive isotopes [MeEr 53], measurement of the radioactivity of the activated cutting elements of the tool during machining [Ar 83], etc. These sensors are basically used to determine the tool life. Jetly developed a pneumatic system for tool wear sensing [Je 84], by the fact that back pressure increases with the decrease in distance between flapper and nozzle through which high pressure air is passed. Suzuki and Weinmann [SuWe 85], have developed on-line tool wear sensors by measuring the change in distance between the tool and the workpiece using a stylus mounted on the tool holder. The stylus movement is sensed by a displacement transducer.

Sajanwala etc. describes the design and testing of a pneumatic feedback system that can be mounted on a center lathe to improve the dimensional stability during turning by on-line tool wear sensing and compensation [SaCh 89]. The proposed system consists of a pneumatic sensor to sense the tool wear during machining, a pilot-controlled direction controlled valve to amplify the signal obtained from the pneumatic sensor and an actuating mechanism to move the tool

for compensation of dimensional inaccuracies.

George Chryssolouris presented an approach to the operation of manufacturing processes which is based on game theory [GeVe 91]. The input settings of the process are selected by evaluating, on-line, a set of feasible alternatives with respect to several criteria. Relevant performance measures such as process cost and production rate can be directly influenced in this approach by establishing appropriate definitions for the decision criteria. He compared PROcess DEcision MAKing (PRODEMA) approach is compared with an adaptive control (AC) scheme through simulations of a turning process. Results show that PRODEMA provides the ability to operate a process in accordance with overall performance measures.

2.5 Significance of Neural Networks in On-line Monitoring

For better on-line monitoring of machining processes, it is necessary to have the actual process of tool wear. Tool wear can be estimated from the signals of sensors. In order to retrace the tool wear information from the measured signals, it is necessary to form the mathematical model of the cutting process. These formulations may be based on either empirical analysis or physical models. The empirical approaches are limited by their nature and applicable only in some specific conditions. But the physical models are applicable in wider range but they are restricted by the difficulties in formulating the accurate models. Empirical relationships for the tool wear states is not useful in on-line monitoring of machining process as the inherent variability property of wear process of tool. So it is not possible to estimate the actual wearing process of the tool, if we cannot formulate the accurate model of the cutting process, as it depends on the so formed model.

Intelligent evolution of sensor information is necessary to perform the diag-

nostic problem, as it is difficult to have the actual physical model to describe the state of the process. Sensor feedback is essential if automated machines are to produce the correct size and surface finish the first time. But unmanned factories will also demand integrated systems that replace the constant supervision of skilled machinists. The solution for this problem can be solved by using neural networks. Neural networks have long been studied in the hope of finding solutions for problems with unknown or complex internal relationships. A neural network takes an input numeric pattern and outputs an output numeric pattern. Adaptation, or the ability to learn, is the most important property of neural networks. Neural networks have capabilities of fast learning and pattern recognition i.e., they work like a human brain. In neural networks, knowledge is stored in the connections between each neurode which is used for pattern recognition. This obviates the requirement for searching for patterns in a separate knowledge base and results in pattern recognition systems which are practical in real time environments. Neural networks are able to make decisions based on noisy and incomplete information.

A neural network can be trained to map a set of input patterns onto a corresponding set of output patterns simply by means of exposure to examples of the mapping. This training is performed by a gradient descent algorithm, which gradually adapts the internal weights of the network, so as to reduce differences between the actual network outputs, for a given set of inputs and the desired network outputs. Neural networks which learn mappings between sets of patterns are called Mapping Neural Networks. A key property of mapping networks is their ability to produce reasonable output vectors for input patterns outside of the set of the training examples.

In neural networks the information contained in the input is recoded into an internal representation by the hidden units (higher level feature detectors), which perform the mapping from input to output. This eliminates the non-linear relationship physical models of tool wear of the cutting tool, which are very difficult

to obtain. A set of weights that we obtained after training the network, can be used to on-line monitor the machining process. Larger the training set, the more accurate the estimate. Intelligent functions like learning, efficient knowledge representation and retrieval, pattern recognition and generalization will make the neural networks as the best alternative for on-line monitor.

2.6 Applications of Neural Networks

The word ‘applications’ should be treated with care. In evaluating an application, it is important to be aware of how well developed an application is at the time of evaluation. The categories of applications are:

- Candidate application - are those problems that could, in principle, be solved by the type of technology that neural networks offer. In this type, the problem has been identified and the problem requires a mapping or optimization that is similar to other problems addressed by neural networks.
- Applications underdevelopment- are those problems for which studies have been performed or are underway. In these studies, a neural network is usually trained to learn a simplified version of the problem first, and then expanded to address the entire problem.
- Proven applications - are those for which neural networks are actually used to solve problems. Such proven applications of neural networks by looking at the network as a pattern-mapping systems that could be placed in a wide variety of domains-medicine, manufacturing, image systems, speech systems, autonomous control, and diagnostics, among others. The important applications are paraphrased as follows.

Oleg Jakubowicz and Sridhar Ramanujam of the State University of New York, Buffalo, have developed a neural network system intended to help technicians to identify faults in circuit boards. This network not only helps to identify

the failure but also directs technicians to the next appropriate board test. They used unusual neural network architecture reminiscent of a counterpropagation network. It consists of an input layer, Kohonen middle layer, and an output layer. The input layer of the network comprises a set of binary (on-off) neurodes and has one such neurode for each symptom that can be exhibited by the tested board. In addition, a series of other input neurodes identify pins on the board observed to be good or bad by the technician. This input layer is then fully connected to a second, self-organizing Kohonen layer. The Kohonen layer is used to develop a self-organizing set of feature maps, that map the various symptoms found in the board. The Kohonen network models the input data in such a way that the input patterns are clustered into groups of symptoms. During the training of this layer, the overall output of the network is disregarded. Once the Kohonen middle layer is trained, the weights between it and the input layer are frozen and training begins on the network's third (output) layer. The output layer is essentially a back-propagation network layer that uses a delta-rule training procedure. The training procedure for the third layer involves presenting the input patterns to the network and letting the Kohonen layer generate its learned-feature map for that symptom combination. This feature map is transmitted to the output layer, where the resulting network output is compared to the desired output. The standard delta rule is used to adjust the weights. Essentially, the output layer's task is to learn the association between the symptom feature maps as well as the probable causes for those symptoms.

G Chryssolouris, M Lee, J Pierce, M Domroese have used neural networks for the design of manufacturing systems [ChLe 90]. The design of manufacturing systems is often performed by means of simulations. Usually, the parameters of a model of the system are varied by trial and error until the simulated performance of the model reaches a desired level, as expressed by some combination of performance measures. They used neural networks to learn the inverse of the simulation function: given desired performance measure levels, the neural

network outputs appropriate values for the system parameters. This eliminates highly iterative guesswork from the manufacturing system design process. This approach is applied to the resource requirements design problem, that is, determining the appropriate number of resources for each work center of a job shop. They used a feed forward neural network with back propagation technique to train the network. Results show that neural networks are capable of learning the mapping from desired performance measures to suitable designs.

Liu and Ko used artificial neural networks for on-line monitoring of drill wear [LiKo 90]. The authors assumed that the drill wear condition can be observed indirectly by measuring both thrust and vertical acceleration during the drilling process. The input vector of the neural network is obtained by processing the signals of both the thrust and vertical acceleration while the output is the wear states. The learning process of the neural networks is done by using back propagation technique. The results of using single kind of sensor and sensor fusion were discussed and compared. They achieved a success rate over 95 % for on-line classification of drill wear.

Computer-integrated manufacturing requires models of manufacturing processes to be implemented on the computer. Mechanistic models developed from the principles of machining science are useful for implementing on a computer. However, in spite of the progress being made in mechanistic process modeling, accurate models are not yet available for many manufacturing processes. Empirical models derived from experimental data still play a major role in manufacturing process modeling. Generally, non-linear regression techniques are used for developing such models. However, these techniques suffer from several disadvantages like the structure of the regression model needs to be decided a priori. S Yerramareddy etc. developed empirical models from experimental data for a machining process [SuSt 91]. In this paper the authors compared non-linear regression models with ANN models to assess the applicability of ANN as a model building tool for computer-integrated manufacturing.

NETTalk, developed by Sejnowski and Rosenberg [SeRo 87] at Johns Hopkins University, learns to translate segments of English text into phonetic notation for pronouncing the text. The phonetic notation that is output can be given to a speech generator and pronounced out loud automatically. The network consists of three fully interconnected layers. The input layer has 203 input units, the hidden layer 80 units, and the output layer 26 units. The bottom layer inputs English text, and the top layer is presented with pronunciation notation for sounds to be pronounced from the input text. The NETTalk study illustrates the extent to which pronunciation rules can be mastered by a three-layer back-propagation net with only a single feature detection layer.

Several organizations are trying to apply neural networks to information-processing problems in commerce and industry that have proved intractable or far too expensive with algorithmic computers. Preliminary results have been encouraging.

Behavioristics Inc., Silver Spring, Md., has demonstrated a neural network for scheduling airline flights. Airline sell seats at different fares depending on how far in advance a reservation is made. The system, called the Airline Marketing Tactician, optimizes over time and allocation of seats between discount and standard fare classes, to maximize the airline's profit. At present, several airlines are considering the system.

Murray Smith has shown how accurately neural networks can score applications for bank loans [KeTi 91]. A neurocomputer loan-scoring application has been developed for a major finance company, which plans to install it in the field.

Jeffrey Elman, associative professor of linguistics of the University of California at San Diego, has shown that neural networks can pick individual words out of connected streams and devise representations for them. The neural network speech-recognition system with the highest accuracy and largest vocabulary was developed by Teuvo Kohonen, research professor in technical physics at the Helsinki University of Technology, under contract to Asahi Chemical Co. of

Tokyo, Japan.

Kunihiko Fukushima, a senior research scientist of NHK Laboratories in Tokyo, and Sei Meyake, a research director at the Automated Telecommunications Research Center in Osaka, have demonstrated a network, the Necognitron, that can identify hand-printed characters with 95 % accuracy, regardless of shifts in position, changes in scale, and even small distortions.

Adaptive control problems are also solved using neural networks. Andres Pelionez, research associate professor of physiology and biophysics of New York University, New York City, who has demonstrated the ability to move a finger of a robot arm in a beautifully coordinated perfect straight line at any angle.

Conclusions

It is concluded that neural networks can be applied in a number of research areas. One can see from above, that neural networks can become a viable alternative to modeling tool. Consequently we were encouraged to apply neural networks in the area of manufacturing.

Chapter 3

On-Line Monitoring Methodology

Overview

A basic methodology of on-line monitoring of a turning process is discussed in this chapter. This chapter is divided into four sections. The first section discusses the proposed on-line monitoring system for a turning process. The second section describes the adaline model used to find the relationship between force and cutting parameters (feed, cutting speed, and depth of cut) and results are discussed. The third section describes the feedforward back-propagation network, which is used to predict the tool wear and surface roughness. The fourth section discusses about the machining of advanced ceramics and applications of neural networks in ceramics machining.

3.1 Description of an On-line Monitoring System

The productivity of the factory of the future and the quality of the products it produces will, in part, depend on the intelligent utilization of computers to monitor and control the machines that produce the products. This demands that the supervising personnel and the computer systems should provide with accurate and reliable information about the state of the machines and the quality of the operations they perform. This requires, therefore a much better understanding of the machining processes and the functions they perform, so that sensors and the associated decision-making systems can be designed to provide the necessary information.

In recent years, there has been an increase in the demand for high quality products. Consequently, the manufacturing engineers are faced with the difficult task of improving the productivity without comprising quality. The complexity of modern day components adds to the difficulty of the task. The use of high machining speeds to increase productivity exacerbates the stability problems and might even lead to tool breakage in certain situations. This emphasizes proper control of the machining process through on-line monitoring. The success of manufacturing systems with a high level of automation and flexibility is the capability to strictly control the quality of the product, to guarantee working processes with a known reliability and availability of the whole system. The high flexibility required of manufacturing systems also involves increasing severity of operating conditions, which the various parts of the system are subjected to. The state of the cutting tool plays an important role in the manufacturing systems for metal-cutting machining. According to Balakrishnan, one can achieve a fully untended machining if one can monitor the following tasks successfully [BaMa 85]:

1. Workpiece monitoring (eg., surface finish, automatic setup, gauging)

2. Tool monitoring (eg., tool life sensing and its breakage, offset measurement)
3. Process monitoring (eg., adaptive control, chip congestion detection)
4. Machine monitoring (eg., vibrations of gearbox, spindle and table)

A machining process is said to be fully monitored if one can successfully implement the first two cases i.e., to increase the efficiency of the manufacturing process, one has to know the tool life and its reliability and quality of the product. Decision concerning optimal tool replacement policies for new manufacturing cycles must be taken as quickly as possible in order to minimize the tool change time and maximize the production rate.

An on-line machining process monitoring is a decision-making process to determine the instant when a tool change is necessary so as to assure the product quality or to avoid tool breakage. It can be expected that the control strategy for this decision making process will be knowledge intensive. The task to control the on-line monitoring is complex because of its physical contexts. A high-performance computer program has to be developed to automate such a decision making process. The recent advances in parallel computing, and the development of hardware neural networks, endow artificial neural networks (ANN) with potentially fast computational capabilities. By considering these advantages, a neural network is implemented as an on-line monitor to machining processes, as shown in Figure 3.1.

A neural network monitor developed to monitor a machining process is divided into three stages:

The first stage is selection of a sensor(s)(eg., a force transducer). By using this sensor(s), one can reliably monitor the machining process. From section 3.3.4, we stated that a machining process can be monitored efficiently by studying the force signals from the cutting operation. So the force ratio, apart from speed, feed, depth of cut, nose radius and time, is presented to input layer of a feedforward neural network.

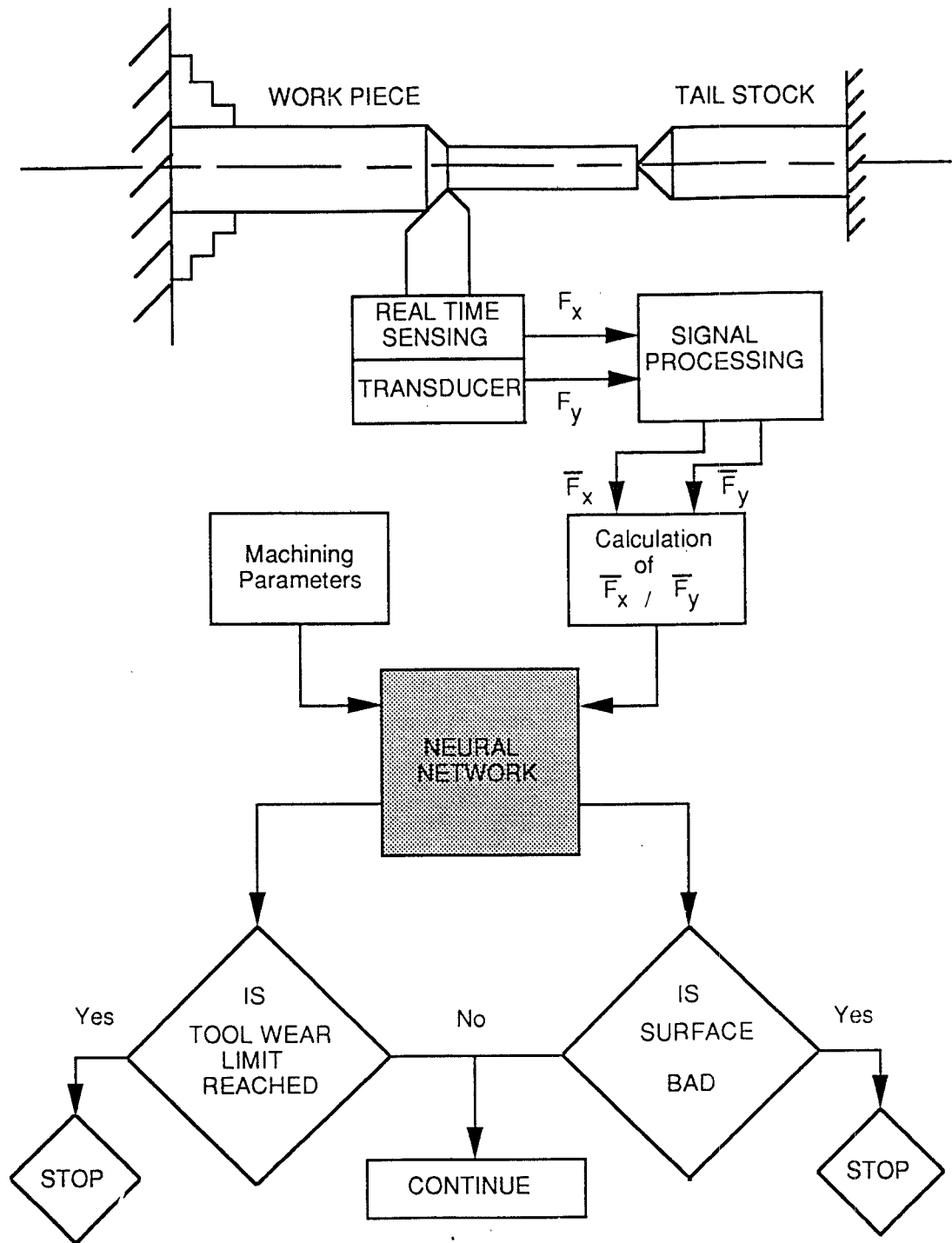


Figure 3.1: Block Diagram of an On-Line Monitoring of a Machining Process

The second stage is training the neural network, which is done off-line. After selecting a proper neural architecture, the network is trained with the training data examples where each example (or pattern) consists a pair of input and output variables. Once the root mean square (RMS) error for all training patterns has reached a predetermined value (or the number of iterations reached some fixed value), training is stopped and weights of the node connections and the node thresholds are frozen. These weights and thresholds are used for on line monitoring of machining processes. The values of connection weights and thresholds depends on the training data. More the training data, better the network performance and also the performance depends on the quality of the training data.

The third stage is to implement the network in real-time machining operation, as shown in Fig. 3.1. The force signals from sensors are low pass filtered and force ratio (feed force to tangential force) is determined. This ratio, along with the other machining parameters, is fed into the trained neural network and tool wear and surface finish are predicted. By looking at the predicted values, the human operator (or control system) can act accordingly to optimize the machining process, i.e., If any one of the output parameters (i.e., tool wear or surface finish) has reached the target value, then the operator (or control system) can stop the machining process or change the tool. The neural network can be referred to as an inference engine, to effectively find a major tool wear mode when the detected signal has been traced for a certain time. When a successful monitoring is completed, the inference engine should take this information in the working memory (training phase) and compare it with similar rules previously stored in the knowledge base. The neural networks allows incremental improvement as new data is made available. Therefore, the network constantly keeps the information stored in the database updated. As the reliable information is accumulated, more aspects of the true nature of the machining process will be revealed.

3.2 Identification of Relationship Between Force and Cutting Parameters

Cutting force and cutting speed are two main quantities for an efficient machining operation, because, their product is proportional to the power at the cutting edge which in turn determines the metal removal rate and the required capacity of the machine tools. Cutting force plays as significant a role in machining as do cutting speed and tool life. Metal removal rate per h.p is very often taken as an indicator of productivity. It is, however, very little realized that the metal removal rate per h.p and unit cutting force are inverted concepts, i.e., metal removal rate per min per h.p is talking only about the unit cutting force. Consequently, the close tie between metal removal rate per h.p and unit cutting force proves the significance of the cutting force in metal cutting for both machine shop and research department. The lower the unit cutting force, the better the machinability of the material in question. Because, the larger the volume of the material removed per min per h.p, the better its machinability. So one has to analyze the cutting forces in order to know the machinability of the material in question.

It is well known, that the cutting forces developed during machining can be controlled by varying the cutting parameters, i.e., cutting speed, feed and depth of cut. While a deep cut and a small feed (i.e., a large slenderness ratio) increases the unit cutting force and cutting speed [Kr 66]. An increase in the permissible cutting speed is an advantage and an increase in cutting force however is a disadvantage because no increase in metal removal results from it, but rather larger deformations of the machine and workpiece takes place. The combined increase of cutting speed and cutting force caused by a large slenderness ratio requires also an increase in machine power. A deep cut favors tool life (for a given cutting speed), a shallow cut favors the metal removal rate of the machine and its accuracy. To find the relationship between the cutting force and cutting parameters,

we developed a model using ‘ADALINE’ neural network, whose description is as follows:

3.2.1 Introduction to Adaline Network

The Adaline (ADaptive LINEar Element) network was developed by Bernie Widrow at Stanford University shortly after Rosenblatt developed the perceptron. This network is more or less similar to perceptron. Adaline uses a slightly more sophisticated learning procedure than perceptron which is known as Widrow-Hoff Least-mean-squared (LMS) learning rule. Also it is known as delta rule (as it tries to minimize the Delta or Difference between the actual and observed output of the same neuron). As mentioned above the basic difference between adaline and a two layered perceptron is the learning law, i.e, adaline learns using L.M.S error reduction where as perceptron uses a simple weighted difference ($\Delta(i) = \text{target_activ}(i) - \text{output_activ}(i)$). But the real difference between them is the way the learning laws are applied (i.e., calculation of the delta). In perceptron delta is calculated using the real (binary) output of the node and the desired output, where as in adaline, delta is calculated using the neural activation before the binary transformation is applied. This allows the weights to change in a way that is more sensitive to the real distance between the neurons activation and the desired output. Adaline is used for both continuous valued and binary inputs.

Architecture of Adaline and Madaline Networks

The adaline and madaline structure are similar to the structure of perceptron, as shown in Figure 3.2. A few differences between them are at the micro-structural level. In perceptron, hard limiting non-linearity is used whereas in adaline it is replaced by a threshold logic non-linearity, where they add a variable threshold instead of subtracting it (but since the thresholds can be positive or negative, this

is just a difference in writing down the equations, not in any real performance). They both have binary transfer function applied to the sum of the inputs times their weights plus a threshold value, yielding (-1,1) values for the nodes.

At Meso-Structural level, the adaline is limited to a single output node and the madaline can have many. They are each two-layered networks.

Dynamics of Adaline and Madaline structures

The dynamics of adaline and madaline is similar to perceptron and it is quite simple. We begin by presenting a pattern. This means that if the input layer for the network is 'n' units long, we input a vector of length 'n'; one value for each neuron in the input layer. Now we move to next layer up. For each neuron in this layer, we calculate an input which is the weighted sum of all the activation from the first layer. The weighted sum is achieved by vector multiplying the activations in the first layer by a connection weight matrix. This is not the activation of this neuron, though. To obtain the activation, we subtract a threshold value (which is fixed for each neuron), and apply a transfer function. The transfer function is defined for each different network. In the case of adaline it is binary, real or threshold-logic transfer function as shown in Fig. 3.2.

Thus the activation of the j^{th} neuron in the second layer would be

$$Y_{activ_j} = F(W_{ij} * input_{activ_i} - \theta_j)$$

where,

Y_{activ_j} = activation of the j^{th} neuron in the second layer

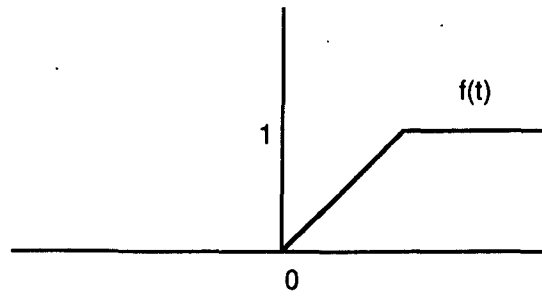
F = transfer function

W_{ij} = connection weight between i^{th} input node and j^{th} hidden node

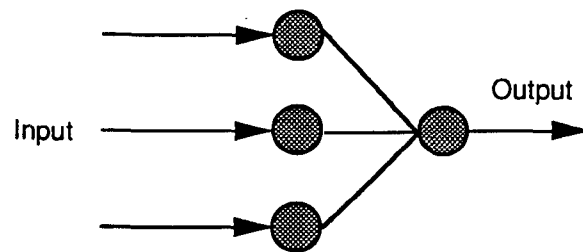
$input_{activ_i}$ = input to the i^{th} neuron in the input layer

θ_j = threshold of the j^{th} neuron in the second layer

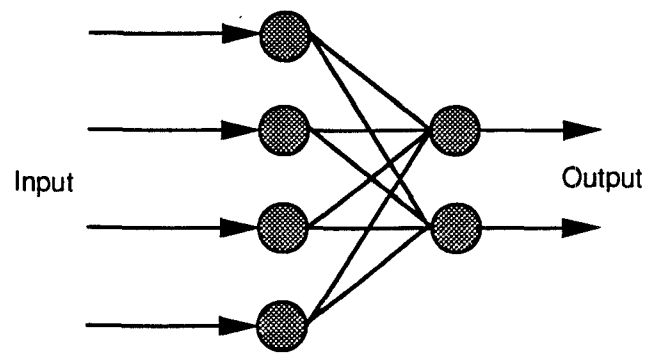
If more than one layers are used, this process is repeated, with each layer acting as the input layer to the layer just above it. As a result of differences



(A) Micro-Structure: Threshold Function



(B) Meso-Structure: Adaline



(C) Meso-Structure: Madaline

Figure 3.2: The Meso and Micro Structure of an Adaline and Madaline

among the value of the connection weights, different inputs can produce different outputs in the top most layer.

The learning Law for the Adaline

After presentation of a pattern, and passing signals through the network in the usual manner, we can observe the actual output value at each node in the output layer and compare it with the targeted or desired value. The L.M.S error 'E' is defined as,

$$E = \frac{1}{2} * \left(\sum_{j=1}^N (target_activ - output_activ)^2 \right)$$

where,

target_activ = targeted activation at output node j

output_activ = actual activation at output node j

N = total number of nodes in the output layer

In the above case, E represents the total error observed over the entire layer of output nodes. This is the error we want to minimize.

3.2.2 Adaline Model

Adaline model can be applied in many fields. In the following we will discuss how it can be applied to find the relationship between cutting force and cutting parameters (i.e., feed, depth of cut, and speed) in a turning operation. We developed a program that trains an adaline unit using the Delta Rule. This program deals with single adaline (not madaline as we have only one output i.e., force) trying to learn to get the desired relationship between force and cutting parameters, like a human brain. Here we assumed a linear relationship between force and cutting parameters as adaline gives better results for linear networks. The structure used for adaline network is illustrated in Figure 3.3. First we will train the network (i.e., adaline) with sixteen different sets of patterns, as shown

Feed (mm/rev)	Depth (mm)	Speed (rpm)	Force (N)
0.1	0.5	100	1533.738
0.1	0.5	100	1533.762
0.3	0.5	100	1601.835
0.3	0.5	100	1600.863
0.1	1.0	100	3067.872
0.1	1.0	100	3055.482
0.3	1.0	100	3204.723
0.3	1.0	100	3170.703
0.1	0.5	200	1534.827
0.1	0.5	200	1545.876
0.3	0.5	200	1604.982
0.3	0.5	200	1636.524
0.1	1.0	200	3067.062
0.1	1.0	200	3048.018
0.3	1.0	200	3203.277
0.3	1.0	200	3142.812

Table 3.1: Cutting Force Data

in Table 3.1. Each pattern will have three input parameters i.e, feed, depth of cut, and speed, and one output parameter i.e, force.

The visual patterns are simulated as 3×1 patterns on a retina i.e, the adaline processing element has a total of 3 inputs (i.e, feed, depth of cut, and speed) to deal with. Rather than treat the desired output as a mentor input of weight one, as explained above, this simulation simply keeps the correct answer in a separate array for consultation when adjusting the weights.

First step in the simulation is to initialize the weights with random values (make sure that all weights should not be equal to one or some positive value). Next step is to initialize the range of the parameters. This is useful to normalize

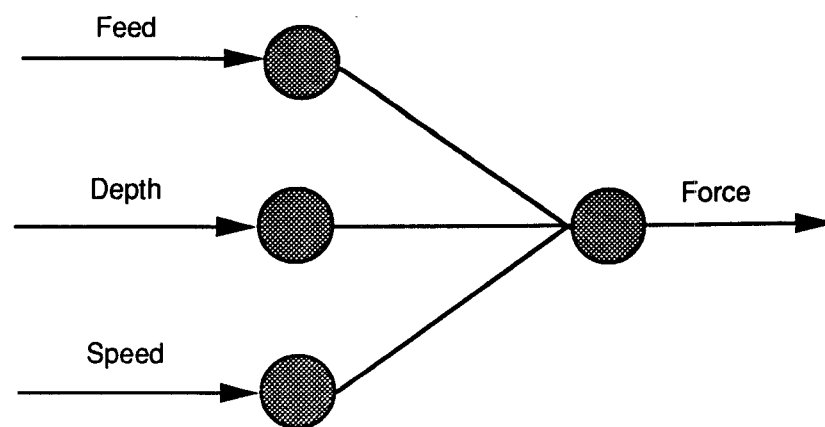


Figure 3.3: The Architecture of Adaline Network

the data between 0 and 1. Normalization of data is done because we have different parameters with different units. Therefore it is difficult to find the relationship between them as we have to simulate the data with these parameters. So to get rid of this problem and to have a better relationship, we will make each parameter value lie between 0 and 1 (it is a ratio i.e, no units).

Next step is to initialize the inputs . Here we will normalize all the input and output data between 0 and 1. Now we train the adaline by presenting it with the first pattern i.e., with three input parameters and one output parameter (which is the desired value of the force) and compute the output using the relation,

$$Actual_output_i = W_{1,i} * feed + W_{2,i} * depth + W_{3,i} * speed$$

where,

i = number of pattern (i.e., example)

$W_{j,i}$ = weights of interconnections and j varies from 1 to 3.

Now we will compare the actual output with the desired output. If the error 'E' ($E = \text{desired output} - \text{actual output}$) is less than the permissible, there is no need to adjust the weights otherwise we need to adjust the weights according to the Delta Rule until we get the error less than the permissible i.e., until it recognizes the pattern with the correct output. The weights are adjusted as follows,

$$W_{new} = W_{old} + \eta * E * X$$

where,

X = input vector

η = learning constant

After recognizing the first pattern, the second pattern is presented and the weights are adjusted using the Delta Rule until it is also recognized (i.e., the error is less than the permissible). At this point we need to confirm that adjusting the weights for the second pattern did not ruin adaline's recognition of the first pattern. If it does not correctly remember the first pattern, we readjust

the weights using the Delta Rule until it recognizes the first pattern. Then with this current weights, we present the third pattern and the weights are adjusted. Again we want to recheck that if it recognizes the first two patterns. So we will present again the first one and find the error. If this error is greater than the permissible, we readjust the weights until the error is less than the permissible. After that the algorithm will do the same thing for the second pattern. This process continues for all sixteen patterns in the training data.

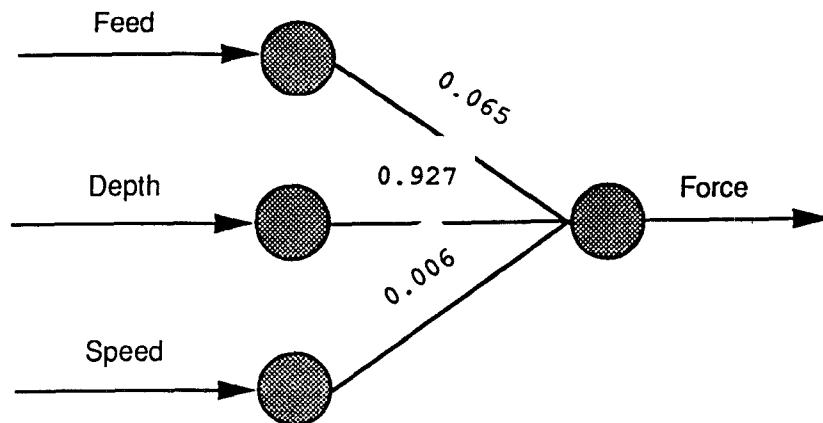
After the check and re-check process for all sixteen patterns, adaline learns to find the relationship between the force and the cutting parameters. Adaline is fairly typical of many analog systems in this adjust and readjust problems.

We can say that the adaline recognizes all patterns (i.e., all patterns have been learned). We have a unique set of weights, (as shown in Figure 3.4) that represents the relationship between the output and input space. We can validate the program by giving some known data i.e., giving input parameters (feed, depth of cut, and speed) and find out the corresponding output using these weights as follows,

$$Actualoutput = W_1 * feed + W_2 * depth + W_3 * speed$$

Now we compare this value with the known value of the force for that particular pattern. This is done for sixteen patterns and a graph is plotted with number of patterns on x-axis and output values on the y-axis, as shown in Figure 3.5. To check the validity of the program, we rearranged the input data (i.e., training data) and used the same data twice, so as to get 32 training data patterns. After doing the simulation for this 32 patterns, we had one more unique set of weights, which are nearly equal to the set we got from the simulation of 16 patterns. From this one can say that, there will be no change in the network's output if the same information is given twice to the network.

From the network's output one can say that the force is most effected by depth of cut, which is a well known fact in the machining area. Consequently, this neural network model can be used successfully in predicting the cutting force



The weights of interconnections after convergence are:

0.065 0.927 0.006

The network's outputs after convergence are:

Patt#	Desired Output Newtons	Actual Output Newtons	Error
0	1533.738	1533.738	0.000
1	1533.762	1533.738	0.024
2	1601.835	1632.070	30.235
3	1600.863	1632.070	31.207
4	3067.872	3079.570	11.698
5	3055.482	3079.570	-24.088
6	3204.723	3185.821	18.902
7	3170.703	3185.821	-15.118
8	1534.827	1607.900	-73.073
9	1545.876	1607.900	-62.024
10	1604.982	1652.488	-47.506
11	1636.524	1652.488	-15.964
12	3067.062	3089.985	-22.923
13	3048.018	3089.985	-41.967
14	3203.277	3198.320	4.957
15	3142.812	3198.320	-55.508

Figure 3.4: Adaline Network's Output

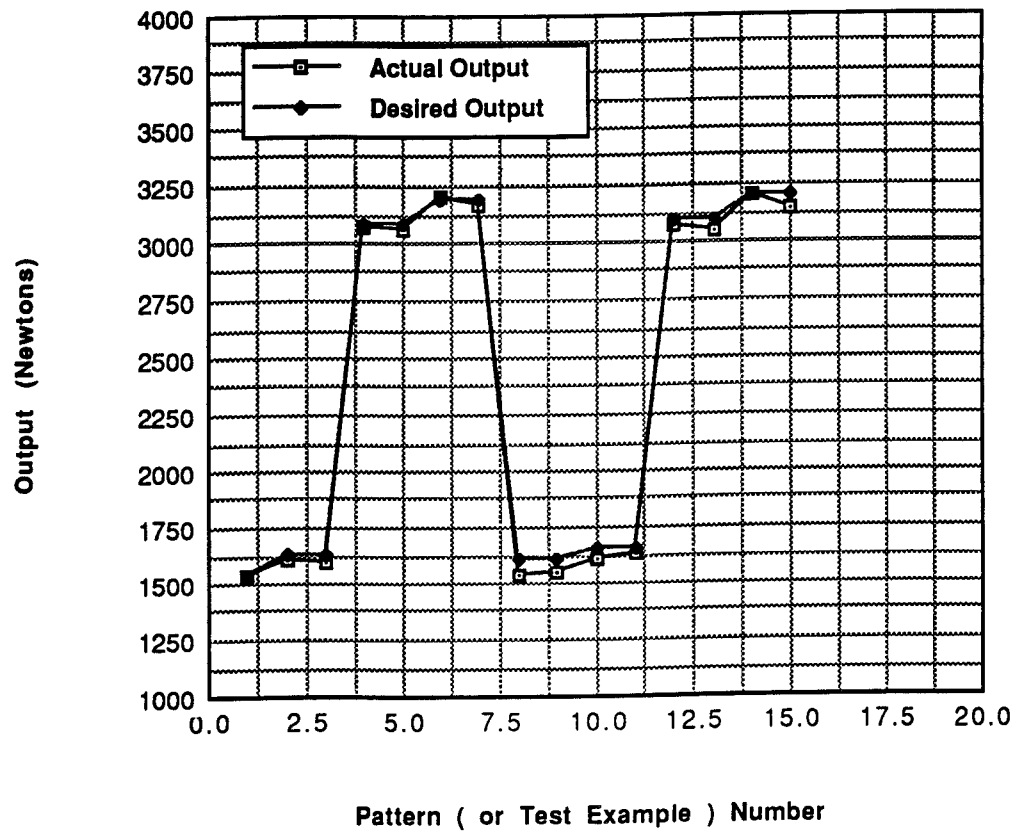


Figure 3.5: The Plot Between Actual and Desired Output of an Adaline

during a machining operation.

3.3 Monitoring of Tool Wear and Surface Roughness

3.3.1 Basic Concepts of Tool Wear

Tool wear is usually undesirable and has to be minimized. We need to control the progress of tool wear, as it is related to the quality and cost of the machining process, production rate, automation of manufacturing process etc. The mechanism of tool wear in metal cutting is a relatively complex phenomenon, usually occurring by several different processes, among them, adhesion, abrasion, diffusion, fatigue, electrochemical action, thermal cracking, mechanical chipping and fracture and tool deformation, depending on conditions in the cutting zone.

Adhesion wear occurs when the welded asperities between the contacting surfaces fracture, tearing away bits of material from either surface. This type of wear mechanism is found principally on the tool's rake face and it is normally associated with the relatively low cutting speeds that lead to formation of an unstable built-up edge. When the built-up edge breaks down, it tears away microscopic fragments of the tool with it and small fragments of the tool's edge are continually and progressively removed as the built-up edge breaks down. This may be accelerated if an intermittent cutting condition occurs or vibrations of the machine tool during machining and it is not influenced by high temperatures.

Abrasion is the most common wear mechanism on the cutting tool. It is caused by the work material containing high surface concentrations of hard particles. Under these conditions, the hard particles abrade the tool's surface in a manner similar to a grinding wheel. This wear is more susceptible to sharp edges than hard inclusions, that are smooth and spherical. Abrasive wear is not confined to any single face of the tool: it may be seen on both the rake and the flank faces

simultaneously. The abrasive action tends to produce a flat surface on the tool and causes such conditions as flank and notch wear.

Diffusion is a problem resulting from the interaction of metallic workpieces and the cutting-tool materials during machining. It depends heavily on the metallurgical relationship between the materials. The diffusion process is highly temperature-dependent, and when machining at high cutting speeds, and therefore at high temperatures, there is actual atomic transfer across the interface between the tool and the chip. *Fatigue wear* occurs when the tool surface is repeatedly subjected to a loading and unloading cycle. This leads to small portions of the tool material becoming detached from the tool's surface. An intermittent cutting action may lead to fatigue wear. Any fatigue problems can always be lessened by changing the cutting-tool materials that are appreciably harder than the workpiece. *Electrochemical action* takes place by electrochemical reaction between the workpiece and the tool, when a cutting fluid is used. This reaction results in the formation on the tool's face of a rather weak layer of low shear strength. This effect is normally considered desirable as it lowers the frictional force on the tool, which in turn reduces the forces required for cutting whilst simultaneously lowering the tool's temperature. However, it may also cause small amounts of tool material to be removed by the action of the chip flow.

Thermal cracking occurs in interrupted cutting, when the tool edge is subjected to rapidly fluctuating temperature. Welding of the built-up edge on the tool face may also result in cracks being formed on the tool face or edge due to differential expansion between the built-up edge and the tool. For the softer cutting tool materials, e.g., high speed steel, *Tool Deformation* may occur when cutting forces are high, and thus accelerate tool failure. *Mechanical Chipping and Fracture* frequently occur with the more brittle cutting tools, especially the ceramics, during interrupted cutting and also swarf action on the exposed edges of the tool.

Figure 3.6 summarizes the criteria for wear and failure of a hard metal insert.

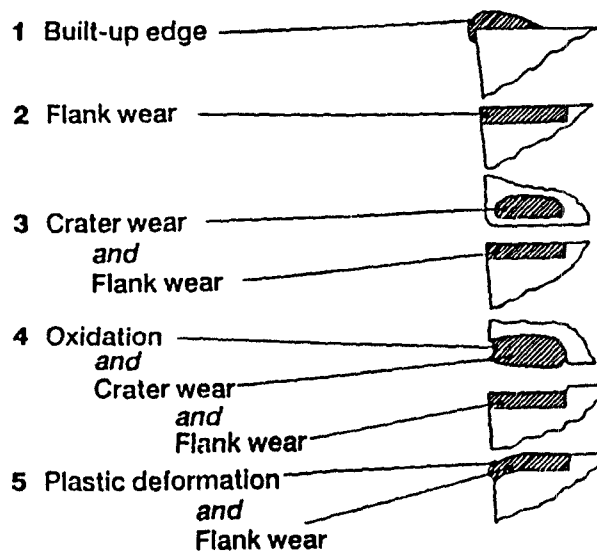
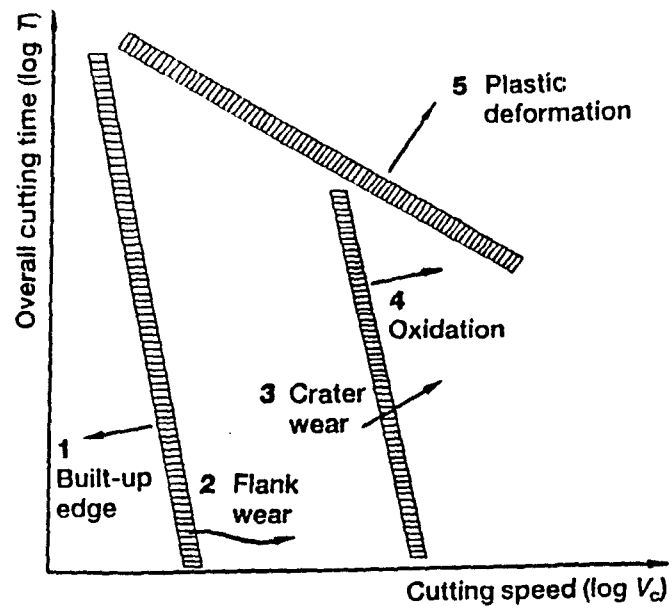


Figure 3.6: Tool Wear Criteria and Failure Mechanisms

An insert edge's life is assessed according to three criteria, namely its ability to hold workpiece tolerances and to maintain specific requirements for surface finish and an efficient chip-breaking ability. If the insert no longer satisfies these criteria, its useful life is ended and it should be discarded. A tool can be discarded by assessing the degree of wear by the flank wear on the insert, but there are many other factors that can also cause a shortened tool life; in this respect, the degree of cratering is much more significant.

Tool Wear Classification

It has been a well known accepted criterion that the condition of the tool can be obtained by the amount of flank wear on the flank face. It has been found that progressive flank wear develops and the wear progress follows a fixed pattern, and there are three stages of flank wear that occur during the life of a tool. These are the initial or primary wear, progressive or secondary wear, and catastrophic or tertiary wear, as shown in Figure 3.7.

Initial or primary wear : When a new tool is first used to machine a component, there is a rapid breakdown of the edge, shown by the initial high wear-rate in the graph of wear against time. This wear-rate is dependent upon the cutting conditions and the workpiece material and it will decrease as the cutting speed is increased. It is often stated that the high wear-rate in this region is caused by crumbling of the edge, rather than the insert being worn in.

Progressive or secondary wear : Once the initial wear has occurred, there is a steady progressive stage of insert wear, with a much less dramatic increase than in the initial stage, when realistic cutting speeds are applied. Towards the end of the progressive wear zone, it is usual to replace the tool. Sometimes, the criteria to replace a tool depends on the quality of the machined part. Once this extent of flank wear is reached, it can be assumed that to all practical purposes the life of the tool has ended.

Catastrophic or tertiary wear : The final stage of wear is rapid, leading to

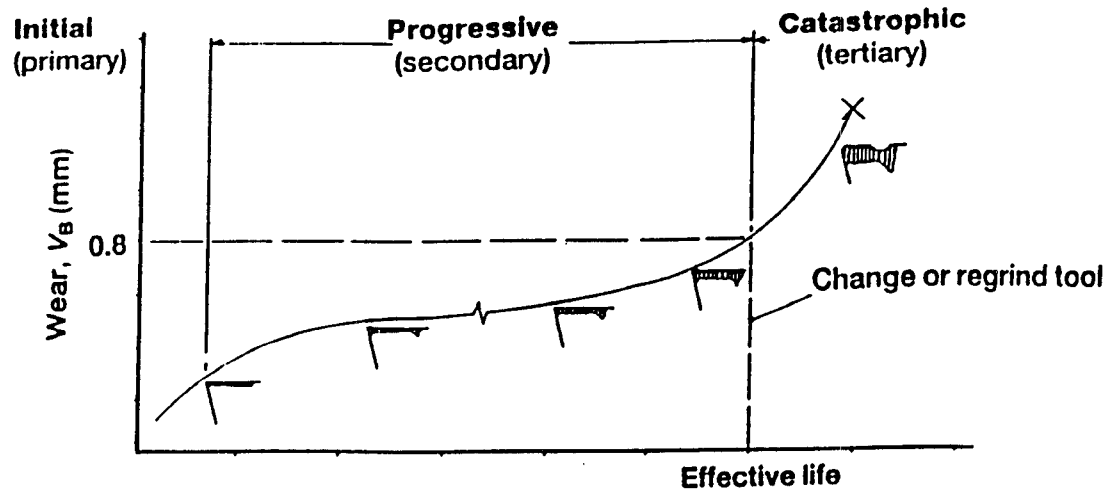


Figure 3.7: Tool Wear Classification

catastrophic failure of the insert. This failure is caused by a combination of high flank wear and a large crater formation which reach a point where the tool is sufficiently weakened for the tool forces to cause it to fracture. If such a rapid edge breakdown occurs during the final pass along the surface of the work, it is likely that the workpiece will have to be scrapped. When the workpiece has a high value, which has been increased as a result of the machining, any savings made by using tools beyond the end of the progressive wear zone to produce extra components will be more than lost by the value of the component scrapped.

3.3.2 Basic Concepts of Surface Roughness

Surface roughness is an important quality of machined surfaces. It not only has an important effect on the wear, fatigue, reflection and adhesive properties of many materials, but also affects their properties and feel. Surface roughness is characterized as the macro and micro geometrical properties of a surface.

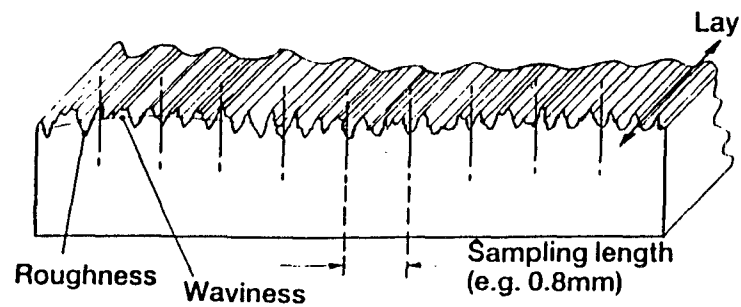
Since it is more difficult to characterize a three dimensional quantity, it has been traditionally characterized by the properties of a two dimensional profile taken from a representative part of the surface. The geometrical properties of this profile are called the surface roughness characteristics. The service life of a component, which is affected by the wear-rate of mating surfaces, is dependent upon the specified level of surface finish. Rough surfaces, with large peaks and valleys, will of course have smaller areas of contact and, as a result, will wear more quickly than smooth mating surfaces.

Figure 3.8 represents the main parameters of a three dimensional surface profile. A typical machined surface has a combinations of roughness, waviness and, lay as shown in Fig. 3.8. The roughness is the primary texture, which is produced by the combination of the tool's path and the tool motion. The roughness with its typical roughness height and roughness spacing, is usually produced by the basic surface forming process. The waviness results from such factors as deflections of the machine or work, vibrations, chatter, heat treatment and warping strains and it is the medium frequency component upon which the surface roughness is superimposed. The waviness consists of the more widely spaced irregularities. Lay, on the other hand, is a result of the production method used and it determines the direction of the predominant surface pattern. On this surface, the lay is in the front-to-back direction. Surfaces produced by machining processes ordinarily have a strong lay pattern; i.e., they are unidirectional. The surface texture is normally assessed by taking a sampling length at right angles to the lay. Roughness and waviness comprise the surface texture. Figure 3.9, shows two quantities that are of primary important: a measure of surface height indicated by the roughness average parameter, R_a , and a measure of the spacings of the peaks and valleys of the surface roughness, indicated by the wavelength parameter, D . The measure of R_a is given by:




$$R_a = [(\sum \text{areas } r + \sum \text{areas } s)/L]100/V$$

where,

(a) A three-dimensional representation of a surface profile – lay



(b) The factors that produce surface irregularities

- 1 Roughness  caused by high-frequency components
- 2 Waviness  caused by medium-frequency components
- 3 Form  caused by low-frequency components

(c) The determination of primary surface texture (R_a)

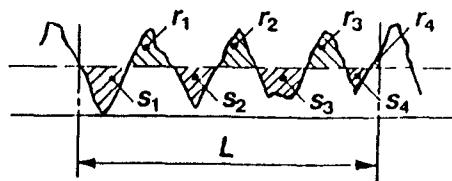


Figure 3.8: A Three Dimensional Surface Profile

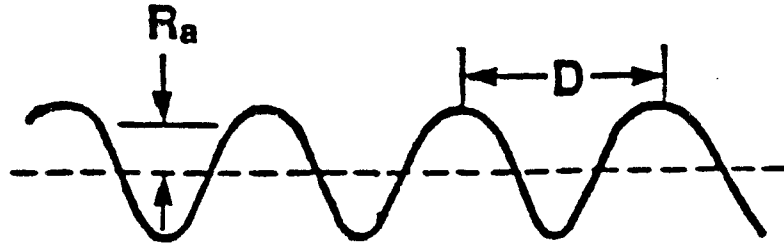


Figure 3.9: The Important Quantities of a Surface Profile

L = sampling length

V = vertical length

and areas r and s shown in Fig. 3.8 and R_a is measured in μm .

The parameter R_a is given as a numerical value: the higher the R_a value, the rougher the surface texture. Whenever a surface-texture assessment is necessary, it is important to consider what sampling length should be used for the trace of the machined surface, as this choice will determine the magnitude of the assessed parameters.

Recent advances in sensor technology have made available a variety of techniques to measure surface roughness. These range from a variety of optical techniques, that yield a direct profile of the surface such as in profilometry and interferometry, to techniques that a parameter which is related in a complex fashion to surface topography as in light scattering, speckle pattern analysis, ultrasonic and capacitance methods [Br 77]. While providing different choices in complexity, measurement range, and cost for measuring surface roughness, it would be more cost effective to use simple sensing techniques, but increase the wealth of information gained by incorporating advanced computing methods in

analyzing sensor data. Artificial neural networks would provide such a computing approach.

Due to the importance of measuring surface roughness a variety of approaches have been developed to analyze surfaces [ZaYe 91, ZaKa 91A, ZaKa 91B]. These techniques range from purely quantitative ones (basic statistical parameters) that give one or more numbers of the geometrical properties of the surface to those that yield continuous functions. The latter include time, and frequency domain techniques which describe the properties of the surface as a function of the height variable, the shift distance along the profile, or the frequency content of the profile. Surface roughness is based on two statistical parameters, center line average (CLA) and root mean square (RMS). The CLA is defined as the average of the absolute value of profile heights from a center line, while the RMS is defined as the root mean square value of profile heights from a center line.

Furthermore, in many sensory data processing applications, there is a requirement to classify sensory data into different classes based on their invariant properties. This is needed in on-line production environments to automate the classification and inspection of products. This can be achieved by using neural networks as an on-line monitor to machining processes, as they work well on noisy data.

3.3.3 Feedforward Neural Networks

Feed forward neural networks are characterized by layered architectures and strictly feedforward connections between the neurons and no lateral, self, or back-connections are allowed. These networks are all good pattern classifiers, and all undergo supervised learning, i.e., they are taught to classify input into one of the several *a priori* categories. This group includes the perceptron, the ADALINE and MADALINE networks, the back-propagation network (also known as back-propagating perceptron and multi-layer perceptron), the Boltzman machine, and the Cauchy machine. These networks differ primarily in the way in which they

learn. The first four networks, basically have the same learning laws and they are good for pattern recognition, signal filtering, data compression, and hetero-associative pattern matching. The Boltzman machine, and the Cauchy machine learning method draws on an optimization approach which comes from statistical modeling of thermodynamic processes. In this section, we will discuss about the back-propagation network in detail, and for other networks reference will be found in [Ri 87].

Back-Propagation Network

The back-propagation network (BPN) is the most well known and widely used network among the different neural networks available at present. The BPN is a perceptron with a different transfer function in the artificial neuron and a more robust and capable learning rule. This is an extension of single-layer perceptron with the addition of hidden layer and differs from single-layer perceptron with the learning law, i.e., it uses Generalized Delta Rule for weight changes.

In 1986, back-propagation network (Rumelhalt, 1986; Segnowski, 1986) came into prominence in neural networks as an exciting new means of pattern recognition. The term back-propagation refers to the training method by which the connection weights between each neuron of the network are adjusted. During operation, all information flow is feed forward. This network takes a long time to learn their pattern class and training sets have to be presented many times (between 100 - 100,000) in order to get stable interconnection weights between the neurons, to classify the input pattern correctly.

The Underlying Concept For The BPN

The important characteristic of the BPN is that, it forms a mapping from a set of input stimuli to a set of output nodes using features (hidden layer) extracted from the input patterns. This network can be designed and trained to accomplish

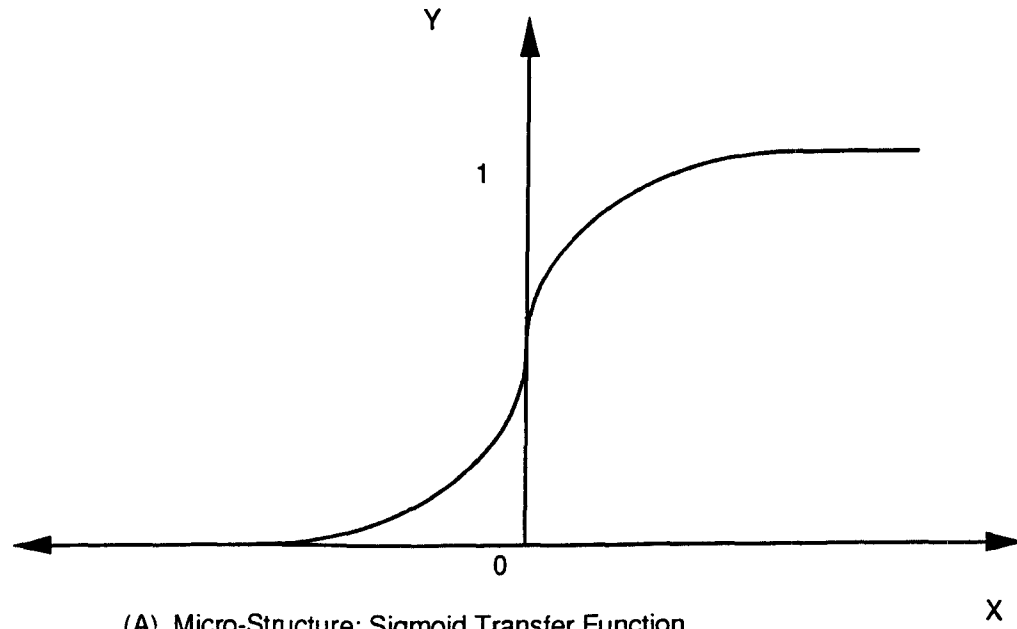
a wide variety of mappings, some of which are very complex, as the nodes in the hidden layers of the network learn to respond to features found in the input. By features, we mean the correlation of the activity among different input nodes. Because nodes in the BPN learn to respond to features as the network is trained with different examples, the network develops the ability to generalize. For example, the BPN can learn to distinguish between straight, concave, and convex curved lines, even if the lines to be tested occur in different locations in the input array than those used for training. The ability to make the complex distinctions, even when the presented pattern is different from those on which the network was trained, is due to the feature detection and generalization abilities which the network was trained into the middle or hidden layer nodes. To be successful for application, the hidden layer node of BPN must be trained to recognize the right sets of features (right sets means to recognize an appropriate and sufficient sets of features). These features must be sufficiently general, so that the network can respond correctly even when its input is different from those it has previously encountered.

Architecture of the BPN

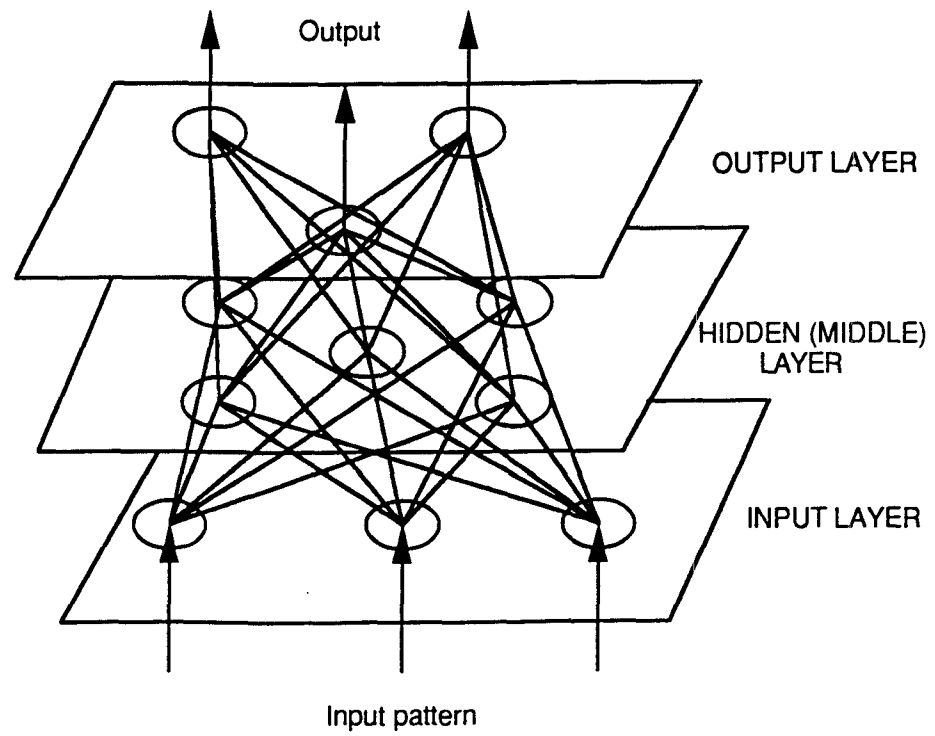
The micro-structure and the meso-structure of BPN are shown in Figure 3.10. The two structures are explained briefly in the following paragraphs.

The Micro-Structure

The BPN uses a sigmoid shape (S-shape) transfer function for each node, as it is defined by a continuous function and is asymptotic for both infinitely large positive and negative values of the input sum 'X' of each node, which is an independent variable. This type of transfer function allows us for more complex pattern recognition problems. The sigmoidal shape of transfer function means that for the most value of an independent variable, the value given by the transfer



(A) Micro-Structure: Sigmoid Transfer Function



(B) Meso-Structure

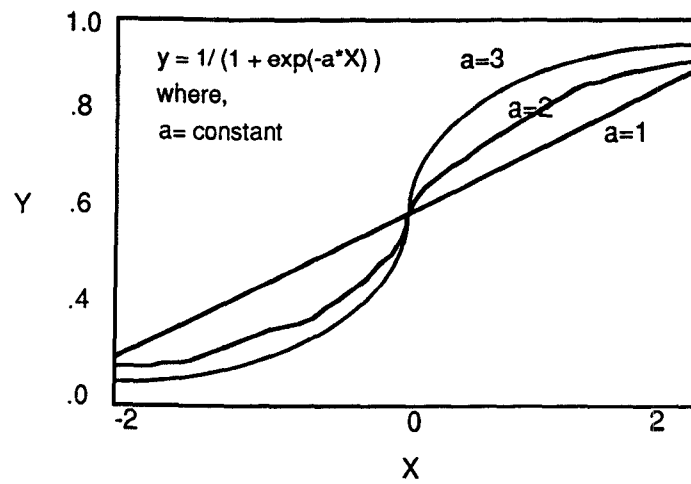
Figure 3.10: The Meso and Micro Structures of a BPN

function is between two asymptotic values, 0 or -1 and 1. One important factor about the sigmoid function is that its derivative is always positive, and is close to '0' for either large positive or large negative values of 'X'. Figure 3.11 illustrates both the sigmoid function and its derivative. As shown in the Figure 3.11, the derivative attains its maximum value at 'X' equal to 0, which is important in helping the BPN learning law work effectively. This is because when we are dealing with equations of learning law, we find that the changes made to the weight connections between each neuron, is proportional to the derivative of the activation. If this derivative value is zero then the weight changes are small, which is desirable, as the derivative is zero when the activation value of neuron is near '0' or '1', i.e., one of the two stable states. The derivative is the largest when the activation is in the middle range, and so the change in the weights is also fairly large, which drives the output of the neuron to one of the stable states. Thus, the transfer function not only gives us smooth and differential behavior, it actually helps the learning law work the way we want it to.

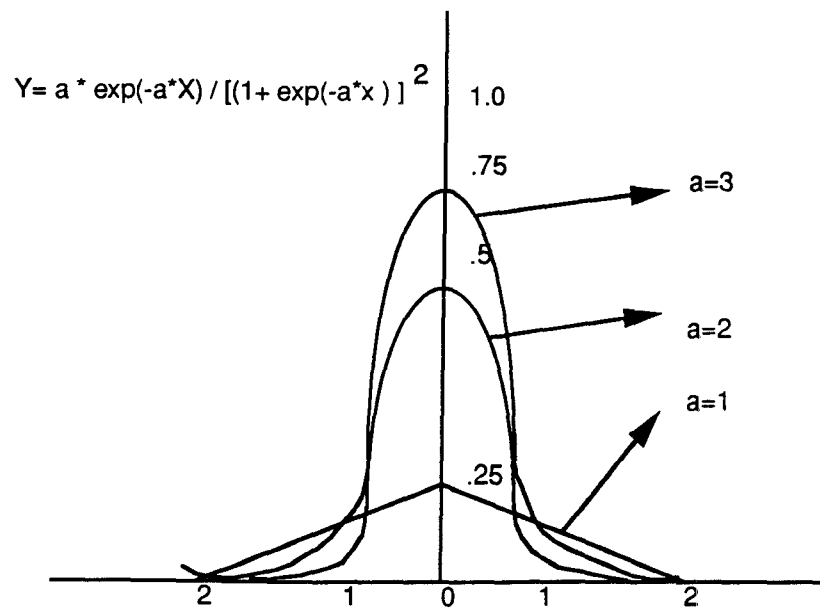
Note that as the activation of a neuron approaches either '0' or '1', the derivative approaches zero (as shown in Fig. 3.11). Because the learning or weight change is proportional to this derivative, a weight may change more slowly than desired for neurons which yield either a large or small activation. This can cause difficulties in training the network.

The Meso-Structure

The BPN requires at least three layers of nodes, where as a single-layer perceptron requires two. In BPN, the layers are referred to as input, hidden or middle and, output layers as shown in Fig. 3.10. The hidden layer nodes are crucial in allowing the BPN to extract features and generalize. Only feed forward connections are allowed and each node in input layer is connected to the hidden layer and each hidden layer is connected to only output layer. Usually BPN is fully connected.



Sigmoidal Transfer Function



Derivative of the Sigmoidal Transfer Function

Figure 3.11: Sigmoid Function and It's Derivative

Dynamics of the BPN

Like other types of networks, BPN has separate stages for learning and operation. Once the network has been trained, the learning process is stopped, and connection weights are fixed. In operation, all information flows forward from the input neurons to the neurons in the hidden layer(s), and from these to the output neurons. No information is passed backward (or back propagated) during the actual network operation, and the back propagation refers strictly to the learning stage. Now finding a set of connection weights for some problem is not easy; it requires application of the back propagation learning rule for several thousand iterations to achieve a good set of connection weights and neuron thresholds.

The basic structure of an BPN has an input layer, hidden layer(s), and an output layer. The network will have variable thresholds on the hidden nodes and the output nodes and variable weight connections between individual nodes. The weight values could be very hard to come by unless some learning algorithm (like back propagation learning rule) is used to drive the collection of weights toward a useful set of values.

Now let us put a pattern to the nodes of input layer. So that we will have some activation value to each node. Next we move our attention to the next layer up. For each neuron in this layer, we calculate an input which is the weighted sum of all the activation from the input layer. The weighted sum is achieved by a vector multiplying the activation in the input layer by a connection weight matrix. Now we add a threshold value to this weighted sum. Then we apply a sigmoid function to get the activation of that neuron. The same thing applies to all neurons in the hidden layer(s) and output layer.

In general the formula for the activation of the j^{th} neuron in a layer is:

$$Y_{activ_j} = F(X) = F(W_{ij} * Input_{activ_i} + \theta_j)$$

where,

- Y_{activj} = activation of the j^{th} neuron in the receiving layer
- $F(X)$ = transfer function (sigmoid function)
- W_{ij} = connected weight between i^{th} neuron in the sending layer and j^{th} neuron in the receiving layer
- $Input_{activi}$ = input to the i^{th} neuron in the sending layer
- θ_j = threshold of the j^{th} neuron in the receiving layer

The goal of the BPN learning law is to find useful value for the weights and threshold values which will enable the desired classification. These weights may not be unique, in fact they won't be. As with most of the BPN applications, there are lots of different weight combinations which will work. However, finding a useful set of connection weight values is a challenge. The weight value could be very hard to come by unless some learning algorithm is used to drive the collection, towards a useful set of weight values over time. We can have multiple possible sets of similar weight values, and that within each set, there may be an infinite range of connection weights and thresholds. Each solution within a given set will probably be roughly proportional to all other solutions. This means that there is no single right answer, and there might not be any single best answer. These are likely to be a very large number of good or workable answers.

Learning Law for the BPN

Like other networks, the BPN is taught to create a mapping from the input to an activation pattern in the output layer. The BPN undergoes supervised training. The BPN learns to distinguish among different pattern categories simultaneously. Each pattern will have a different type of influence on the change in the connection weights. Thus, it is important that the patterns in the training set presented in such a way that the changes in the connection weights move over time to values which optimize the network's response to all the pattern classes. So we need a rule that will tell us how to change the weights depending

on whether or not our training is correct. We somehow need to make the weights on the processing element to approach the desired response. This can be done by modifying the process element, so it can monitor its own output (by comparing the actual output with the desired output and computing the error 'E') for that particular input pattern. The rule used for training a BPN is 'The Generalized Delta Rule'.

The Generalized Delta Rule (GDR), which was first derived by Werbos in 1974 for a multilayer perceptron network, is an extension of the learning rule for the single layer perceptron network proposed by Widrow and Hoff (1960). The basic idea of GDR is to learn the target function via gradient descent minimization of a global error function 'E'. The rule first uses the input vector to produce its own output vector and then compares with the desired output, or target vector. If there is no difference between desired and target outputs, no learning takes place. Otherwise the weights are changed to reduce the difference. This can be viewed as a nonlinear optimization problem where the goal is to find the set of network weights ' W_{ij} ' that minimize the total error.

The connection weight adjustment is done as follows:

$$\Delta W_{j,i} = \eta \delta_j a_i$$

where,

$\Delta W_{j,i}$ = adjustment of weight $W_{j,i}$, which goes to unit 'j' from unit 'i'

η = learning constant

δ_j = the error value of the target unit 'j'

a_j = the output value for the originating unit

So the new weight between node 'i' and node 'j' is:

$$W_{j,i}(n+1) = W_{j,i}(n) + \eta \delta_j a_i$$

where, n = current iteration number.

The error value, denoted by δ_j , is simple to compute for the output layer, but somewhat complicated for the hidden layers. If unit 'j' is in the output layer, then its error value is:

$$\delta_j = (t_j - a_j)F'(X)$$

For a hidden layer:

$$\delta_j = [\sum_{n=1}^k \delta_n W_{n,j}]F'(X)$$

where,

t_j = the target value for unit j

a_j = the output value for unit j

$F'(X)$ = the derivative of the sigmoid function, $F = F(X)(1 - F(X))$

X = weighted sum of inputs to j

k = number of output nodes in output layer

The algorithm of BPN is given in appendix A. The basic principle of a GDR is same as the Least Mean Square (LMS) principle. The LMS rule attempts to insure that the aggregate statistical LMS error in the network is minimized. In this case, the error in the weights of the processing element is based on some ideal value for the weights. We compute the current error i.e., the deviation from this ideal value, based on the weights for this input. We then adjust the weight vector by computing a delta vector that is parallel to the input vector and has a magnitude as desired in the previous equation. The weights are adjusted by adding this delta vector to our current weight vector. This process has a simple geometric interpretation . It can be shown mathematically that the aggregate mean squared error is a function of the weight vector; specifically, this function is a quadratic function of the weight vector. So, if we were to plot the mean-squared

error Vs the possible weight vectors, we would get a parabola as illustrated in Figure 3.12. One prominent feature of the parabola is that it has a bottom, i.e., the point where the bowl bottoms out into a minimum value. This point represents the minimum mean squared error or LMS error (or Delta Rule). The weight vector that corresponds to this minimum error is our ideal weight vector. It is the best weight vector we can possibly have for this particular input pattern.

The delta rule moves the weight vector from wherever it is on the surface of its parabola towards the bottom of the bowl. It does this by moving along the negative gradient of the parabola, which is the most direct route to the bottom of the bowl. We know that the gradient always points along the direction of the steepest descent of the curve at any given position. This means that the learning algorithm always takes the most efficient route from the current position of the weight vector to the ideal position, based on the current input pattern. So the Delta Rule not only minimizes the mean squared error but does so in the most efficient fashion possible.

The next step is to choose the value of the learning constant η . We certainly want η to be positive. If it is negative, the direction of our delta vector is away from the ideal vector, i.e., we will be degrading the response rather than improving it. Also η should be less than 1; otherwise, the network cannot be stabilized. If η is greater than 1, we will overshoot the input vector and end up somewhere on the other side of it. So η should always be between 0 and 1. Also η is a measure of the speed of convergence of the weight vector to the ideal one. If our weight vector is very far from its correct position, we want it to take fairly large steps, otherwise we might end up training it for a very long time. On the other hand, when we get close to the correct position, the specific difference between the actual input sample and sample used to train the system begins to introduce significant noise levels into the network. At this point, fine distinction between the actual and desired output can cause the weight vector to jump around excessively and be quite slow to find the correct settling point.

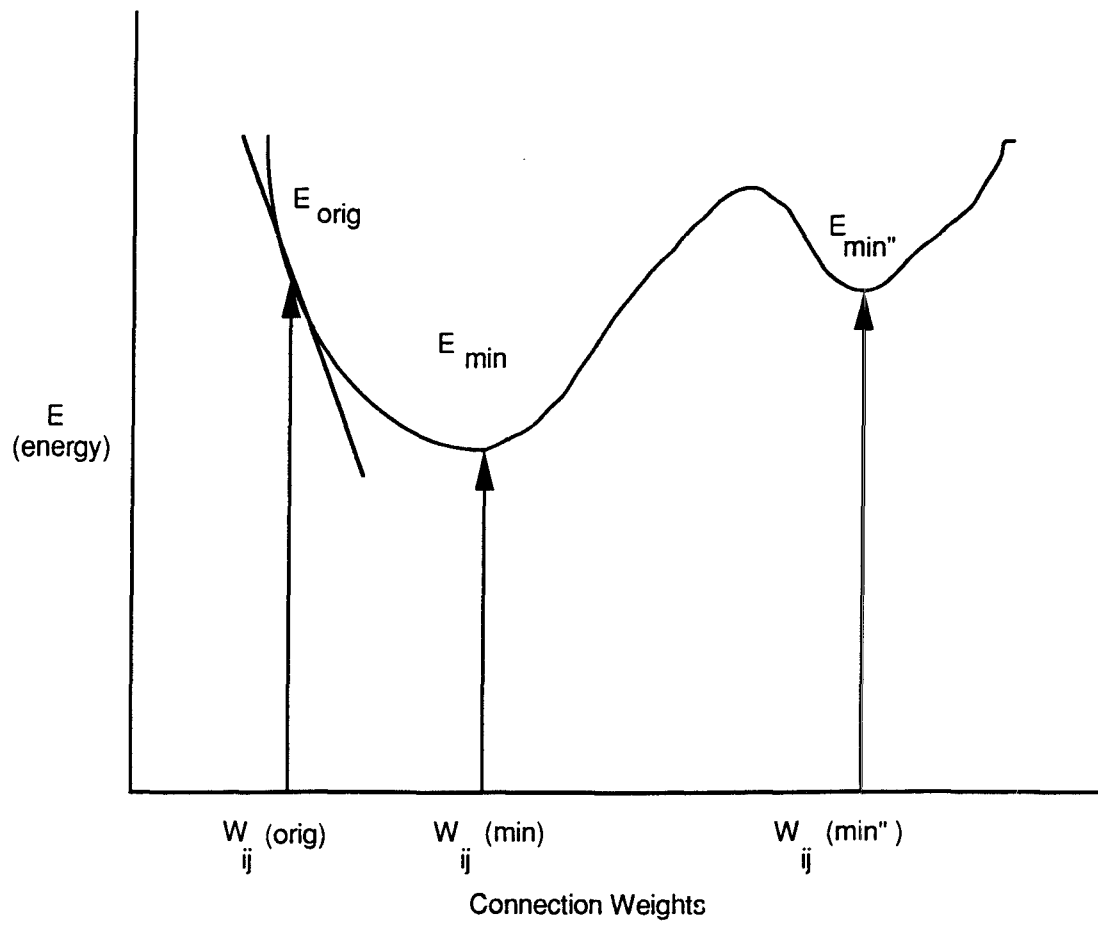


Figure 3.12: The Plot Between Error and Weight Vector

A good value of η depends on how good our training data are.

3.3.4 Factors Related to Tool Wear and Surface Roughness

Successful automation of machining operations relies, to a great extent, on the ability to recognize process abnormalities and initiate corrective action. In the absence of human operators, this function has to be performed with sensors and associated decision-making systems which are able to interpret incoming sensor information and decide on the appropriate control action. As stated in the literature review, a machining operation can be recognized by a direct or an indirect approach. In the indirect approach, signals related to the cutting force, tool vibration characteristics, cutting temperature, and acoustic emission etc., are measured and recorded during machining operation.

It is well known that a tool will produce relatively higher loads as it begins to wear during the cutting operation. For an effective process monitoring it is important that the signal used should vary progressively as the tool wears, and not just at the point when it breaks. Force measurement is currently the most reliable and accurate sensing method in metal cutting operation. Hence, it has been the preferred method for monitoring a machining process. The cutting force generally increases with flank wear due to increase in the contact area of the wear land with workpiece. It is generally assumed that the components of cutting force are related to flank and crater wear [DaUl 87, Filp 69, MiDe 67]. It is also known that the thrust force and the feed force are affected the most by flank and crater wear. At light cutting conditions the crater wear is negligible and the tool failure is mainly caused by crater wear. Except for the small changes caused by flank wear, the cutting force for light cutting is relatively constant. The variations in the cutting force become much more pronounced at heavier cutting conditions. According to Wright [YeWr 83], however, cutting force information by itself is inadequate for tool wear detection because its magnitude also depends on the

cutting velocity. Another problem is that although flank wear tends to increase the cutting force, the accompanying crater wear tends to reduce it [YeWr 83], so that the magnitude of the cutting force may not show any sensitivity to tool wear.

Ramalingam proposed a model, in which the ratio of the feed force to the cutting force (tangential force) is used as the measure of flank wear of a cutting tool during turning [ShRa 90]. The first derivative of this ratio is used to signal tool change. According to Ramalingam, direct use of cutting force to monitor tool wear is complicated by variation in cutting conditions. For the same cutting conditions, both cutting force and feed force increases as the flank wear increases. But feed force is more sensitive to the flank wear compared to the tangential cutting force. Feed force is significantly affected by feed rate, cutting speed, and depth of cut. As the feed force increases more rapidly than tangential cutting force, the resulting force direction in the x-z plane will change with increasing flank wear. It is shown that the feed to cutting force ratio is insensitive to speed and depth of cut, and it decreases linearly with feed. But in actual situations, feed is held constant and can be used as a process control variable. The unknown parameters are workpiece size variations and the speed change (along with changes in material properties). The former causes the depth of cut to vary. So a small variation in cutting speed and depth of cut do not cause noticeable changes in the force ratio. The curve representing the variation feed force with time is similar to the flank wear characteristic curve. The cutting force ratio is also a good indicator of the wear regime. When the tool reaches the catastrophic wear stage, not only the cutting force ratio increases but also its derivative is high.

By using cutting force ratio as an indicator of wear regime, one can eliminate the problem of using the magnitude of force to the sensitivity of tool wear. At high temperatures, the magnitude of cutting force may decrease due to work hardening at the tool-workpiece interface. Because of this, the magnitude of

cutting force may decrease, but the feed to cutting force ratio will not be effected (i.e., it will increase with flank wear) as both components are affected by work hardening.

Apart from force ratio, other machining parameters, like feed, speed, nose radius of the cutting insert or tool etc., also affect the machining process. When a high cutting speed is chosen, the insert may deform plastically as a result of the high tangential force. This is usually caused by several interrelated factors. For example, the high cutting speed causes the temperature of the tool to increase locally, a factor which is further exacerbated when a small nose radius is used. This local increase in temperature, coupled with the high speed, causes the tool geometry to change. So the chip flow pattern is modified accordingly, leading to tool wear. By using a larger nose radius on the insert, the heat can be more easily dissipated.

Fortin, concluded that it is possible to increase the cutting tool life substantially by a proper variation of the cutting feed rate throughout the cutting process [ClMa 90]. When machining tough materials, tool wear appears quickly with the creation of a crater on the rake face. This decrease in tool-cross section induces material chipping and premature tool breakage which can be avoided by varying the feed rate. When high feed rates are utilized in conjunction with other wear-promoting factors, namely a small nose radius and a low conductivity insert, the axial force will also plastically deform the insert. In this case, the clearances and plan approach angles will change, rather than the rake angle as in the high speed case. Once again, tool wear will increase, leading to a premature tool failure. Brittle fracture is associated with very high feed rates and depths of cut.

3.3.5 On-line Monitor Model

A neural network model is used as an on-line monitor for the machining processes. Developing the neural network model involves two stages. First, an appropriate

architecture for the network has to be determined. Then the network with the appropriate architecture has to be trained on the training data, which is obtained through a series of experiments. A three-layer fully-connected feed forward neural network (described in section 3.3.3) is used to learn mappings between input and output parameters. In this thesis work, a back-propagation neural network algorithm is implemented in a sun sparc station. A schematic of the artificial neural network (ANN), with the input layer, the hidden layer and the output layer, is shown in Figure 3.13. Each of the input nodes correspond to a single input parameter, and similarly each output node is uniquely associated with one of the output parameters. The determination of the hidden nodes is mainly based on experience and trial and error experimentation. The role of the hidden nodes is to perform feature extraction on the patterns presented at the input layer. Essentially, this involves noise rejection in the raw sensor patterns, and produce new features with a higher signal to noise ratio.

Once the architecture is determined, the neural network has to be trained on the examples (or patterns), where each pattern is a set of input and output parameters. All the values of patterns are normalized between 0 and 1. This is because, each node represents a nonlinear sigmoid function, as explained earlier, and gives an output which lies between 0 and 1. So the data we are presenting to the network should lie between the same limits; i.e., 0 and 1. We know that the main function of a neural network is to find a set of weights, which represent the mapping from the input space to the output space. It is very difficult to converge to a set of weights, if the input and output parameters are not normalized. Also, normalization is required because the parameters used for training will have different units of measurement and to prevent saturation of the sigmoid function. The parameters are normalized using the relation:

$$V_{nor} = [(0.9 - 0.1)/(V_{max} - V_{min})(V_{cur} - V_{min})] + 0.1$$

where,

V_{max} = maximum value of the parameter

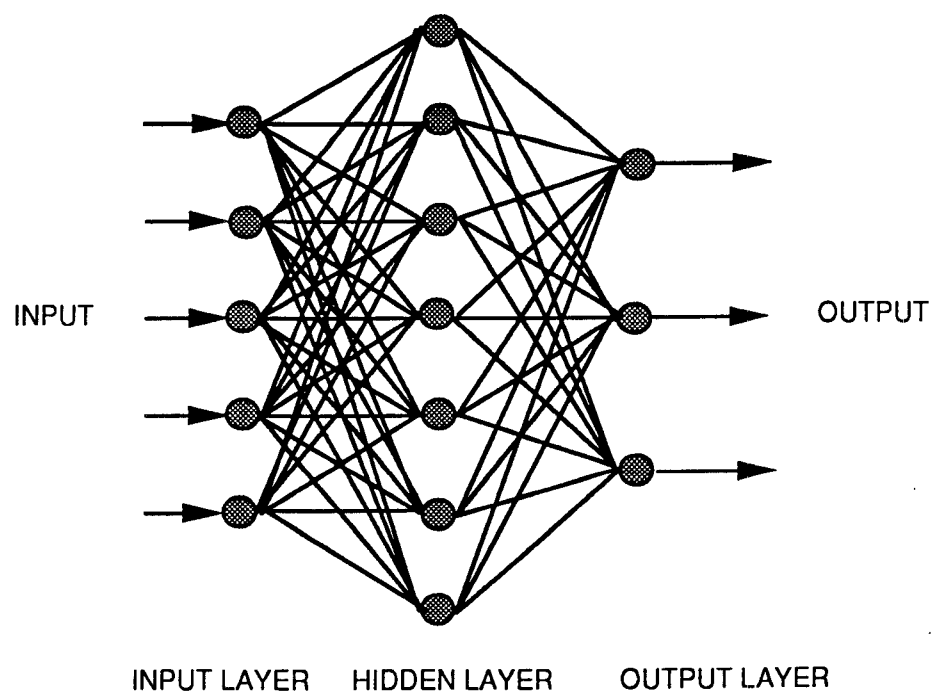


Figure 3.13: A Schematic View of an ANN

V_{min} = minimum value of the parameter

V_{cur} = current value of the parameter

V_{nor} = normalized value of the parameter

Once the data is normalized, the network weights and node thresholds are randomized. Training involves presenting the patterns (input-output pairs) to the network, calculating the error, propagating the error back through the network (using the back-propagation algorithm), and then modifying the weights by small increments to reduce the error. This process is repeated until the network is stabilized (i.e., the weights do not change) or the overall error (for all testing patterns) is reduced below a threshold. Unlike the conventional batch algorithms, back-propagation is an incremental learning algorithm, because at any stage during this process, training can be stopped and the network would still serve as a model of the function being learnt, even though it may not be quite accurate. This feature of the algorithm allows incremental improvement as new data is made available. If a trained network is already available, there is no need to randomize the weights of the network. Though the original training sets have to be reused, the network can start from its current state, thus reducing the training time.

The six input parameters for the proposed network are: the cutting force ratio (ratio of the feed force to the tangential force), speed, feed, depth of cut, nose radius and time . Three different types of networks are proposed in this work. They are as follows:

- Tool wear is predicted in the first type. The training data consists of six input parameters and one output parameter (tool wear). The architecture of this network is: six input nodes, one output node, and four hidden nodes (as shown in Figure 3.14).
- Surface roughness is predicted in the second type. The training data consists of six input parameters and one output parameter (surface roughness).

The architecture is the same as that of the first type (as shown in Figure 3.14).

- Both tool wear and surface roughness are predicted in the third type. The training data consists of six input parameters and two output parameters (tool wear and surface roughness). The architecture of this network is: six input nodes, two output nodes, and four hidden nodes (as shown in Figure 3.15). This type of network is chosen to remove false alarms for both tool wear and surface roughness.

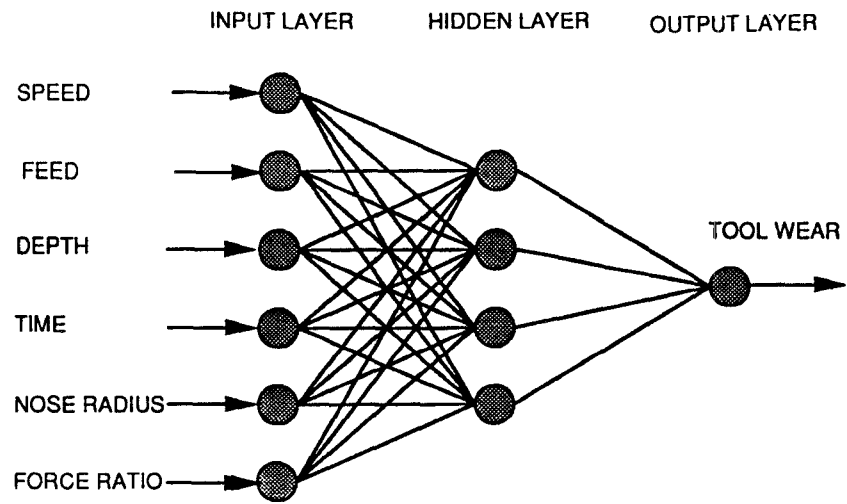
The data for training the above networks is taken from [Ar 89] and is shown in tables 3.2 and 3.3.

Feedforward back-propagation network is trained with three different training data and the results are illustrated in figures 3.16, 3.17, 3.18. Figure 3.19 illustrates the correlation between the predicted and the actual values for the first type of network. Figures 3.20 and 3.21 give the same information for the second and the third type. From the results we concluded that the percentage of predicting the tool wear (i.e., $(\text{error}/\text{desired value}) * 100$) from the trained network is about 93%. Similarly predicting the surface roughness was about 82%. In the third type of network it is 91% for tool wear and 83% for surface roughness.

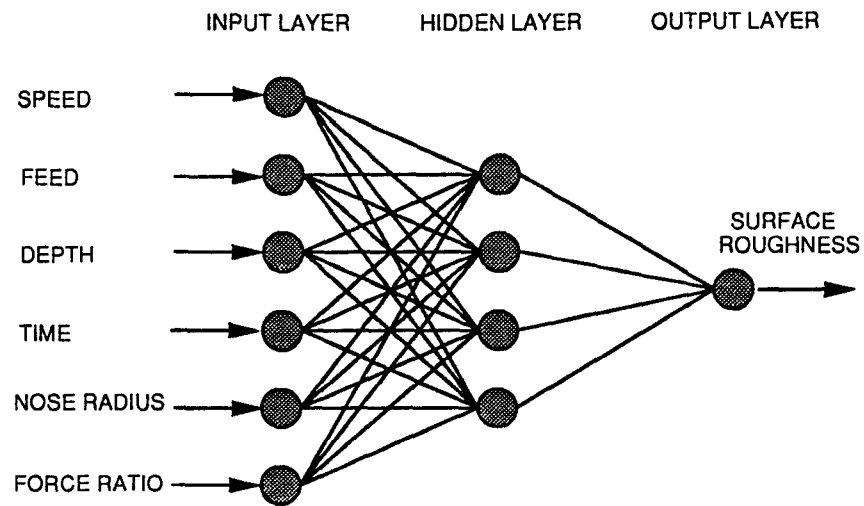
3.4 ANN Applications to Machining of Ceramics

3.4.1 Overview

Machining of advanced structural ceramics has recently come into prominence due to the need for high-strength materials for high temperature applications. A comprehensive survey of the industry has confirmed that the high cost of machining is a primary impediment to the widespread utilization of advanced ceramics.



(A) Tool Wear Model



(B) Surface Roughness Model

Figure 3.14: The Architecture of an On-Line Model

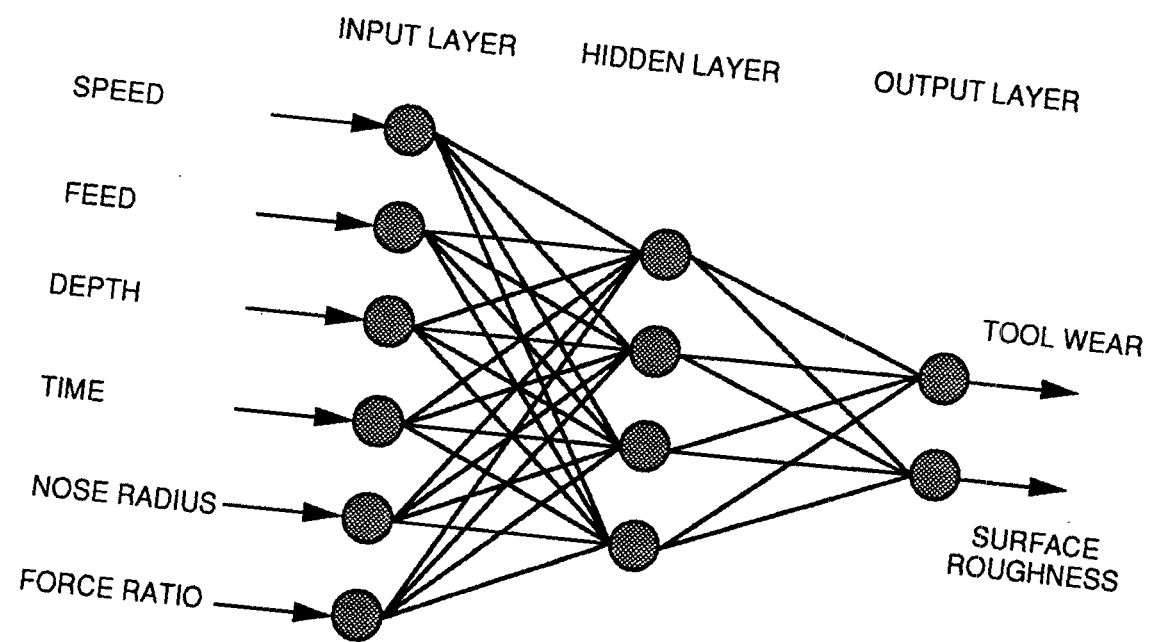


Figure 3.15: The Architecture of a Force and Surface Roughness Model

Speed (rpm)	Feed in.	Depth in.	Radius in.	time min.	Force Tan	Force Rad	Force Fed	Finish mu in.	Wear in.
325	0.003	0.040	.0156	0	44	14	26	-	0.0000
325	0.003	0.040	.0156	2	48	12	24	35	0.0030
325	0.003	0.040	.0156	5	48	12	24	30	0.0035
325	0.003	0.040	.0156	10	38	12	26	35	0.0037
325	0.003	0.040	.0156	20	44	14	22	60	0.0038
325	0.003	0.040	.0156	30	48	14	28	65	0.0058
325	0.003	0.040	.0156	40	50	18	28	70	0.0054
325	0.003	0.040	.0156	60	48	16	28	55	0.0056
100	0.006	0.020	.0312	0	54	28	24	.	0.0000
100	0.006	0.020	.0312	2	56	30	28	40	0.0021
100	0.006	0.020	.0312	5	50	30	24	26	0.0034
100	0.006	0.020	.0312	10	48	26	20	60	0.0051
100	0.006	0.020	.0312	20	52	30	20	45	0.0071
100	0.006	0.020	.0312	30	46	28	20	49	0.0079
100	0.006	0.020	.0312	40	50	26	18	45	0.0089
100	0.006	0.020	.0312	60	57	40	22	50	0.0097
325	0.003	0.020	.0312	0	28	14	10	.	0.0000
325	0.003	0.020	.0312	2	28	12	10	18	0.0038
325	0.003	0.020	.0312	5	28	14	10	40	0.0038
325	0.003	0.020	.0312	10	28	16	12	35	0.0047
325	0.003	0.020	.0312	20	28	16	12	35	0.0059
325	0.003	0.020	.0312	30	30	16	12	37	0.0064
325	0.003	0.020	.0312	40	26	16	12	29	0.0067
325	0.003	0.020	.0312	60	30	18	12	42	0.0095

Table 3.2: Experimental Data

Speed (rpm)	Feed in.	Depth in.	Radius in.	time min.	Force Tan	Force Rad	Force Fed	Finish mu in.	Wear in.
325	0.006	0.040	.0312	0	92	30	38	.	0.0000
325	0.006	0.040	.0312	2	85	23	40	27	0.0035
325	0.006	0.040	.0312	5	88	26	40	50	0.0043
325	0.006	0.040	.0312	10	88	23	36	55	0.0049
325	0.006	0.040	.0312	20	82	26	36	70	0.0051
325	0.006	0.040	.0312	30	86	28	40	100	0.0058
325	0.006	0.040	.0312	40	88	26	40	125	0.0060
325	0.006	0.040	.0312	60	90	28	42	145	0.0072
100	0.003	0.040	.0312	0	54	20	34	.	0.0000
100	0.003	0.040	.0312	2	56	22	38	23	0.0025
100	0.003	0.040	.0312	5	56	20	38	20	0.0031
100	0.003	0.040	.0312	10	64	22	40	23	0.0048
100	0.003	0.040	.0312	20	58	24	38	23	0.0060
100	0.003	0.040	.0312	30	64	24	34	23	0.0067
100	0.003	0.040	.0312	40	60	22	34	35	0.0068
100	0.003	0.040	.0312	60	52	24	34	35	0.0076
100	0.003	0.020	.0156	0	42	16	30	.	0.0000
100	0.003	0.020	.0156	2	42	16	30	37	0.0031
100	0.003	0.020	.0156	5	44	18	30	50	0.0035
100	0.003	0.020	.0156	10	42	20	30	55	0.0037
100	0.003	0.020	.0156	20	50	20	32	60	0.0044
100	0.003	0.020	.0156	30	52	20	30	65	0.0047
100	0.003	0.020	.0156	40	26	14	14	70	0.0050
100	0.003	0.020	.0156	60	28	16	14	60	0.0055

Table 3.3: Experimental Data

TRAINING COMPLETE after 15001 iterations

Weights of Neurode Interconnections:

Hidden Layer:

Mid Neurode # 0					
0.334	0.080	0.220	47.976	-0.161	0.187
Mid Neurode # 1					
-1.750	-1.722	6.349	-8.754	0.120	8.718
Mid Neurode # 2					
1.349	0.191	-1.013	-9.983	-6.427	7.256
Mid Neurode # 3					
2.038	1.479	3.933	1.077	3.135	4.061

Output Layer :

Out Neurode # 0			
3.071	-1.830	-1.190	-2.882

Neurode Threshold Values:

Threshold for 0 mid node is: -5.371

Threshold for 1 mid node is: 6.192

Threshold for 2 mid node is: 7.606

Threshold for 3 mid node is: -2.260

Threshold for 0 out node is: 2.683

Pattern Number	Actual Output (mm)	Desired Output (mm)	Error (mm)
95	0.112	0.109	-0.007
96	0.113	0.117	0.004
97	0.124	0.107	-0.017
98	0.164	0.111	-0.053
99	0.123	0.116	-0.007
100	0.112	0.109	-0.003
101	0.113	0.116	-0.003
102	0.116	0.114	-0.002
103	0.118	0.114	-0.004
104	0.136	0.137	0.001
105	0.112	0.106	-0.006
106	0.113	0.114	0.001
107	0.215	0.227	0.012

Figure 3.16: Results of Tool Wear Network

TRAINING COMPLETE after 15001 iterations

Weights of Neurode Interconnections:

Hidden Layer:

Mid Neurode # 0					
-5.883	0.306	-1.258	15.252	-3.428	-3.522
Mid Neurode # 1					
4.829	-1.436	0.717	-11.226	4.539	4.104
Mid Neurode # 2					
-0.153	-9.603	4.369	-0.335	-2.072	6.483
Mid Neurode # 3					
8.671	-10.788	7.976	15.287	-2.943	6.387

Output Layer:

Out Neurode # 0			
-3.941	-5.053	-1.894	2.333

Neurode Threshold Values:

Threshold for 0 mid node is: -0.350

Threshold for 1 mid node is: -1.704

Threshold for 2 mid node is: 2.769

Threshold for 3 mid node is: -6.253

Threshold for 0 out node is: 2.823

Pattern Number	Actual Output (mu in)	Desired Output (mu in)	Error (mu in)
95	79.14	65.000	-14.140
96	69.243	70.000	0.757
97	105.33	90.000	-15.330
98	45.280	65.000	19.720
99	103.48	100.000	-3.480
100	82.440	68.000	-14.440
101	74.190	65.000	-9.190
102	72.340	108.000	-35.660
103	81.620	150.000	68.380
104	83.272	85.000	1.728
105	83.483	65.000	-18.483
106	82.854	65.000	-17.854
107	62.696	45.000	17.696

Figure 3.17: Results of Surface Roughness Network

```

TRAINING COMPLETE after 15001 iterations
-----
Weights of Neurode interconnections:
Hidden Layer:

Mid Neurode # 0
  -3.054   -2.051   -1.398   -3.004    0.285    4.449
Mid Neurode # 1
  -0.118   -0.074    0.177   43.264    0.042    0.116
Mid Neurode # 2
  -1.104    1.091   -1.011    0.011   -2.482   -1.136
Mid Neurode # 3
   1.244    1.061    0.607   -2.860   -2.019    2.192

Output Layer:

Out Neurode # 0
  -15.953    9.161   12.875    2.303
Out Neurode # 1
   -7.852    9.988    2.830   -6.403
-----
Neurode Threshold Values:

Threshold for 0 mid node is:  7.789
Threshold for 1 mid node is: -2.894
Threshold for 2 mid node is: -1.527
Threshold for 3 mid node is:  3.300
Threshold for 0 out node is:  3.293
Threshold for 1 out node is:  3.885
-----
Pattern Actual Desired Error      Actual Desired Error
Number  Output Output      (mm)      Output Output      (mu in)
        (mm)      (mm)
-----
95      0.114  0.109  -0.005      77.000  65.000 -12.000
96      0.134  0.117  -0.017      69.650  70.000  0.350
97      0.121  0.107  -0.014      99.560  90.000 -9.560
98      0.093  0.111  0.018      47.580  65.000  17.420
99      0.124  0.116  -0.008     115.560 100.000 -15.560
100     0.109  0.109  0.000      77.490  68.000 -9.490
101     0.127  0.116  -0.011      81.000  65.000 -16.000
102     0.129  0.114  -0.015     120.600 108.000 -11.400
103     0.127  0.114  -0.013     118.950 150.000  31.050
104     0.121  0.137  0.016      65.320  85.000  19.680
105     0.106  0.106  0.000      73.980  65.000 -8.980
106     0.125  0.114  -0.009      70.490  65.000 -5.490
107     0.202  0.227  0.025      55.487  45.000  15.487
-----

```

Figure 3.18: Results of Tool Wear and Surface Roughness Network

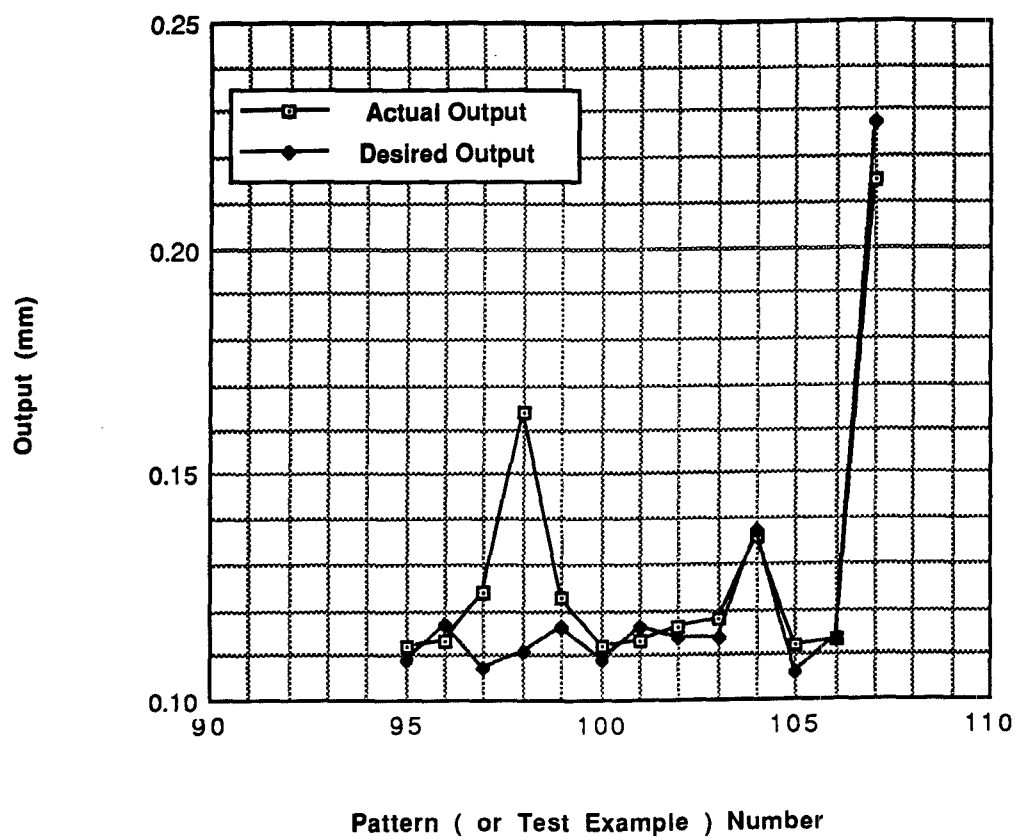


Figure 3.19: Plots of Tool Wear Model

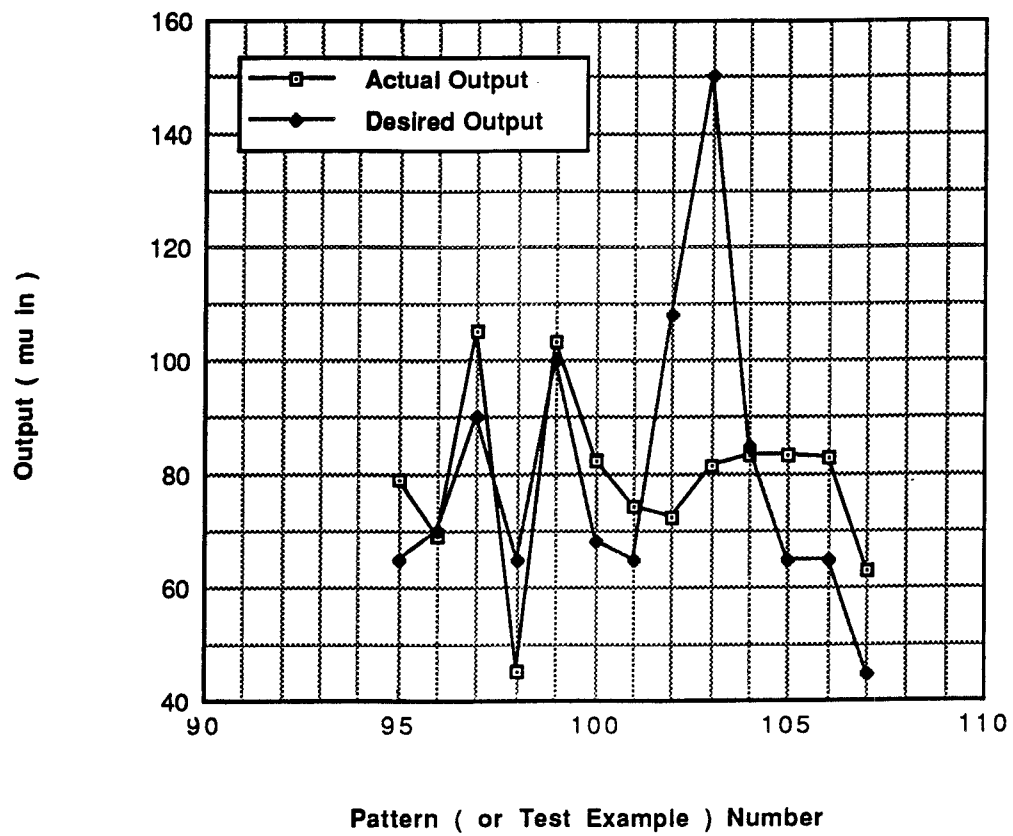


Figure 3.20: Plots of Surface Roughness Model

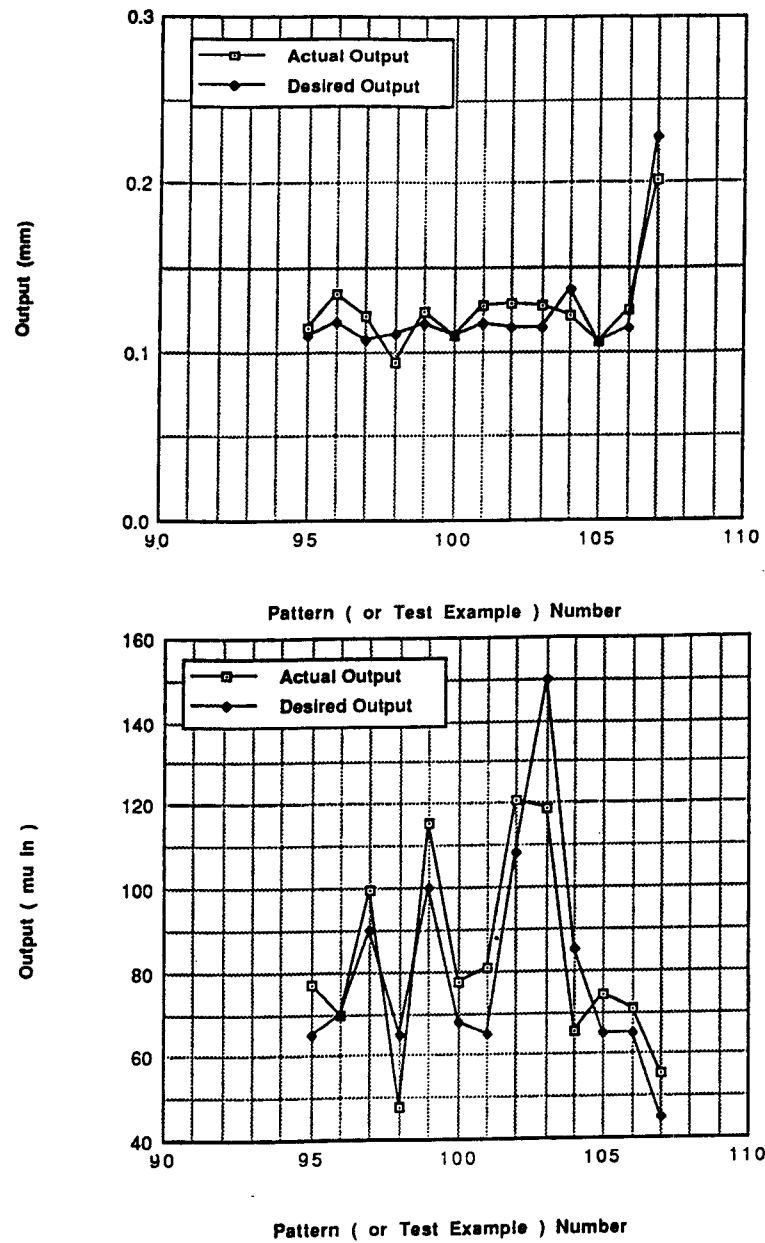


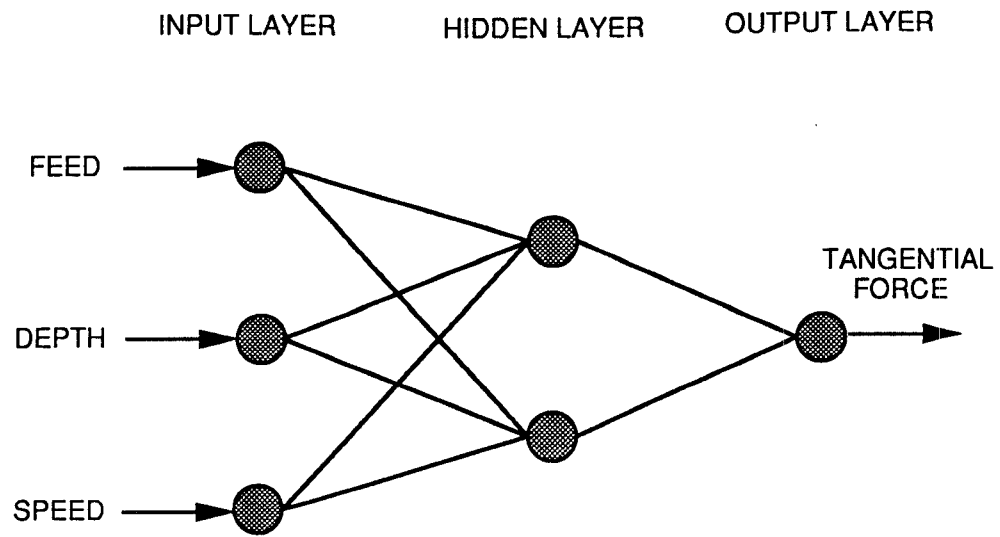
Figure 3.21: Plots of Tool Wear and Surface Roughness Model

Improved processing methods in the manufacturing of ceramic materials have achieved high strength, fracture toughness, and uniform properties. The goal of the ceramic machining is to provide measurement methods, data, and mechanistic information needed by industry to develop innovative and cost-effective planned activities like obtaining machining data for optimization of grinding, elucidating the chemo-mechanical interactions during machining etc. Now-a-days, strategic material considerations and economic factors are forcing the choice towards the use of ceramic components.

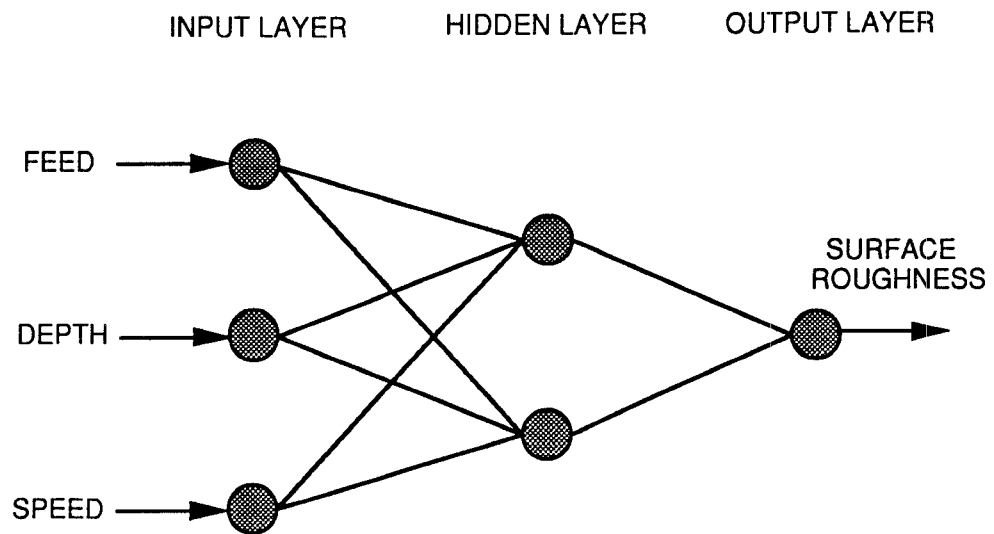
3.4.2 Effect of Cutting Parameters on Tangential Force and Surface Roughness

For a ceramic materials, a feedforward back-propagation network is used to find the relationship between the cutting parameters, and the tangential force, and the surface roughness. Machining of ceramics is more difficult than machining of metals. One can see from results of ceramic machining (chapter 4), that the feed force is more than the tangential force, which is reverse in the case of a metal machining. This is mainly because of the hardness of ceramics leading to a cutting mechanics dominated by fracture instead of plastic deformation. The structure of the neural network used for this work is illustrated in Figure 3.22. The architecture of the network is: three input nodes, one output node, and two hidden nodes. Each of the input nodes correspond to a single input parameter, and similarly each output node is uniquely associated with the one of the output parameters. Two different types of networks are used for this work. They are as follows:

- In the first network tangential force is predicted by giving feed, depth and speed to the input layer.
- In the second network surface roughness is predicted by giving feed, depth, and speed to the input layer.



(A) Tangential Force Model



(B) Surface Roughness Model

Figure 3.22: Architecture of Neural Network Used for Ceramics

PARAMETER VALUE	CHANGED PARAMETER	ACTUAL OUTPUT	CHANGED OUTPUT	PERCENTAGE CHANGE
0.40 mm/rev	0.36 mm/rev	1.88 lbs	1.523 lbs	-18.99%
4 mils	3.6 mils	1.88 lbs	1.836 lbs	-2.3%
600 rpm	540 rpm	1.88 lbs	1.0156 lbs	-45.98%

Table 3.4: Effect of Cutting Parameters on Tangential Force

After determining the architecture, the network is trained with the training examples, where each example is a set of input and output parameters. The training examples were obtained from ceramics tests, as explained in chapter 4. The steps for training are similar to the earlier case, i.e., steps used to train the network for steel bars. The results are summarized in Figures 3.23 and 3.24.

3.4.3 Conclusions

1. Considering the first type network (where tangential cutting force is predicted), we see from the results that the effect of depth of cut is very small. This can be explained in the following way. Each cutting parameter value is decreased by an amount of 10% of original value and finding its effect on the output from the trained neural network. Only one cutting parameter value is changed. The results are as follows:

From above table we can say that depth of cut has the least effect on the process (about 2.3%) and speed has the highest positive influence (about 45.98%) on tangential force. Feed has medium effect (about 18.99%). The remaining percentage is due to other factors, such as tool wear etc.

The best quality of the machined part can be obtained at a low feed (0.2 mm/rev) and a low cutting speed (78.5 ft/min).

2. Considering the second type of network (where surface roughness is predicted), we see from the results that the effect of depth of cut on surface

TRAINING COMPLETE after 5472 iterations

Weights of Neurode Interconnections:
Hidden Layer:

Mid Neurode # 0
-4.314 -2.858 -2.974
Mid Neurode # 1
-5.642 -3.880 -7.926

Output Layer:

Out Neurode # 0
2.941 -4.442

Neurode Threshold values:

Threshold for mid node 0 is: 2.105

Threshold for mid node 1 is: 11.328

Threshold for out node 0 is: 2.055

Network's outputs:

	ACTUAL OUTPUT lbs	DESIRED OUTPUT lbs	ERROR
PATTERN # 0 :-	0.747	0.743	0.004
PATTERN # 1 :-	0.747	0.749	0.002
PATTERN # 2 :-	0.105	0.092	-0.013
PATTERN # 3 :-	0.105	0.119	0.014
PATTERN # 4 :-	0.024	0.010	-0.014
PATTERN # 5 :-	0.024	0.029	0.005
PATTERN # 6 :-	0.433	0.460	0.027
PATTERN # 7 :-	0.433	0.410	-0.023
PATTERN # 8 :-	0.189	0.200	0.011
PATTERN # 9 :-	0.189	0.181	-0.008
PATTERN # 10 :-	0.965	0.990	0.025
PATTERN # 11 :-	0.965	0.939	-0.026
PATTERN # 12 :-	1.931	1.882	-0.049
PATTERN # 13 :-	1.931	1.982	0.051
PATTERN # 14 :-	2.198	2.250	0.052
PATTERN # 15 :-	2.198	2.150	-0.048
PATTERN # 16 :-	1.523	1.881	0.358
PATTERN # 17 :-	1.836	1.881	0.045
PATTERN # 18 :-	1.016	1.881	0.865

Figure 3.23: Results of Neural Networks for Ceramics (Tangential Force)

TRAINING COMPLETE after 10501 iterations

Weights of Neurode Interconnections:

Hidden Layer:

Mid Neurode # 0

-6.181 1.837 6.688

Mid Neurode # 1

0.747 -5.458 -1.009

Output Layer:

Out Neurode # 0

-4.015 -2.896

Neurode Threshold values:

Threshold for mid node 0 is: -1.351

Threshold for mid node 1 is: 0.910

Threshold for 0 out node is: 2.495

Network's outputs:

	ACTUAL OUTPUT (mu in)	DESIRED OUTPUT (mu in)	ERROR
PATTERN # 0 :-	0.935	0.938	0.003
PATTERN # 1 :-	0.935	0.940	0.005
PATTERN # 2 :-	0.959	0.968	0.019
PATTERN # 3 :-	0.959	0.949	-0.010
PATTERN # 4 :-	1.564	1.549	-0.015
PATTERN # 5 :-	1.564	1.578	0.014
PATTERN # 6 :-	1.987	1.969	-0.018
PATTERN # 7 :-	1.987	2.000	0.013
PATTERN # 8 :-	0.053	0.056	0.003
PATTERN # 9 :-	0.053	0.055	0.002
PATTERN # 10 :-	0.671	0.631	-0.040
PATTERN # 11 :-	0.671	0.709	0.038
PATTERN # 12 :-	1.158	1.121	-0.037
PATTERN # 13 :-	1.158	1.190	0.032
PATTERN # 14 :-	1.124	1.170	0.046
PATTERN # 15 :-	1.124	1.080	-0.044
PATTERN # 16 :-	0.802	1.120	0.318
PATTERN # 17 :-	1.086	1.120	0.034
PATTERN # 18 :-	1.503	1.120	-0.383

Figure 3.24: Results of Neural Networks for Ceramics (Surface Roughness)

PARAMETER VALUE	CHANGED PARAMETER	ACTUAL OUTPUT	CHANGED OUTPUT	PERCENTAGE CHANGE
0.40 mm/rev	0.36 mm/rev	1.12 μm	0.8015 μm	-28.44%
4 mils	3.6 mils	1.12 μm	1.086 μm	-3.1%
600 rpm	540 rpm	1.12 μm	1.503 μm	+34.19%

Table 3.5: Effect of Cutting Parameters on Surface Roughness

roughness is very small. This can be explained in the same way as above.

The results obtained are as follows:

From above table we can say that as speed is decreased, surface roughness increases (about 34.19%). Again the depth of cut has least effect on surface roughness (about 3.1%). Feed has a medium effect (about 28.44%).

The best surface quality can be obtained by having a low feed (0.2 mm/rev) and a high cutting speed (118 ft/min). From the results it is seen that the percentage of predicting tangential force and surface roughness is 95% and 98% respectively, from the trained neural network model.

Chapter 4

Experimental Verification

Overview

In this chapter, a series of machining tests have been performed to investigate the feasibility of using sensor signals for on-line monitoring of machining processes. This chapter is divided into three sections. The first section discusses the experimental setup used to perform the machining tests. Measurements of tool wear and cutting force were discussed. The second section discusses the design of an instrumented transducer. Machining of ceramics is discussed in section three. The last section describes the analysis of data collected during the machining tests.

4.1 Experimental Arrangement

4.1.1 Description of the Experiments

Experiments were conducted in the machine lab of the Mechanical Engineering Department on the College Park campus. The experimental arrangement used to perform the machining tests is illustrated in Figures 4.1 and 4.2. A tool holder with a carbide insert and a designed force transducer was mounted on

the cross slide of the lathe for cutting force and tool wear measurements. The force transducer was connected to two strain indicators to measure the amount of deflection that the cutting tool influenced in two directions (i.e., tangential and feed directions). The cutting force signals were amplified using an amplification ratio of 1:100. The amplified force signals were sampled simultaneously using a 4×4 card from a rapid system, which was connected to an IBM computer. The data record of the signal obtained with each channel of the rapid system contained 2048 points. All the data acquired during machining tests was stored on floppy disks for further analysis. The signals from the amplifier was connected to two multimeters (which measures the voltage output) to monitor the cutting tests and to validate data by the rapid system.

Several issues were considered before starting of the actual machining test. They are as follows:

- The flank wear is the major concern in these cutting tests.
- The material used in the experiments is tough 1018 carbon steel.
- As the amount of metal removal is large at high cutting speeds, five carbon steel bars of length 58 cms, cut from the same stock of diameter 10.16 cms, were used for machining tests. This ensures the same tool-work pair microstructure.
- Since one wear test requires several layers of the workpiece to be removed, i.e., a test involves several diameters. Constant surface speed control is necessary to maintain a constant cutting speed. By using large diameter carbon steel bars, the percentage change in cutting speed can be kept within 5%.



Figure 4.1: Experimental Arrangement for Conducting the Machining Tests



Figure 4.2: Setup of Data Acquisition System

4.1.2 Test Procedure

Continuous Wear Tests

A test insert is used and the steel bar is machined along its length without the coolant. The coolant not used to accelerate the flank wear of the insert. The cut was interrupted and the flank wear was measured using an optical microscope (as explained in the next section). To match the wear with the signals measured, three types of cutting force signals were recorded: signals just after start of the cut, signals in between the cut and signals just before the cut was interrupted. After each test cut, a separate insert was used to clean-up the test bar diameter to avoid unnecessary rubbing of the test insert's flank during the clean-up cuts. Whenever the diameter of the bar became small, a new test bar of the same material was used in order to maintain constant cutting speed. More than eight inserts were used in the tool wear tests. Because of the high cutting speeds (170 m/min) and depth of cut (1 mm) used, some thermally-induced tool failures were encountered. In some instances, tool chipping, and spalling of the cutting edge necessitated stopping of the machining test.

Artificial Wear Tests

This group of experiments were performed in a manner similar to that described in the previous section. The Same material, i.e., carbon steel, was used for the test bar to run the tests. To study the cutting force variation under progressively increasing flank wear, carbide inserts were artificially ground to obtain different flank wear land sizes. To assure realistic simulation of cutting with flank wear, the tools with artificial flank wear were used in turning until an observable increase in flank wear occurred. This can be achieved by running the test for some time before collecting the force signals from the sensor. These conditions of artificially worn tools were used to assess the feasibility of real-time tool wear sensing by measuring the cutting force components during turning. The cutting force signals

were sampled and recorded through an IBM computer.

4.1.3 Tool Wear Measurement

Tool wear measurement was performed in the Measurements Lab of the Mechanical Engineering Department at the College Park campus. Tool wear was measured using an optical microscope. Figure 4.3 illustrates the basic structure of the optical microscope.

Test Procedure

The issues considered before starting the tool wear measurements are as follows:

- In order to utilize the optical microscope fully and efficiently, i.e., to get the best pictures of the tool flank, the correct magnification and exposure time should be used. The distance between the optical microscope and the test place is minimized.
- Before starting the experiment, photographic films are kept ready to take pictures of worn out flank of the tool. Polaroid 55 films were used to get the pictures of magnified tool wear.
- One can get the best picture of tool wear by using necessary filters during tool wear measurement.

After the test was interrupted, the tool insert should be placed in correct position on the microscope table, i.e., the flank face of the insert should be focussed to the light coming from the microscope light source. Pictures of tool wear were obtained by adjusting the knobs of the microscope. By using the polaroid films, pictures of flank wear were taken and stored for further analysis. A magnification of 100 and an exposure time of 4 seconds was used.

The next step is to measure the wear land. Mean wear land was measured using vernier calipers by taking an average of wear land measurements made.

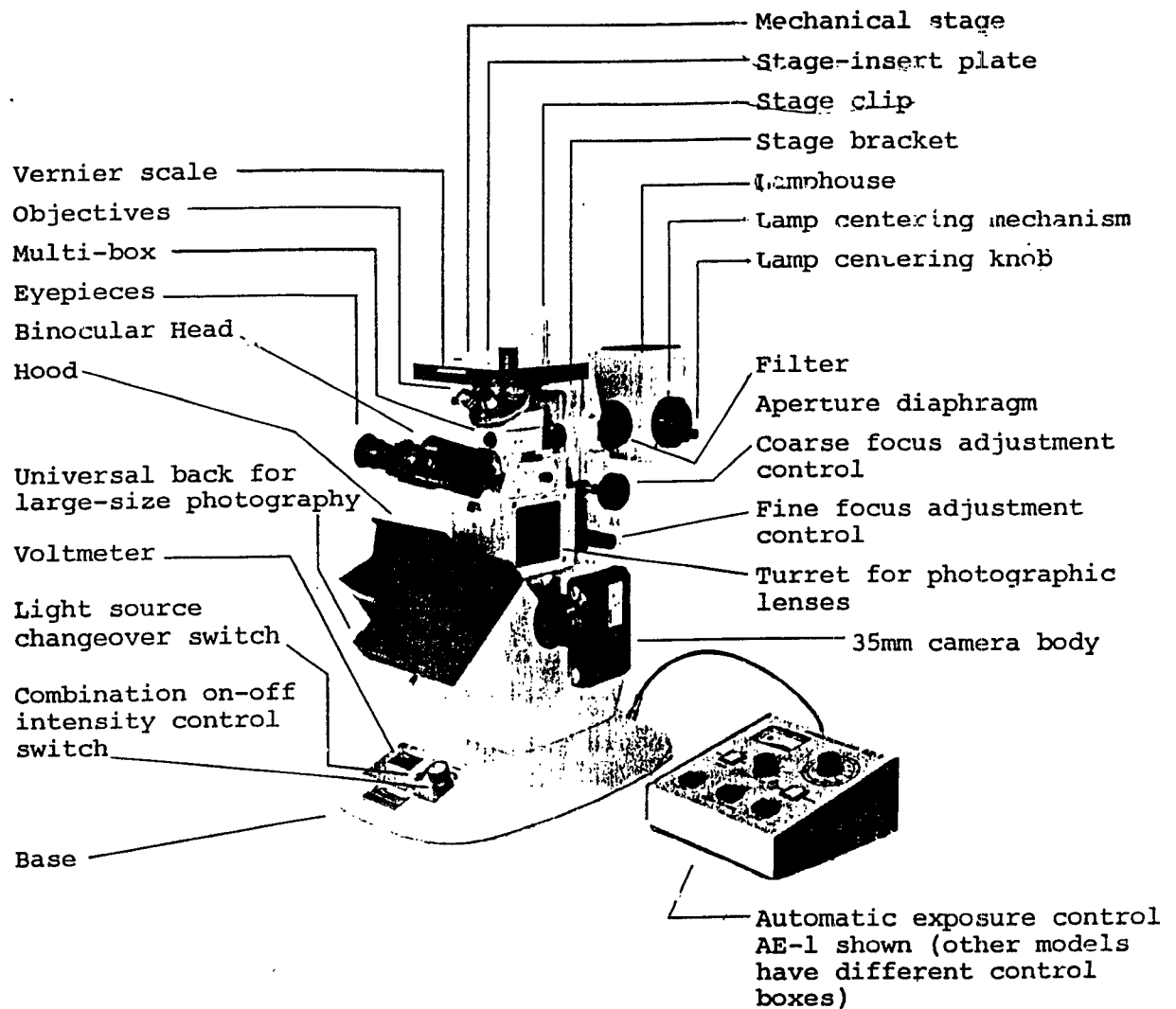


Figure 4.3: Structure of an Optical Microscope

Time (min)	Wear (mm) (169.56 m/min)	Wear (mm) (113 m/min)
0	0.0000	0.0000
5	0.0695	0.0480
10	0.1096	0.0511
15	0.1198	0.0632
20	0.1316	0.0753
25	0.1416	0.0988
30	0.1926	0.1196
35	0.2066	0.1257
40	0.2691	0.1365
45	-	0.1476
50	-	0.1587
55	-	0.1601
60	-	0.1723
65	-	0.1784
70	-	0.1878
75	-	0.1923
80	-	0.2120

Table 4.1: Tool Wear Land Values at Two Different Cutting Speeds

The mean wear measurement reported is prone to error, as the natural wear land shape is irregular. As the mean wear measured has a subjective component (a judgement of the average wear land), some inaccuracy is inevitable. The measured wear lands for two cutting speeds are shown in Table 4.1.

After finishing the tool wear measurement, care should be taken to keep the inserts in the same position as before the cut. This is achieved by using a cutting tool, whose indentation (or groove) for insert matches exactly with the shape of insert. Though we tried to minimize this effect completely, but some inaccuracy

is unavoidable. Before starting the next cut, care should be taken to fix the insert tightly to the cutting tool. Otherwise this may cause damage to the insert, or spoil the surface of the machined work piece, and in an extreme case it may spoil the tool cross slide.

4.1.4 Cutting Force Measurement

In order to obtain the data to investigate the feasibility of cutting force signals for on-line tool conditioning, cutting tests were carried out. The cutting force measuring system consists of:

1. A transducer or sensor, which measures the two cutting force signals from real-time machining tests,
2. An amplifier system to amplify the measured signals,
3. A digital data acquisition system.

The experimental setup for cutting force measurements is illustrated in Figure 4.1 and the data acquisition system is shown in Figure 4.2. An instrumented transducer is designed to measure the cutting force signals. During machining, the voltage outputs from the force sensor system were recorded on IBM computer disks and further analyzed to obtain the actual output voltage. After manipulation, the two force components were obtained under the given conditions.

Two levels of depth of cut, feed and spindle speeds were used to conduct this experiment. These levels are 0.07 mm/rev and 0.184 mm/rev for feed, 0.5 mm and 1 mm for depth of cut, and 375 rpm and 625 rpm for spindle speeds. Table 4.2 shows the cutting force components for one setting of machining parameters. Table 4.3 shows the cutting force variations for artificial flank wear land.

Time (min)	Tangential Force (Newtons)	Feed Force (Newtons)	Force Ratio
0	0.000	0.000	0.000
3.5	868.569	242.544	0.272
7.0	893.625	255.722	0.286
10.5	946.239	288.278	0.300
14.0	927.866	278.782	0.301
17.5	982.152	309.666	0.315
21.0	1004.784	319.138	0.3176
24.5	1071.514	332.727	0.310
28.0	1085.71	364.023	0.335
31.5	1070.547	329.728	0.308
35.0	1102.416	396.966	0.360

Table 4.2: Measured Tangential and Feed Forces (feed=0.14 mm/rev, depth of cut=1.00, speed=625)

Tool Wear (mm)	Tangential Force (Newtons)	Feed Force (Newtons)	Force Ratio
0.162	739.62	132.184	0.179
0.202	799.083	174.228	0.218
0.270	912.833	202.518	0.221
0.310	978.811	279.606	0.286
0.370	1147.51	382.966	0.334

Table 4.3: Variation of Tangential and Feed Forces With Tool Wear (feed=0.14 mm/rev, depth of cut= 1.00 mm, speed=470 rpm)

4.2 Design of an Instrumented Transducer

An instrumented transducer basically consists of two parts: a tool holder and a force sensor system. The force sensor system consists of strain gage layouts, a bridge amplifier system, and a digital data acquisition system. The attached strain gages function as sensors to convert the cutting force signals into measurable electric voltage signals through the bridge amplifier system. A quantitative relation between the cutting force components and the measured voltage outputs from the digital data acquisition system, established from a calibration process, recovers the magnitudes of the cutting force components from the measured voltage signals.

Strain Gage Selection

Precision strain gages, CEA-06-062uw-350, are used for designing a transducer. The basic considerations involved in selecting strain gages are:

- *Heat Dissipation* is an important factor in selecting a strain gage. For an accurate strain measurement, small sizes of strain gages are preferred. However, larger sizes would be more preferable if the effect of temperature variation on the resistance of the strain gage is of concern.
- *Gage factor* is defined as the ratio of the resistance change to the strain received.

Design of Gage Layout

The cutting force produced during machining can mathematically be treated as a spatial vector, which is conventionally decomposed into its three components. The cutting force along the cutting speed direction, F_z , is known as tangential force, the component along the workpiece is known as feed force, F_x and the component of force perpendicular to the machined surface i.e., along radius, is

known as radial force. In this thesis work we find only two components of force, namely tangential and feed forces. In order to vary the resistance changes which active strain gages undergo when subjected to acting strains, Wheatstone bridges are usually employed as shown in Figure 4.4 [Za 86]. The basic principle of the Wheatstone bridge is that an initially balanced bridge with $R_1 * R_3 = R_2 * R_4$ (or the bridge stratus before the cutting force is applied). The voltage output of the bridge, ΔE_{out} , due to the resistance changes of ΔR_1 , ΔR_2 , ΔR_3 , ΔR_4 is given by

$$\Delta E_{out} = E_{in} \frac{r}{(1+r)^2} \left(\frac{\Delta R_1}{R_1} - \frac{\Delta R_2}{R_2} + \frac{\Delta R_3}{R_3} - \frac{\Delta R_4}{R_4} \right)$$

where,

E_{in} = bridge supply voltage and $r = \frac{R_2}{R_1}$.

The term $(\frac{r}{(1+r)^2})$ is an index to indicate the bridge circuit efficiency. The maximum bridge circuit efficiency will occur at $r=1$, i.e., at $R_1 = R_2$

The strain gages were calibrated prior to the tests to establish the transformation matrix between the voltage signals and the cutting force generated during the machining process. Figure 4.5 illustrates the set up for calibration process. The relationship between the applied forces and the voltage outputs of the strain gages are shown in Tables 4.4 and 4.5. The transformation matrix, T_r , is defined as the relation between the forces and voltages:

$$[T_r] = \frac{Input}{Output} = \frac{V_{out}}{F}$$

The above relation can be written in matrix form as:

$$[T_r] = \begin{bmatrix} \frac{\Delta V_x}{\Delta F_z} & \frac{\Delta V_z}{\Delta F_x} \\ \frac{\Delta V_y}{\Delta F_z} & \frac{\Delta V_z}{\Delta F_x} \end{bmatrix}$$

From the calibration process, we have:

$$[T_r] = \begin{bmatrix} T_{11} & T_{12} \\ T_{13} & T_{14} \end{bmatrix} = \begin{bmatrix} 0.0283 & 0.0000 \\ -0.0003 & 0.0253 \end{bmatrix}$$

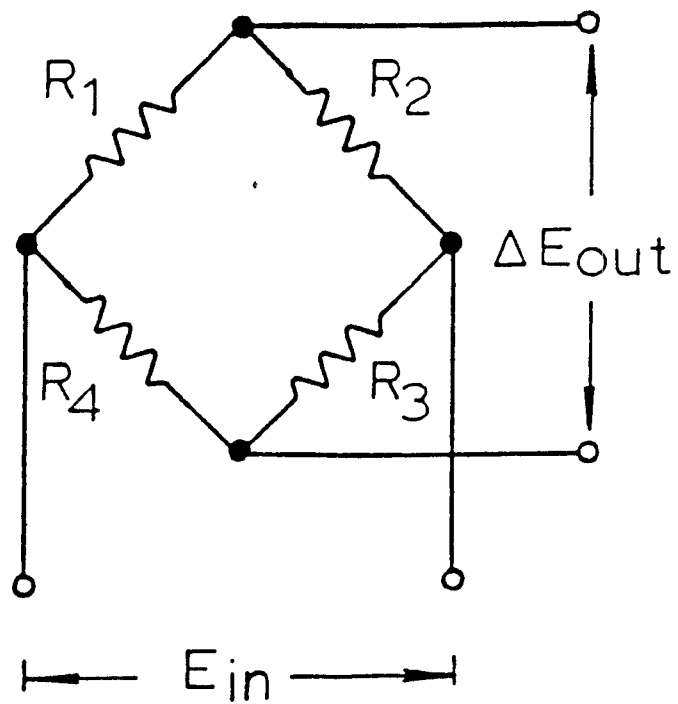


Figure 4.4: The Wheatstone Bridge Circuit

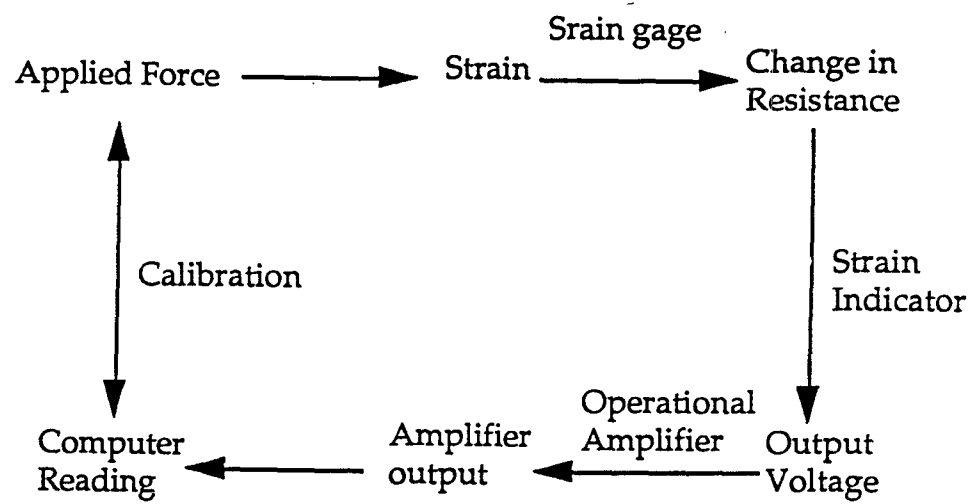


Figure 4.5: Strain Gage Calibration

Load (lbs)	Voltage Tangential (volts)	Voltage Feed (volts)
10.0	0.1186	0.4204
20.0	0.1227	0.7054
30.0	0.1192	0.9452
40.0	0.1119	1.2148
50.0	0.1097	1.4136
60.0	0.1036	1.7138
70.0	0.1038	1.9307
80.0	0.0976	2.2067
90.0	0.1039	2.4667

Table 4.4: Calibration in Feed Direction

Each element of $[T_r]$ is a relation of the i^{th} voltage output and the j^{th} cutting force component. The force matrix is calculated as:

$$[F] = (inv([T_r])) [V]$$

The above reaction can be written as:

$$\begin{bmatrix} F_z \\ F_x \end{bmatrix} = \begin{bmatrix} 35.3357 & 0.0000 \\ 0.4190 & 39.5257 \end{bmatrix} \begin{bmatrix} V_z \\ V_x \end{bmatrix}$$

4.3 Case Study: Machining of Ceramics

4.3.1 Experimentation

Machining tests of ceramics were conducted to investigate the possible chemo-mechanical effects on ceramics materials and to study the effect of cutting parameters on forces and surface roughness. The experiments were conducted in

Load (lbs)	Voltage Tangential (volts)	Voltage Feed (volts)
10.0	0.4137	0.2173
20.0	0.7137	0.2199
30.0	0.9815	0.2209
40.0	1.2855	0.2193
50.0	1.5800	0.2206
60.0	1.8426	0.2199
70.0	2.1349	0.2189
80.0	2.4064	0.2198
90.0	2.4064	0.2198

Table 4.5: Calibration in Tangential Direction

the Advanced Design and Manufacturing Lab of the Mechanical Engineering Department at the College Park campus. The experimental setup used to perform the tests is illustrated in Figure 4.6. We used a CNC machine to conduct the machining tests. A tool holder with Polycrystalline Diamond Compact (PDC) tool insert and a designed force transducer were fixed on the machine tool table with the help of a fixture. The workpiece, aluminium oxide ceramic, is mounted to the spindle of the machine, which can be rotated at any desired speed. The setup for measuring the cutting forces is the same as that used for machining steel bars (as explained earlier).

Ceramic bars of length 10.16 cm and diameter 1.91 cm are used for machining tests. A 2-factorial design of cutting parameters was used for the tests i.e., two levels of each cutting parameters were used. Specifically, 0.2 mm/rev and 0.40 mm/rev for feed, 4 mils and 8 mils for depth of cut, and 400 rpm and 600 rpm for spindle speed were used. Two cutting force signals (feed force and tangential force) were monitored during the tests (as explained earlier). The experimental

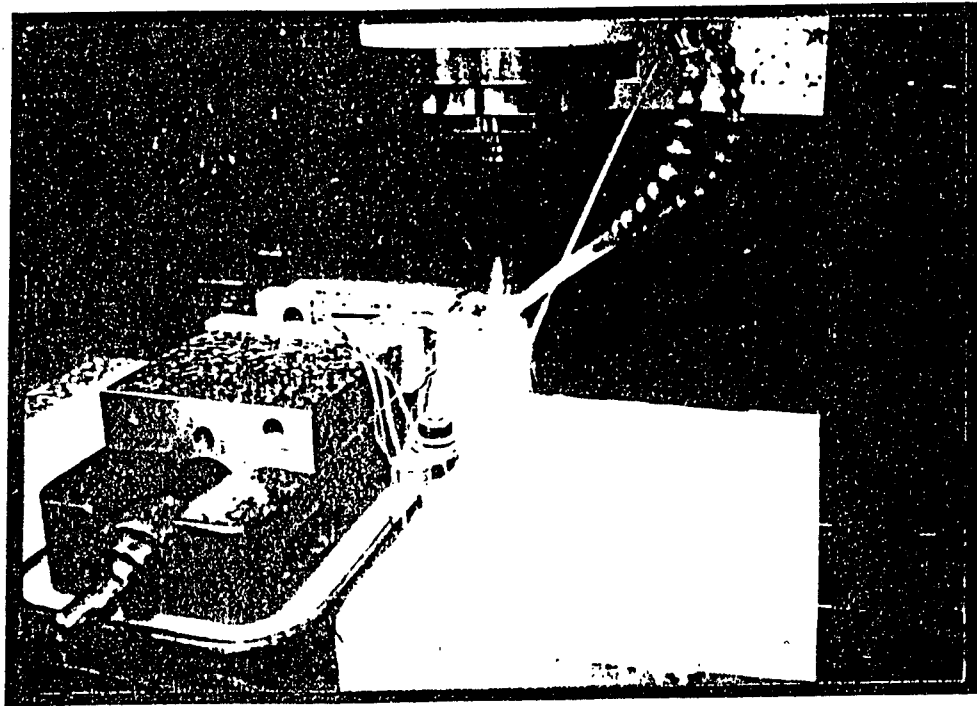
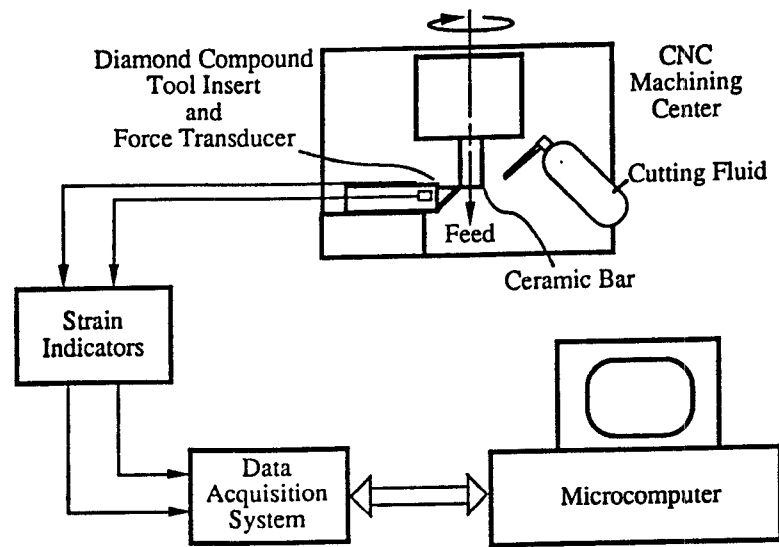


Figure 4.6: Experimental Setup For Ceramic Tests

Speed (rpm)	Feed in/min	Depth mils	Tangential Force lbs	Feed Force lbs	Surface Roughness μm
400	0.2	4.0	6.05	12.13	2.83
400	0.2	8.0	5.22	20.73	1.57
400	0.4	4.0	6.51	12.85	2.77
400	0.4	8.0	4.25	19.74	1.57
600	0.2	4.0	6.01	12.72	2.40
600	0.2	8.0	4.25	19.74	1.57
600	0.4	4.0	4.26	12.08	1.97
600	0.4	8.0	5.81	18.40	1.53

Table 4.6: Experimental Results for Type 1 Fluid

results are summarized in Tables 4.6 and 4.7. Figures 4.7 and 4.8 are the graphical representations of the experimental results, i.e., the measured tangential cutting force and surface roughness. Figure 4.9 shows the tool wear of the insert during the initial stage (4 mts machining) and at the final stage (16 mts of machining).

4.4 Data Analysis

This section analyzes the data collected during the turning tests. Taylor's tool life equation is derived from the experimental tool wear curves.

4.4.1 Derivation of Taylor's Equation

Definition of Taylor's Equation

F.W Taylor was the first to investigate the relationship between tool life and cutting speed. Taylor's well known *tool life – cutting speed* relation is as follows:

$$VT^n = C_T$$

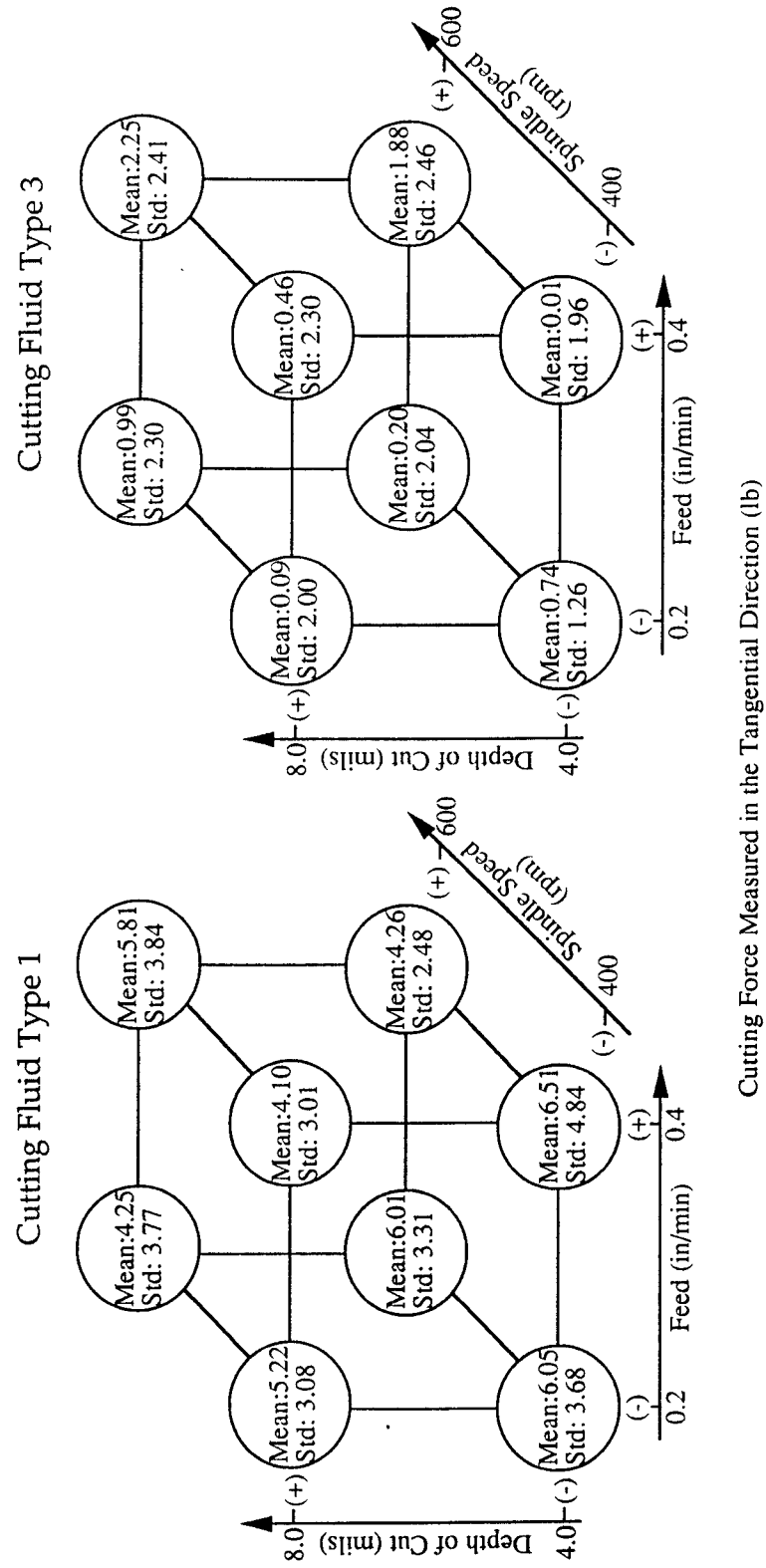


Figure 4.7: Factorial Design of Tangential Cutting Force

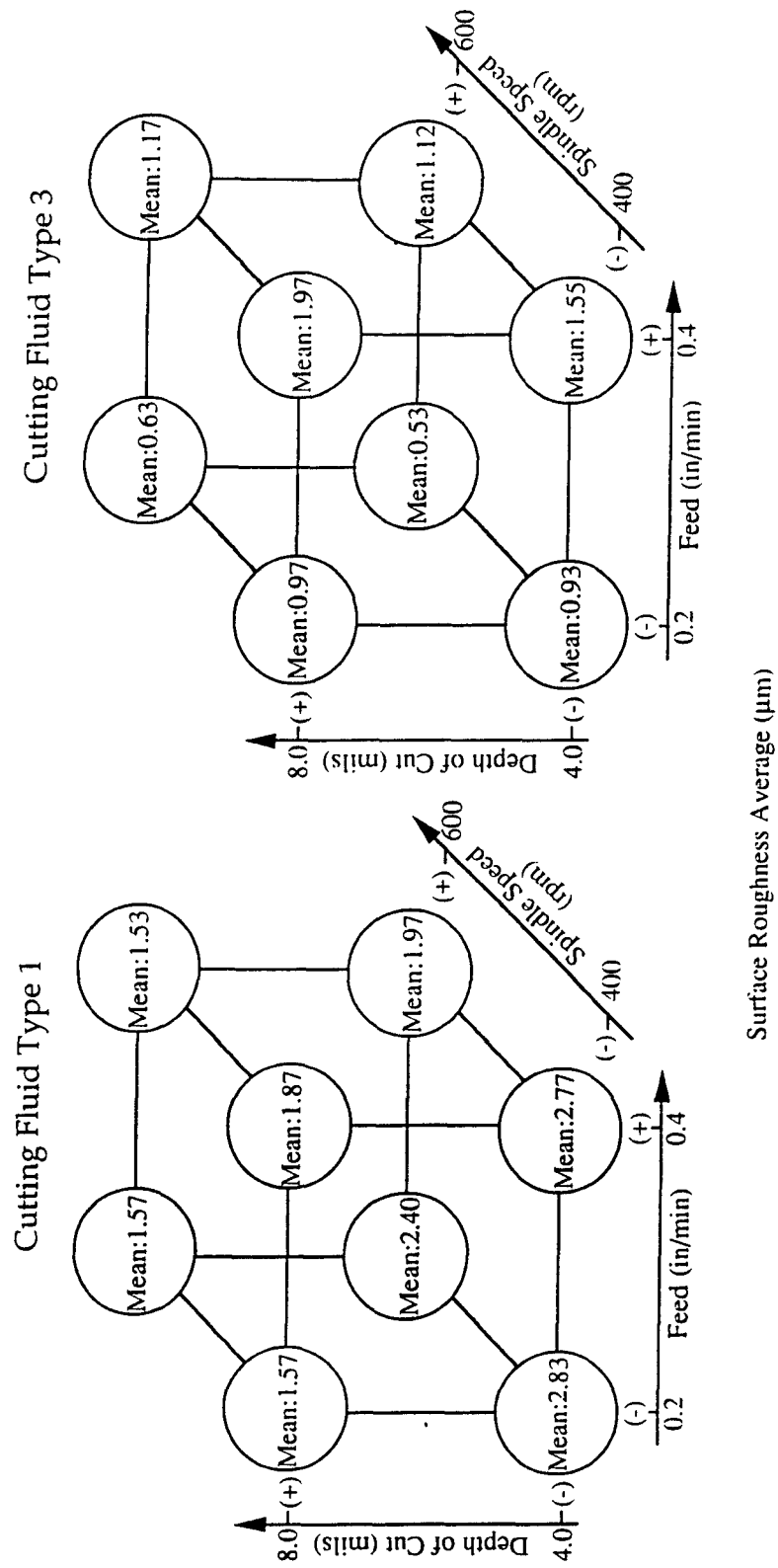


Figure 4.8: Factorial Design of Surface Roughness

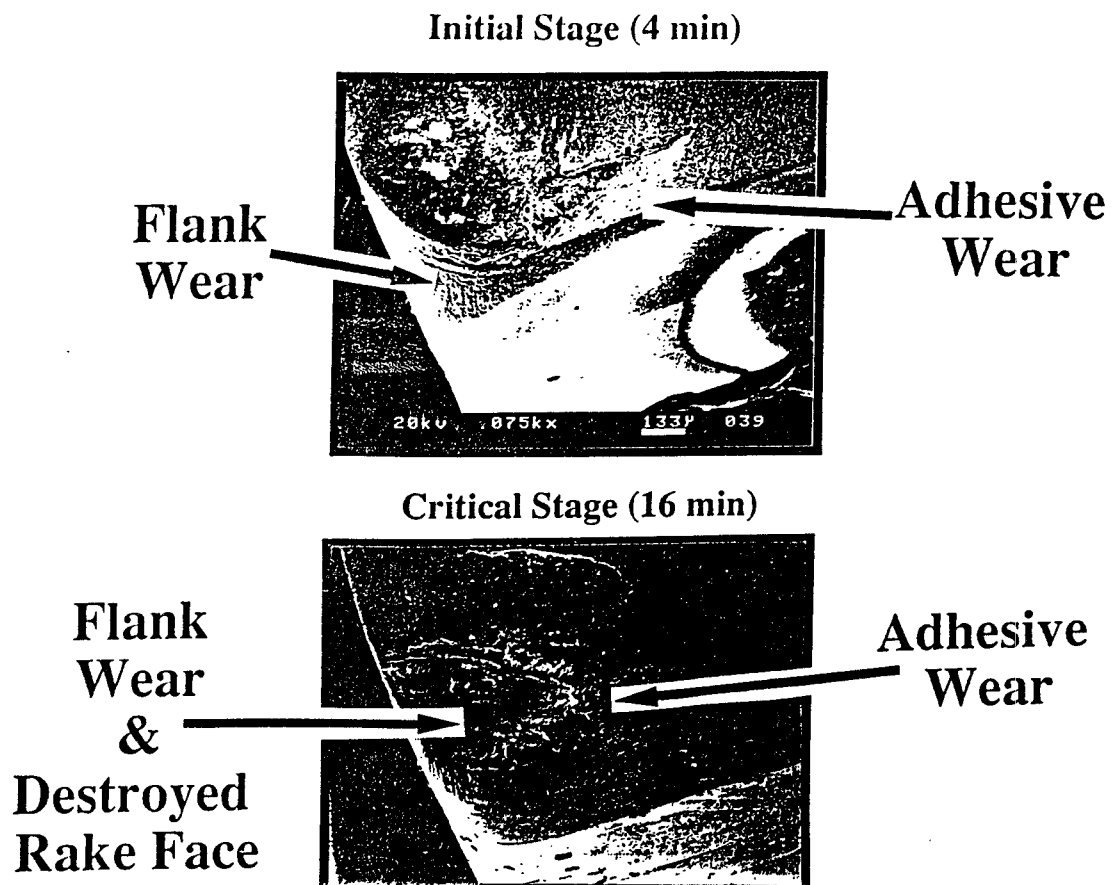


Figure 4.9: Tool Wear of Diamond Insert

Speed (rpm)	Feed in/min	Depth mils	Tangential Force lbs	Feed Force lbs	Surface Roughness μm
400	0.2	4.0	0.74	16.00	0.93
400	0.2	8.0	0.09	24.00	0.97
400	0.4	4.0	0.01	18.4	1.55
400	0.4	8.0	0.46	30.00	1.97
600	0.2	4.0	0.20	15.44	0.53
600	0.2	8.0	0.99	21.20	0.63
600	0.4	4.0	1.88	19.03	1.12
600	0.4	8.0	2.25	24.70	1.17

Table 4.7: Experimental Results for Type 3 Fluid

where,

V = cutting speed in m/min

T = tool life in minutes

n = exponent mainly depending on tool material

C_T = Taylor's constant, mainly depending on the tool and workpiece material. Very often it is a function of the machining data.

The physical significance of C_T is that it is the cutting speed for a tool life of 1 min. However, the equation does not take into consideration the magnitude of the chip-cross sectional area and thus a different Taylor's constant is required for each chip cross-sectional area. If the tool life is too short, the tool must be reconditioned very often. On the other hand, when tool life is too long, the cutting speed is too low, causing losses due to high machining time. According to Taylor's findings the ratio of tool life between grinds and time for regrinding the tool should, be between 7 and 35, i.e., tool life should be at least seven times, but not more than thirty five times the grinding time.

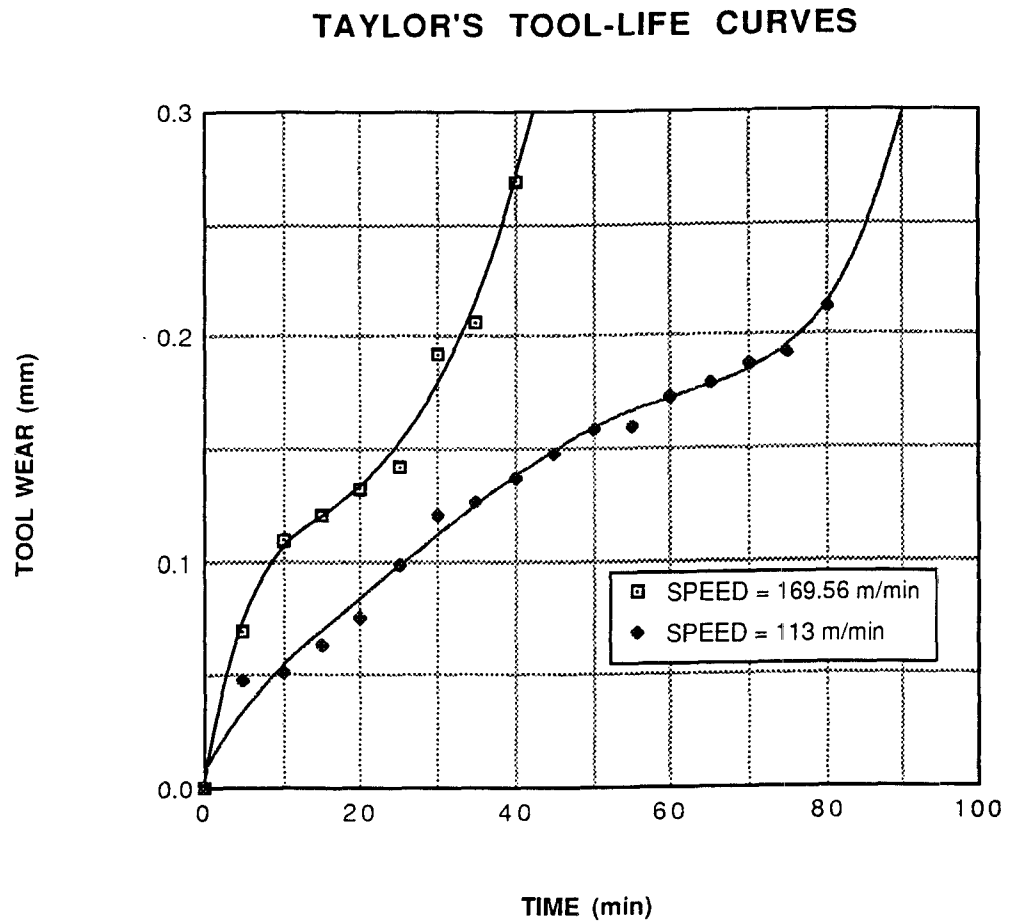


Figure 4.10: Tool Wear Curves Obtained from Experimental Results

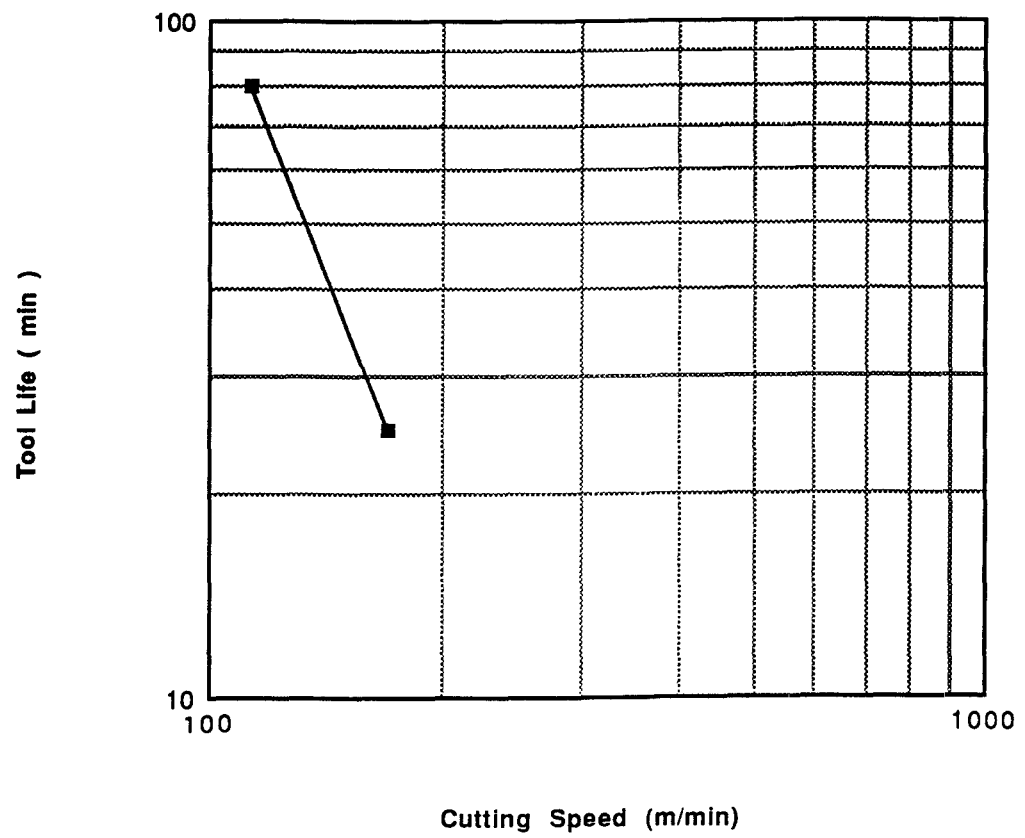


Figure 4.11: Log-Log Plot of Cutting Speed and Tool Life

Derivation

Tool wear curves are drawn using the results in Table 4.5 and are illustrated in Figure 4.10. The following results are deduced from the figure, for a tool wear of 0.20 mm.

- For 0.20 mm of tool wear for a cutting speed of 169.56 m/min, the tool life is 25 minutes.
- For 0.20 mm of tool wear for cutting speed of 113 m/min, the tool life is 80 minutes.

From Figure 4.11 (for tool wear of 0.20 mm), we have:

$$\begin{aligned}\tan \alpha &= \frac{\log Tl_1 - \log Tl_2}{\log v_2 - \log v_1} \\ &= \frac{\log \frac{Tl_1}{Tl_2}}{\log \frac{v_2}{v_1}} \\ &= \frac{\log \frac{80}{25}}{\log \frac{169.56}{113}} \\ &= 2.866\end{aligned}$$

We know that, $\tan \alpha = \frac{1}{n}$

$$\Rightarrow n = 0.3488$$

Therefore, Taylor's constant C_T is derived as:

$$C_T = 169.56 * 25^{0.349} = 521.44$$

So, we can write Taylor's tool life equation as:

$$VT^n = 521.44$$

Conclusions

We conducted experiments to validate the developed neural network program. Two types of experiments were performed: machining of steels and machining of

the advanced ceramic materials. A force transducer is designed and monitored the cutting forces successfully. It is found that unlike the case of machining of metals, the feed force is more than the tangential force in the case of machining of ceramics. Tool wear and surface roughness are measured to analyze the machining process. Taylor's tool life equations were derived from the measured tool wear. All the data collected from the experiments were used for training and testing of the developed neural network (as explained in chapter 3).

Chapter 5

Framework of an On-line Monitoring System

Overview

This chapter describes about the framework of an on-line monitoring system and is divided into five sections. The first section discusses the basic requirements of a monitoring system; namely, a sensor system, an ANN on-line monitor, and an expert system. The second section discusses the implementation of developed components of an on-line monitoring system. The advantages of instrumented transducer and on-line tool wear monitoring are also discussed. The third section describes the merits and demerits of feedforward back-propagation neural networks. A control system for the machining process is discussed in the fourth section. Finally, a brief description of the developed program using feedforward back-propagation network (which predicts the tool wear and surface roughness) is discussed.

5.1 Basic Requirements of an On-line Monitoring System

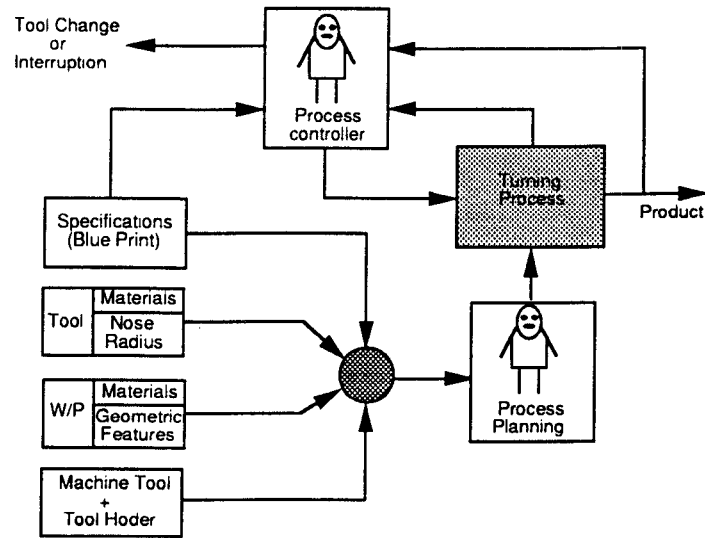
The demand for a shortened product cycle time, reduced waste, and a consistent high level of quality is a constant concern in the manufacturing environment. The implementation of an on-line monitor for the manufacturing processes from conception to the finished product stage in an untended machining environment will allow the manufacturer to better meet that demand. Figure 5.1 presents the outline of an monitoring process. Figure 5.1.a represents the conventional turning operation where a human operator is needed to intelligently control the turning process. Figure 5.1b represents an ideal control scheme where the on-line model is so accurate that the prediction of cutting tool wear status assures a prompt tool change. By using an intelligent process controller, an open loop control is sufficient to replace the human operator. However, such an on-line model does not exist presently. To replace an intelligent human operator, one can go for an on-line monitoring system as shown in Figure 5.2 [ZaHw 90A, ZaHw 90B, ZaHa 90].

The basic requirements of an on-line monitoring system are:

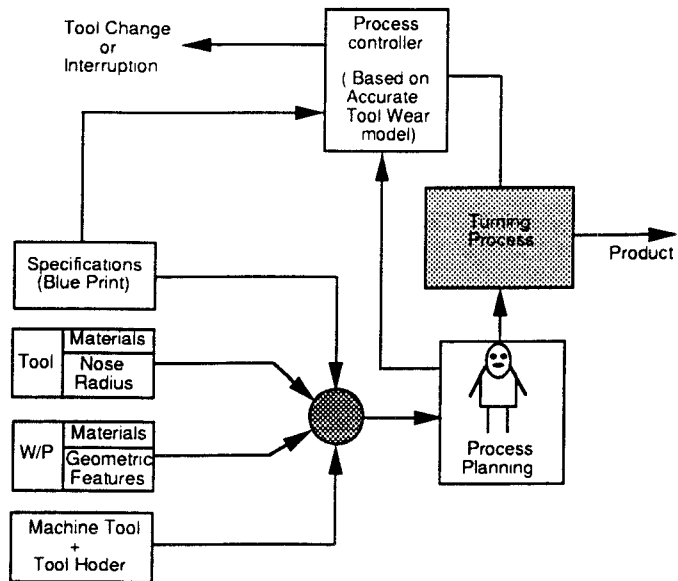
1. sensor signal detection system
2. an on-line monitor system
3. an expert system

5.1.1 Sensor Signal Detection System

A successful on-line monitoring system heavily relies on the reliability of the measured signal(s) from the sensor(s). The basic principle of the indirect on-line monitoring techniques is to retrace the state of the machining process (for example tool wear, surface roughness) through accessible process variables. A sensor is developed because of it's suitability for monitoring the cutting process



(A) Conventional Turning Operation Procedure



(B) Ideal Control with an Open-Loop System

Figure 5.1: Conventional and Ideal On-Line Monitoring Turning-Process

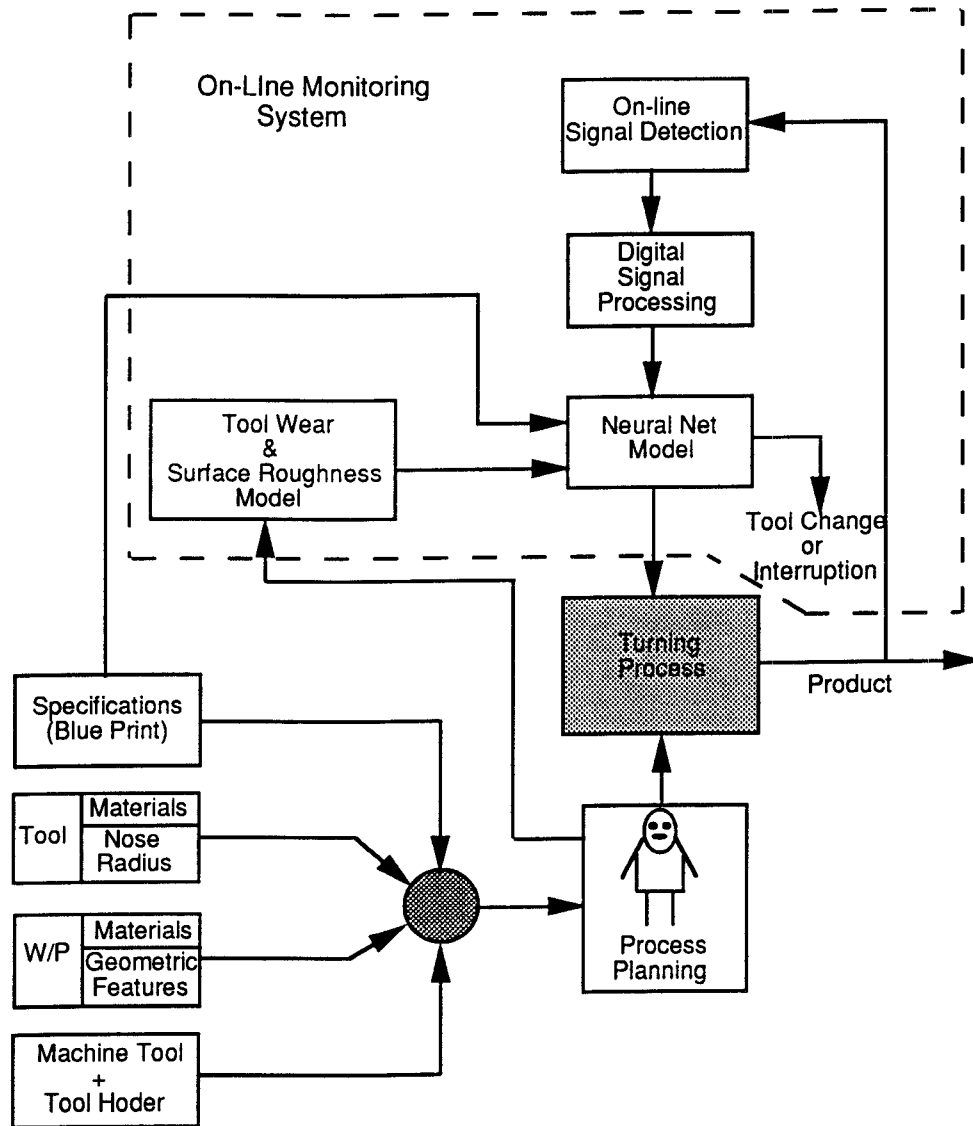


Figure 5.2: Methodology of an On-line Monitoring of a Turning Process

and the tool wear process (as stated before in chapter 3). This sensor captures wear-related signals during the machining process (e.g., turning process). The detected analog signals are digitized, fed to a buffer, stores into disks, and finally analyzed based on a desired signal processing scheme. The results from the digital signal processing are organized into a coding system (on-line monitor).

5.1.2 On-line Monitoring System

The integration of signal detection system and process model will become the framework of a monitoring system to retrace the machining process variables (such as, tool wear and surface roughness etc.) during a machining process. The next step is to develop a machining process classification and coding system, which combines the theoretical and experimental work and organizes the practical examples of an on-line machine monitoring into a database for future reference. As explained earlier, Artificial Neural Networks (ANN) have received increased attention as a modeling tool. By exploring the advantages of ANN in machining processes, like the fact that they are suitable for incremental learning, we developed a Neural Network monitor (as explained in chapter 3).

A Neural Network monitor performs the following functions:

- It senses and processes tool-cutting information.
- It learns by studying the signal values obtained from sharp and worn out cutting tools.
- It stores a large amount of knowledge of the machining process in its interconnections from different cutting operations and automatically gives the alarm signals from its memory.
- It reacts by sending alarm signals to the control unit of machine, informing it if the tool is worn, broken or not-in-use.

- It automatically coordinates machining and monitors commands from the control unit of the machine.
- It communicates between the operator and the machine via the control panel, informing operating personnel about cutting-tool conditions and presenting an interface for control of all functions.

5.1.3 Expert System

An on-line machining process is a decision-making process to determine the instant when a tool change is necessary so as to assure the product quality or to avoid tool breakage. It can be expected that the control strategy for this decision-making process will be knowledge intensive. The tasks to control the on-line monitoring is complex because of its physical nature. So, a high-performance computer program has to be developed to automate such a decision-making process. The expert system proposed in machining process focuses on the following aspects:

1. Combine the coding system and database together and transform the combined form into a knowledge base. The causal rules incorporated in the knowledge base in the symbolic representation can provide a fast response control.
2. Combine quality control program with on-line tool wear monitoring system.
3. Develop an inference engine to effectively find a major tool wear mode when the detected signal has been traced for a certain time.

5.2 Implementation of an On-line Monitoring System

The implementation of an on-line monitoring system proposed above is shown in Figure 5.3. We developed an instrumented transducer for signal detection system and an Artificial Neural Network monitor to measure the tool wear and surface roughness.

The advantages of using on-line monitoring to detect the condition of the tool are:

- Tool wear is monitored and tool changes initiated when necessary, avoiding damage to the machine or the workpiece.
- If there is a breakage, a signal will be produced to stop the machine tool within milliseconds.
- The system will detect if a tool or the workpiece is missing, thus eliminating wasted machine time and the likelihood of unpredictable crashes.

The cost advantages of tool monitoring are:

- Tool life can be optimized, which means that tools need only be changed when they are worn and so reduces tool costs.
- Down-time is lessened, and this increases the machine's tools output.
- Repairs to the machine tool and cutting tools may be reduced to a minimum so the maintenance costs are lower.
- The metal-cutting operation is monitored automatically, limiting operator involvement.

In the following paragraphs we will discuss some important features of designed transducer and tool wear measurement.

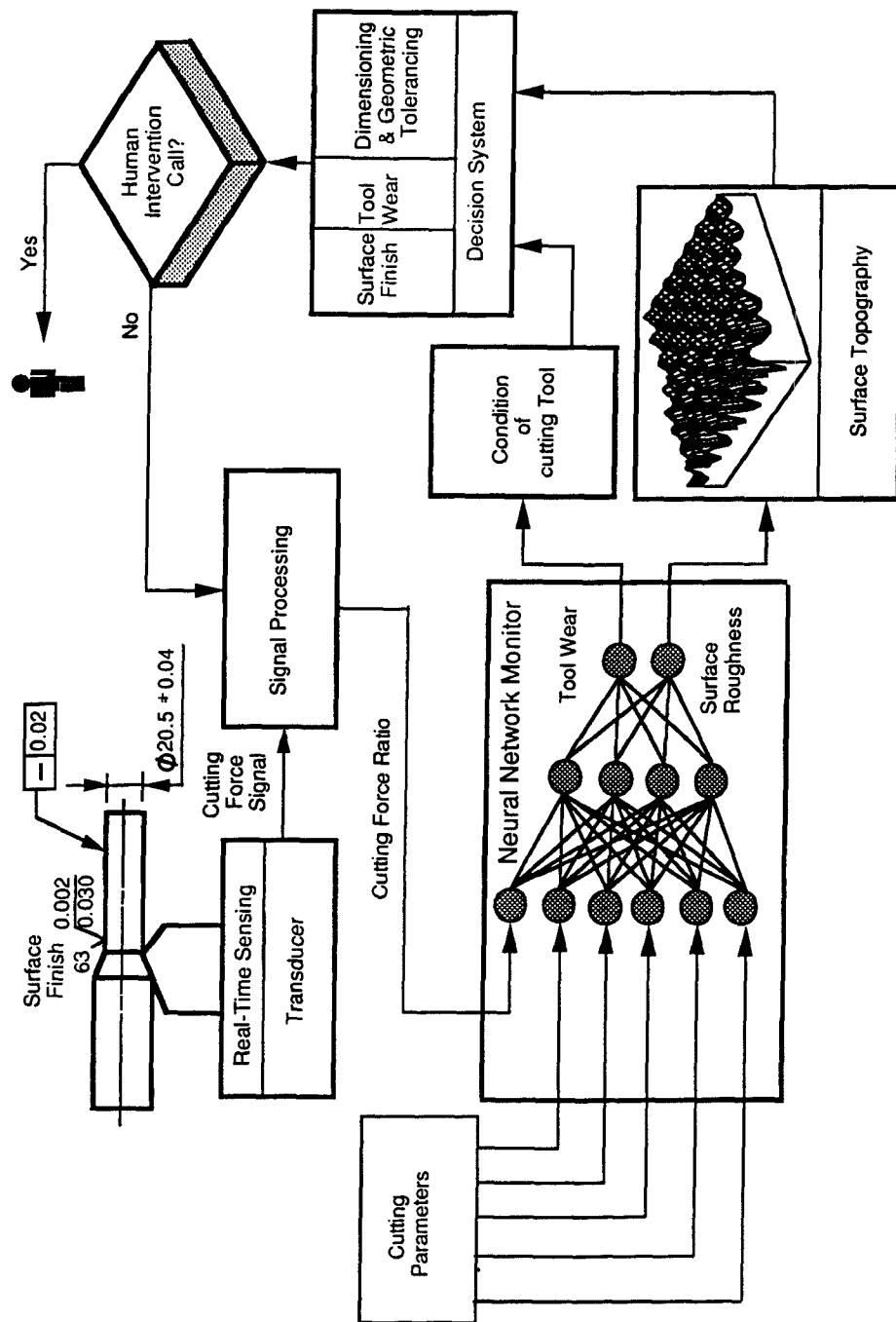


Figure 5.3: Implementation of an On-Line Monitoring in a Turning Process

5.2.1 Instrumented Transducer Vs Commercial Dynamometers

Cutting force measurement is usually done using two methods: by using a commercial dynamometer or by designing an instrumented transducer. Dynamometers with high quality are available in the market. But they have two major drawbacks:

- They are expensive and designed for laboratory work but not for on-line monitoring unless the machine tool is changed accordingly.
- Its effectiveness is mainly dependent on the user's knowledge of locating the dynamometer in the machining system to have an explicit transfer function of the signal transition process. This requires a careful study of system dynamics and the interaction between the two structures of dynamometer and the machine tool.

The second method is to design an instrumented transducer, where sensors such as strain gages are attached directly to the machining system part, such as tool holder. The main advantage of the second method is that the detected signals may truly reflect the dynamic characteristics of the machining processes. But however the accuracy depends upon the calibration and noise filter processes.

5.2.2 Tool Wear Measurement

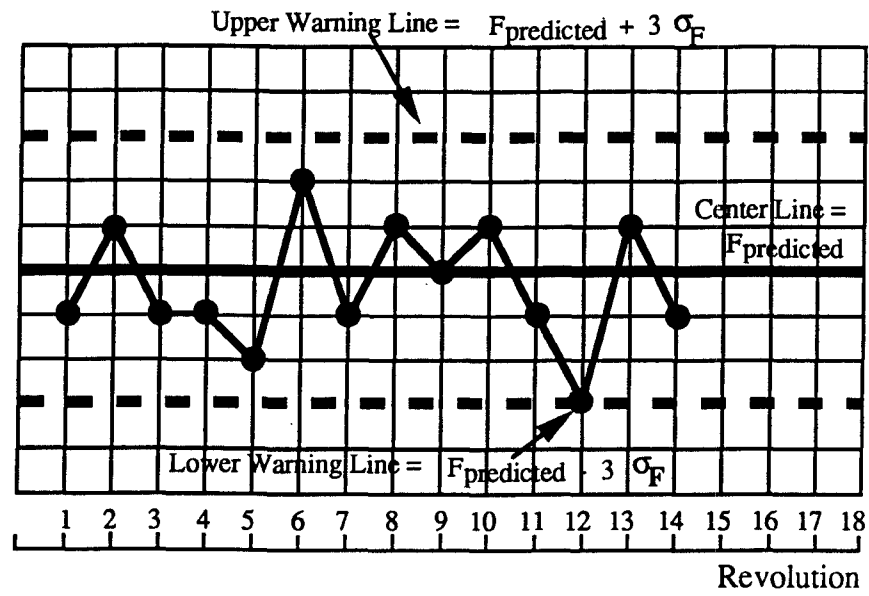
A method of identifying the correct tool to be used for a specific operation can be considered to be an essential requirement. If the wrong tool is selected, at the very least the workpiece will be scraped or, still worse, the machine tool itself may be seriously damaged. An efficient automatic tool-handling system, in which the tools may be accessed at random as required, will depend upon each tool being identified correctly at every stage of the part manufacture. Two

methods can be used to identify the condition of the tool: a direct method and an indirect method. As explained earlier, direct methods, such as an optical scanning technique, electrical resistance etc., have some practical problems. This leads to indirect measurement of tool wear. We successfully used Artificial Neural Networks in predicting the tool wear (shown in chapter 3).

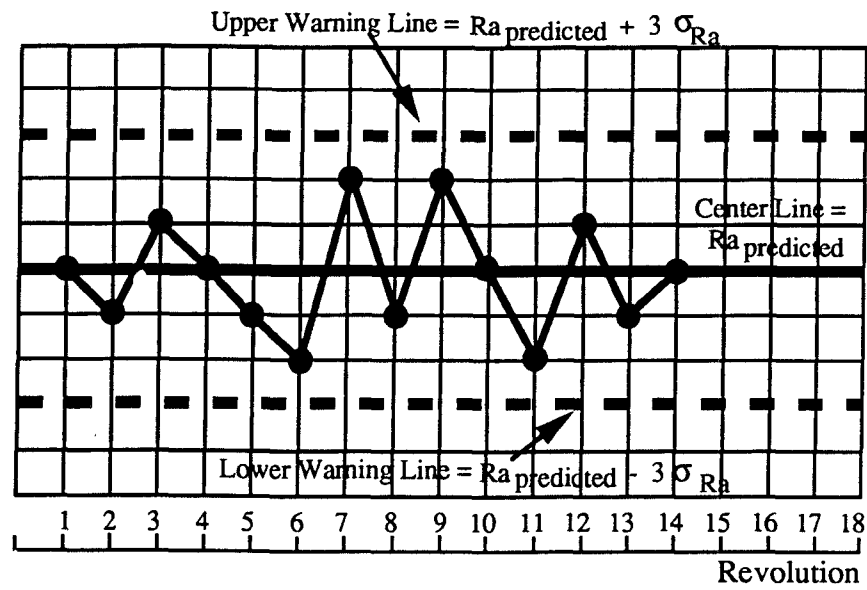
5.2.3 On-Line Monitoring of a Turning Process

The implementation of the developed monitor in a real time machining operation requires an additional piece of information, i.e., the estimate of the dynamic variation of the monitoring target(s), such as the cutting force and/or surface finish. This additional information enables the monitor to distinguish the external noise related to the machining process abnormalities from the natural, or inherent process variation. In this work, estimates of the dynamic variations of the two monitoring targets were obtained by two steps. The first step was to obtain the two estimates at each of the 8 machining tests. These estimates are presented in Figures 4.7 and 4.8. In the second step, two pooled estimates, representing the natural variation of the cutting force and surface finish during machining, were calculated from the 16 estimates. These two pooled estimates were the standard deviation of the cutting force variation (2.12 Newtons) and the standard deviation of the surface finish ($0.52 \mu\text{m}$), respectively.

To demonstrate applicability of the proposed on-line monitoring system, a prototype monitoring system using the developed neural network monitor has been implemented to monitor the machining of advanced ceramic materials. The objective of such a monitoring system is to control the appearance of micro-cracks on the machined surface, the dimensional accuracy, and the finish quality of the machined surface. An assumption is made that a large cutting force leads to a severe surface damage. Therefore, the two monitoring targets are the cutting force and the surface finish generated during machining. Figure 5.4 illustrates the two monitoring charts.



(A) Cutting Force



(B) Surface Finish

Figure 5.4: On-line Monitoring Charts of Ceramics Machining

The center lines of these two charts represent the predicted cutting force and surface finish through training. The upper and lower limits on the two charts are the warning lines of the monitoring system. They are constructed based on the 3σ principle, which indicates that the machining process is in its normal status as long as the detected cutting force signal, or the measured surface finish, is within the upper and lower limits. Whenever a detected signal is out of the two limits, the machining process goes on to an abnormal status. If the detected cutting force signal hit the upper warning line. The machining operation should stop immediately in order not to spoil the surface of the work-piece. The worn tool must be replaced by a new tool and the machining process resumed to its normal status. When the detected cutting force signal hit the lower warning line, a severe tool breakage may occur. The breakage caused the depth of cut during machining to drop to a level wherein there was almost no contact between the workpiece and the tool. However, false alarmings sometimes happen, especially during the cutting force monitoring due to the complexity of cutting mechanism, such as effects of built-up edges and nonhomogeneous distributions of workpiece material properties. As a result, the decision-making advisor built in the monitoring system will call on a human intervention only under circumstances when the two monitoring targets are hitting their warning lines. There is, however, an associated side-effect, i.e., the decrease of the sensitivity of the monitoring system for an effective detection of the process abnormalities. Further improvements are being made to balance the need between the false alarm elimination and the sensitivity of detection.

5.3 Advantages and Disadvantages of Feedforward Neural Networks

5.3.1 Advantages

The advantages of using feedforward neural nets as an on-line monitor are:

- They can be used in a wide variety of applications, ranging from classification and pattern recognition, to optimization and control.
- Unlike conventional batch algorithms, they are incremental learning algorithms, because at any stage during the training process, training can be stopped and the network would still serve as a model of function being learned, even though it may not be quite accurate.
- They can be used in developing the empirical models based on experimental and observational knowledge.
- They are best suited for fast computations on parallel architectures.
- They have good generalization capabilities.
- They can learn from experience and gives accurate results from incomplete and noisy data.
- The hidden layers perform feature extraction on the patterns presented at the input layer i.e., they find a correct fit of input space and output space.
- They do not require any a priori knowledge of a mathematical function that maps the input patterns to the output patterns. They need only examples to train the network.

5.3.2 Disadvantages

The disadvantages of using feedforward neural nets as an on-line monitor are:

- The largest drawback with feedforward back-propagation algorithm appears to be its convergence time. Training sessions can require hundreds or thousands of iterations for some problems. Realistic applications may have thousands of examples in a training set, and it may take days of computing time (or more) for complete training. Usually, this lengthy training needs to be done only during the development of the network, because most applications require a trained network and do not need on-line retraining of the net.
- Lack of proper guidelines for networks architecture (number of hidden layers and number of nodes in each layer) hinders the use of these networks fully. However, the flexibility of the network's paradigm is enhanced by the large number of design choices available: choices for the number of layers, interconnections, processing units, the learning constant, and data representations.
- The feedforward back-propagation algorithm uses a gradient search technique to minimize a cost function equal to the mean square difference between the desired and actual node outputs. The main problem while minimizing a cost function is sticking to a local minima rather than the global minima. This can be avoided by using some techniques.

5.4 Control System

When a programmer writes a CNC program, it is not possible to choose optimum values for speed, feed rate etc. because of variables such as tool wear, non-homogenous materials and variations in dimensions of rough stock. It is normal for a programmer to choose median values for these factors in any calculations, but even here problems may arise, like tolerances etc. Whatever route chosen in writing the program for a part, the programmer faces the quandary of whether to utilize high rates and sacrifice some tool-cutting predictability to increase

workpiece output, or to use normal rates and produce a dependable cutting action with lower production cycle times. The adaptive control systems eliminate the need of this guessing game.

5.5 Description of Code

Depending on the problem to be solved, the number of input and output nodes are selected. The size of the middle layer is really our choice. If the middle layer is too large, it will encourage the network to memorize the input patterns rather than generalizing the input into features. This reduces the ability of the network to handle unfamiliar inputs after the training is complete. On the other hand, a middle layer too small will drastically extend the number of iterations required to train the network and is likely to reduce the accuracy of recall. Therefore to balance the both needs we have to select an optimum number of nodes in the hidden layer. This is done by trial and error or by experience (as explained in chap. 3). Once the architecture of the network is selected, it is ready to train the data. Number of patterns are selected from a real-time data, to train the neural network. Before the training starts, values for η , a learning constant ($0 < \eta < 1$), and α , momentum term constant ($0 < \alpha < 1$), should be selected.

In this thesis work, a program has been developed using feedforward back-propagation network to predict the machining process variables, like, the tool wear, surface roughness etc. Definitions of various global variables and constants, and functions used in the program are listed below:

Global Variables and Constants

`max_num_pats` = maximum number of patterns in the
data file

`in_x_size` = number of nodes per row of input layer

`in_y_size` = number of nodes per column of input layer
`in_size` = number of nodes in input layer
 (i.e., `in_x_size * in_y_size`)
`hid_size` = total number of nodes in hidden layer
`out_size` = number of nodes in output layer
`wts_hid[hid_size][in_size]` = hidden layer weights
 (weights between input and hidden layer nodes)
`wts_out[out_size][hid_size]` = output layer weights
 (weights between hidden and output layer nodes)
`hid_out[max_num_pats][hid_size]` = output of hidden layer nodes
`out_out[max_num_pats][out_size]` = output of output layer nodes i.e.,
 network's output
`hid_error[hid_size]` = error of hidden layer nodes
`out_error[out_size]` = error of output layer nodes i.e.,
 network's output errors
`out1_error[max_num_pats]` = square of error of an output layer node
`hid_wt_change[hid_size][in_size]` = storage for last weight change for a
 hidden node
`out_wt_change[out_size][hid_size]` = storage for last weight change for an
 output node
`thre_hid[hid_size]` = threshold values for a hidden node
`thre_out[out_size]` = threshold values for an output node
`hid_thre_change[hid_size]` = storage for last threshold change for a
 hidden node
`out_thre_change[out_size]` = storage for last threshold change for an
 output node

`in_pats[max_num_pats][in_size]` = normalized value of an input variable
`out_pats[max_num_pats][out_size]` = normalized value of an output variable
`total_out_error[max_num_pats]` = half of the sum of the square of errors
of an output node, over all output nodes
in output layer
`total_out_error[max_num_pats]` = sum of errors of all nodes in an output
layer
`grand_error` = LMS error for all patterns, which is
the measure of whether the net
is done or not
`iterations` = number of times the network seen the
training data
`numpats` = total number of patterns (or examples)
in the data file

Functions

`forward_pass` : propagates input activity forward through network
`out_forward` : process forward pass for an output layer i.e.,
determines the network's output
`hid_forward` :process forward pass for hidden layer i.e.,
determines the hidden layer nodes output
`back_error_prop` : propagates the error backward through network
`error_network` : computes the error for output nodes i.e.,
network's error
`out_wt_adjust` :adjust the weights between hidden layer nodes and
output layer nodes

`hid_wt_adjust` : adjust the weights between input layer nodes and
 hidden layer nodes
`check_out_grand_error` : this function checks whether the net is done or not
`net_initialize` : initializes the network
`wt_randomize` : randomizes the weights on hidden and output layers
`read_data` : reads the data from a file i.e., training data
 (or testing data)
`print_hid_wts` : prints the hidden layer's weights
`print_out_wts` : prints the output layer's weights
`main` : main control program

The structure of the developed program is explained below. There are four stages in the network:

- Network initialization
- Forward pass
- Back error propagation
- Termination of training and Validating the network

Each stage will be discussed briefly in the following paragraphs.

Figure 5.5 outlines the four stages and presents the basic working principle.

Network Initialization

Initialization of the network is done by invoking the function *net_initialize*. This function in turn invokes two other sub-functions, namely *wt_randomize* and *read_data*.

wt_randomize : It expects a random number seed from the programmer. It randomizes the weights of hidden and output layers between -1 and +1. It also randomizes the threshold values for nodes in hidden and output layers between -1 and +1.

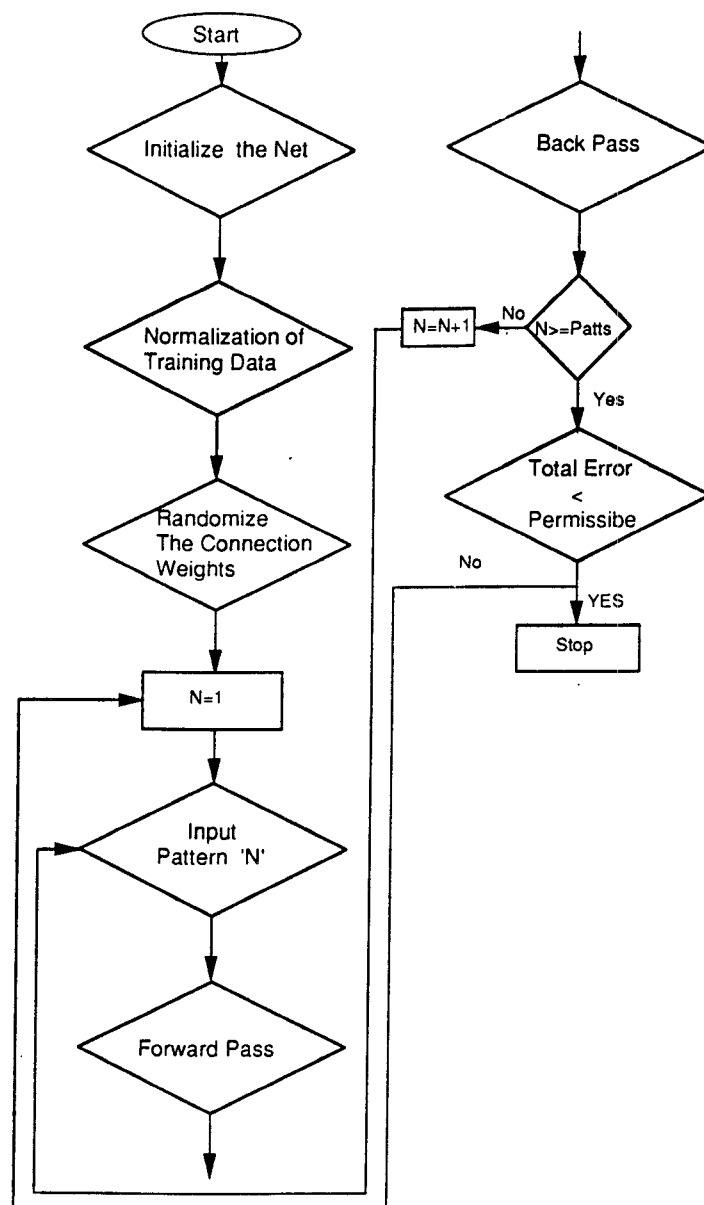


Figure 5.5: Basic Working Principle of a BPN Network

read_data : It reads the data from a file. The format of the data file is as follows:

line#1: *in_x_size*, *in_y_size* and *out_size*

line#2: total number of patterns

line#3: maximum and minimum values of the first variable in the data #1

line#4: line#n: maximum and minimum values of the last(nth) variable in the data #1

line#n+1: values of input variables

line#n+2: values of output variables

.. and so on for all patterns in the data file

After reading the values from the file, it normalizes each value between 0 and 1 (as explained in chap. 3) and stores the value in *in_pats* and *out_pats*. If it finds any fault with the data file, it sends an error message. After initialization is done, the network is ready for the forward pass during the training.

Forward Pass

It invokes two sub-functions, namely *hid_forward* and *out_forward*. We now present the first pattern to the network, after the weights on the interconnections are randomized (in network initialization). The input layer receives the pattern and thus passes it along to each node in the hidden layer, which is done by *hid_forward* function. Each of these nodes computes an activation in the following fashion.

First the summed input, X_j , is determined by multiplying each input signal by the weight (randomized initially) on the interconnections:

$$X_j = F_j(\sum W_{ij} * I_i)$$

for the j^{th} node in hidden layer from the i^{th} node in input layer and the sum is taken over i = number of nodes in the input layer. The terms W and I are the

weights and input signals, respectively. This sum is added to the thresholds (which are randomized initially) and used as an input to the activation function of the node, i.e., sigmoidal function. It determines the activity, or excitation level, generated in the node as a result of an input signal of a particular size. The sigmoid function is given as:

$$F(X) = 1/(1 + e^{-(X+T)})$$

where,

T = threshold of a node.

In this way we compute the total activation of the hidden layer nodes, and simply use this activation as the output of each node in the hidden layer. This ensures that our outputs are constrained to be in the range of 0 and 1. Following through the network, these outputs from the hidden layer nodes are treated as inputs to the output layer. Each node here receives the signals from all hidden layer nodes and, in exactly the same fashion as before, an activation is computed for each output layer node. This is done by the *out_forward* function. These activations become the output for the network as the nodes generate their output signals.

Back Error Propagation

Back error propagation is done by calling the *back_error_prop* function. This function in turn calls its four sub-functions, namely, *error_network*, *error_hid*, *out_wts_adjust*, and *hid_wts_adjust*.

At this stage we compare the output of the network to our desired output or to the target output (in *error_network* function). It is very likely that the pattern is wrong in its initial stage, so we need to adjust the weights to complete this iteration of the network. Weight changes for the output layer nodes is done by calling the *out_wts_adjust* function using the generalized delta rule.

Figure 5.6, shows the training process (weight changes) for a single weight

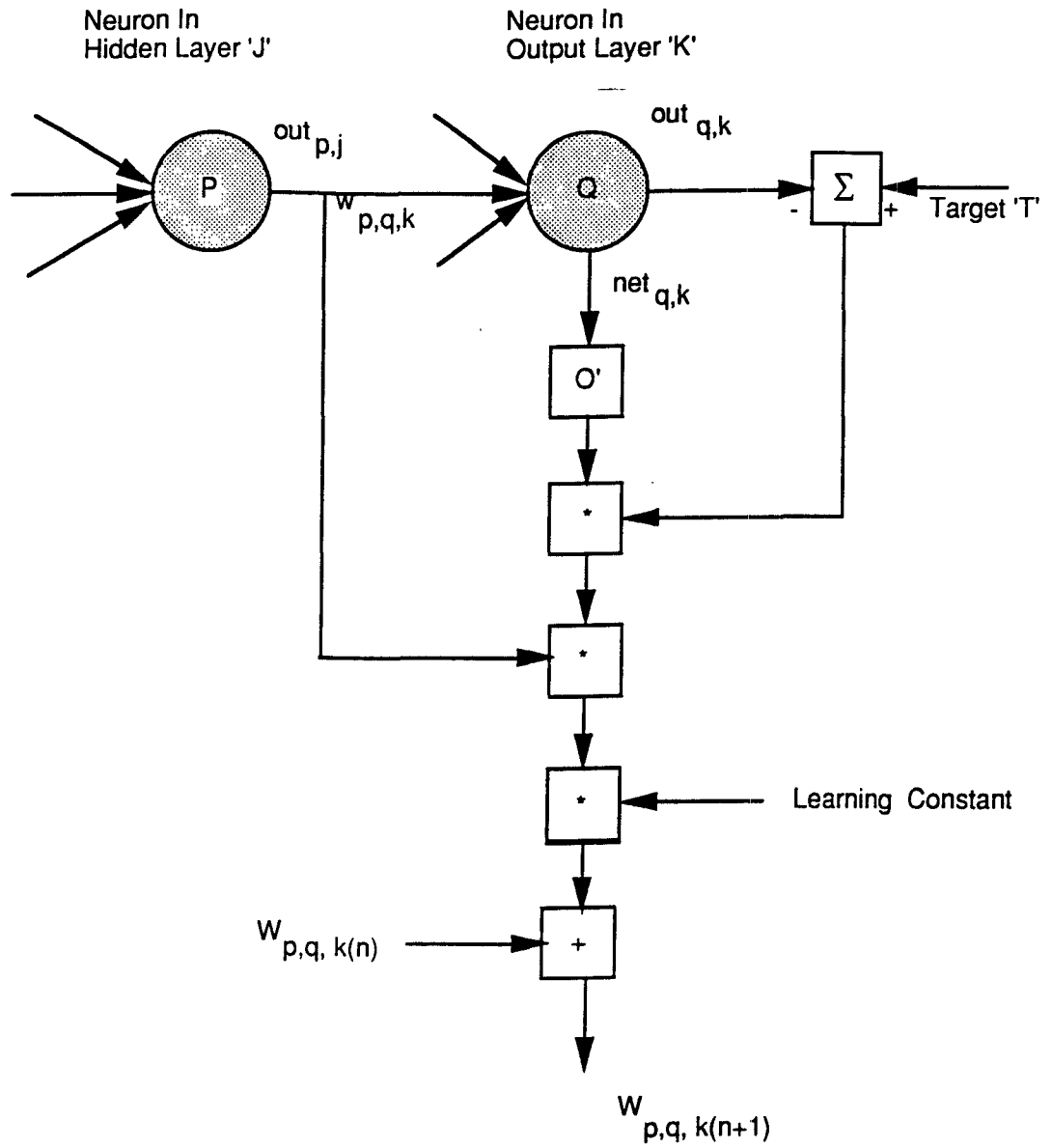


Figure 5.6: Basic Principle of Weight Changes in Output Layer

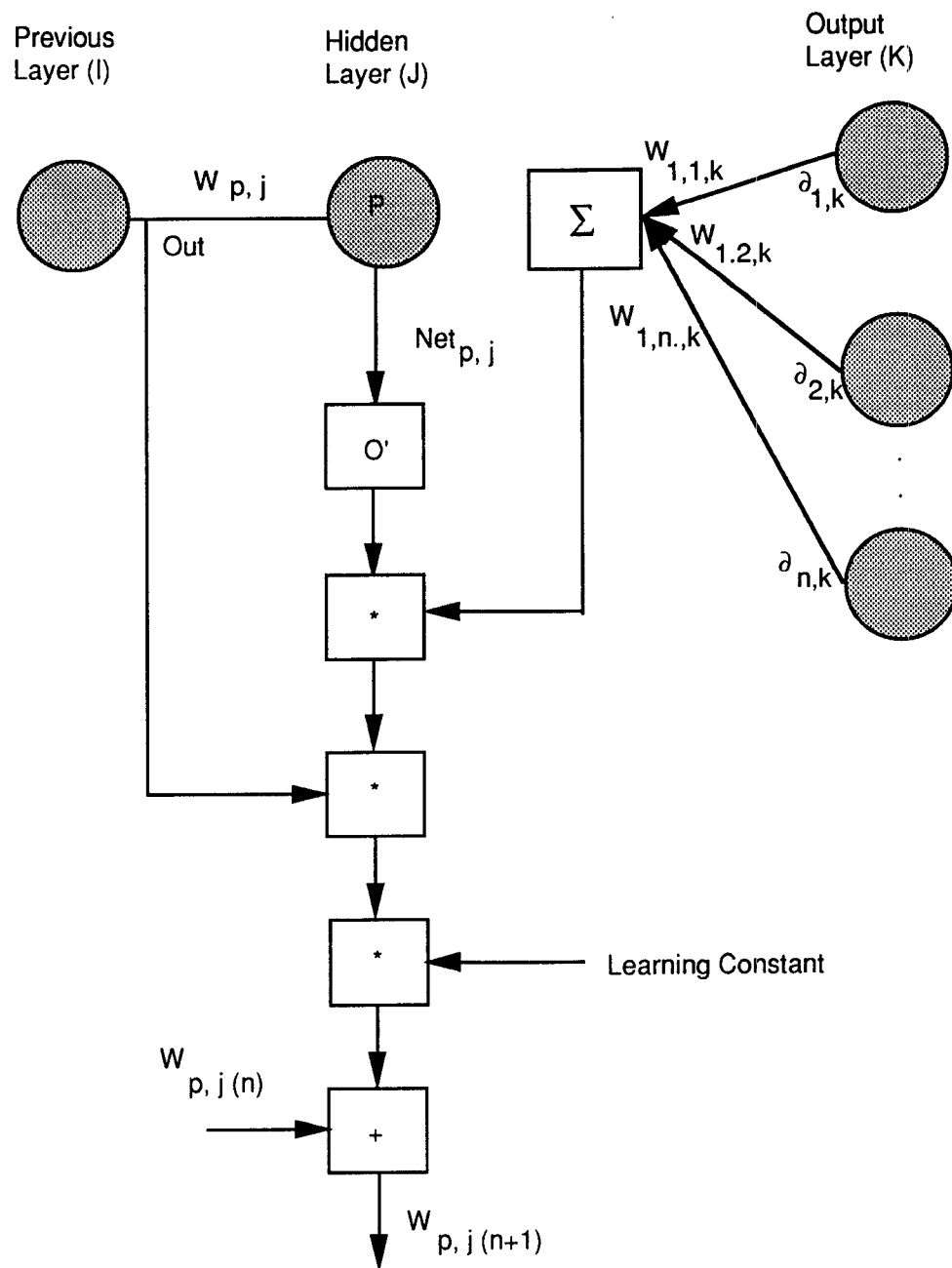


Figure 5.7: Basic Principle of Weight Changes in Hidden Layer

from node 'p' in the hidden layer 'j' to node 'q' in the output layer 'k'. The output of a node in layer 'k' is subtracted from its target value to produce an error signal. This is then multiplied by the derivative of the squashing function (sigmoid function) calculated for node 'k' of that layer, thereby producing the delta value:

$$\text{delta} = \text{out_out}[\text{patt}][\text{node}] * (1 - \text{out_out}[\text{patt}][\text{node}] * \text{out_error}[\text{node}])$$

Therefore,

$$\begin{aligned} \text{wts_out}[\text{midnode}][\text{outnode}](n + 1) &= \text{wts_out}[\text{midnode}][\text{outnode}](n) \\ &+ \eta * \text{delta} * \text{hid_out}[\text{midnode}] \\ &+ (\alpha * \text{out_wt_change}[\text{outnode}][\text{midnode}]) \end{aligned}$$

mid node's threshold change is:

$$\begin{aligned} \text{ch_thre} &= \eta * \text{out_out}[\text{patt}][\text{outnode}] \\ &* ((1 - \text{out_out}[\text{patt}][\text{outnode}]) \\ &* \text{out_error}[\text{outnode}]) \\ &+ (\alpha * \text{out_thre_change}[\text{outnode}]) \end{aligned}$$

New threshold value is:

$$\text{thre_out}[\text{outnode}](n + 1) = \text{thre_out}[\text{outnode}](n) + \text{ch_thre}$$

Considering the middle layer, we can apply the delta rule only if we know the input signals to each node and the resulting output. But we don't know the desired output. We can find the hidden node error by the following method.

Logically, we can see that if the output nodes generated the wrong answer, it can be due to their own incorrect weights. But it can also be due to middle layers nodes that generate the wrong input signals to the output layer. To assign this blame we backpropagate the errors for each of the output-layer nodes to the middle layer using the same interconnections and weights as the middle layer used to transmit outputs to the output layer. We then compute the error for each node in the middle layer based on their portion of the blame for the error

of the output layer. This is computed as:

$$e_i = f'(X) * [\sum W_{ij} * E_j]$$

where,

e_i = error in the i^{th} middle-layer node and the sum is taken over j

j = j^{th} output layer node

The remaining term is the derivative of the activation function of the middle-layer node for the net input it receives. Application of this derivative serves two purposes. First, it contributes to the stability of the network and it ensures that, as the output approaches 0 and 1, only very small changes can occur. Second, it helps to compensate for excessive blame attached to the middle-layer node. When the connection between a middle layer node and an output layer node is very strong (high values for the weight interconnection) and the output layer node has a very large error, the weights of the middle layer node may be assigned a very large error too, even if that node had a very small output and thus could not have contributed much to the error of the output node. By applying the derivative of the signal function, this error is moderated, and only small to moderate changes are made to the weight's of the middle-layer node. So we have propagated the error from the output layer back to the middle layer and computed for each of the nodes of the middle layer. We then use this error in the generalized delta rule and adjust the weights as before. Figure 5.7 shows how the hidden (or middle) layer weights are changed. The weights of the hidden layer nodes are changed as follows:

$$\text{delta} = \text{hid_out}[\text{midnode}] * (1 - \text{hid_out}[\text{node}] * \text{sum})$$

where,

sum= sum of product of output errors and output weight interconnections.

Therefore,

$$\begin{aligned}
wts_hid[insize][midnode](n+1) &= wts_hid[insize][midnode](n) \\
&+ \eta * delta * in_pats[patt][midnode] \\
&+ (\alpha * hid_wt_change[midnode][outnode])
\end{aligned}$$

mid node's threshold change is:

$$\begin{aligned}
ch_thre &= \eta * error_hid[midnode] \\
&+ (\alpha * out_thre_change[outnode])
\end{aligned}$$

New threshold value is:

$$thre_hid[midnode](n+1) = thre_hid[midnode](n) + ch_thre$$

Termination and Validation

After adjusting the weights of hidden and output layer nodes, we can find the error for that pattern. Similarly we can find the error for all other patterns. After showing all patterns to the network, the error for each pattern is identified and it is then squared. Sum of half of the squared errors for all the patterns is stored in the `grand_error`. Then this `grand_error` is compared with the permissible error. The network will stop from training if the `grand_error` is within the permissible error, otherwise all patterns are presented to the network until the `grand_error` is less than permissible error. Also the network can be terminated by supervising the number of iterations i.e., if the number of iterations reaches a predetermined value, the network will be stopped from training. Once the training is done, the connection weights and node thresholds are frozen and functions `print_hid_wts` and `print_out_wts` are called to print the weights and the thresholds to output file. Then the testing data (which is not shown to the network during the training) is used to validate the trained network.

5.6 Comparision With Other's Work

Several people showed interest in applying neural networks in the area of machining. Rangwala used neural networks to integrate information from different sensors in order to know the status of the tool [RaDo 87]. He assumed that magnitude of the cutting force is inadequate for tool wear detection because its magnitude is also dependent on the cutting velocity and also the flank wear tends to increase the cutting force, the accompanying crater wear tends to reduce it. In order to eliminate this effect he assumed that the status of the tool can be enhanced by complementing the acoustic emission information with the cutting force information. But in this thesis cutting force ratio (feed force/tangential force) is used instead of magnitude of cutting force (the explanation for using this ratio is given in chapter 3). So in this thesis work we assumed and showed that the tool can be monitored successfully by monitoring the cutting force signals alone. This eliminates the use of multiple sensors, which indirectly reduces the amount of noise to the network and it is economical. Also Rangwala did not discussed in his paper how we can monitor a machining process on-line, which is one of the major emphasis of this thesis. Surface roughness of the machined work piece is monitored simultaneously with the tool wear in this thesis work, which is very much useful to reduce the false alarms during machining. Chryssolouris, in his work compared different modeling techniques like multiple regression analysis, group method of data handling (GMDH), neural networks [ChGu 88]. He proved that neural networks are better than other two.

One of the major contributions of this thesis was application of neural networks in machining of advanced ceramic materials, which is gaining importance in modern industries because ceramics have good wear and abrasion resistance, hardness and light in weight. Applying neural networks in this area is first of its kind. We successfully monitored the cutting force and surface roughness.

5.7 Limitations and Generalization of the Neural Network Monitor

- The performance of the developed neural network monitor depends on the quality of the experimental data used for training. The architecture of the network i.e., number of hidden layers and number of hidden nodes in each layer, can be different from the one used in this thesis if one uses another set of training data for developing the monitor. This is because of no proper guidelines for determining the network architecture and also due to variations in the quality of the training data.
- A further concern is the reliability of the trained network. Neural networks must be tested for performance after training is complete. In order to use this trained network to monitor the machining process, one should use appropriate data and experiments for this testing. Testing data should reflect the data to be encountered while the neural network is in use, and testing experiments should use as broad a spectrum of inputs as possible, including inputs that may result from unlikely circumstances, like noisy signals from sensors and variations in material properties.
- The developed monitor would be machine dependent and it requires whole database of the machining process in order to monitor the machining process correctly. As mechanism of removing material differs from work-piece to work-piece, e.g., machining of steel and machining of ceramics, one has to build different monitors for each kind.

5.8 Conclusions

A framework of an on-line monitoring system is proposed. The basic requirements of an on-line monitoring system were discussed. The advantages of on-line

monitoring system were analyzed. It is found that using an instrumented transducer for measuring the cutting forces is more advantages than using a commercial dynamometer. Advantages and disadvantages of neural networks were also discussed. The developed neural network program was successfully utilized to predict tool wear and surface roughness. Limitations and generalization of the developed network monitor are discussed and comparisons between this work with others work are also made.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

The conclusions made from this thesis are as follows:

1. The advantages of artificial neural networks were explored. A structure for an intelligent monitoring system for a machining process was proposed and evaluated. The proposed system utilized a force sensor as the detector of the machining process and a neural network as the learning and decision-making component. Through appropriate selection of architecture of the network, noise can be minimized and the process variables (like tool wear, surface roughness etc.) can be predicted accurately. In this thesis work, application of neural networks for on-line monitoring, learning, and pattern recognition were demonstrated. The results presented here show that these networks possess the ability for learning and noise suppression. As a result, a neural network enables a high success rate (above 90%) for on-line monitoring of a machining process under a range of process conditions.

2. We explored two types of Artificial Neural Networks: Adaline Network, and Feedforward Back-Propagation Network. The effect of cutting parameters on cutting force is modeled using an adaline network. In this thesis work, we concluded from the results that the depth of cut has the highest effect on the cutting force in machining of metals. The success rate of predicting the cutting force is over 90% under different cutting parameters.
3. We successfully monitored tool wear and surface roughness, using a feed-forward back-propagation network. Because the feed force increases more rapidly than tangential cutting force, the resulting force direction in the plane will change. For that reason we used force ratio (feed force to tangential force) instead of using total cutting force directly. The cutting force ratio is also a good indicator of tool wear regime. The success rate of monitoring the tool wear and surface roughness was above 92%.
4. Use of a commercial dynamometer is not a solution for force measurement on the shop floor. In this thesis work, a practical force sensor was assembled into the commercial machine. This force transducer was successfully utilized in measuring the cutting forces in real-time machining.
5. Tool wear information is indispensable in untended machining. An optical microscope was used to measure the tool wear during the machining tests. Taylor's tool-life equations under several machining conditions were derived from the measured tool wear.
6. We applied neural networks in the machining of advanced ceramics. A feed-forward back-propagation network was successfully implemented to predict the effect of cutting parameters on tangential cutting force and surface roughness. We found from experiments that unlike the case of machining of metals, the feed force is more than the tangential force. This is mainly due to the high hardness of ceramic materials leading to a cutting mechanics dominated by fracture instead of deformation. From results, it can be

seen that the depth of cut has the least effect and speed has the greatest effect on tangential cutting force and surface roughness. We found that an efficient way of machining, which reduces the tangential cutting force, is using low feed and low cutting speed. A low feed and high cutting speed is used for better surface finish. We used a feed rate of 0.2 mm/rev with cutting speeds of 78.5 ft/min and 118 ft/min. The success rate of monitoring tangential force and surface roughness is over 95%.

7. In a feedforward back-propagation network, it is shown that the error decreases with the number of hidden nodes. It is stated earlier that the nodes in the hidden layer learn to respond to features found in the input. If we use a large number of nodes in the hidden layer, the error can be decreased, but the network memorizes the input patterns rather than generalizing the input. For this reason, we used four hidden nodes in the hidden layer.

6.2 Future Work and Recommendations

The following recommendations are proposed for future work:

1. In this work we proposed an on-line monitoring system for a turning process. In future one can explore artificial neural networks in other machining processes such as milling, grinding, etc. Care should be taken in the application of neural networks to milling and grinding operations, as they involve a multi tooth cutting process. Consequently, the cutting force generation is more complex compared to a single cutting process. Furthermore, the generation of surface topography will require the knowledge of each cutting edge and the sequence of positions that each cutting edge will traverse. It is evident that the trajectory made by the cutting tool is complex, because it involves overlapping between each cutting edge's path.

2. The on-line model (especially the tool wear model) can be implemented in a Computer Aided Surface Topography Simulator (CASTS), which predicts the surface texture in order to know the effect of tool wear on surface roughness. However, it is important to consider effects of random tool motion on the surface texture formation. It has been well known that built-up edge phenomenon, tool wear, and properties of workpiece materials such as, hardness and ductility are factors which may introduce random excitation during machining.
3. Our next goal is hardware implementation of neural networks in on-line monitoring of tool wear. The research focus should be on the development of a parallel system to perform on-line monitoring of the surface topography generation, which may represent a breakthrough for real-time control of machining processes at the factory level. In this regard, one of the promising targets would be to implement an application-specific Very Large Scale Integration (VLSI) system. The building of a special purposed system using a VLSI array processor approach will lead to the dream of real-time control of machining processes to come true.

Bibliography

- [Ar 83] Arsovski, S. M, "Wear sensors in the adaptive control systems of machine tools", *Int. J of Prod Research*, 21 (3), 1983.
- [Ar 89] K. F. Arnold, "Measurement of cutting forces as they relate to a calibrated Flank Wear", *M. S. Thesis*, Dept. of Mechanical Engg., University of Missouri-Columbia, MO, 1989.
- [BaMa 85] Balakrishnan, P., and Macbain, J. C., "Monitoring Systems For Unmanned Machining", em SME technical paper MR 85-479, 1985.
- [Br 77] Brecker, J., "A Capacitive-Based Surface Texture Measuring System", *Annals of the CIRP*, Vol. 25, No. 1, pp. 375-377, 1977.
- [ChGu 88] G. Chryssolouris, and M. Guillot, "An A.I. Approach to the Selection of Process Parameters in Intelligent Machining", *Sensors and Controls for Manufacturing*, PED-vol. 33, ASME Winter Annual Meeting, Chicago, Nov. 27 - Dec. 2, 1988.
- [ChLe 90] G. Chryssolouris, M. Lee, J. Pierce and m. Domroese, "Use of neural networks for the design of manufacturing systems", *Manufacturing Review*, Vol. 3, No. 3, September 1990.
- [ClMa 90] Clement Fortin and Marek Balazinski et. al, "Study of the influence of Feed Variation on Tool Wear", *Fundamental Issues In Machining*, PED-Vol. 43, 1990.

- [DaUl 87] Danai, K., and Ulsoy, A.G. 1987, "An Adaptive Observer for On-Line Tool Wear Estimation in Turning, Part 1-theory, Part 2-Results",
- [Filp 69] De Filippi, A., and Ippolito, R., 1969, " Adaptive Control in Turning: Cutting Forces and Tool Wear Relationships for P10, P20, p30 Carbides", CIRP Annals, Vol. 17 pp. 377-379.
- [GeVe 91] George Chryssolouris, Velusamy Subramaniam and Michael Doroese, "A Game theory Approach to the operation of Machining Processes".
- [Je 84] Jetly, S. "Measuring tool wear on-line: some practical considerations", *Mfg Engg*, July 1984.
- [KeTi 91] Kevin G. Coleman, Timothy J. Graettinger and William F. Lawrence, "Neural Networks for Bankruptcy Prediction: The power to solve financial problems", *AI Review*, July - August 1991.
- [Ko 78] Koren, Y., "Flank Wear Model of cutting tools using Control Theory, *ASME J. of Engg. for Industry*, Vol. 100, N0.1, February 1978, pp. 103-109.
- [Kr 66] M. Kronenberg, *Machining Science and Application*, Pergamon Press Inc., 1966.
- [LiKo 90] T. I. Liu and E. J. Ko, "On-line Classification of Drill Wear via A. I. Techniques".
- [McRu 86] J. L. McClelland, D. Rumelhart et al., "*Parallel Distributed Processing*", The MIT Press.
- [MeEr 53] Merchant, M. E., Ernst, H. and Krabacher, E. J, "Radioactive cutting tools for rapid tool-life testing", *Trans. of ASME*, May 1953.

- [MiDe 67] Micheletti, G. F., De Filippi., and Ippolito, R., "*Proceedings of the CIRP International Conference on Manufacturing Technology*", Ann Harbor, Michigan, Sep. 1967, pp. 513-524.
- Mechanical Systems and Signal Processing*, Vol. N0 2, pp. 211-240
- [RaDo 87] Rangwala, S., and Dornfeld D., "Integration of Sensors via Neural Networks for Detection of Tool Wear States, "*Intelligent and Integrated Manufacturing Analysis and Synthesis*, ASME Winter Annual Meeting, Boston, MA, Dec. 12-13, 1987.
- [Ri 87] Richard Lippmann, "An Introduction to Computing with Neural Nets", *IEEE ASSP Magazine*, April 1987.
- [Ru 76] Rubenstein, C., "An Analytical of Tool Life Based on Flank Face Wear," *ASME J. of Engg. for Industry*, Vol. 98, February 1976, pp. 221-226.
- [SaCh 89] A. Sajjanwala, S. K. Choudhury and V. K. Jain, "On-line tool wear sensing and compensation during turning operation", 1989.
- [SeRo 87] Sejnowski, T. and C. Rosenberg. 1987. "Parallel networks that learn to pronounce English text", *Complex Systems* 1(1): 145-68.
- [ShRa 90] T. Shi and S. Ramalingam, "Real-Time Flank Wear Sensing", *Fundamental Issues In Machining*, PED-Vol. 43, 1990.
- [SuWe 85] Suzuki, H. and Weinmann, K. J. "An on-line tool wear sensor for straight turning operations", *J of Engg for Ind, Trans of ASME*, 107, November 1985.
- [SuSt 91] Sudhakar Yerramareddy, Stephen C-Y. Lu, "Developing Empirical Models from Observational Data Using Artificial Neural Networks", Accepted for publication, *Journal of Intelligent Manufacturing Systems*.

- [Ta 06] Taylor, F. W., "On the Art of the Cutting Metals", *S Trans. J of the ASME*, Vol. 28, 1906, pp. 204.
- [UsSh 84] Usui, E., Shirakashi, T., and Kitagawa, T., "Analytical prediction of Three dimensional Cutting Process-part3: Cutting Temperature and Crater Wear of Carbide Tool, *ASME J. of Engg. for Industry*", Vol. 100, May 1978, pp. 236-243.
- [YeWr 83] Yen, D. W., and Wright P. K., "Adaptive Control in Machining - A new approach based on the physical constraints of Tool Wear Mechanisms", *ASME J. of Engineering for Industry*, Vol. 105, No. 1, Feb. 1983, pp. 31-38.
- [Za 86] G. M. Zhang, "Dynamic Modeling and Dynamic Analysis of the Boring Machining System, " Ph.D. thesis, University of Illinois at Urbana-Champaign, Jan. 1986.
- [ZaYe 91] G. M. Zhang, S. Yerramareddy, S. M. Lee and S. C-Y. Lu, "Simulation of Surface Topography Formed During the Intermittent Turning Process," *Journal of Dynamic Systems, Measurement, and Control*, Transactions of ASME, June, 1991, Vol. 113, pp. 273-279.
- [ZaKa 91A] G. M. Zhang, S. G. Kapoor, "Dynamic Generation of the Machined Surface, Part 1: Mathematical Description of the Random Excitation System," *Journal of Engineering for Industry*, Transactions of ASME, May 1991, Vol. 113, pp. 137-144.
- [ZaKa 91B] G. M. Zhang, S. G. Kapoor, "Dynamic Generation of the Machined Surface, Part 2: Mathematical Description of the Tool Vibratory Motion and Construction of Surface Topography," *Journal of Engineering for Industry*, Transactions of ASME, May 1991, Vol. 113, pp. 145-153.

- [ZaHw 90A] G. M. Zhang, T. W. Hwang, "Analysis of the Cutting Dynamics in Microscale, "Symposium on Fundamental Issues in Machining, 1990 ASME Winter Annual Meeting, PED-Vol. 43, pp. 25-37.
- [ZaHw 90B] G. M. Zhang, T. W. Hwang, "Analysis and Control of Geometric Tolerancing through Surface Topography Generation," Symposium on Automation of Manufacturing Processes, 1990 ASME Annual Meeting, DSC-Vol. 22, pp. 31-38.
- [ZaHa 90] G. M. Zhang, T. W. Hwang, and G. Harhalakis, "Control of Surface Topographies Formed During Machining, " Proceedings of the Second International Conference on Computer Integrated Manufacturing, pp. 339-345, Troy, NY, May 1990.

Appendix A

The Back-Propagation Training Algorithm

The back-propagation training algorithm is an iterative gradient algorithm designed to minimize the mean square error between the actual output of a multilayer feed-forward perceptron and the desired output. It requires continuous differential non-linearities. The following assumes a sigmoid logistic non-linearity is used where the function

$$f(\alpha) = 1/(1 + \exp^{-(\alpha-\theta)})$$

- 1. Initialize Weights and Offsets.**

Set all weights and node offsets to small random values.

- 2. Present Input and Desired Outputs**

Present a continuous valued input vector x_0, x_1, \dots, x_{N-1} and specify the desired outputs d_0, d_1, \dots, d_{M-1} . If the net is used as a classifier, then all the desired outputs are typically set to zero, except for that corresponding to the class the input is from. That desired output is 1. The input could be new on each trial or samples from a training set could be presented cyclically until weights stabilize.

- 3. Calculate Actual Outputs**

Use the sigmoid non-linearity from above and calculate outputs y_0, y_1, \dots, y_{M-1} .

- 4. Adapt Weights**

Use a recursive algorithm starting at the output nodes and working back to the first hidden layer. Adjust weights by

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i$$

In this equation $w_{ij}(t)$ is the weight from hidden node i or from an input node j at time t , x_i is either the output of node i or is an input, η is gain term, and δ_j is an error from node j . If node j is an output node, then

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

where d_j is the desired output of node j and y_j is the actual output. If node j is an internal hidden node then

$$\delta_i = y_i(1 - y_i) \sum_0^k \delta_k W_{ik}$$

where k is over all nodes in the layers above node i . Internal node thresholds are adapted in a similar manner by assuming they are connection weights on links from auxiliary constant-valued inputs. Convergence is sometimes faster if a momentum term is added and weight changes are smoothed by

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i' + \alpha (W_{ij}(t) - W_{ij}(t-1))$$

where $0 < \alpha < 1$

5. Repeat by Going to Step 2

