REMOTE MONITORING AND DIAGNOSIS FOR CONTROL OF EDM PARAMATERS

By

Charl Cilliers

A dissertation submitted to the faculty of Engineering in the partial fulfillment of the requirements for the degree of

MAGISTER INGENERIAE IN MECHANICAL AND MANUFACTURING ENGINEERING

at the

Rand Afrikaans University

Supervisor: Prof Z. Katz

June 2002

Abstract

This thesis presents the concept of remote (Internet) data acquisition and processing for control of EDM parameters. It incorporates aspects of data analysis while performing diagnosis of process behavior. A working neuro-macro model of the EDM process is utilized for process correlation and diagnosis recommendations.

The approach allows selective ways to vary the EDM process performance. The focus in this work is on the process efficiency. The process efficiency is related to the input parameters through the neuro-macro model of the EDM process. A comprehensive description of on-line monitoring and processing of data is included. Experimental results based on the suggested analysis and diagnosis for possible process adaptation particularly with respect to the EDM process efficiency is presented.

List of figures

Table of contents Page

Chapter 1

Introduction

The EDM machining process behavior is hard to predict and optimize. There are many dependant and independent variables [1] that have an influence on the machining process, making it complex and difficult to model. The optimization of machining processes holds a great potential to reduce the cost of production.

Attempts to model the process on a microscopic scale [2] have contributed to the knowledge of the process on a theoretical level. Practical variations in the machining environment make it necessary to adapt the process model for each specific machine. Fuzzy control systems [3] and adaptive systems [4] have been developed and have proved to be effective in improving the machining process.

The remote (Internet based) monitoring [5] of EDM holds potential for data exchange in a centralized controlled manufacturing environment. Integrating such environments can lead to standardization of machine control and improve the database of machine conditions, relevant to a specific machine.

From a comprehensive database of machining conditions, it is possible to train a neural network [6][7], giving the system experience on simultaneously running EDM machines in different environments.

This work attempts to suggest a neuro macro EDM process model based upon remote (Internet) data analysis, which is correlated to on-line process parameter diagnosis.

1.1 Research Objectives and Approach

In essence, all machining processes are controlled by a set of input parameters. Machining parameters controls the fundamental principles involved, and any variation in the machining input parameters can influence the process outputs. In particular when dealing with a very time consuming process, the output parameters can change significantly. In particular, machining time, machining quality, power consumption and ultimately, cost of manufacturing [8].

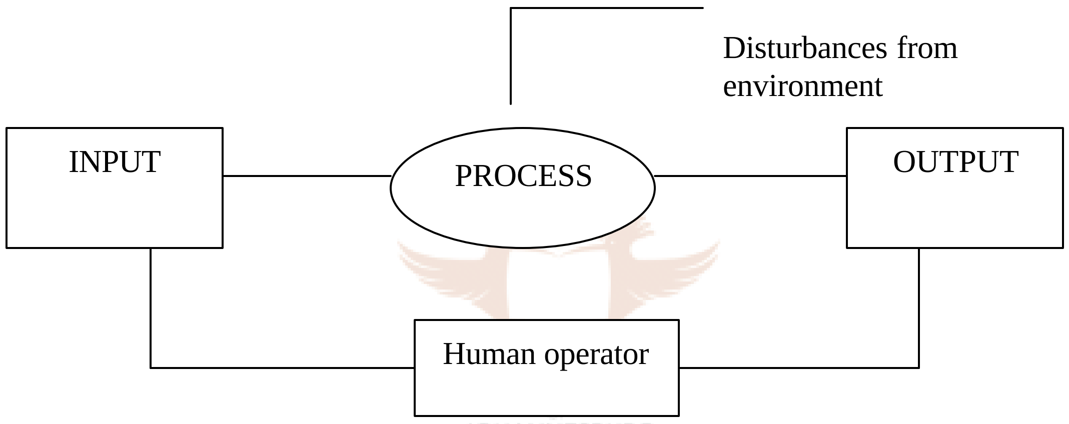


Figure 1.1 Machine Input – output relationship[8]

The EDM machining process is prone to very relevant traditional machining obstacles. The following are machining features that have significant relevant importance in the EDM process.

- Machining time
- Surface finish
- Machining geometry
- Material characteristics
- Machining Efficiency

These variables are found in most metal removal processes. These features have influence on productivity, product quality, process adaptability and ultimately profitability.

The objectives of this research are

- To identify the main process characteristics
- To develop an approach for improvement of process control
- Create a macroscopic model for the EDM process.
- Create an Internet based remote communication system specifically for process monitoring.
- Create a diagnosis system based on the EDM process model, incorporating the concept of remote monitoring

The approach taken in the research is based on process adaptation for machining variations. The system requires to adapt through the EDM macro model based on experience gained from experimentation. The process must be monitored remotely, instantaneously and frequently, and diagnosed with improvement recommendations for the control of the EDM parameters.

It is not the aim of the thesis to describe the control system for the EDM process. Control parameters are generated. The focus of this thesis is in determining the relation between the input parameters and process efficiency, chosen because it is a measure of how well the process is performing.

When controlling a system, a set of control input parameters will cause a resultant output parameter. The control method in this thesis is manual operator control. By changing the input parameters, the operator controls the change in the output parameter.

Because the mathematical relationship of input parameters related the output efficiency is non linear, a neural network is suggested for input – output correlations.

1.2 Non-traditional Machining

Non-traditional machining has made it possible to fabricate components that were either difficult or impossible to produce by conventional material removal processes[9]. Nontraditional machining refers to a wide variety of mechanical, electrical, thermal, and chemical energy based material removal processes that can be used to machine super alloys and ceramics, wood, plastics, and textiles.

The following processes fall under the category of non-traditional machining:

- Ultrasonic machining,
- Abrasive water jet machining (AWJM)
- Water jet machining (WJM)
- Chemical machining (CM)
- Electrochemical machining (ECM)
- Electrical discharge machining (EDM)
- Electrochemical grinding (ECG)
- Electron-beam machining (EBM)
- Laser-beam machining (LBM)
- Plasma machining

These processes find application when traditional machining processes are not satisfactory, economical, or even impossible, for the following reasons:

- The hardness and strength of the material is very high (typically above 400 HB) or the material is too brittle.

- The work piece is too flexible, slender, or delicate to withstand the cutting or grinding forces, or the parts are difficult to fixture – that is, to clamp in work holding devices.
- The shape of the part is complex, including such features as internal and external profiles or small-diameter holes of fuel-injection nozzles.
- Surface finish and dimensional tolerance requirements are more rigorous than those obtained by other processes.
- Temperature rise and residual stresses in the work piece are not desirable or acceptable.

These requirements led to the development of chemical, electrical, laser, and other means of material removal in the early 1940s.

1.3 EDM process

1.3.1 Introduction

The principal of electrical discharge machining, also called electro-discharge or spark-erosion machining, is based on the erosion of metals by spark discharges. Sometimes it is used to produce a part, such as producing a slot in a very hard metal, and sometimes it is used to "rescue" a part such as removing a broken tap.

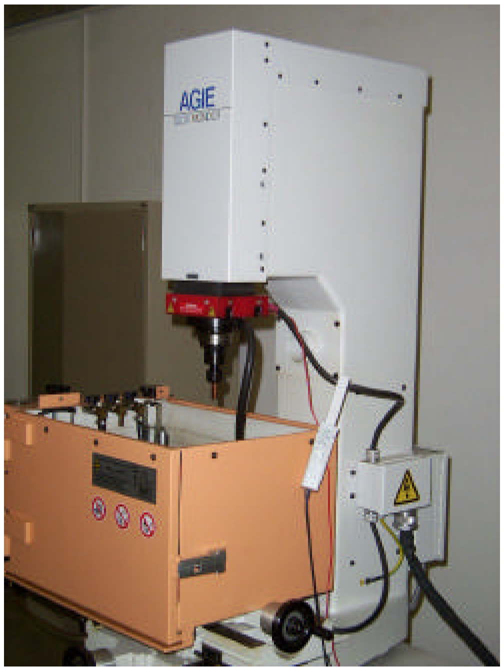


Figure 1.2 AGIE EDM machine

The basic idea is to move an electrode very close to the work piece, and repeatedly produce a spark between the two. This is best done while immersed in a dielectric liquid rather than in air, and it helps if the proper distance can be automatically maintained. Note that the electrode is eaten as well as the work piece, and some compensation must be made for this. Very good finish can be achieved, though at reduced speed. EDM is not a fast method; some jobs can take days to produce holes, so its use is limited to jobs that cannot easily be done in other ways (e.g. oblong slots or complex shapes, sometimes in very hard material). Note too, the work must be conductive so it does not work on materials such as glass or ceramic, or most plastics.

1.3.2 Control circuits

EDM is well suited for automated control. In fact, it seems almost the only way to go. The development of control circuits started with vacuum tube circuits, which were replaced with transistor circuits and today practically, all EDM systems are controlled by some form of microcomputer. The basic control system is illustrated in figure 1.3.

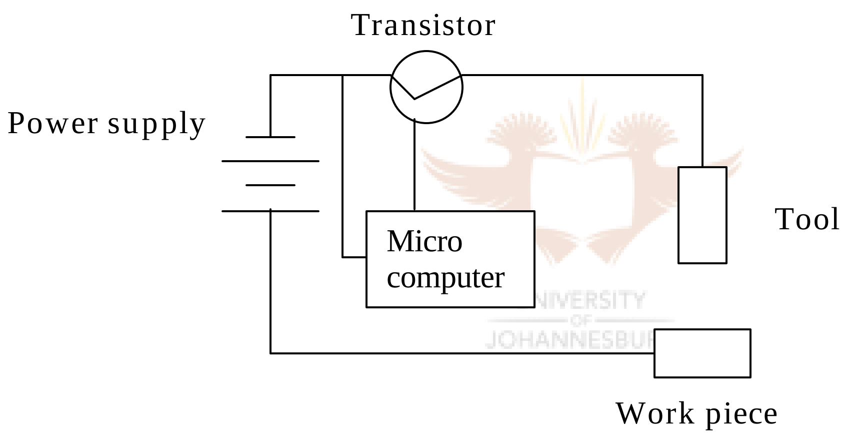


Figure 1.3 EDM microcomputer controller

1.3.3 EDM flushing

Flushing plays a very important role in the EDM process in particular the process efficiency and stability. The reason is simply that any particles that accumulate between the electrode and work piece, can cause an electrical short-circuit. The EDM process is based on the dielectric breakdown principal, where a high-energy plasma channel forms between the electrode and work piece. If there is a short circuit, the electrons will not flow through the plasma channel but rather through the accumulated metal residue, causing an inefficient pulse

There are various flushing strategies, each more applicable to a different machining scenario. The following basic flushing guidelines have been formulated.

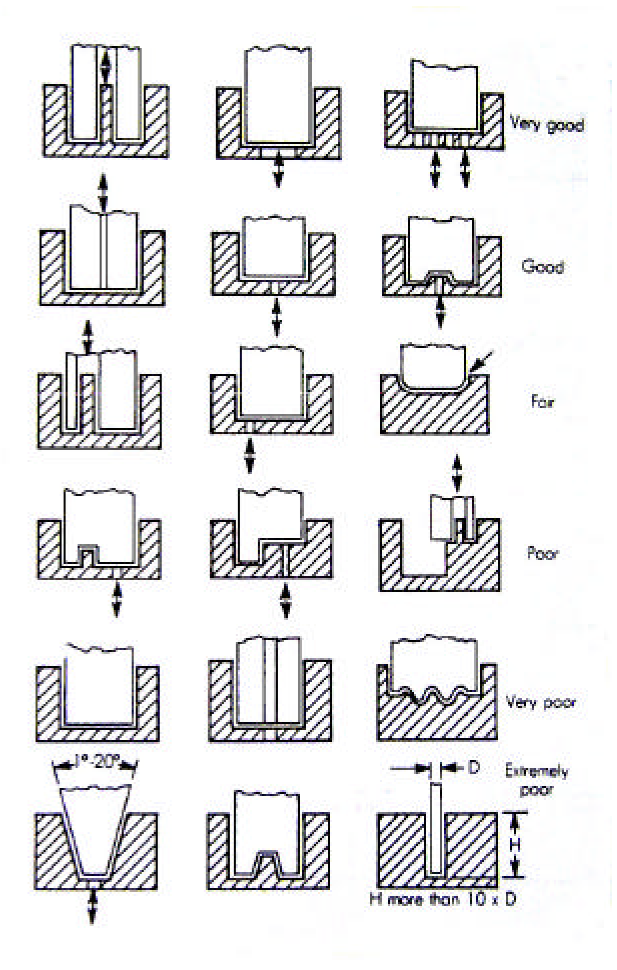


Figure 1.4 Flushing guidelines[9]

1.3.4 EDM pulse characteristics

Figure 1.5a shows the current through the electrode and the work piece and the voltage across the electrode and the work piece as they appear on an oscilloscope, as functions of time. The upper trace shows current with current on for about one third of the time. The current builds up quickly and then starts to level off when the power supply cuts off.

The voltage trace shows the breakdown voltage of the gap at *A*. The voltage then drops to *B*, which is the initial voltage of the arc. The voltage drops from *B* to *C* as the current increase because of the negative voltage current relationship in the electric arc. At *C*, the current is interrupted, the voltage goes to zero and reverses to *D*; but since there is no breakdown in the opposite direction, there is no current reversal and no machining. The voltage returns to zero, at *E*, and waits for the next pulse.

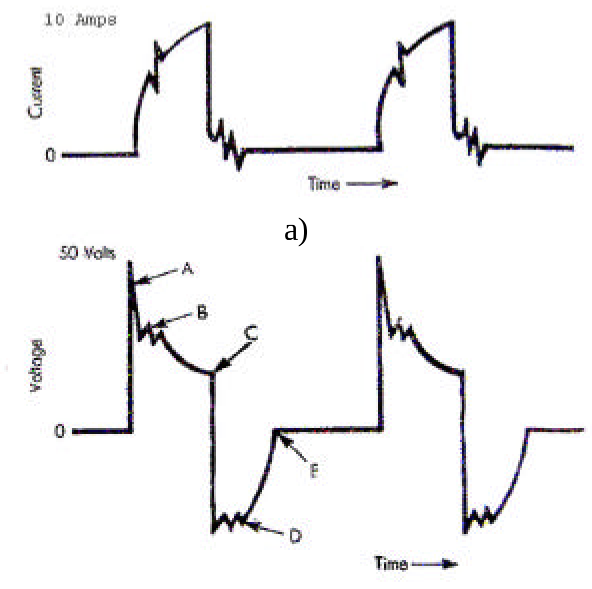


Figure 1.5 EDM pulse characteristics[9]

The energy dissipated in the system is the voltage times the current times the time. This is fairly constant during the pulse, as shown in figure 1.5b in which as oscillogram is taken of the same pulses shown in – except that the vertical axis is voltage and the horizontal axis is current. Any point along vertical and horizontal axes represents no power. A represents the breakdown voltage, but no power. B represents the power going into the work pieces. *C*, *D*, *E* and *F* represents traces at either no voltage or no current, and therefore no power. The line *B* is very nearly a section of the curve, volts time amps equals a constant, which would indicate constant power during the current pulse.

1.4 EDM Process parameters

The following parameters are relevant for the EDM process.

1. Pulse characteristics

- Voltage
- Current
- On time
- Off time
- Frequency
- Duty ratio
- Power
- Efficiency

2. Machining characteristics

- Surface finish
- Metal removal rate (MRR)

3. Electrode characteristics

- Electrode material
- Electrode geometry
- Electrode surface area
- Electrode wear

4. Flushing characteristics

- Flushing system
- Flushing rate
- Dielectric fluid

5. Machining characteristics

- Machining time
- Machining depth

1.5 State of the art.

When concerned with manufacturing, the introduction of Computer integrated manufacturing (CIM) [10][11] has played a huge roll in optimizing and streamlining the manufacturing process. CAE, CAD,CAM and CAT are buzzwords in the industry and have been incorporated into most manufacturing disciplines. Numerical control, flexible manufacturing systems, and group technology are computer based and are widely recognized as the future of manufacturing optimization.

Various optimization strategies have been applied to the EDM machining process. EDM modeling has always been illusive as it is machine dependant and environment dependant. It is therefore not surprising that control strategies of the EDM process was by nature an adaptive or fuzzy type of strategy[3]. By not understanding the process fully, the best that one can do is to adapt the process in a manner that experimentally proved sufficient.

These control systems proved extremely successful. Most commercially available machines use this kind of control method. Research has gone into the microscopic characteristics of the EDM process. This entails mathematical models for thermal distributions and research into the capacitance breakdown principal [2]. From a control point of view, this kind of research is applicable only if it has relevance to the macroscopic control parameters.

Other researches have gone into some of the EDM parameters such as electrode geometry and material, flushing systems, dielectric fluid, multiple electrodes, rotating electrodes etc.

Chapter 2

Concepts of Remote (Internet) communication

2.1 Introduction

Remote communication systems allow for the sharing of information and the communication between people, systems and devices[13]. The ability to communicate digital information over a secure network was initially developed by defense organizations and later expanded to companies and today it is freely available to the public.

The concept of remote communication has led to the development of a standardized communication language (protocol), which in general is referred to as TCP/IP.

2.2 Open system networking

The term open system networking in its broadest definition refers to a network based on a well-known and understood protocol (such as TCP/IP) that has its standards published and readily available to anyone who wants to use them. Open system networking also refers to the process of networking open systems (machine-specific hardware and software) using a network protocol.

Open systems make it possible for different computer manufacturers and software vendors to write communication software that will work between different software packages and different hardware configurations.

2.3 The OSI reference model

The Open Systems Interconnection Reference Model was adopted as a standard for open systems. The OSI Reference Model (OSI-RM) uses seven layers, as shown in Figure 2.1. The TCP/IP model is shown next to the OSI reference model to show interrelations between the two.

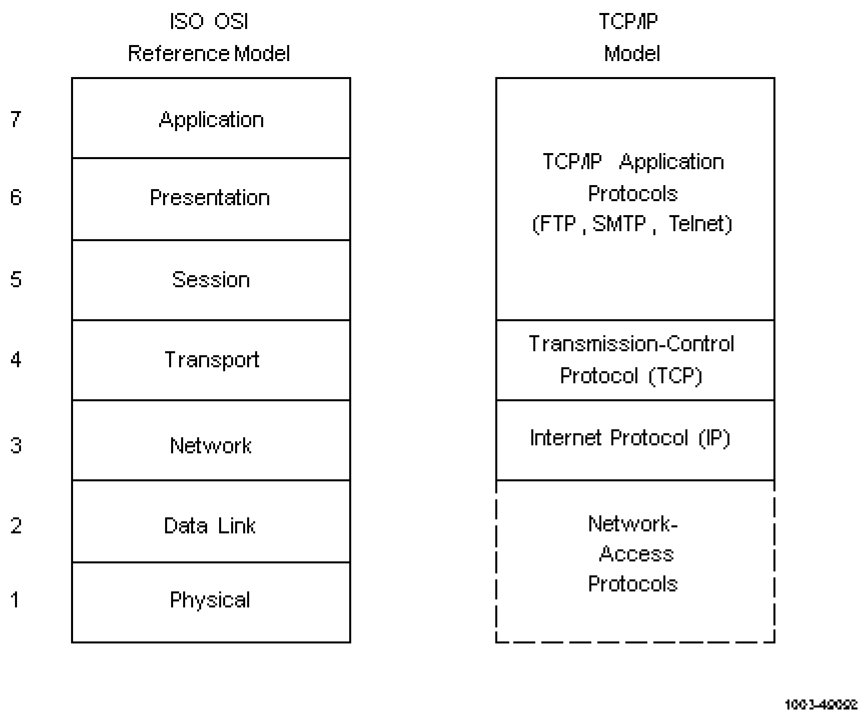


Figure 2.1 OSI reference model[13]

<u>The Application Layer</u>

The application layer is the end-user interface to the OSI system. It is where the applications, such as electronic mail, USENET newsreaders, or database display modules, reside. The application layer's task is to display received information and send the user's new data to the lower layers.

In distributed applications, such as client/server systems, the application layer is where the client application resides. It communicates through the lower layers to the server.

The Presentation Layer

The presentation layer's task is to isolate the lower layers from the application's data format. It converts the data from the application into a common format, often called the canonical representation. The presentation layer processes machine-dependent data from the application layer into a machine-independent format for the lower layers.

The presentation layer is where file formats and even character formats (ASCII and EBCDIC, for example) are lost. The conversion from the application data format takes place through a "common network programming language" (as it is called in the OSI Reference Model documents) that has a structured format.

The presentation layer does the reverse for incoming data. It is converted from the common format into application-specific formats, based on the type of application for which the machine has instructions. If the data comes in without reformatting instructions, the information might not be assembled in the correct manner for the user's application.

The Session Layer

The session layer organizes and synchronizes the exchange of data between application processes. It works with the application layer to provide simple data sets called synchronization points that let an application know how the transmission and reception of data are progressing. In simplified terms, the session layer can be thought of as a timing and flow control layer.

The session layer is involved in coordinating communications between different applications, letting each know the status of the other. An error in one application (whether on the same machine or across the country) is handled by the session layer to let the receiving application know that the error has occurred. The session layer can resynchronize applications that are currently connected to each other. This can be necessary when communications are temporarily interrupted, or when an error has occurred that results in loss of data.

The Transport Layer

The transport layer, as its name suggests, is designed to provide the "transparent transfer of data from a source end open system to a destination end open system," according to the OSI Reference Model. The transport layer establishes, maintains, and terminates communications between two machines.

The transport layer is responsible for ensuring that data sent matches the data received. This verification role is important in ensuring that data is correctly sent, with a resend if an error was detected. The transport layer manages the sending of data, determining its order and its priority.

The Network Layer

The network layer provides the physical routing of the data, determining the path between the machines. The network layer handles all these routing issues, relieving the higher layers from this issue.

The network layer examines the network topology to determine the best route to send a message, as well as figuring out relay systems. It is the only network layer that sends a message from the source to a target machine, managing other chunks of data that pass through the system on their way to another machine.

The Data Link Layer

The data link layer, according to the OSI reference paper, "provides for the control of the physical layer, and detects and possibly corrects errors that can occur." In practicality, the data link layer is responsible for correcting transmission errors induced during transmission (as opposed to errors in the application data itself, which are handled in the transport layer).

The data link layer is usually concerned with signal interference on the physical transmission media, whether through copper wire, fiber optic cable, or microwave. Interference is common, resulting from many sources, including cosmic rays and stray magnetic interference from other sources.

The Physical Layer

The physical layer is the lowest layer of the OSI model and deals with the "mechanical, electrical, functional, and procedural means" required for transmission of data, according to the OSI definition. This is really the wiring or other transmission form.

When the OSI model was being developed, a lot of concern dealt with the lower two layers, because they are, in most cases, inseparable. The real world treats the data link layer and the physical layer as one combined layer, but the formal OSI definition stipulates different purposes for each. (TCP/IP includes the data link and physical layers as one layer, recognizing that the division is more academic than practical.)

2.4 The TCP / IP Suite

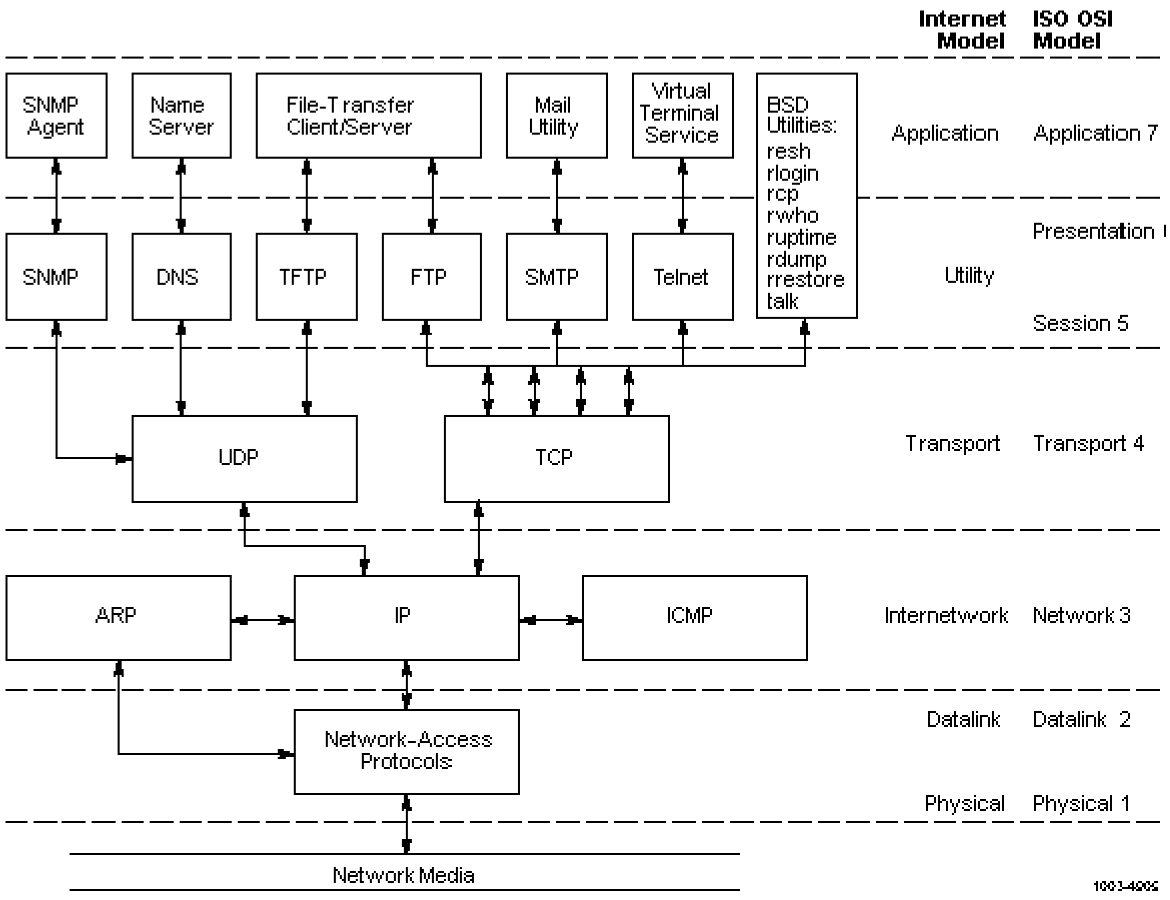


Figure 2.2 TCP/IP protocol suite.[13]

The mail, file transfer, and virtual terminal services use the services of the underlying TCP/IP protocols. The BSD protocols require the TCP services.

The following list is not exhaustive but mentions the primary user applications that TCP/IP provides.

Telnet

The Telnet program provides a remote login capability. This lets a user on one machine log onto another machine and act as though he or she were directly in front of the second machine. The connection can be anywhere on the local network or on another network anywhere in the world, as long as the user has permission to log onto the remote system.

You can use Telnet when you need to perform actions on a machine across the country. This isn't often done except in a LAN or WAN context, but a few systems accessible through the Internet allow Telnet sessions while users play around with a new application or operating system.

File Transfer Protocol

File Transfer Protocol (FTP) enables a file on one system to be copied to another system. The user doesn't actually log in as a full user to the machine he or she wants to access, as with Telnet, but instead uses the FTP program to enable access. Again, the correct permissions are necessary to provide access to the files.

Once the connection to a remote machine has been established, FTP enables you to copy one or more files to your machine. (The term *transfer* implies that the file is moved from one system to another but the original is not affected. Files are copied.) FTP is a widely used service on the Internet, as well as on many large LANs and WANs.

Simple Mail Transfer Protocol

Simple Mail Transfer Protocol (SMTP) is used for transferring electronic mail. SMTP is completely transparent to the user. Behind the scenes, SMTP connects to remote machines and transfers mail messages much like FTP transfers files. Users are almost never aware of SMTP working, and few system administrators have to bother with it. SMTP is a mostly trouble-free protocol and is in very wide use.

Transmission Control Protocol

Transmission Control Protocol (the TCP part of TCP/IP) is a communications protocol that provides reliable transfer of data. It is responsible for assembling data passed from higher-layer applications into standard packets and ensuring that the data is transferred correctly.

User Datagram Protocol

User Datagram Protocol (UDP) is a connectionless-oriented protocol, meaning that it does not provide for the retransmission of datagrams (unlike TCP, which is connection-oriented). UDP is not very reliable, but it does have specialized purposes. If the applications that use UDP have reliability checking built into them, the shortcomings of UDP are overcome.

Internet Protocol

Internet Protocol (IP) is responsible for moving the packets of data assembled by either TCP or UDP across networks. It uses a set of unique addresses for every device on the network to determine routing and destinations.

TCP/IP description

In general, TCP provides a reliable two-way communication method between two programs running on the same or different computers. The reference to "reliable" means that each byte of information is guaranteed to reach its destination, or the originating program will be notified of failure. Additionally, the transmission is ordered. This means that data is received in the same order it is sent. Each TCP connection relies on four numbers: the originating host address and port, and the target host address and port. In terms of the Internet, these are the IP address and port numbers used to contact another machine.

When a document is requested via a web browser for example, the host name is looked up on the Domain Name Server and translated into an IP address. The standard port for World Wide Web traffic is 80, so by default the browser knows to connect to the remote host address on port 80. An Internet user can specify a different port by using a colon (:) and the number after the host name, for example: http://www.webserver.com:8088/index.html. The browser would then translate www.webserver.com into an IP address such as 207.199.193.192, and then open a TCP connection to that server on port 8088.

Each time a connection is made, two bits in the TCP packet header are used to negotiate the creation. The SYN bit is set in the first packet sent to a target host in an attempt to make a connection. If the target host receives the packet successfully and allows connection, it will reply with the SYN and ACK bits set. Once the originating host has received that packet, it sends another packet to the target host, this time with the ACK bit set and the SYN bit unset. This is known as the "three way handshake." From that point on only the ACK bit will be set.

TCP/IP works on a Client-Server principal. If two computers are used, one has to be set up as a server and the other as a client. This means that the Client will have to make a connection request to the server. When more than one computers is used, one computer has to be set up as a server and the rest has to be set up as Clients. This allows the control of data flow to the Server, therefore isolating the clients from each other.

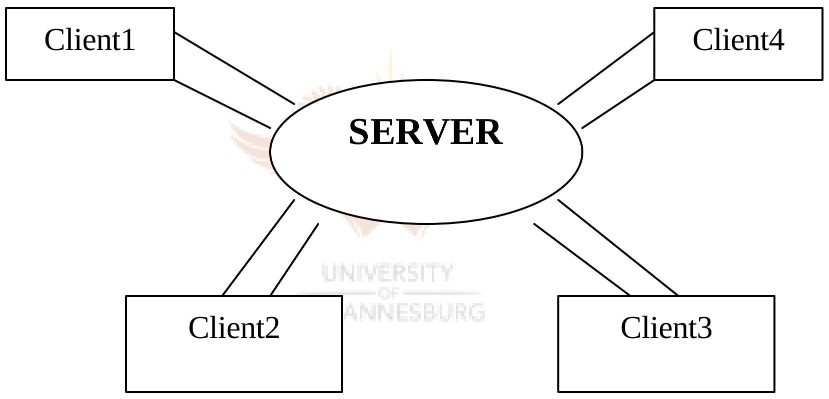


Figure 2.3 Server-Client setup

TCP/IP client-server setup

The following flow diagram illustrates the data flow between server and client applications. This model is applicable to all program languages. It serves as a framework for reliable data communication

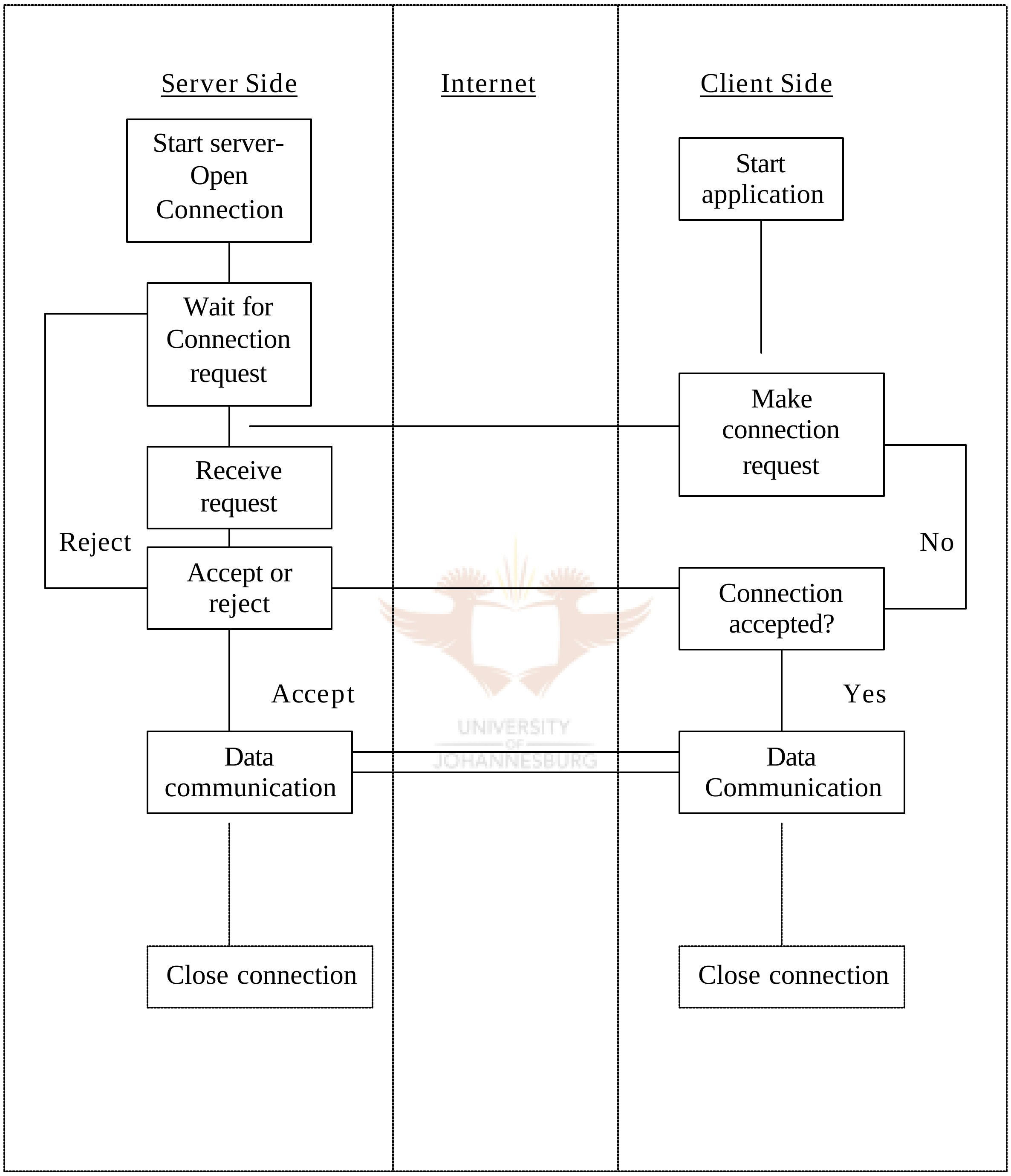


Figure 2.4 TCP / IP communication

2.4 Remote system application

The development of the above protocols has led to globalized data communication and exchange of information. In more recent times, the use of the Internet and all of its protocols and suites proved to be a tool to optimize systems and environments. Internet conferencing, remote user intervention, user training and user monitoring became synonymous with the information portal. People are communicating over secure connections and companies are streamlining through programs such as SAP, which allows for total control of all aspects of business through remote users onto a server.

To take it a step further is to apply remote systems to machines and machine environments. The control and monitoring of manufacturing environments through computer integrated manufacturing (CIM), has proved very useful and has found practical application in the manufacturing industry[10].

The application of remote Internet communication in this thesis aims to allow manufacturing systems to optimize machining parameters through sharing of machining experience and a global server. The environments must be integrated into a centralized database of experience, which in turn will be available to any other machining environment.

The layout suits the TCP/IP protocol well, since there is a centralized database or server that contains the machining experience. Remote users can log onto the server. The server will determine if a valid user is trying to log on, or if the request should be rejected, and further more, the user will be isolated from other users. Any communication that takes place is only between server and client.

The advantages of such a system is obvious. Information is centralized, security is maintained, data integrity can be verified, and the maintenance of such a system resides with one administrator.

The database must be updated remotely and continuously. The database is used to create a practical model of the EDM machine, which takes the form of a Neural network. This model is remotely available to any machining environment for machining parameter optimizations.

Chapter 3

Neural Networks

3.1 Introduction

In general, neural networks are simply mathematical techniques designed to accomplish a variety of tasks. Neural networks can be configured in various arrangements to perform a range of tasks including pattern recognition, data mining, classification, and process modeling[14]. The latter is the primary interest in this context and although the types and topologies of neural networks (NN) vary greatly in the field, by far the most commonly used, particularly in process control, is the feed – forward, back propagation neural network.

The theory and even implementation of neural networks has been around better than 50 years; however, only recently have neural networks found widespread, practical application. This is due primarily to the advent of high speed, low cost computers that can support the rather computationally intensive requirements of a neural network of any real complexity.

The conceptual constructs of a neural network stemmed from our early understanding of the human brain. The brain is comprised of billions of interconnected neurons (upwards of 10^{11} neurons in the human brain). The fundamental building blocks of this massively parallel cellular structure are really quite simple when studied in isolation. A neuron receives incoming electrochemical signals from its dendrites and collects these signals at the neuron nucleus. The neuron nucleus has an internal threshold that determines if the neuron fires and an electrochemical signal are sent to all neurons connected to the firing neuron on its output connections or axons. Otherwise, the incoming signals are ignored and the neuron remains dormant.

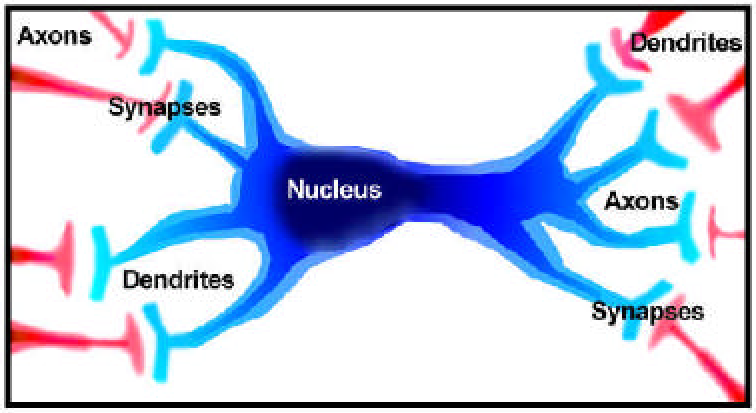


Figure 3.1 Neuron[19]

The basic architecture of a Neural Network typically consists of an input function, which can take the form of binary, continuous or normalized data; an overall control strategy which describes the number and functions of layers and connections in the Network; a processing architecture which consists of a transfer function description, summation functions, and a relative learning strategy; a method for identifying and "learning" from past errors in estimates; and finally a mechanism for feeding error corrections back into the network

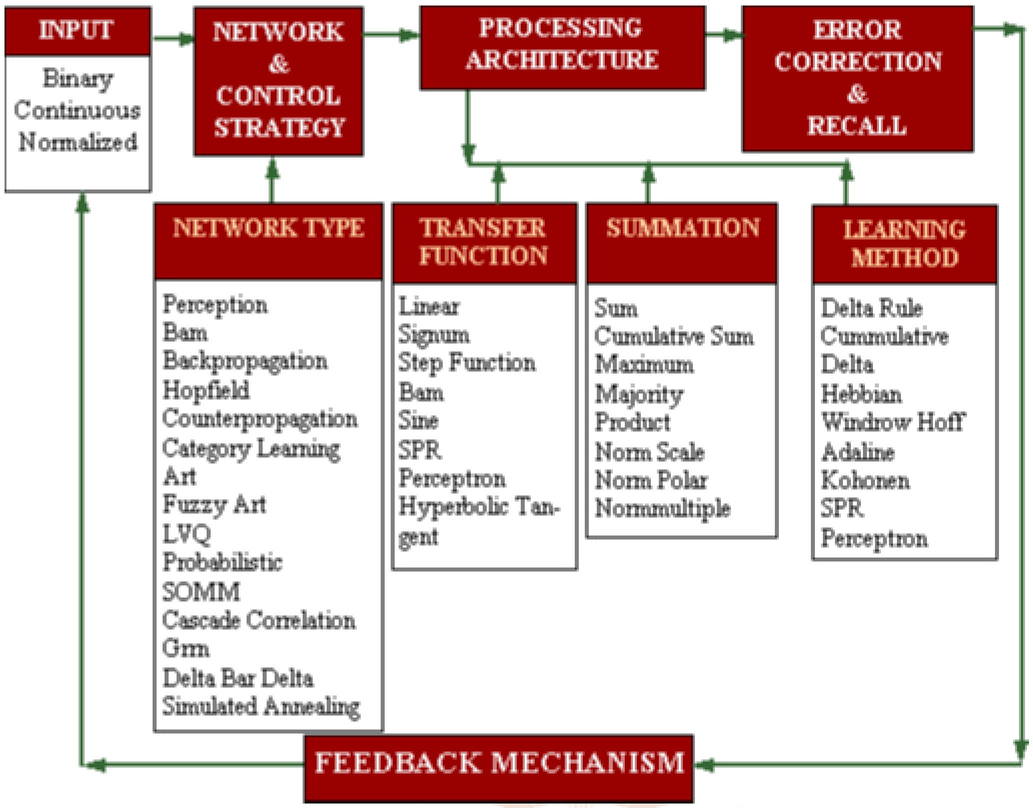


Figure 3.2 Neural network Topology[19]

3.2 Neural Networks for process modeling

In the process modeling and prediction the feed-forward, back-propagation NN is used just like many other modeling techniques. It receives process inputs and predicts process outputs. The error between predicted outputs and actual outputs are used to validate the effectiveness of the model and fine tune the model to more accurately predict the process dynamics in the future. Using some clever techniques, the neural network modeling approach can predict both static and dynamic conditions and account for process lag and dead-times

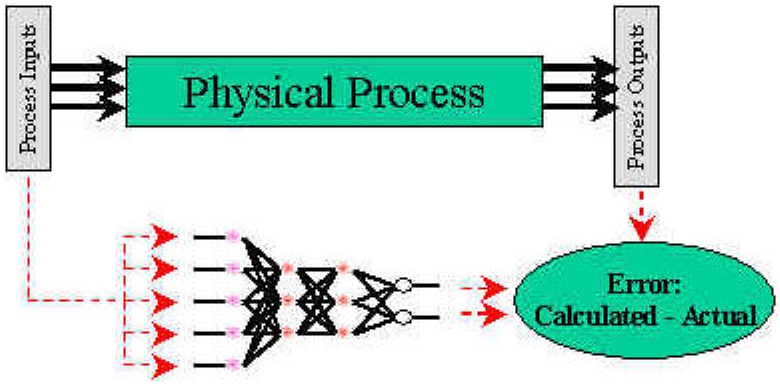


Figure 3.3 Neural network process modeling[19]

One of the advantages of using the neural network approach is that a first principals understanding of the physical process is not necessary. Given the inputs and outputs a neural network model can be constructed and trained to accurately predict process dynamics. This technique is especially valuable in processes where a complete understanding of the physical mechanisms at work is very difficult or even impossible to acquire. Neural networks are also used in tandem with proven models and have been found to discover higher order and dynamic effects that were not accounted for in the original model.

The Neuron Node

At the heart of every neural network is what is referred to as the neuron node (sometimes called a processing element), which is analogous to the neuron nucleus in the brain. As was the case in the brain, the operation of the neuron node is very simple; however, also as is the case in the brain, when all connected neurons operate as a collective they can provide some very powerful learning capacity.

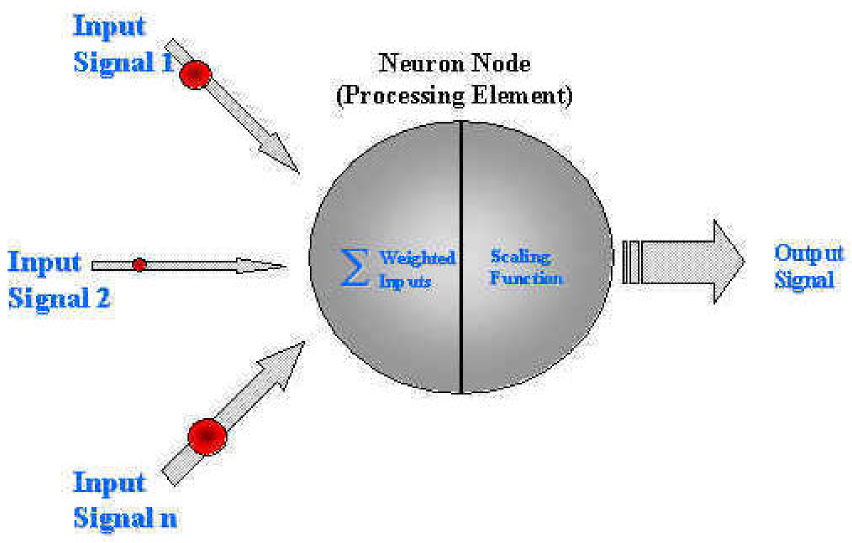


Figure 3.4 Neuron node[19]

Input signals are applied to the node via input connections y_i (dendrites in the case of the brain). The connections have "strength" which changes as the system learns. In neural networks the strength of the connections are referred to as weights (w_{ji}). Weights can either excite or inhibit the transmission of the incoming signal. Mathematically, incoming signal values are multiplied by the value of that particular weight.

$$y_j = f(\sum_{i=1}^{n} w_{ji} x_i - \Theta_j)$$

At the neuron node, all weighted-inputs are summed. This summed value is then passed to a scaling function (f). The selection of the scaling function is part of the neural network design. Some examples of scaling functions are shown below. Functions having rounded corners (thus no points of discontinuities) such as the sigmoid or hyperbolic tangent are good selections for natural phenomena.

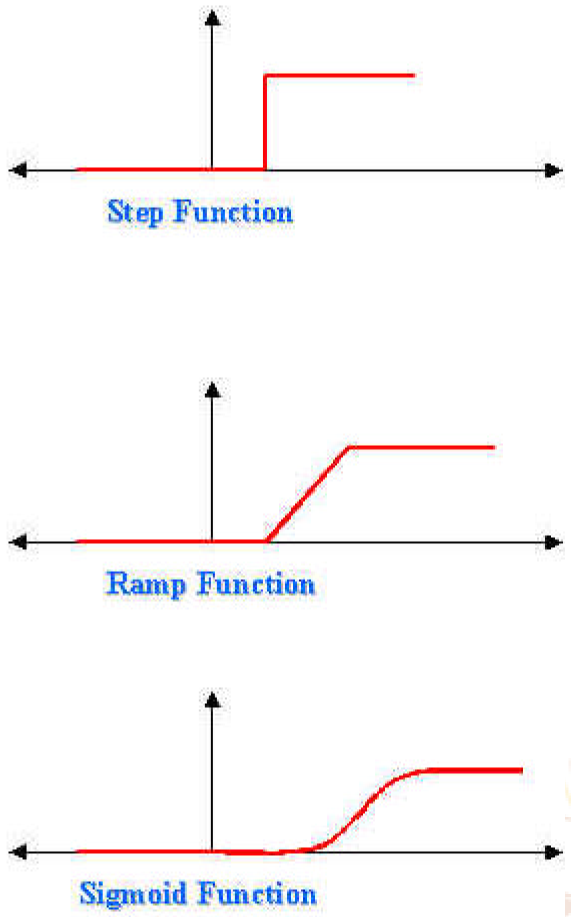


Figure 3.5 Activation functions[19]

In each of the cases above, the sum of the weighted inputs represents the horizontal axis. The curve represents the output of the function for each value on the horizontal axis. θ is the external threshold, also called an offset or bias. The function output is then sent to each node that is connected to an output weight of the firing neuron.

3.3 Neural Network Topology:

In a feed-forward NN the network is constructed using layers. All nodes in a given layer are connected to all nodes in a subsequent layer. The network requires at least two layers, an input layer and an output layer. In addition, the network can include any number of hidden layers with any number of hidden nodes in each layer (not necessarily the same in each hidden layer, in fact, typically not). An example of a typical feed-forward NN is as follows:

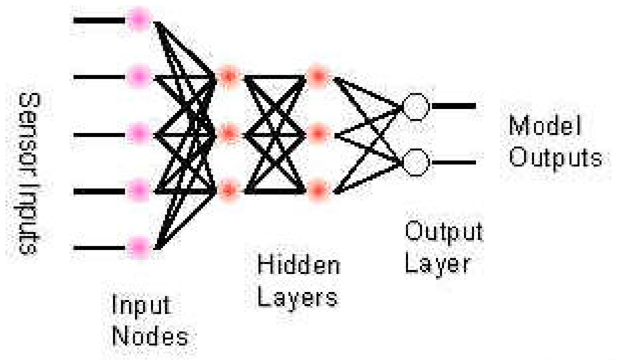


Figure 3.6 Feed forward Neural network[19]

In this topology the sensor information comes from the process instrumentation and is sent to the neural network as a series of model inputs. Using the mathematical mechanics described above the values at each node site are calculated and the signal from the input vector propagates through the network layer by layer until finally the output layer is reached. The output layer, or output vector, has a node for each process variable that is being predicted. The output vector represents the predicted output.

3.4 Training a Neural Network

Thus far, we discussed the feed-forward characteristics of the NN. There has been no mention of how a neural network arrives at and/or adapts its weight matrices to accurately predict process dynamics. The *training* of a neural network is the process by which the neural network is presented with actual data from the process, and uses this data (either off-line or continuously on-line) to find the most appropriate set of weights for each connection. The most appropriate set of weights being defined as those weight values, which minimize the output error.

Similar to the way we progressed forward through the neural network layer by layer to calculate the predicted values we use the "back-propagation" technique to run backwards through the network layer by layer to make small modifications to weight values aimed at minimizing the output error.

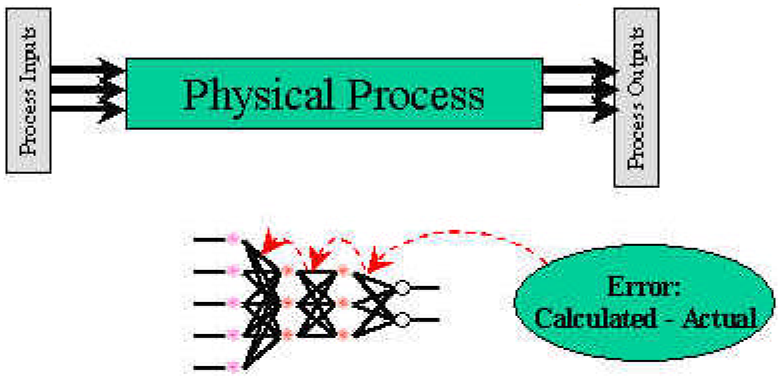


Figure 3.7 Back propagation learning[19]

The task of training the network can be thought of in terms of an optimization problem. The variables to manipulate are the various weight factors while the variable we are seeking to minimize is the NN output error. One way to determine the appropriate weight modifications is to use the following equation:

$$\left\{\begin{matrix}\text{new weight}\\\text{factor}\end{matrix}\right\} = \left\{\begin{matrix}\text{old weight}\\\text{factor}\end{matrix}\right\} + \left\{\begin{matrix}\text{learning}\\\text{rate}\end{matrix}\right\} * \left\{\begin{matrix}\text{correction}\\\text{term}\end{matrix}\right\} + \left\{\begin{matrix}\text{momentum}\\\text{coefficient}\end{matrix}\right\} * \left\{\begin{matrix}\text{previous}\\\text{weight change}\end{matrix}\right\}$$

There are various methods for determining the correction term. However, once the correction term has been calculated the learning rate determines how radical the incremental change should be. The momentum coefficient in this approach encourages the direction and relative magnitude of the previous change.

The learning rate provides the effect of gravity if we think of the optimization problem in terms of a marble rolling over some curved surface. The learning rate encourages the marble to follow the fall line of the surface.

The momentum coefficient mirrors the effect of physical momentum of an object in motion (mass times velocity). This mechanism can enable the training equation to work through local minimums. Consider an error solution that follows a two dimensional curve. Further suppose that the original weight matrix predicts an error on the upper left side of the curve:

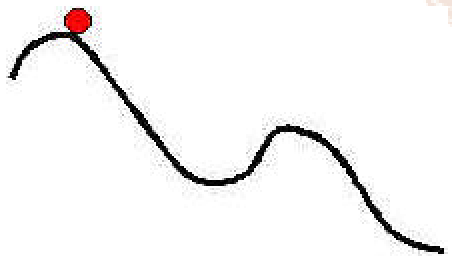


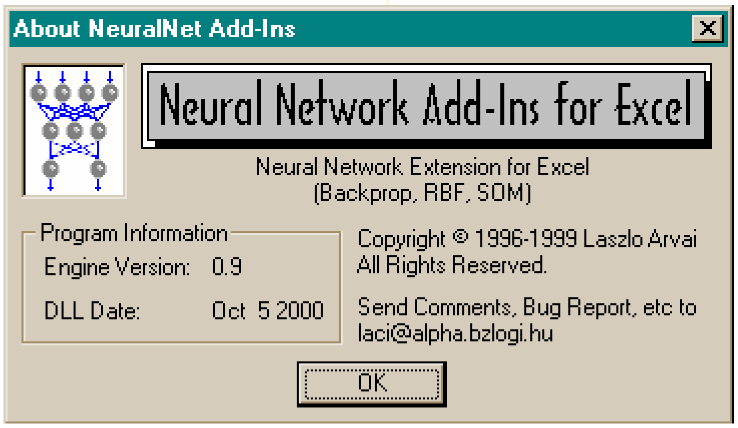
Figure 3.8 Function Momentum[19]

Without gravity (or in neural network term: a zero learning value) the ball doesn't move. If we add gravity but no momentum, the ball will follow the fall line to the dip in the middle of the hill. However, this is not the lowest point on the curve. By adding techniques such as momentum, the ball will pick up speed through the decent that will carry it through the local minimum and find the lower elevation on the other side.

Other techniques that periodically disturb the system can be beneficial as well. Resetting the weights, adding noise to the input signals, etc, can do this. The physical analogy to our ball on the hill case here is something like an earthquake that would cause the ball to be jostled out of the dip and continue down the hill to the lower elevation. By strategically timing these types of events based on the rate of change of the error, the true global minimum can be identified.

3.5 Neural network software

The Neural network software being used is an Excel add-in, which interacts with data on a specific data sheet.



About NeuralNet Add-Ins

Neural Network Add-Ins for Excel

Neural Network Extension for Excel
(Backprop, RBF, SOM)

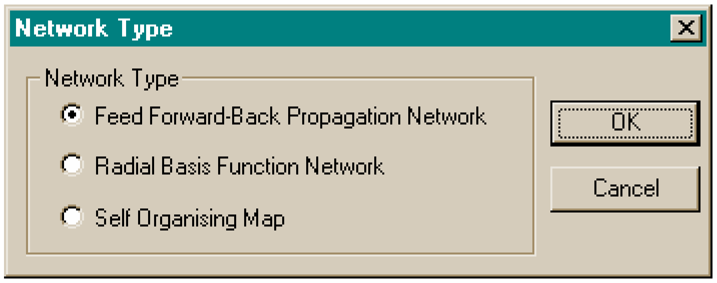
Program Information
Engine Version: 0.9
DLL Date: Oct 5 2000

Copyright © 1996-1999 Laszlo Arvai
All Rights Reserved.

Send Comments, Bug Report, etc to
laci@alpha.bzlogi.hu

Figure 3.9.1 Splash screen

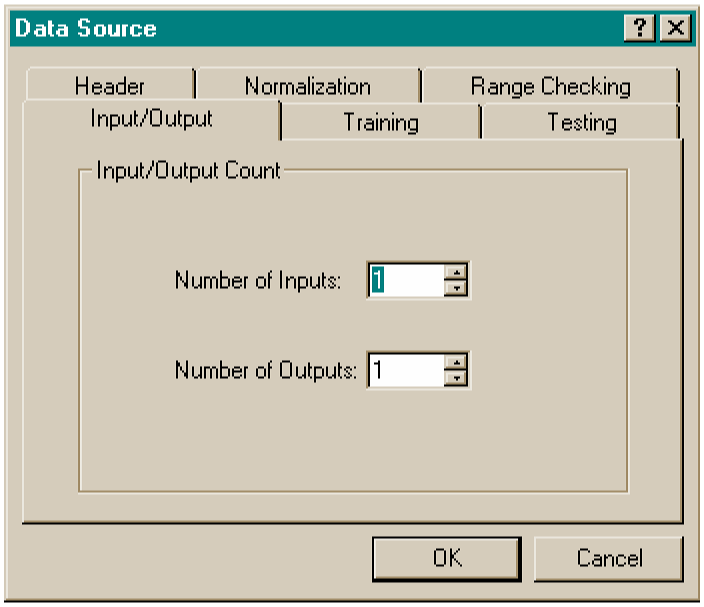
This program makes provision for 3 primary types of neural networks. It has built in algorithms to support customs architectures for each type of network.


Network Type

Network Type
- (●) Feed Forward-Back Propagation Network
- ( ) Radial Basis Function Network
- ( ) Self Organising Map

Figure 3.9.2 Network selection screen

Since the focus is placed on process modeling, the network type that is used is the feed forward back propagation neural net. The network has to be initialized first and therefore the number of inputs and outputs has to be specified. The custom dialogue box for this also has other options concerning the network.


Data Source

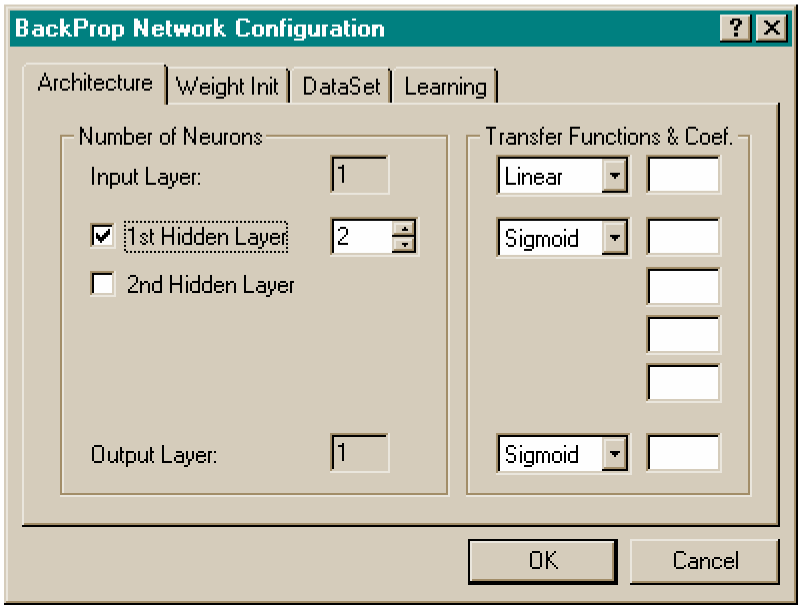
Input/Output Count

Number of Inputs: 1

Number of Outputs: 1

Figure 3.9.3 Input – Output setup

When setting up the network, a decision has to be made on the amount of hidden layers in the network and the threshold function to be applied to each neuron. In this dialogue box it can be seen that each layer can be organized as the user wishes. With experimentation, the best combination of hidden layers and transfer functions can be found.


BackProp Network Configuration

Architecture | Weight Init | DataSet | Learning

Number of Neurons

Input Layer: 1

[✓] 1st Hidden Layer: 2

[ ] 2nd Hidden Layer

Output Layer: 1

Transfer Functions & Coef.

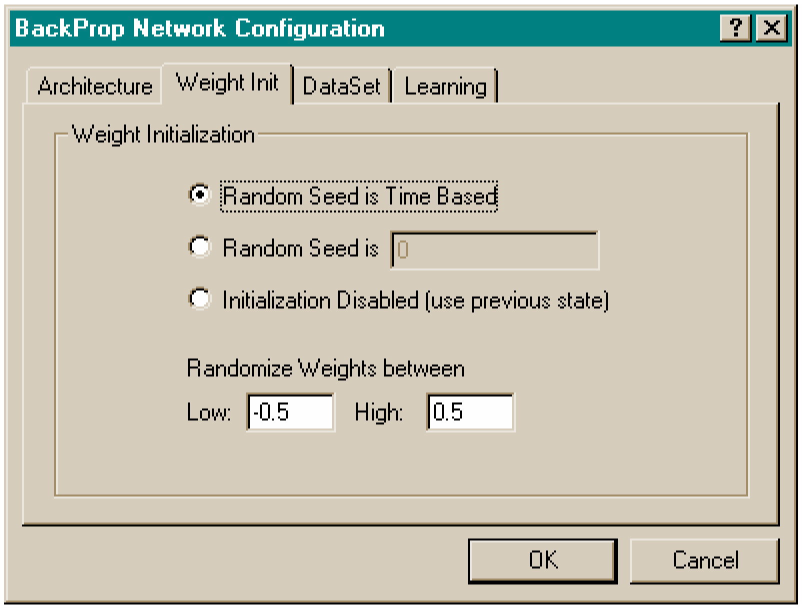
Linear

Sigmoid

Sigmoid

Figure 3.9.4 Network architecture setup

The neuron weights associated with the network can be randomly initialized or initialized from a previous state. Random initialization gives the network a different starting position and when training the network, it is often possible that the network will produce different training results. It is sometimes necessary to train the network a couple of times until a good result is achieved.


BackProp Network Configuration

Architecture | Weight Init | DataSet | Learning

Weight Initialization

[● Random Seed is Time Based]

○ Random Seed is 0

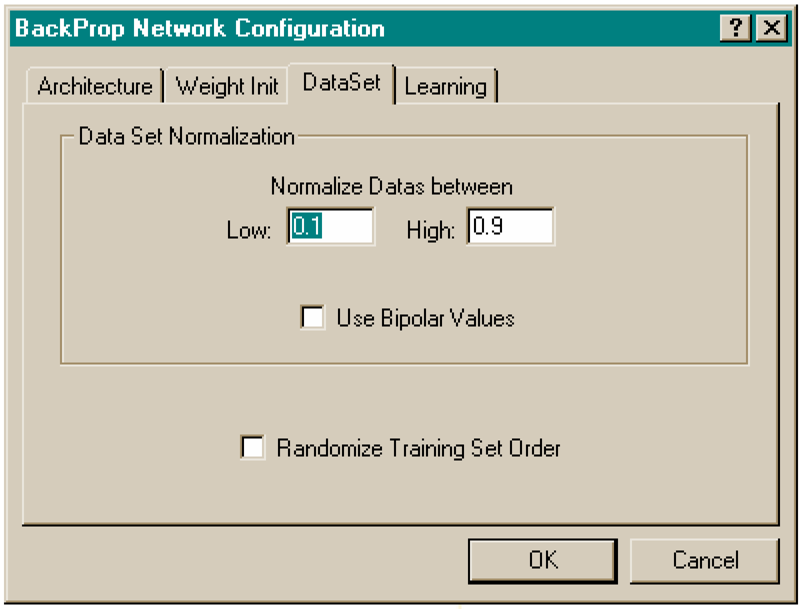
○ Initialization Disabled (use previous state)

Randomize Weights between

Low: -0.5 High: 0.5

Figure 3.9.5 Weight initialization screen

Normalization maps the data into an interval. This is done to format the data into a range that the neural network can deal with.


BackProp Network Configuration

Architecture | Weight Init | DataSet | Learning

Data Set Normalization

Normalize Datas between

Low: 0.1 High: 0.9

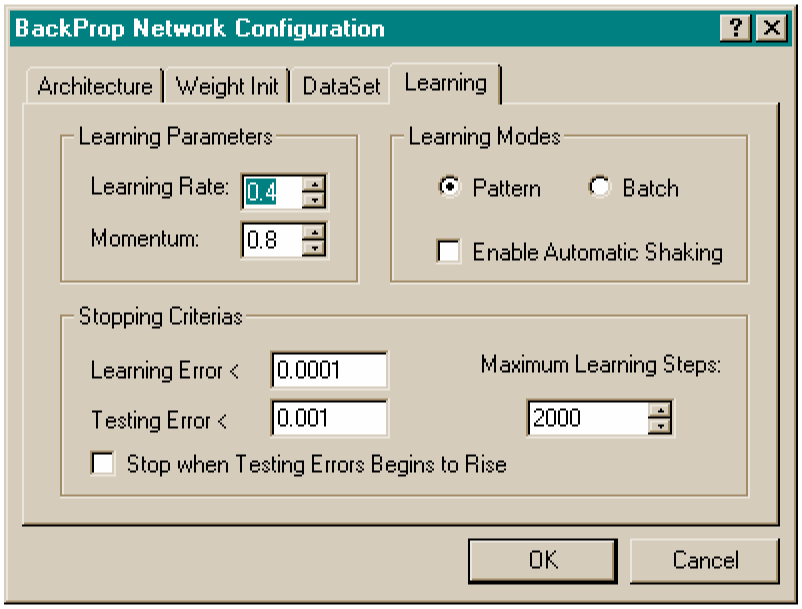
☐ Use Bipolar Values

☐ Randomize Training Set Order

OK Cancel

Figure 3.9.6 Dataset normalization screen

The final step in setting up the neural network is by defining the learning rate, Momentum and stopping criteria.


BackProp Network Configuration

Architecture | Weight Init | DataSet | Learning

Learning Parameters

Learning Rate: 0.4

Momentum: 0.8

Learning Modes

● Pattern ○ Batch

☐ Enable Automatic Shaking

Stopping Criterias

Learning Error < 0.0001

Testing Error < 0.001

Maximum Learning Steps: 2000

☐ Stop when Testing Errors Begins to Rise

OK Cancel

Figure 3.9.7 Learning setup screen

The neural network can now be trained from data in the excel sheet. To optimize the training of the network, the above parameters can be manipulated until the desired training criteria have been met.

3.6 Neural network application for modeling of EDM parameters.

The neural network has to be provided with a training data set. This data is utilized to train the neural network through back propagation. The training set consists of a set of input parameters related to an output parameter.

For the EDM process, it was decided to focus on 5 input parameters, and a single output parameter. The input parameters have to be easily variable, and parameters that the EDM operator can easily change. The output parameter is not related to any single input parameter, but rather is a characteristic of the process that is dependant on a change in a combination of process parameters.

Input parameters

- On time
- Off time
- Electrode diameter
- Current
- Flushing

Output parameter

- Efficiency

Chapter 4

Monitoring of the EDM process

4.1 Motivation for Remote Monitoring

A recent trend in manufacturing is for global competitiveness and information sharing. Companies set up international communication networks to improve communication between its global subsidiaries and to standardize manufacturing facilities. There are many obvious advantages to this. Quality control can be standardized, Machining procedures can be standardized, and Staff training can be kept up to date.

This has traditionally been done through very expensive communication networks. More and more trends are moving to Internet communication. The Internet is a readily available, cheap communications channel that ensures reliable secure communication.

4.2 Experimental layout

For experimental purposes, the experimental layout was done on a local area network. This means that communications was through a local area server. The setup can be done on any network topology that conforms to TCP/IP standards.

The following equipment is required for the experimental setup

- ADC 200

Figure 4.1 Pico ADC-200 oscilloscope

The ADC 200 is a digitizing oscilloscope. Its basic function is to interpret an analogue signal and to translate it to a discrete digital signal. This is presented in an array of 500 data points, making it easy to manipulate the data and to process it for efficiency calculation. Each data point can be separately processed, and the average efficiency calculated.

The interface with the pc is through Visual Basic libraries. The oscilloscope plugs into the pc through the parallel printer port. All the oscilloscope functions can be accessed through simple Visual Basic commands.

- Current Probe

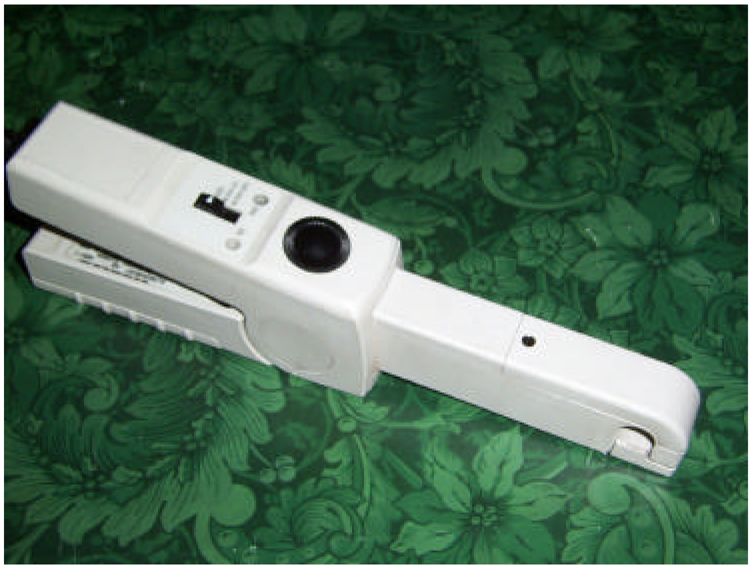


Figure 4.2 HP current probe

The HP probe is clamped to the current supply cable. It measures the current flowing through the electrode. This implies that the power supply characteristics is ignored, and only the machining power is measured

- PC (Pentium based) x 2

For optimum performance, it is recommended that at least a Pentium 2 based computer be used. This increases the sampling rate of the oscilloscope. For neural net training, it is recommended that a very fast computer be utilized, as it takes a lot of time for back propagation training.

- LAN
- Server software
- Client software
- Neural Network software

The experimental layout is given in figure 4.3

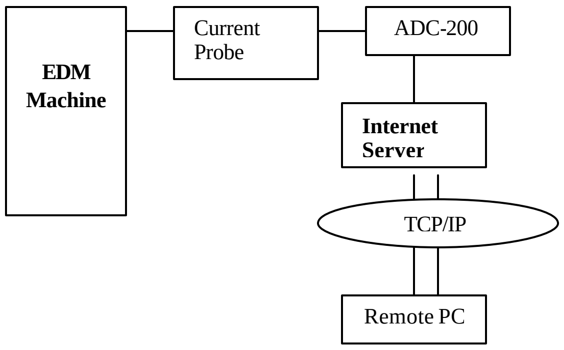


Figure 4.3 Experimental setup

4.3 Efficiency Definition

The EDM process is constantly monitored, with the efficiency calculated at discrete intervals. The definition of process efficiency is given as follows:

$$Efficiency(h) = \frac{T_{ie}}{T_i} \times 100$$

Where T_{ie} = Total efficient current On time

T_i = Total current On time

Efficient On time is defined as the time where the EDM pulse is at the working current. This implies that dielectric breakdown occurs and that there are no short circuit or open circuit pulses. Instantaneous process efficiency is calculated accordingly.

4.4 Efficiency calculation

The ADC-200 oscilloscope gives a discrete digital output. The oscilloscope signal is mapped into an array of 500 data points. Each point represents a current value at a specific time. By plotting these points in series, the oscilloscope signal can be seen.

The efficiency calculation is based on the amount of data points that are efficient, divided by the total amount of data points. Only the on-time data points are considered, since the on time is the power cycle of the EDM machine.

By considering the following EDM pulse characteristics, it is possible to formulate an algorithm to calculate the EDM efficiency.

- The on-time is slightly greater that zero
- The off time is zero
- Short circuit pulse is greater than effective pulse
- Open circuit pulse are equal to zero, which means that no power is flowing

The following algorithm represent the above stated pulse characteristics for calculation of EDM efficiency

```
For k = 1 to 500
begin
If   current > 0 then total on pulses = total on pulses+1:
                         If  lower threshold < current < upper threshold
                              Then effective on pulses = effective on pulses +1
End
Efficiency = effective on pulses / total on pulses * 100
```

This algorithm determines the efficiency for the total sample time of the oscilloscope. The sample time can include varying amounts of pulses depending on the machining frequency and the oscilloscope settings.

4.5 Statistical manipulation of efficiency data

The EDM efficiency is by definition very unstable. When looking at for example five or six pulses, it is very difficult to determine the efficiency of the machine accurately. It is necessary to look at much more pulses to determine the efficiency accurately. A number of 131 samples [5] are required to calculate the efficiency within a 95% confidence.

It was decided to use 300 samples for efficiency calculation. This is also a good amount since this would provide an efficiency calculation every three minutes, depending on the computer sample time and the computer capability.

The following algorithm was formulated for efficiency calculation.

```
        For k = 1 to 300
                Begin
                Get sample from PC
                Calculate sample efficiency
                Sumtotal = Sumtotal + efficiency
        End
Efficiency = Sumtotal/300
```

4.6 Monitored data for diagnosis

Each time the data is monitored; additional data is added to the data training set for the neural network. The data consists of the input parameters and the output parameter (process efficiency). The more data there is available to the neural network, the better the neural network will be able to predict the output efficiency of the process. The neural network is only trained periodically; therefore the data needs to be accumulated over a period of time, or over an experimental period, for later use in training the neural network.

When performing process diagnosis, the monitored data is compared to acceptable process for diagnosis of the process state. This comparison will allow the system to decide whether the process needs to be adjusted by generated recommendations to improve the process efficiency.

There are therefore two monitoring situations. The one is for experimental data acquisition and the other is for process diagnosis. They can be performed separately or simultaneously. For simplification, the monitoring of the process in this work was done separately for data acquisition and experimentation. Ultimately, a system that learns from every sample query will in time, provide the best process model, since it will have more experience to be trained on.

Chapter 5

Experimental data acquisition and Back propagation learning

5.1 Relevant parameters

The parameters presented in section 1.3.4 are all relevant for the EDM machining process, but it was decided to focus on six parameters, five Input parameters and one output parameter. A prerequisite was that the output parameter has to be monitored on-line. Metal removal rate and surface finish could not be monitored on-line, but can possibly be included in further exploration of this topic.

Input parameters

- On time
- Off time
- Electrode diameter
- Current
- Flushing

Output parameter

- Efficiency

Electrodes

The decision was made to use round copper electrodes with flushing holes. Round electrodes create the most stable machining conditions and most continuous flushing patterns.

Parameters kept constant

The following parameters are kept constant throughout the experimentation process.

- Electrode material
- Electrode shape
- Flushing system
- Dielectric oil
- Work piece material
- Machining time

5.2 Software design

The software program has to be able to generate experimental results that could be used in the Excel neural network program. All the relevant parameters will have to be recorded in a file. The following flow diagram illustrates the experimental data acquisition method.

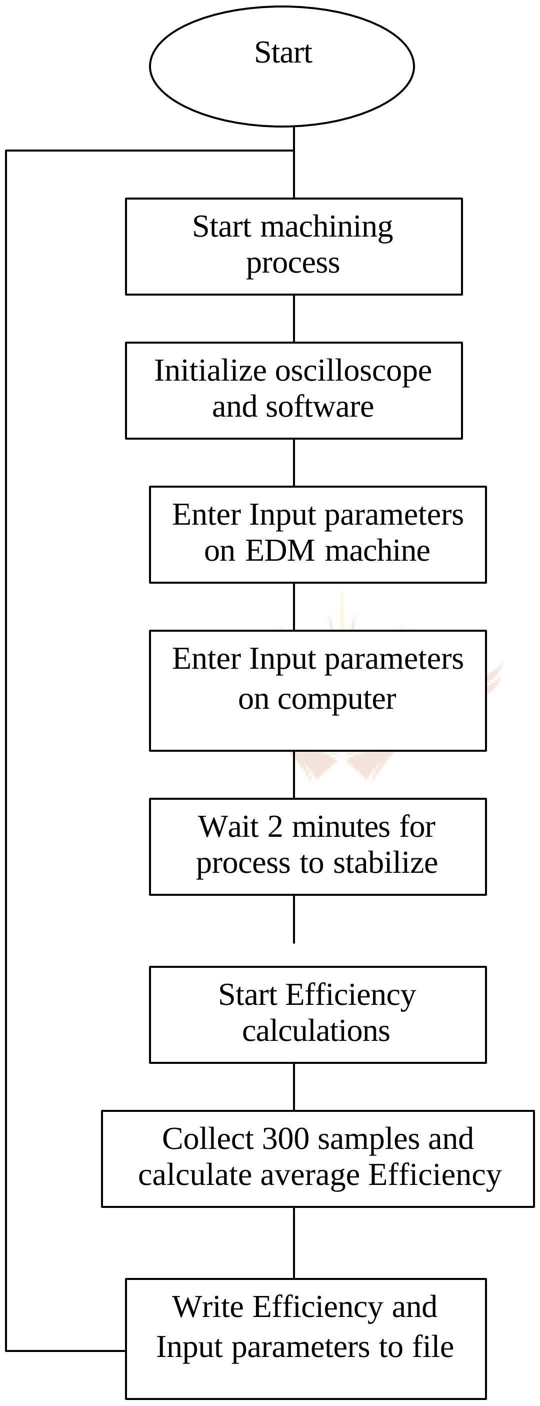

Start
Start machining process
Initialize oscilloscope and software
Enter Input parameters on EDM machine
Enter Input parameters on computer
Wait 2 minutes for process to stabilize
Start Efficiency calculations
Collect 300 samples and calculate average Efficiency
Write Efficiency and Input parameters to file

Figure 5.1 Experimental setup flow diagram

5.3 Experimental tests

Experimental results and plots are presented in Appendix A and B

Test 1

The aim of these tests was to find the influence of an increase in current combined with the influence of the increase of the frequency. Four current changes were made and for each current setting, the on and off time were increased through the range of the EDM machine

The results showed an increase in efficiency for higher and lower current settings and a drop in efficiency for higher frequency settings.

Test 2

The aim of these experiments was to find the influence of the electrode diameter on the efficiency. Four different electrodes were utilized ant three different current settings.

The results shows optimum efficiency when 10mm diameter electrodes are used. For increased current, a larger diameter electrode provides increased efficiency.

Test 3

The aim of these experiments is to find the influence of the off time on the process efficiency. The on time is kept constant at 42μs, while the off time is changed at different current settings.

The process shows the best efficiency at an intermediate current setting. The off time has different influences on different current settings, but it can be said that at the optimum current setting, a higher off time produces a better efficiency.

Test 4

The aim of these experiments is to find the influence of the Duty ratio on the process efficiency. The off time is kept constant at 42µs, while the on time is changed at different current settings.

Test 5

The aim of these experiments is to find the influence of the on time on the process efficiency at a lower off time of 24 µs. The off time is kept constant at 24µs, while the off time is changed at different current settings.

5.4 Neural network training

Once all the experimental results was fed to the Excel spreadsheet, the neural network was trained with the following neural network settings

Input parameters: 5

Output parameters: 1

Training steps: 20 000

Hidden layers: 3

1st hidden layer: 5

2nd hidden layer: 6

Random seed: 0,8

These parameters were found by iteration, until the best training result was achieved

Chapter 6

Process efficiency diagnosis

The former remote analysis could present the process status in terms of parameter values and relationships to its performance. An additional analysis or investigation into the cause or nature of such status results in a process diagnosis. It aims at obtaining final decision on process status in relation to its performance or its possible optimization procedure. It could recognize by “signs or symptoms”, the difference between the present status and desired situation

6.1 Diagnosis criteria

When diagnosing a process with many independent variables, it is crucial that the criteria for a diagnosis be well defined. In this work, all focus is set on the process efficiency.

What must be taken into account is the statistical variation of the process efficiency. This means that the process is by definition, within a certain limit, unstable. By what amount, is important when setting limits or thresholds for diagnosis.

It was decided that the diagnosis would be set at 10% of acceptable values. An illustration of this is given in fig 6.1. If the process is performing outside the 10% threshold, further steps can be taken by methods of discrete parameter optimization.

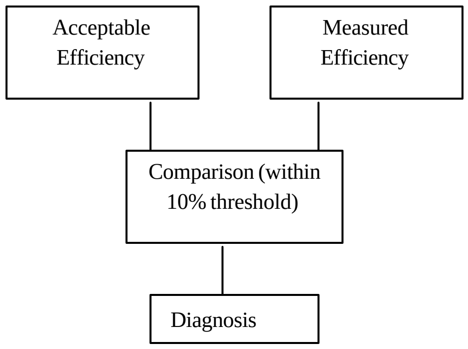


Figure 6.1. Diagnosis concept

6.2 Incorporation of neural network prediction

The diagnosis decision to be made is based on 10% of acceptable values. The acceptable machining level can be set. In this work, an acceptable efficiency of 60% is defined, which can be raised or lowered, depending on machining conditions.

6.3 Discrete parameter optimization

Once it is determined that the process is running outside the required 10% efficiency level, further steps have to be taken to remedy the situation. This means that the input parameters have to be altered to improve the situation.

The neural network can now be utilized to discretely optimize the input parameters. By altering the input parameters one by one over the complete range of its operation, the output of the neural network, which represents the process efficiency, is altered. Optimal values for each separate optimization can be stored, and these are combined for optimization recommendations.

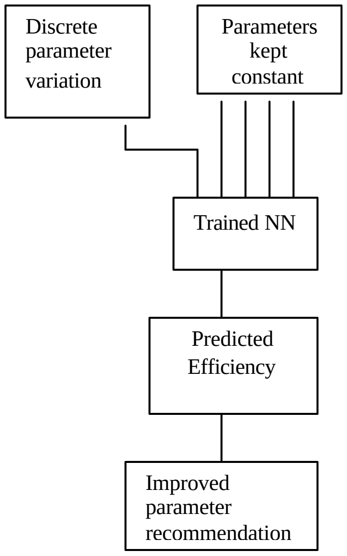

Discrete parameter variation
Parameters kept constant
Trained NN
Predicted Efficiency
Improved parameter recommendation

Figure 6.2 Discrete parameter optimization

6.4 Software design

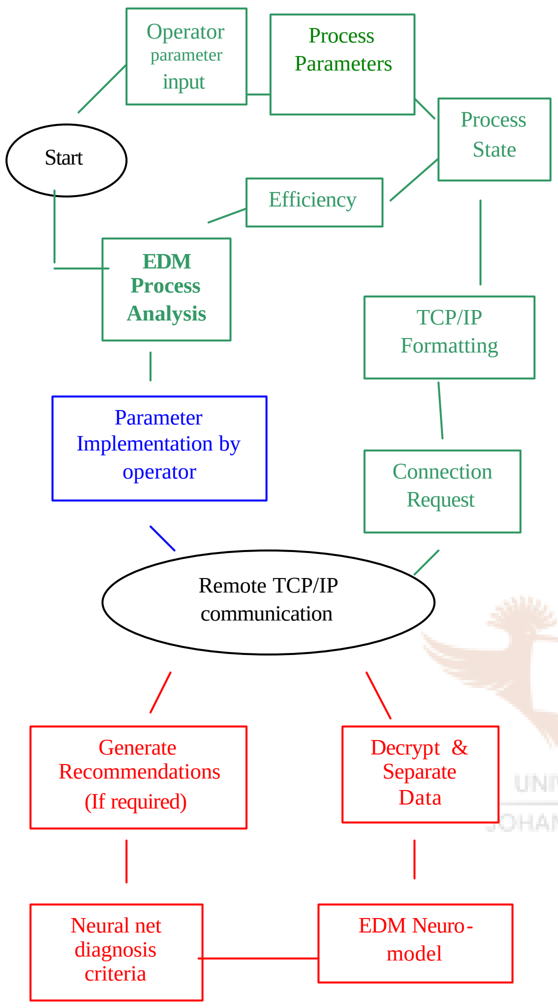


Figure 6.3 Flow diagram of final system design

6.4.1 Client interface

The interface is divided into tabbed windows. Each tab contains grouped functions.

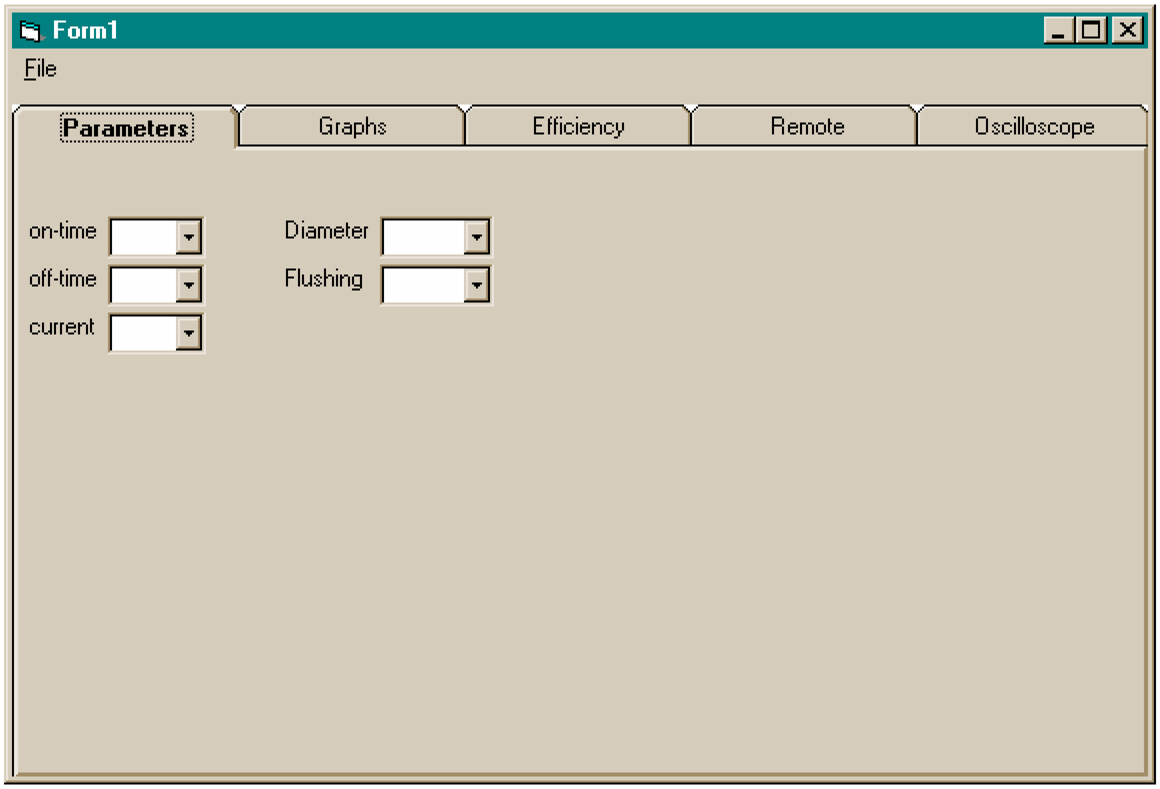
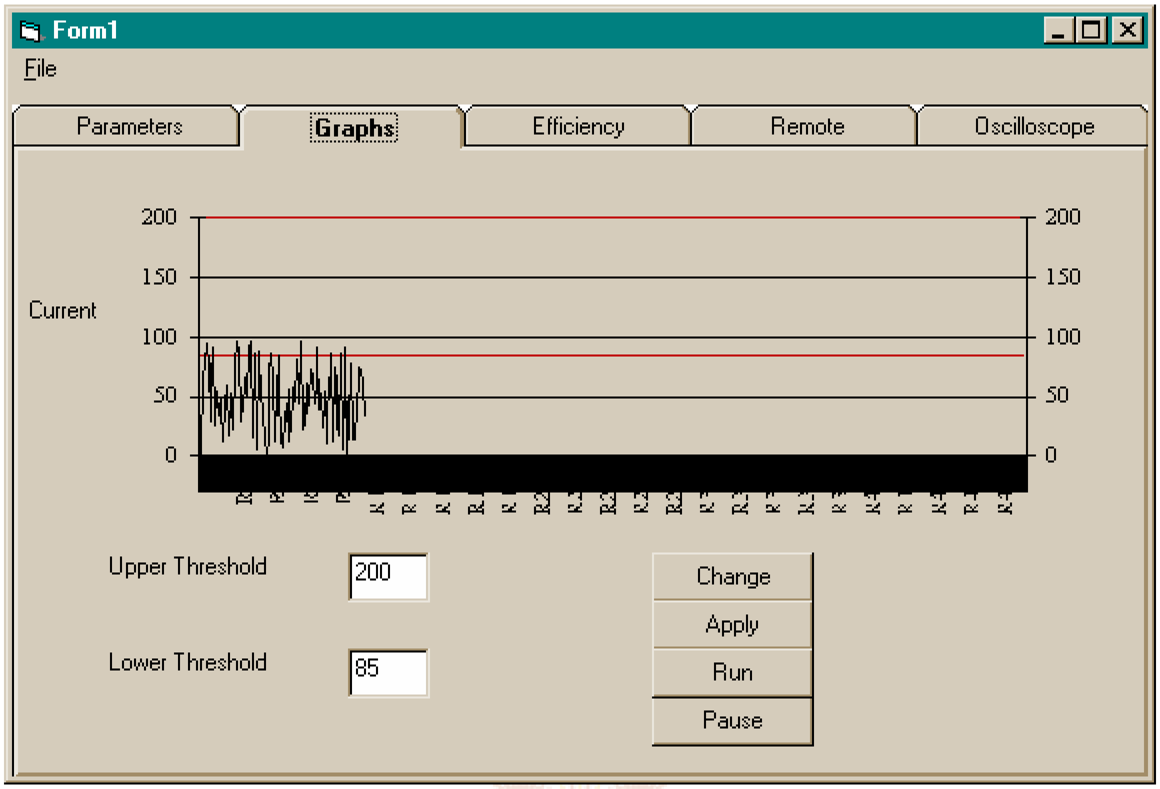


Figure 6.4 User input tab

This tab contains the user input for the process parameters. Drop down text boxes contains setting relevant to the EDM machine.



Upper Threshold 200

Lower Threshold 85

Figure 6.5 oscilloscope output tab

The Graphs tab contains the oscilloscope output. On this graph, all the thresholds can be set and the efficiency calculations fine tuned.

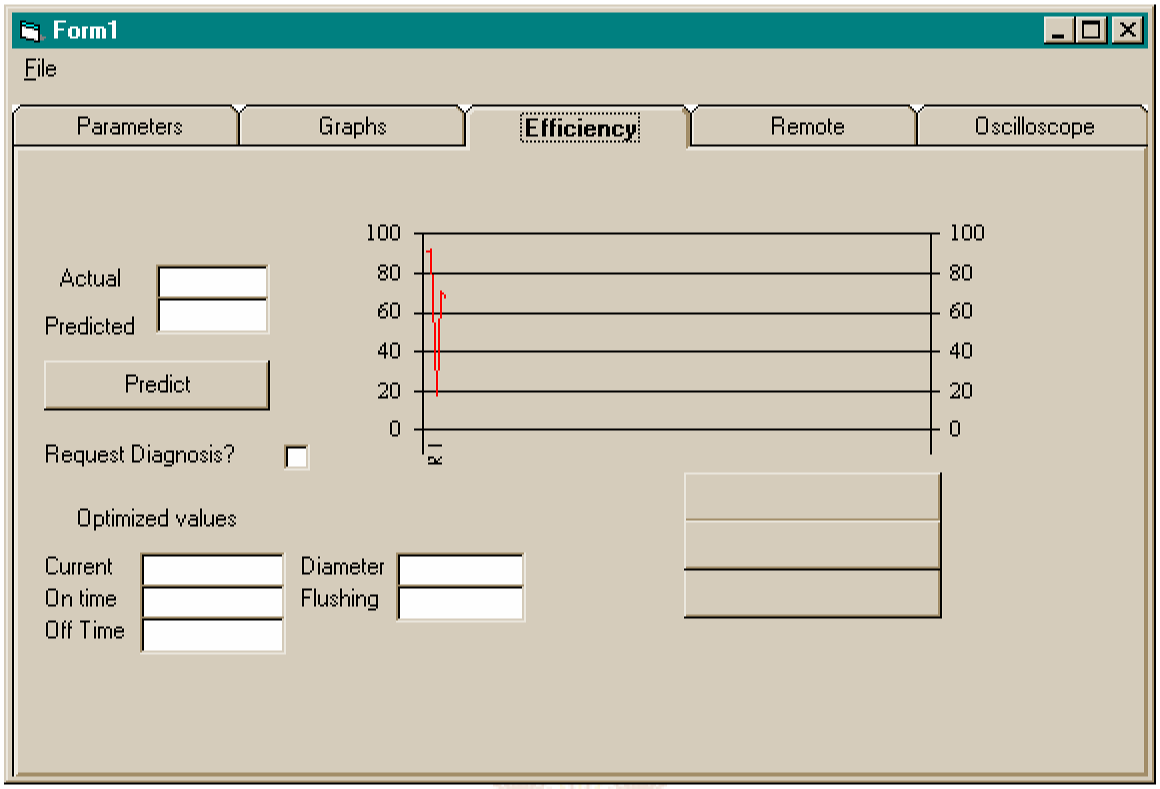


Figure 6.6 Efficiency tab

The efficiency tab is the main process monitoring tab. On this tab, the efficiency as calculated from the oscilloscope is displayed on a time dependant scale. Predicted values as well as the actual values are displayed as is communicated from the program itself and from the remote server. When a diagnosis is requested, the optimizes values are displayed in the appropriate text boxes.

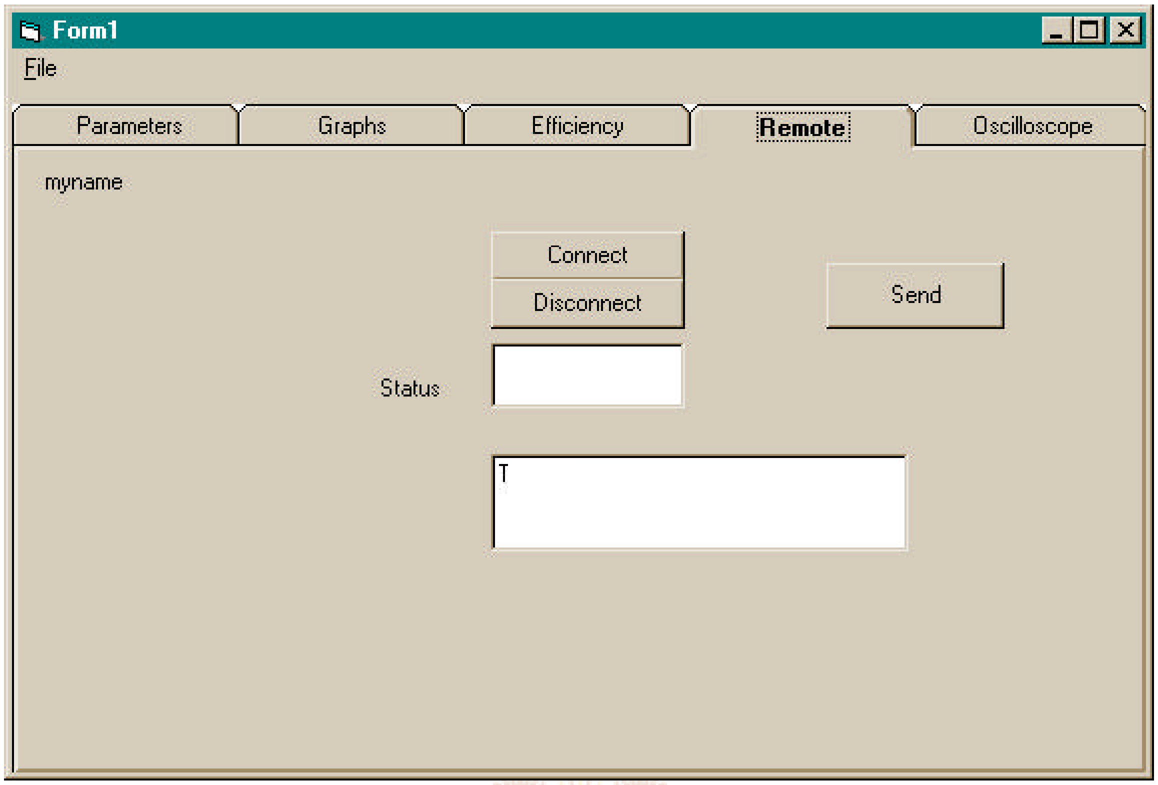
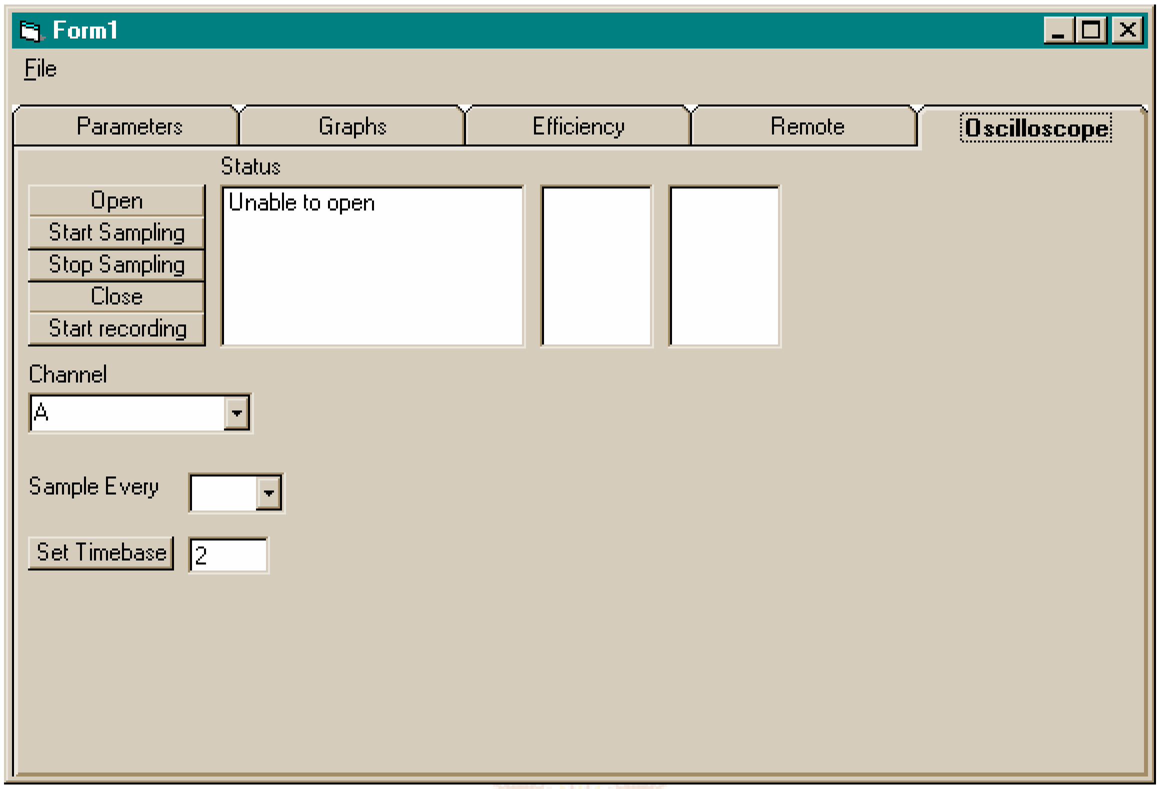


Figure 6.7 Client – Server tab

This tab contains the Remote Client server information and from this page, the internet connection is initialized and can also be stopped



Status

Unable to open

Channel: A

Set Timebase: 2

Figure 6.8 Oscilloscope interface

This page is the interface to the oscilloscope. Form here sampling rates, time base, when to start or stop and oscilloscope status is controlled.

6.4.2 Server interface

The server interface is an Excel module. This means that it is part of an Excel sheet, and can only function in an Excel sheet. The module is activated by pressing the button on the excel sheet. From this interface it is possible to start the internet server, receive process data and predict the process efficiency by use of the trained neural network that is already running in the excel sheet.

The on time, off time and current for the process can be optimized to frequency by pressing the buttons. When a diagnosis request is made, these optimizations are done automatically, and no user intervention is necessary.

The TCP software has also been designed so that only one client can connect at one stage. This can be changed to accept more than one request at a time.

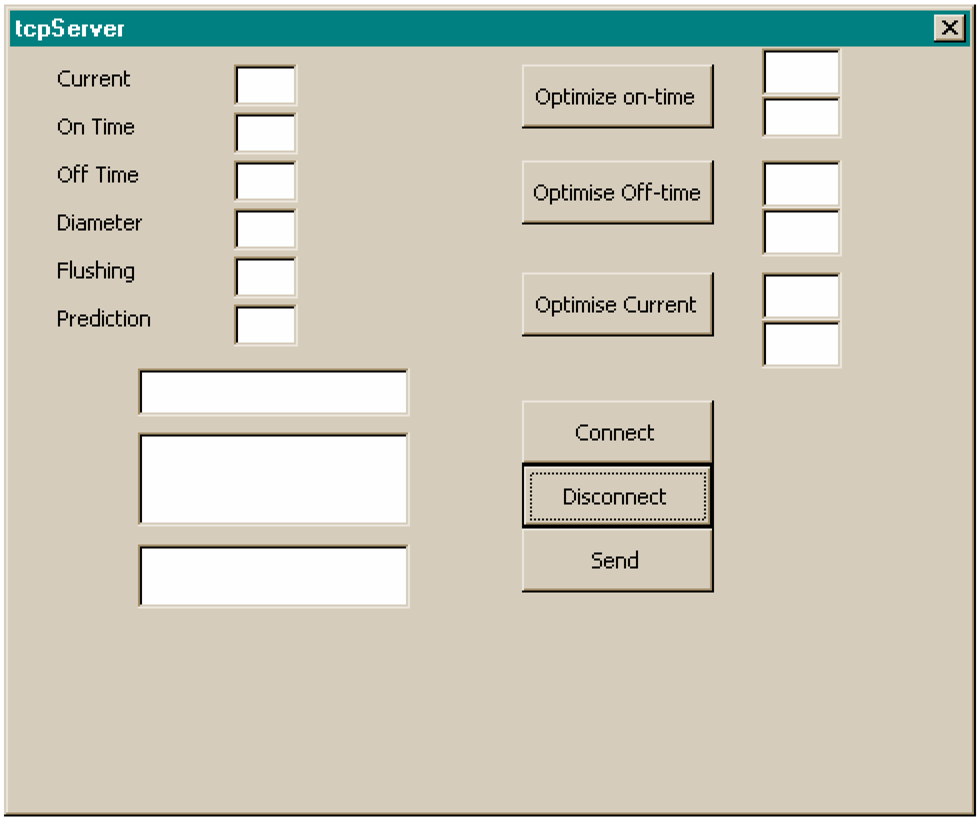


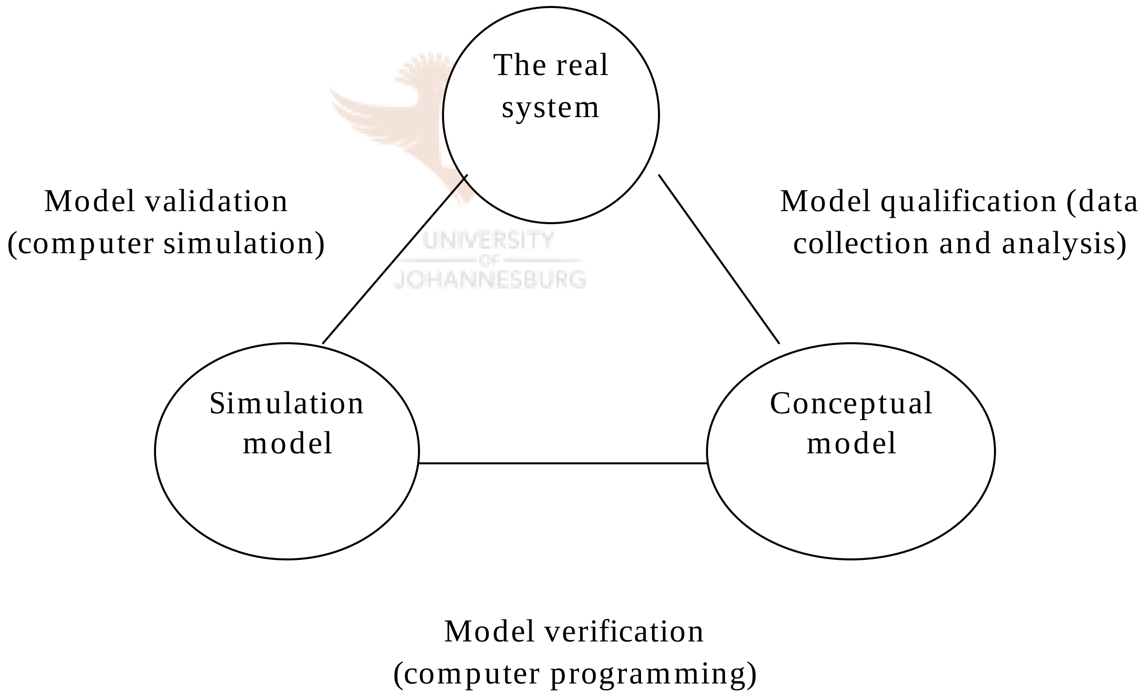
Figure 6.9 Server Interface

Chapter 7

System integration and Validation of Monitoring and Diagnosis System

7.1 Qualification, verification and validation

The model development process should follow a framework, which may be graphically presented in figure 7.1 [7]



The real system
Model validation (computer simulation)
Model qualification (data collection and analysis)
Simulation model
Conceptual model
Model verification (computer programming)

Figure 7.1 Model development framework

Model qualification is the procedure of determining the structure, elements and data of the model, which are required to provide an acceptable level of agreement with the actual system. Model verification involves ensuring the proper functioning of the model developed for the system. This is mainly concerned with the correctness of the model itself. For example, if the model concerned involves computer programming tasks, then it must be made certain that the routines are programmed correctly, and that they function exactly according to the logic as designed. Finally, model validation is concerned with determining whether the model developed is indeed and accurate representation of the real system of interest.

Qualification

The qualification stage of the model involves the activities of data collection, data clarification and parameter calculation, and the analysis of the system structure and functions.

Model verification and validation

Model verification and validation are both concerned with the problem of trying to determine whether the model is accurate. Model verification is basically concerned with the proper construction of the model itself. In computer terms this can involve repeated processes of program writing and debugging. Numerous repeated test runs may have to be conducted to check that all the activity routines in the model function exactly according to the logic defined by their flow charts and that these elements of the model have been properly embodied within the overall modeling frame through the executive control routines, so that they affect the system states in the manner expected. Model validation is concerned with comparing the model to the real system

7.2 Neuro macro EDM model

Figure 7.1 shows the experimental results and the neural network prediction results. The network will predict an output for a set of input with a n average deviation of 3.88% and a maximum deviation of 11.9%.

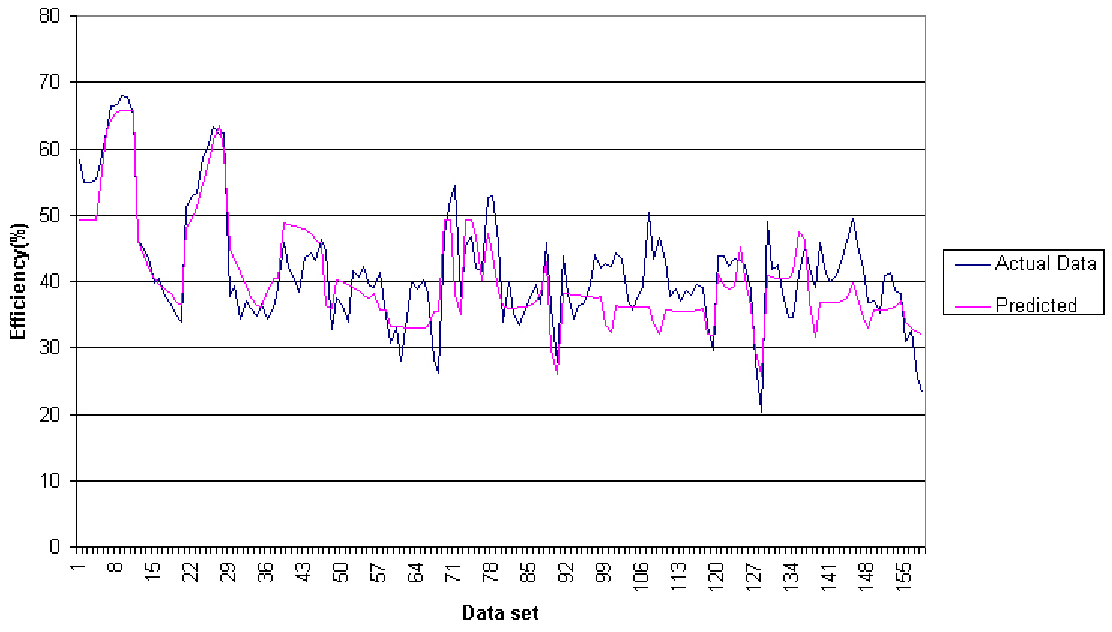


Figure 7.2 Neuro macro model training results

7.3 Discrete parameter optimizations

The parameter optimizations are predicted within the above neuro macro EDM model. Any prediction given is approximates by the neural network. The discrete parameter optimization will provide the EDM operator with improved machining conditions based on these predictions.

7.5 Summary of working system

The system works as a manufacturing environment integrated with a remote process model with the capability of process diagnosis based on an EDM macro model. The EDM operator has a visual interface of the process efficiency performance. The interface gives full control of the monitoring system and allows for fine-tuning of the system.

Once the operator is satisfied that the monitoring system is functioning properly and that the process is stabilized, he can request a process diagnosis remotely. This is done by connecting to the remote server and sending a diagnosis request with the relevant data. The server will automatically run the diagnosis and send a recommended set of process improvements to the operator.

The operator can repeatedly request a diagnosis until he is satisfied that the system has improved to a desired situation.

Chapter 8

Conclusion

The remote diagnosis of the EDM process results in an integrated and remote manufacturing environment, with a centralized database of machining conditions and experience. A trained neural model of the EDM process improves machining by diagnosis of the process state based on empirically gained experience on process performance

The final EDM neuro macro model proved to be consistent with empirical results and the objectives set to improve process control was achieved. Predictions made by the neural network are accurate enough to give a diagnosis of whether the process state is satisfactory, and to provide a diagnosis with improvement recommendations.

Remote monitoring of the process allows for collection of data remotely. This gives the added capability of collecting data from more than one machine globally. More data would lead to more training data for the neural network, which would improve the neural network predictions

The focus of the work was in improving the process efficiency by ways of remote monitoring and diagnosis. The focus can be shifted to any other relevant process output, such as metal removal rate or surface finish.

By ways of experimentation and back propagation neural network training, it is possible to expand the system to include all relevant machining output variables. This would provide a parameter selection system by which a combination of outputs can be optimized relevant to the input parameters.

The objectives set for the thesis as described in section 1.1 was achieved and a working system was created which incorporates all of the aspects focused on in this work.

The main process characteristics were identified and a set of relevant parameters isolated. A method for monitoring and analysis of these parameters were determined and implemented.

The approach followed for improvement of process control was through the use of the neural network. The neural network does not only predict output parameters, but also provide a recommendation of parameters to improve the process. This gives the operator a tool to improve the control of the EDM process.

The neural network also represents a macro model of the EDM process. It relates a set of input parameters with the output efficiency

The Internet based remote communication for process monitoring provides the addition al tool of a centralized remote database of information. The remote aspect of the system was implemented and functioned with real time data communication and analysis of data.

The working system incorporates all of the above aspects into an elegant working computer program. Diagnoses of the parameters are done online in real time using the neural network through the Internet.

References

[1] Lonardo P.M., Bruzzone A.A., 1999. Effect of Flushing and Electrode Material on Die Sinking EDM, Annals of CIRP Vol. 48/1/1999

[2] R. Snoes, F. van Dyck, Investigations of EDM Operations by Means of Thermo-Mathematical Models. Annals of CIRP Vol 20/1/1977

[3] Boccadoro, M., Dauw D.F. 1995, About the Application of Fuzzy Controllers in High-Performance Die-Sinking EDM Machines, Annals of CIRP Vol. 44/1/1995

[4] Rajurkart K.P. and Wang W.M., New model reference adaptive control system for EDM Anals of CIRP vol 38/1 1989 pp 183-186

[5] Katz Z, Vermaak J. An analysis of the electro discharge machining process through remote monitoring

[6] Katz Z, Naude J.J. (1999) A neural network/expert system based system approach for design improvement of products manufactured by EDM. JMSE Vol 121 August 1999

[7] G and Rajurkar K.P, Artificial Neural network approach in modeling of EDM Indurkhya, Proceedings of the artificial Neural network in engineering vol2. 1992 pp 845-851

[8] B. WU, Manufacturing systems design and analysis, second edition

[9] Gary F Benedict ,Nontraditional manufacturing processes

[10] URembold, B.O.Nnanji, A. Storr, Computer integrated manufacturing and engineering –

[11] Rajurkar K.P. Wang W.M., Improvements of EDM performance with advanced monitoring and control systems Journal of manufacturing science and engineering v119 n04B Nov 1997 p 770-5

[12] Uhlmann E, Fies E, 1999, Intelligent Process Monitoring, Production Engineering Vol. VI/2(1999)

[13] Teach yourself TCP/IP in 14 days – Second edition, Sams publishing.

[14] Alianna Maren, Craig Harston, Robert Pap. Handbook of neural computing applications

[15] Electro discharge machining program – Defense documentation center for scientific and technical information

[16] Wang W.M. and Rajurkar K.P, Monitoring, Modelling and control of EDM ASME PED vol 44, 1990 pp 393 – 406

[17] Uhlmann E, Fries E, 1999, Intelligent Process Monitoring, Production Engineering Vol VI/2(1999)

[18] Peter S. Vail. Computer integrated manufacturing

Internet resources:

[19] www.oxxford.com

Appendix A Experimental results

Current	On time	Off time	Electrode Diameter	Flushing	Efficiency
5.6	4.2	4.2	10	1	58.25
5.6	5.6	5.6	10	1	55.01
5.6	7.5	7.5	10	1	54.95
5.6	10	10	10	1	55.28
5.6	13	13	10	1	58.04
5.6	18	18	10	1	62.56
5.6	24	24	10	1	66.47
5.6	32	32	10	1	66.69
5.6	42	42	10	1	67.99
5.6	75	75	10	1	67.86
5.6	100	100	10	1	65.15
11	7.5	7.5	10	1	46.208
11	10	10	10	1	45.02
11	13	13	10	1	43.43
11	18	18	10	1	39.88
11	24	24	10	1	40.41
11	32	32	10	1	37.78
11	42	42	10	1	36.55
11	75	75	10	1	35.12
11	100	100	10	1	34.01
14.2	7.5	7.5	10	1	51.34
14.2	10	10	10	1	52.85
14.2	13	13	10	1	53.4
14.2	18	18	10	1	58.48
14.2	24	24	10	1	60.77
14.2	32	32	10	1	63.36
14.2	42	42	10	1	61.42
14.2	75	75	10	1	62.1
14.2	100	100	10	1	62.38
5.6	42	4.2	10	1	37.69
5.6	42	5.6	10	1	39.28
5.6	42	7.5	10	1	34.39

5.6	42	10	10	1	37.07
5.6	42	13	10	1	35.82
5.6	42	18	10	1	34.88
5.6	42	24	10	1	36.52
5.6	42	32	10	1	34.27
5.6	42	75	10	1	35.94
5.6	42	100	10	1	39.04
9.4	42	4.2	10	1	45.96
9.4	42	5.6	10	1	42.3
9.4	42	7.5	10	1	40.25
9.4	42	10	10	1	38.36
9.4	42	13	10	1	43.64
9.4	42	18	10	1	44.36
9.4	42	24	10	1	43.27
9.4	42	32	10	1	46.28
9.4	42	75	10	1	44.6
9.4	42	100	10	1	32.78
11	42	4.2	10	1	37.56
11	42	5.6	10	1	36.44
11	42	7.5	10	1	34.01
11	42	10	10	1	41.58
11	42	13	10	1	40.69
11	42	18	10	1	42.26
11	42	24	10	1	39.24
11	42	32	10	1	39.2
11	42	75	10	1	41.32
11	42	100	10	1	34.75
14.2	42	4.2	10	1	30.79
14.2	42	5.6	10	1	33.09
14.2	42	7.5	10	1	28.09
14.2	42	10	10	1	34.51
14.2	42	13	10	1	40.02
14.2	42	18	10	1	38.84
14.2	42	24	10	1	40.28
14.2	42	32	10	1	38.29
14.2	42	75	10	1	28.22

Current	On time	Off time	Electrode Diameter	Flushing	Efficiency
14.2	42	100	10	1	26.32
5.6	24	5.6	8	1	47.19
5.6	24	5.6	10	1	52.4
5.6	24	5.6	14	1	54.36
5.6	24	5.6	20	1	37.79
9.4	24	5.6	8	1	45.5
9.4	24	5.6	10	1	46.81
9.4	24	5.6	14	1	41.93
9.4	24	5.6	20	1	41.61
11	24	5.6	8	1	52.73
11	24	5.6	10	1	52.89
11	24	5.6	14	1	45.61
11	24	5.6	20	1	33.88
5.6	4.2	42	10	1	39.98
5.6	5.6	42	10	1	35.13
5.6	7.5	42	10	1	33.38
5.6	10	42	10	1	36.01
5.6	13	42	10	1	37.75
5.6	18	42	10	1	39.47
5.6	24	42	10	1	36.72
5.6	32	42	10	1	45.78
5.6	75	42	10	1	36.13
5.6	100	42	10	1	27.77
9.4	4.2	42	10	1	43.86
9.4	5.6	42	10	1	38.07
9.4	7.5	42	10	1	34.33
9.4	10	42	10	1	36.42
9.4	13	42	10	1	36.88
9.4	18	42	10	1	39.56
9.4	24	42	10	1	44.18
9.4	32	42	10	1	42.02
9.4	75	42	10	1	42.72
9.4	100	42	10	1	42.23
11	4.2	42	10	1	44.24
11	5.6	42	10	1	43.33

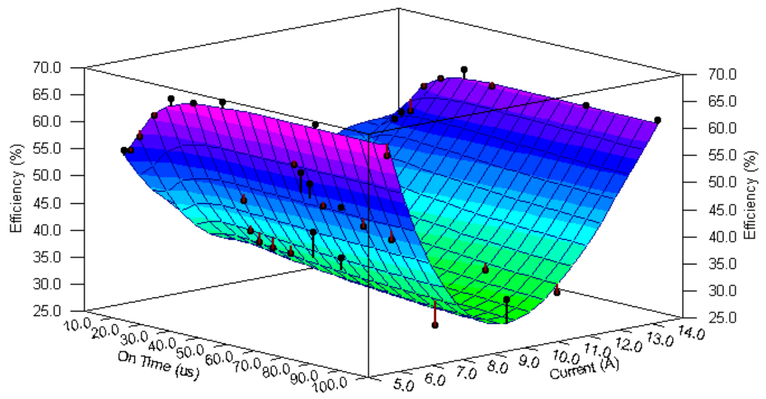
Current	On time	Off time	Electrode Diameter	Flushing	Efficiency
11	7.5	42	10	1	37.35
11	10	42	10	1	35.71
11	13	42	10	1	38.05
11	18	42	10	1	39.33
11	24	42	10	1	50.43
11	75	42	10	1	43.44
11	100	42	10	1	46.6
14.2	4.2	42	10	1	43.16
14.2	5.6	42	10	1	37.66
14.2	7.5	42	10	1	38.78
14.2	10	42	10	1	37.05
14.2	13	42	10	1	38.7
14.2	18	42	10	1	38.03
14.2	24	42	10	1	39.56
14.2	32	42	10	1	39.08
14.2	75	42	10	1	33.43
14.2	100	42	10	1	29.64

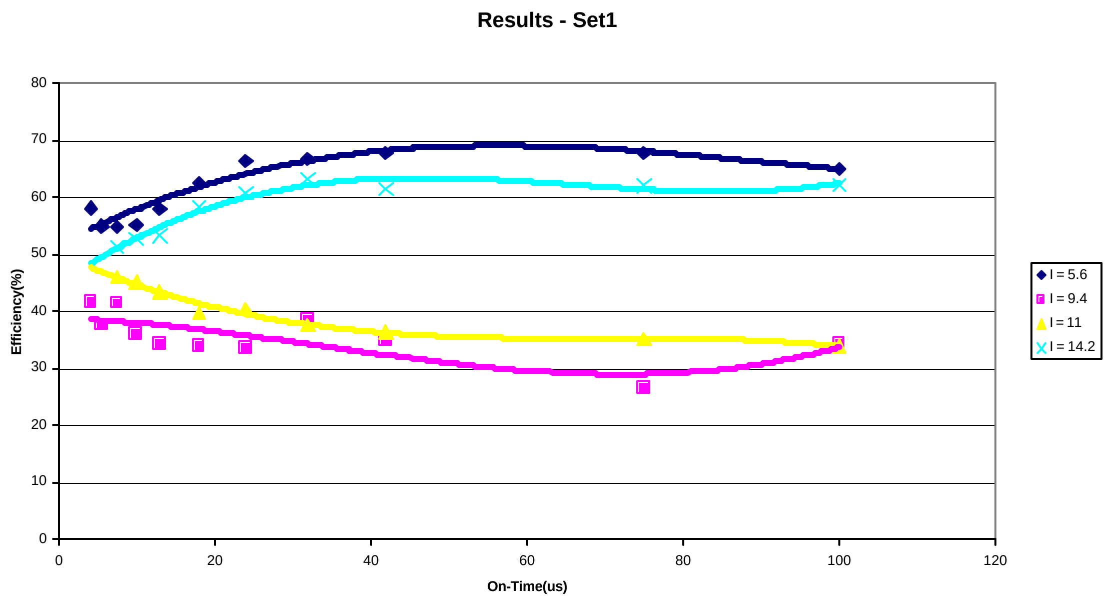
Appendix B Experimental result plots

Test 1

Electrode diameter = 10mm

Duty ratio = 0.5

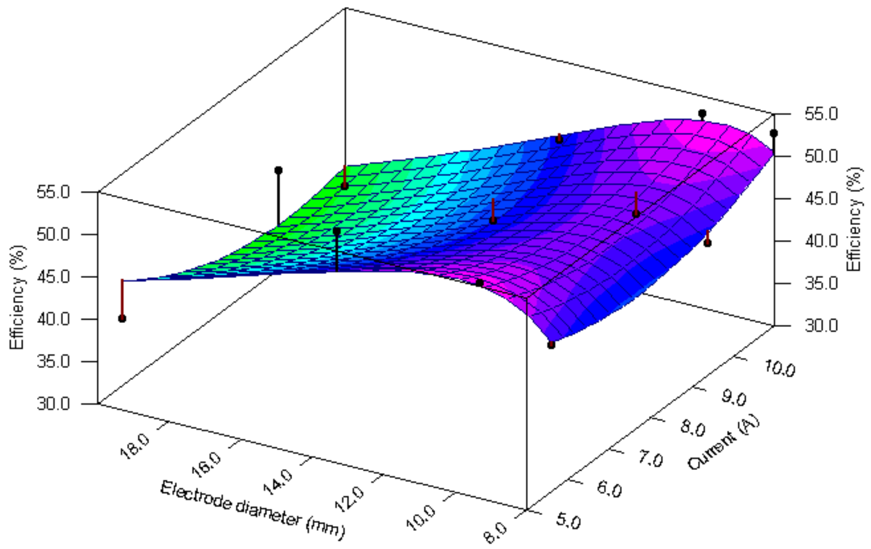


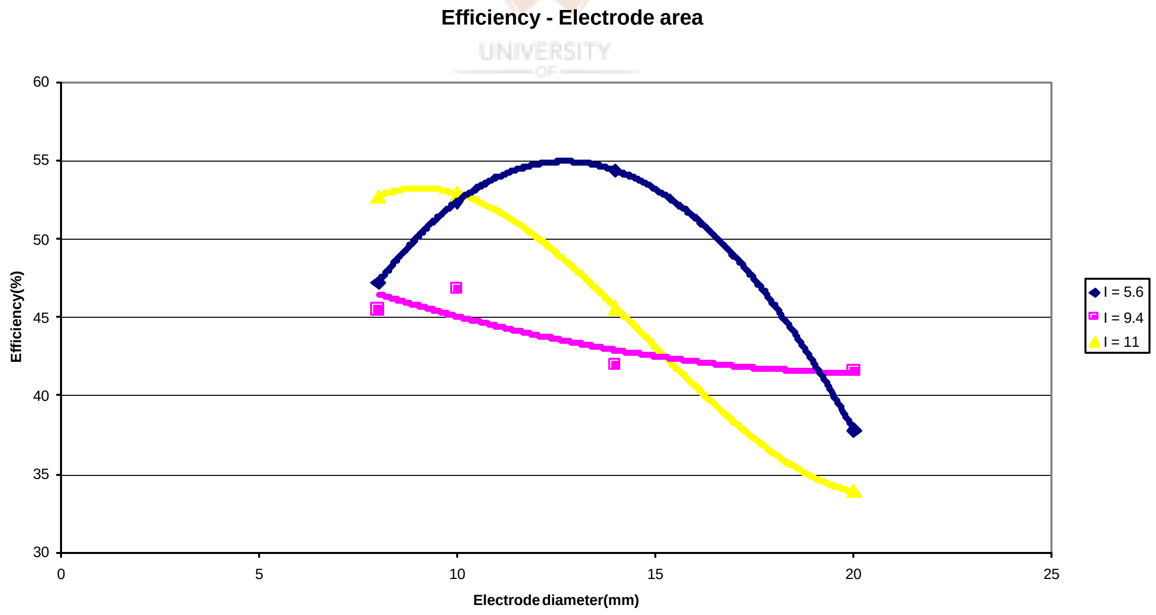


Test 2

On time = 10μs

Off time = 42μs

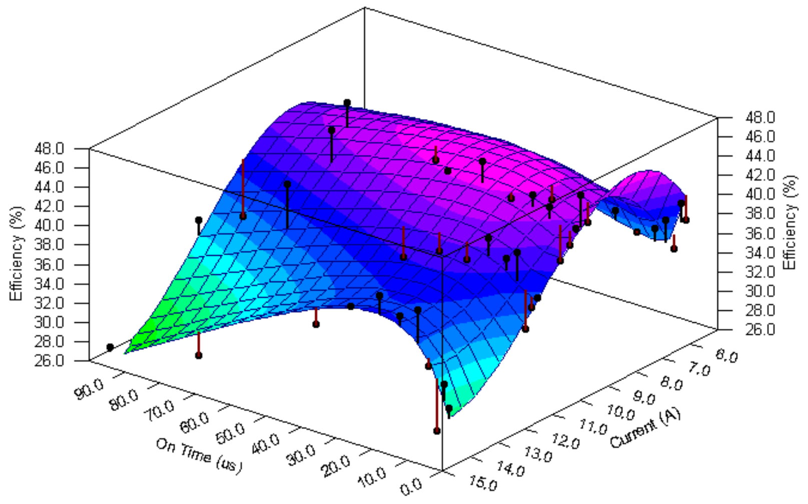


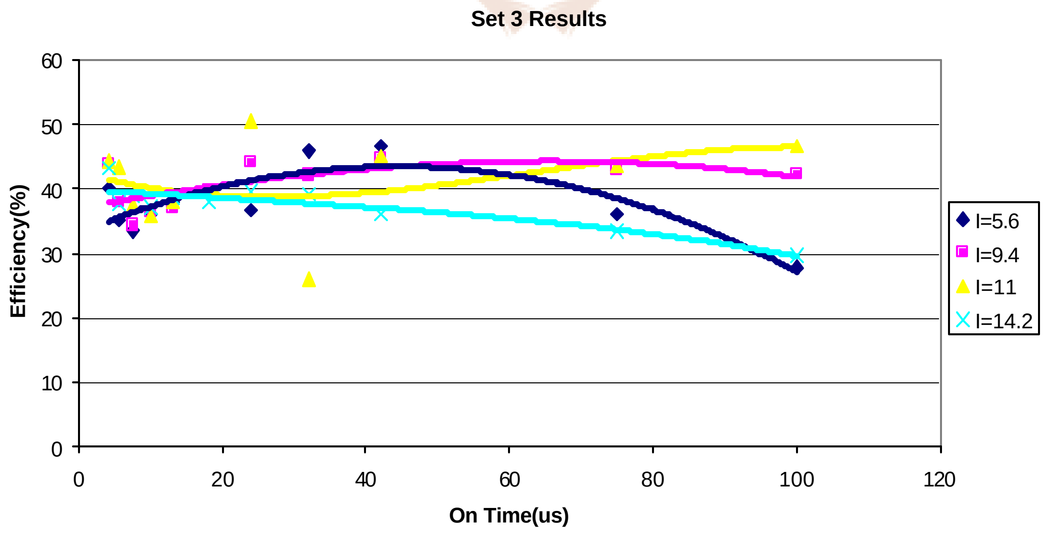


<u>Test 3</u>

Electrode diameter = 10mm

On time = 42μs

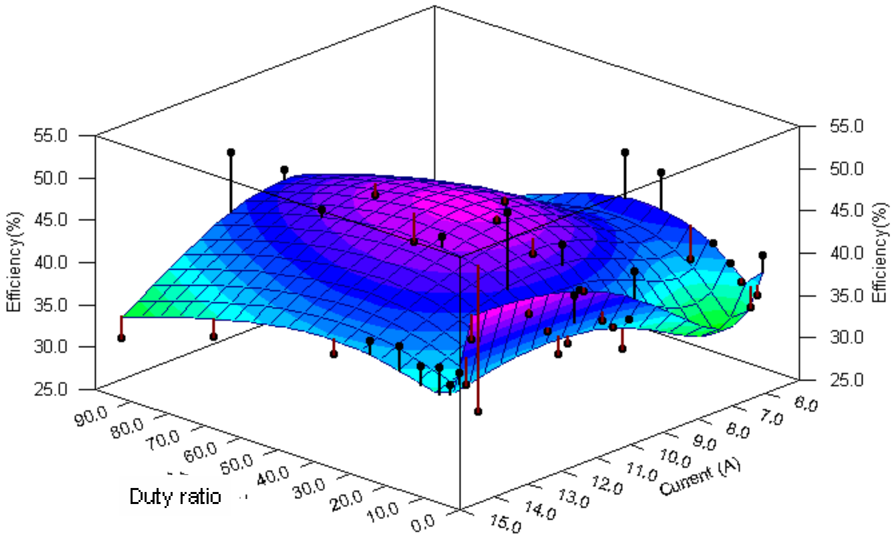


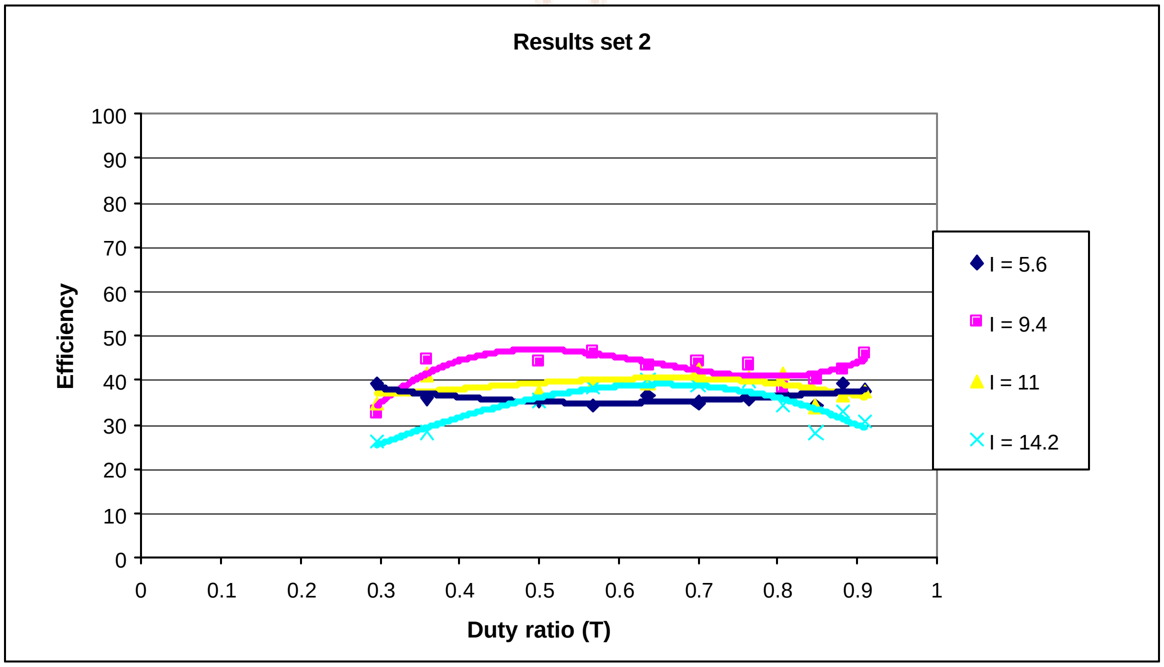


Test 4

Off time = 42μs

Electrode diameter=10mm

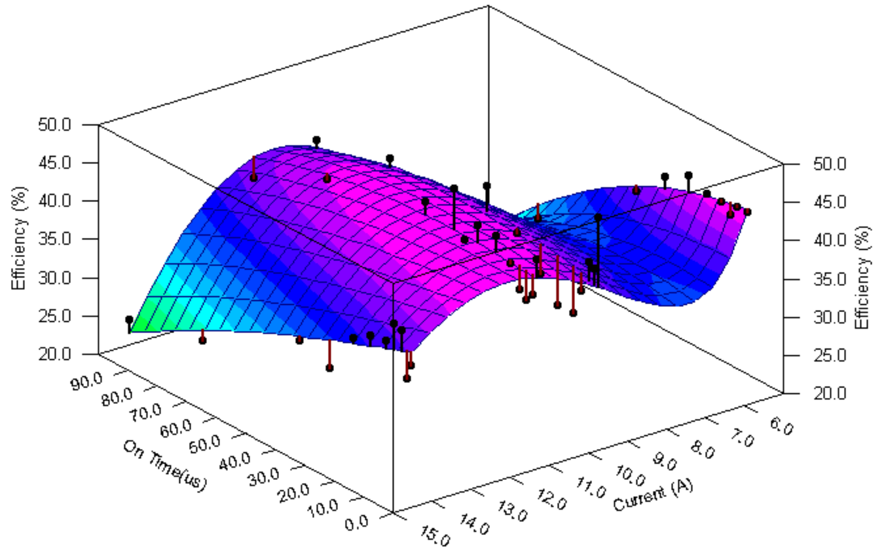




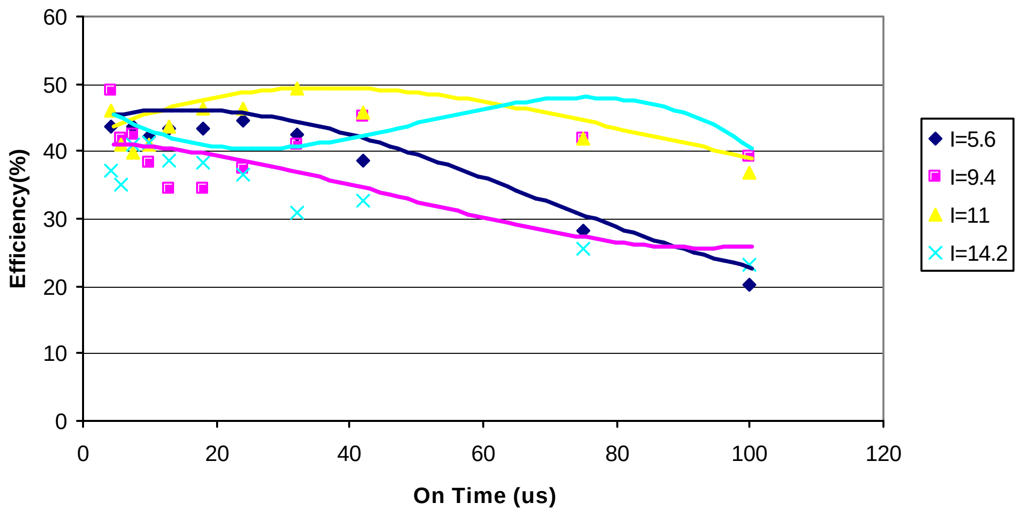
Test 5

Off time = 24μs

Electrode diameter = 10mm



Set 4-2



Appendix C Programming

Server Side

```
Option Explicit
Dim strData As String
Dim myname As String
Dim reqef As Double
Private isConnectedFlag As Boolean
Private sClientChatName As String   'holds the name of connected user
Const msgTitle As String = "VBnet Winsock Chat Server Demo"
Private Sub cmdOptOn_Click()
On Error GoTo errHandler

Dim k As Double
Dim x As Range
Dim efpred As Double
Dim maxef As Double
For k = 10 To 100 Step 1
txtOn.Text = k
Range("i8").Select
Set x = ActiveCell
x.Offset(0, 0) = txtCurr.Text
x.Offset(0, 1) = Format(txtOn.Text, "###,##")
x.Offset(0, 2) = txtOff.Text
x.Offset(0, 3) = txtDia.Text
x.Offset(0, 4) = txtFlush.Text
Range("n8").Select
txtPredict.Text = Format(ActiveCell.Value, "##.##")
efpred = txtPredict.Text
If efpred > maxef Then maxef = efpred: maxOn = k
Next k
maxEff.Text = maxef
maxOn.Text = maxOn
Exit Sub

errHandler: MsgBox "Invalid data, Please re-enter"
End Sub
```

```
Private Sub cmdConnect_Click()
'This method first assures that the server
'Winsock control is closed, then assigns a
'port to control's LocalPort for use as the
'server (the port specified must be numeric -
'it can not be a friendly name from the system's
'services file. Once assigned, invoke the
'Winsock Listen method.
tcpServer.Close
tcpServer.LocalPort = 1544
tcpServer.Listen
'If the connection was successful, the control's
'state wil be 'sckListening'
If tcpServer.State = sckListening Then
Me.Caption = "TCP Server : Listening"
cmdDisconnect.Caption = "Stop Listening"
cmdDisconnect.Enabled = tcpServer.State = sckListening
cmdConnect.Enabled = tcpServer.State = sckClosed
End If
'if there was an error in the connection,
'display it in the txtErr box
txtErr.Text = Err.Description
End Sub

Private Sub cmdDisconnect_Click()
If tcpServer.State = sckListening Or _
tcpServer.State = sckConnected Then
tcpServer.Close
isConnectedFlag = tcpServer.State = sckConnected
Me.Caption = "TCP Server Closed"
cmdDisconnect.Enabled = isConnectedFlag = True
cmdConnect.Enabled = isConnectedFlag = False
End If
End Sub

Private Sub cmdSend_Click()
Call TransmitMessage
End Sub

Private Sub Form_Load()
txtErr.Text = ""
txtSend.Text = ""
txtReceive.Text = ""
```

```
myname = "server"
'Label2.Caption = myname

End Sub
Private Sub Form_Unload(Cancel As Integer)

tcpServer.Close

Set frmServer = Nothing
End Sub

Private Sub CommandButton2_Click()
On Error GoTo errHandler

Dim k As Double
Dim x As Range
Dim efpred As Double
Dim efoffpred As Double
Dim maxoffef As Double
For k = 10 To 100 Step 1
txtOff.Text = k
Range("i8").Select
Set x = ActiveCell
x.Offset(0, 0) = txtCurr.Text
x.Offset(0, 1) = txtOn.Text
x.Offset(0, 2) = Format(txtOff.Text, "###,##")
x.Offset(0, 3) = txtDia.Text
x.Offset(0, 4) = txtFlush.Text
Range("n8").Select
txtPredict.Text = Format(ActiveCell.Value, "##.##")
efpred = txtPredict.Text
If efpred > maxoffef Then maxoffef = efpred: efoffpred = k
'MsgBox "wait!!!"
Next k
txtmaxoffeff.Text = maxoffef
txtmaxoff.Text = efoffpred
Exit Sub

errHandler: MsgBox "Invalid data, Please re-enter"
End Sub
```

```
Private Sub CommandButton3_Click()
On Error GoTo errHandler

Dim k As Double
Dim x As Range
Dim efpred As Double
Dim efcurrpred As Double
Dim maxcurr As Double
Dim maxcurref As Double
For k = 6 To 14
txtCurr.Text = k
Range("i8").Select
Set x = ActiveCell
x.Offset(0, 0) = txtCurr.Text
x.Offset(0, 1) = txtOn.Text
x.Offset(0, 2) = txtOff.Text
x.Offset(0, 3) = txtDia.Text
x.Offset(0, 4) = txtFlush.Text
Range("n8").Select
txtPredict.Text = Format(ActiveCell.Value, "##.##")
efpred = txtPredict.Text
If efpred > maxcurref Then maxcurref = efpred: efcurrpred = k
'MsgBox "wait!!!"
Next k
txtmaxcurreff.Text = maxcurref
txtmaxcurr.Text = efcurrpred
Exit Sub
errHandler: MsgBox "Invalid data, Please re-enter"
End Sub

Private Sub tcpServer_Close()


'we need this flag check as showing a
'msgbox in this event will cause the
'event to fire again on closing the
'msgbox, causing an endless loop.
If isConnectedFlag = True Then
If tcpServer.State = sckClosing Then
'assure we avoid the loop
isConnectedFlag = False
'update the caption
Me.Caption = "TCP Server Closing"
```

```
'and inform the user
MsgBox "The connection to " & sClientChatName & "has been unexpectedly
terminated.", vbExclamation Or vbOKOnly, msgTitle
'close to allow reconnection
tcpServer.Close
cmdDisconnect.Enabled = isConnectedFlag
cmdConnect.Enabled = Not isConnectedFlag
End If
End If
Me.Caption = "TCP Server Closed"
End Sub
Private Sub tcpServer_ConnectionRequest(ByVal requestID As Long)


'Check if the control's State is closed. If not,
'close the connection before accepting the new
'connection.
If tcpServer.State <> sckClosed Then
tcpServer.Close
End If
'Accept the request with the requestID parameter.
tcpServer.Accept requestID
End Sub

Private Sub tcpServer_DataArrival(ByVal bytesTotal As Long)

txtReceive.Text = ""

'holds incoming data
Dim buff As String
'avoid cycles by placing the most-likely
'condition first in the If..Then statement
If isConnectedFlag = True Then
'connection is established, and isConnectedFlag
'is set, so any incoming data is part of the chat
tcpServer.GetData strData
txtReceive.Text = strData
If Len(strData) > 7 Then procparam Else diagnose
'if there is text in txtReceived, (not the
'first line received) then we need a crlf
'between lines. This also provides a place to
'preface the string with the sender's name.
'=If Len(txtReceive.Text) Then
```

```
'=buff = buff & vbCrLf & sClientChatName & " :" & vbTab & strData
'=Else
'=buff = buff & sClientChatName & " :" & vbTab & strData
'=End If
'this assigns the new string to the end of
'txtReceived, and scrolls it into view.
'=With txtReceive
'=.SelStart = Len(txtReceive.Text)
'=.Text = buff
'=.SelStart = Len(txtReceive.Text)
'=End With
'clear the user-input textbox (if desired)
'txtSend.Text = ""
Else
'set the isConnectedFlat to avoid entering
'this condition again during this session
isConnectedFlag = True
'isConnectedFlag was false, so the first data
'received from the connected client will be
'the name of the user. Save this for use when
'posting subsequent data to txtReceived.
tcpServer.GetData strData
'=sClientChatName = strData
'=Me.Caption = "TCP Server : Chatting with " & sClientChatName
'be friendly and transmit your name to the client
'=tcpServer.SendData myname
'change the caption to the disconnect button
cmdDisconnect.Caption = "Disconnect"
txtSend.SetFocus
End If
End Sub
Private Sub tcpServer_Error(ByVal Number As Integer, _
Description As String, _
ByVal Scode As Long, _
ByVal Source As String, _
ByVal HelpFile As String, _
ByVal HelpContext As Long, _
CancelDisplay As Boolean)
```

```
MsgBox "tcpServer Error: " & Number & vbCrLf & Description, _
vbExclamation Or vbOKOnly, msgTitle
CancelDisplay = True
tcpServer.Close
End Sub
'Private Sub txtSend_KeyPress(KeyAscii As Integer)
'If KeyAscii = vbKeyReturn Then
'Call TransmitMessage
'End If
'End Sub

Private Sub TransmitMessage()
Dim buff As String
'in this method, we don't want to
'first test for a valid connection
'(ie If tcpClient.State = sckConnected)
'in order to generate the appropriate
'error message to the user.
On Local Error GoTo TransmitMessage_error
tcpServer.SendData txtSend.Text
'if there is text in txtReceived, (not the
'first line received) then we need a crlf
'between lines. This also provides a place to
'preface the string with the your name.
If Len(txtReceive.Text) Then
buff = buff & vbCrLf & myname & " :" & vbTab & txtSend.Text
Else
buff = buff & myname & " :" & vbTab & txtSend.Text
End If
'assign the new string to the end of
'txtReceived, and scroll it into view.
With txtReceive
.SelStart = Len(txtReceive.Text)
.Text = buff
.SelStart = Len(txtReceive.Text)
End With
'clear the input textbox
txtSend.Text = ""
TransmitMessage_exit: Exit Sub
TransmitMessage_error: Select Case Err
Case sckBadState:
MsgBox Err.Description & "." & vbCrLf & _
"The server is not connected to a client.", _
```

```
vbExclamation Or vbOKOnly, msgTitle
Case Else
MsgBox Err.Description & ".", _
vbExclamation Or vbOKOnly, msgTitle
End Select
Resume TransmitMessage_exit
End Sub
'--end block--'


Public Sub procparam()
Dim leng As Integer
Dim k As Integer
Dim c As Integer
c = 0
leng = Len(strData)
Dim p(6) As Integer
For k = 1 To leng
If Mid(strData, k, 1) = "," Then p(c) = k: c = c + 1
Next k
txtOn.Text = Mid(strData, 1, p(0) - 1)
txtOff.Text = Mid(strData, p(0) + 1, (p(1) - p(0) - 1))
txtCurr.Text = Mid(strData, p(1) + 1, (p(2) - p(1) - 1))
txtDia.Text = Mid(strData, p(2) + 1, (p(3) - p(2) - 1))
txtFlush.Text = Mid(strData, p(3) + 1, (p(4) - p(3) - 1))
backpredict
End Sub

Public Sub backpredict()
Dim x As Object
On Error GoTo errHandler

Range("i5").Select
Set x = ActiveCell

x.Offset(0, 0) = txtCurr.Text
x.Offset(0, 1) = txtOn.Text
x.Offset(0, 2) = txtOff.Text
x.Offset(0, 3) = txtDia.Text
x.Offset(0, 4) = txtFlush.Text
Range("n5").Select
txtPredict = Format(ActiveCell.Value, "##.##")
Exit Sub
```

```
errHandler: MsgBox "Invalid data, Please re-enter"
tcpServer.SendData (txtPredict.Text)
End Sub

Private Sub UserForm_Click()

End Sub

Public Sub diagnose()
reqef = Val(strData)
If reqef > 40 Then MsgBox "Acceptable" Else MsgBox "Not acceptable"
End Sub
```

Client Side

```
Option Explicit
Dim myname As String
Dim isConnectedFlag As Boolean
Dim sClientChatName As String
Dim strData As String
Const msgtitle As String = "VBnet Winsock Chat Client "

Private Sub Check1_Click()
reqflag = 1
End Sub

Private Sub CloseUnit_Click()
opened = False
slow_collect = False
Call adc200_close_unit(port)
List1.AddItem ("ADC-200 closed")
End Sub

Private Sub Combo1_Select_change()
channel = adc200_set_channels(Channel_Select.ListIndex)
End Sub

Private Sub cmdConnect_Click()
'The name of the Winsock control is tcpClient.
'To specify a remote host, you can use
'either the IP address (ex: "14.15.15.16") or
'the computer's "friendly" name (LocalHostName)
'as shown here.
tcpClient.RemoteHost = "localhost" '"152.106.20.219"
tcpClient.RemotePort = 1544
'call the Connect method to open a connection.
'If the call fails, the tcpClient_Error event will fire
tcpClient.Connect
cmdConnect.enabled = tcpClient.State = sckClosed
connect_exit:   Exit Sub
End Sub

Private Sub cmdDisconnect_Click()
If tcpClient.State = sckConnected Then
   tcpClient.Close
```

```
    isConnectedFlag = tcpClient.State = sckConnected
    'tcpLabel.Caption = "TCP Client closed"
    cmdDisconnect.enabled = isConnectedFlag
    cmdConnect.enabled = Not isConnectedFlag
End If
End Sub

Private Sub cmdsend_Click()
Call TransmitMessage


End Sub

Private Sub Command1_Click()
Timer1.enabled = False
End Sub

Private Sub Command11_Click()
Form1.tcpClient.SendData ("39")
End Sub

Private Sub Command10_Click()
Form1.tcpClient.SendData ("41")
End Sub

Private Sub Command2_Click()
Timer1.enabled = True
count2 = 1
Form1.MSChart1.Refresh
'For k = 1 To 100
'Form1.MSChart2.Row = k
'Form1.MSChart2.Data = ""
'Next k

End Sub

Private Sub Command3_Click()
eflag = 1
efcount = 1
End Sub

Private Sub Command4_Click()
For k = 1 To 100
```

```
MSChart2.Column = 1
MSChart2.Row = k
MSChart2.Data = k
Next k
End Sub

Private Sub Command6_Click()
lowthres = txtlow.Text
upthres = txtup.Text
For k = 1 To 500
MSChart1.Column = 2
MSChart1.Row = k
MSChart1.Data = lowthres
MSChart1.Column = 3
MSChart1.Data = upthres
Next k
End Sub

Private Sub Command7_Click()
Timer1.enabled = False
End Sub

Private Sub Command8_Click()
Timer1.enabled = True
End Sub

Private Sub Command9_Click()
timeb = Text8.Text
Call open_unit
End Sub

Private Sub Form_Load()
parflag = 0
upthres = 200
lowthres = 85
txtlow = lowthres
txtup = upthres
timeb = 2
For k = 1 To 500
MSChart1.Column = 2
MSChart1.Row = k
MSChart1.Data = lowthres
MSChart1.Column = 3
```

```
MSChart1.Data = upthres
Next k

efcount = 1
open_unit
count2 = 0
'Channel selection
Channel_Select.AddItem "A", 0
Channel_Select.AddItem "B", 1
Channel_Select.AddItem "BOTH", 2
Channel_Select.ListIndex = 0

'Remote
myname = "EDMremote"
tcpLabel.Caption = "myname"

End Sub


Private Sub Frame1_DragDrop(source As Control, X As Single, Y As Single)

End Sub

Private Sub openunit_Click()
Call open_unit
End Sub

Public Sub tcpClient_DataArrival(ByVal bytesTotal As Long)
'holds incoming data
Dim buff As String
'avoid cycles by placing the most-likely
'condition first in the If..Then statement
If isConnectedFlag = True Then
'connection is established, and isConnectedFlag
'is set, so any incoming data is part of the chat
Form1.tcpClient.GetData strData
Form1.txtReceived.Text = strData
If Len(strData) > 7 Then procparam
'if there is text in txtReceived, (not the
'firs
'tline received) then we need a crlf
'between lines. This also provides a place to
'preface the string with the sender's name.
```

```
'+If Len(txtReceive.Text) Then
'+buff = buff & vbCrLf & sClientChatName & " :" & vbTab & strData
'+Else
'+buff = buff & sClientChatName & " :" & vbTab & strData
'+End If
'this assigns the new string to the end of
'txtReceived, and scrolls it into view.
'=With txtReceive
'=.SelStart = Len(txtReceive.Text)
'=.SelText = buff
'=.SelStart = Len(txtReceive.Text)
'=End With
'clear the user-input textbox (if desired)
'txtSend.Text = ""
Else
'set the isConnectedFlat to avoid entering
'this condition again during this session
isConnectedFlag = True
'isConnectedFlag is false, so the first data
'received from the connected client will be
'the name of the user. Save this for use when
'posting subsequent data to the txtReceived box.
Form1.tcpClient.GetData strData
'+sClientChatName = strData
Form1.txtStatus.Text = "TCP Client : Chatting with " & sClientChatName
End If
End Sub
Private Sub tcpClient_Error(ByVal Number As Integer, _
Description As String, _
ByVal Scode As Long, _
ByVal source As String, _
ByVal HelpFile As String, _
ByVal HelpContext As Long, _
CancelDisplay As Boolean)

MsgBox "fff"
Select Case Number
Case 10061
txtStatus.Text = "Error: " & Number & vbCrLf & Description & vbCrLf & vbCrLf
& _
"The VBnet Winsock Chat Demo server is not running, " & _
"or has not properly established a connection."
'vbExclamation Or vbOKOnly Or vbMsgBoxSetForeground, _
```

```
'msgTitle
Case 2: MsgBox "2"
Case 3: MsgBox "3"
Case Else
txtStatus.Text = "Error: " & Number & vbCrLf & Description
'vbOKOnly Or vbExclamation Or vbMsgBoxSetForeground, _
'msgTitle
End Select
CancelDisplay = True
tcpClient.Close
're-enable to connect button
cmdConnect.enabled = tcpClient.State = sckClosed
End Sub

Private Sub Timer1_Timer()
Call sample
End Sub
Private Sub Form_Unload(Cancel As Integer)

tcpClient.Close
End Sub

Public Sub procparam()
Dim leng As Integer
Dim k As Integer
Dim c As Integer
c = 0
leng = Len(strData)
Dim p(6) As Integer
For k = 1 To leng
If Mid(strData, k, 1) = "," Then p(c) = k: c = c + 1
Next k
onPred.Text = Mid(strData, 1, p(0) - 1)
offPred.Text = Mid(strData, p(0) + 1, (p(1) - p(0) - 1))
CurrPred.Text = Mid(strData, p(1) + 1, (p(2) - p(1) - 1))
diaPred.Text = Mid(strData, p(2) + 1, (p(3) - p(2) - 1))
flPred.Text = Mid(strData, p(3) + 1, (p(4) - p(3) - 1))
End Sub
```