11-24-1999

Development of a near-perfect lens surface generation advisor (NPLSGA)

Victor De Rossi
Florida International University

DOI: 10.25148/etd.FI14062224

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DEVELOPMENT OF A NEAR-PERFECT LENS SURFACE GENERATION ADVISOR (NPLSGA)

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

MECHANICAL ENGINEERING

by

Victor De Rossi

1999

To: Dean Gordon Hopkins
College of Engineering

This thesis, written by Victor De Rossi, and entitled Development Of A Near-Perfect Lens Surface Generation Advisor (NPLSGA), having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

Sabri Tosunoglu

Kuang-Hsi Wu

Ibrahim Nur Tansel, Major Professor

Date of Defense: November 24, 1999

The thesis of Victor De Rossi is approved.

Dean Gordon Hopkins
College of Engineering

Dean Richard L. Campbell
Division of Graduate Studies

Florida International University, 1999

DEDICATION

To my parents and grandmother.

ACKNOWLEDGMENTS

I wish to thank Motorola for setting up the experiments on this thesis, and providing their technical support and assistance that made possible the completion of this work.

I would also like to thank the members of my committee, Dr. Tosunoglu, Dr. Wu, and my major professor Dr. Tansel, for their support, teachings, encouragement and patience. I deeply appreciate their academic contributions to my professional career, and the energy they provided me in order to accomplish this important milestone in my life.

ABSTRACT OF THE THESIS

DEVELOPMENT OF A NEAR-PERFECT LENS SURFACE GENERATION ADVISOR (NPLSGA)

by

Victor De Rossi

Florida International University, 1999

Miami, Florida

Professor Ibrahim Nur Tansel, Major Professor

The Near-Perfect Lens Surface Generation Advisor is a software program designed to optimize manufacturing conditions of acrylic lenses. The program numerically minimizes the total manufacturing cost of machining and hand polishing by choosing the optimal feed rate, tool diameter and step size. Derivations of mathematical equations plus neural networks estimations were used to calculate the total cost as a function of these given variables. Machining time was found from values obtained from feed rate and tool displacement. Polishing time was estimated based on the scallop height, and the total polished surface area. Scallop height was calculated from geometrical expressions depending on the tool diameter, step over, lens curvature, and relative angle between the cutting tool and the material surface.

TABLE OF CONTENTS

CHAPTER PAGE

I. INTRODUCTION 1

II. THEORETICAL BACKGROUND 5
Machining Process 6
Neural Networks 10
Other Estimation Techniques 13
Curve Fitting Methods 14
Genetic Algorithms 16

III. DERIVATION OF EQUATIONS AND OPTIMIZATION 19
Scallop Height Estimation 21
Feed Rate and Step Over Relationship 32
Machining Time Calculations 34
Tool Diameter Suggestion 36
Polishing Time Calculations 37
Optimization 39

IV. COMPUTER PROGRAM DEVELOPMENT 47
User Interface 49
Algorithms 54
Optimization Algorithm 55
Scallop Height Algorithm 57
Machining Time Algorithm 60
Polishing Time Algorithm 62
Assumptions 65

V. EXPERIMENTAL SET-UP 68
Polishing Time Experiment Preparation 71

VI. RESULTS AND DISCUSSION 73
Scallop height of machined surfaces 76
Tool Diameter Recommendation 83
Experimental Polishing time 86
Accuracy Estimation of the Model 89
Performance of the Program 90

VII. CONCLUSION 91

LIST OF REFERENCES 95

APPENDICES 98

LIST OF TABLES

TABLE PAGE

Table 1: Polishing Time Experimental Values to be Curve-fitted 38

Table 2: Dependency of Total Cost with the Rest of the Variables 40

Table 3: Machining Process Data Recollection Sample Table 70

Table 4: Sample of Ranges for Experimentation 71

Table 5: Hand Polishing Data Recollection 75

Table 6: Scallop Height Calculated per Pass and Total Average 84

LIST OF FIGURES

FIGURE PAGE

Figure 1: How the Scallop is formed 3

Figure 2: Difference in Scallop Height between End-mills and Ball-mills 7

Figure 3: End-mill Cutting a Relative Inclined Plane 8

Figure 4: Scallop Height vs. Angle of Inclination Between the Tool and the Material 9

Figure 5: Representation of a Perceptron 11

Figure 6: Neural Network with 4 Input, Hidden and Output Nodes 12

Figure 7: Least-Square Curve Fitting Example 15

Figure 8: Genetic Algorithm Graphical Example 17

Figure 9: Optimization Process to Find Optimum Manufacturing Settings 20

Figure 10: 45-degree Wedge 22

Figure 11: Shape of Scallop Depending on Cutting Direction 23

Figure 12: Calculation of Scallop Height in a Wedge 24

Figure 13: Calculating Projected Angle of Cutting Tool 25

Figure 14: Representation of End-Mill Tool Cutting in $\alpha \neq 0$ or 90 degrees 26

Figure 15: Ellipses representing the interception of two tool passes 26

Figure 16: Interception Points for Scallop Height Calculation 27

Figure 17: Calculating Curvature Factor 30

Figure 18: Chip Load and Overlap 33

Figure 19: Number of Tool Passes in a Quarter Sphere 34

Figure 20: Simplified Flow Chart to Calculate and Minimize Total Cost 39

Figure 21: Total Manufacturing Cost vs. Machine Step Size 42

Figure 22: Sample Function to Optimize, Iteration 1 44

Figure 23: Sample Function to Optimize, Iteration 2 45

Figure 24: Main Window of the Near-Perfect Lens Surface Generation Advisor (NPLSGA) 50

Figure 25: Detailed View Showing Extra Optimization Parameters 51

Figure 26: Contents in the NPLSHA Help Window 54

Figure 27: Optimization Algorithm 55

Figure 28: Average Scallop Height Calculation Flow Chart. 60

Figure 29: Machining Time Calculation Flow Chart 61

Figure 30: Error Message displayed if number of Flutes is entered as a Negative Number 66

Figure 31: Acrylic 45-degrees Wedge Sample 69

Figure 32: Wedge Sample After Machining 70

Figure 33: Wedge Sample after Machining and Polishing 72

Figure 34: Neural Networks Representations to Estimate Polishing Time and Tool Diameter 74

Figure 35: Scallop Calculator Software 75

Figure 36: Scallop Height plotted against Step over with different Tool Diameters 77

Figure 37: Scallop Height plotted against Step over with different Slope Angles 78

Figure 38: Scallop Height plotted against Step over with different Cutting Angles 79

Figure 39: Scallop Height plotted against Step over with different Cutting Angles except $\alpha = 90°$ 79

Figure 40: Scallop Height plotted against Step over with different Radius of Curvature 82

Figure 41: Scallop Height plotted against the Plane Inclination with different Tool Diameters 82
Figure 42: NPLSGA Program Displaying Average Scallop Height in a Lens 83
Figure 43: SHC Displaying the Scallop Height at One Point on the Same Lens 83
Figure 44: Tool Diameter Estimation vs. RPM / Feed Rate 85
Figure 45: Tool Diameter Estimation vs. Chip Load 86
Figure 46: Surface that has had Two Perpendicular Finish Passes 87
Figure 47: Experimental Samples 88
Figure 48: Results from Experimental Setup 89
Figure 49: Approximation of Polishing Time Given by Neural Networks and Curve-Fitted equations 90
Figure 50: Acrylic Sample 1, Step Size = 0.052 100
Figure 51: Acrylic Sample 2, Step Size = 0.078 101
Figure 52: Acrylic Sample 3, Step Size = 0.156 102

CHAPTER I

INTRODUCTION

Designing, developing, and manufacturing of state of the art products, along with timely marketing, are very important factors in today's fast growing industries. Many consumer products have dials or screens. These dials or screens are protected with plastic lenses. For instance, they are used in cellular phones, electronic agendas, pagers, digital cameras, and other electronics devices. These lenses are given very complicated shapes, which abide to cosmetic, optical, or strength reasons.

Acrylic lenses' prototypes are prepared in two stages. First, the material is machined to obtain the desired shape, and second, the part is hand polished to give the lens an acceptable finish.[9] Total cost of the part will be the sum of the machining cost plus the polishing cost. There are several factors that will prevail upon the process of minimizing these costs. The optimum step over and feed rate, along with the best tool diameter will calculate the parameters to machine the specified curved surface in the program.[11] One possibility to reduce the machining cost is to increase the step size. However, this process generates rougher surfaces that will require longer polishing time, creating an increase in the polishing cost. That is why both costs should be equally considered when selecting the optimal step size to prepare the lens surface at the minimum cost. Throughout this thesis, the cost of both processes will be estimated using analytical expressions plus estimations of experimental data.

Machining cost is a function of machining parameters (tool diameter, step size, feed rate, spindle speed, depth of cut), labor, machine cost and overhead.[6] Step size and feed rate will be calculated by the program based on the lens diameter, spindle speed, allowable chip load and number of teeth in the cutting tool. On the other hand, polishing cost depends on the labor rate, and initial surface roughness. The cost of labor per hour is

assigned programmer, and the scallop height will be calculated by mathematical expressions. The scallop is formed with the tool advances parallel to the previous passes with certain overlap.[9] A graphical representation of the process is presented in Figure 1.

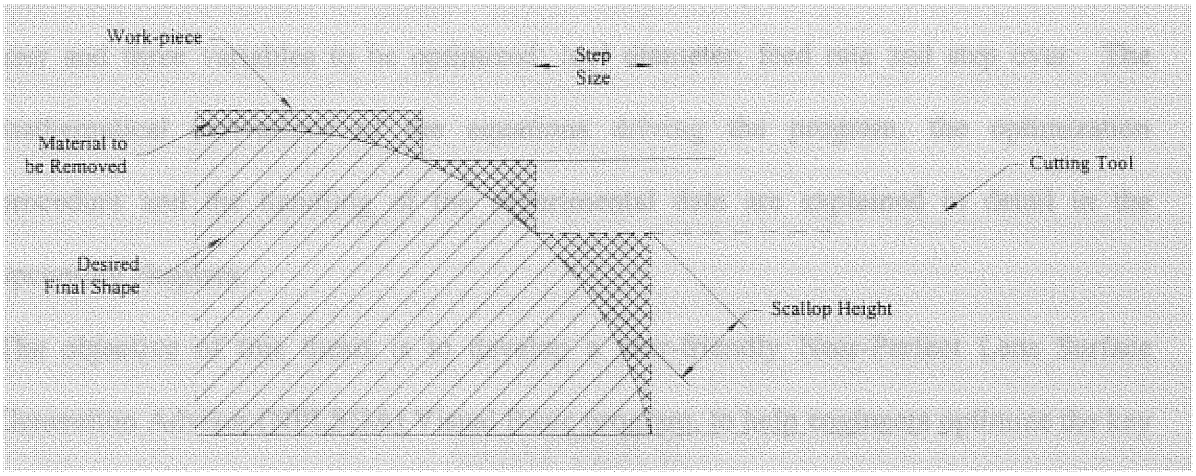

Figure 1: How the Scallop is formed

The scallop height depends on the lens shape, the tool type and its diameter, the overlapping of the passes, the relative angle between the material's surface and the tool, and the direction of the cut.[9] The tool type selected may either be an end-mill or a ball-mill. This study is based under the assumption that a flat end tool is used. It's been proven that end-mills give a better match to the designed curved geometry in much less number of passes than a ball-mill.[1] However, if a ball-mill is used instead, the program has the option to calculate the scallop height with this condition. The lens geometry must be given. Also, the relative angle of the material and the tool, and the direction of the cut have to somehow be specified by the program user. Optimizing the tool diameter and the overlap will result in the best polishing cost.

To relate all these variables with the polishing cost, a systematic experimental procedure is set up. A relationship between the scallop height and the polishing time will result

based on the experimental data. All the data obtained in these experiments is then fed to a neural network to find the best relationship among theses variables.[12]

After finding the machining cost and polishing cost, they are added together to estimate the total manufacturing cost.[11] This cost value will depend on fixed numbers given by the user and three variables to be optimized: tool diameter, feed rate and step over. The mathematical derivations of the equations driving the program, the optimization procedure and the handling of the experimental data are explained in detail in the upcoming chapters.

The objective of this thesis is to develop a user-friendly Near-Perfect Lens Surface Generation Advisor (NPLSGA) Visual Basic program, to help engineers optimize the key parameters to minimize the machining of acrylic lenses.

CHAPTER II

THEORETICAL BACKGROUND

Quality of surface finish is defined by evaluating the smoothness of the surface. The smoothness of the surface depends on scallop height and surface roughness. Generally, even tough the surface looks flat and smooth, there are some hardly visible irregularities, commonly called surface roughness. Surface roughness is the small irregularities on the surface created by the relative machine tool, work-piece vibration, and imperfect surface created by the shear during machining.[23]

The scallop height is the narrow layers left by the passes of a tool. Later in this thesis it is explained how the scallop shapes may differ, depending on the relative angle between the tool and the work-piece, and the distance between passes.[9] The scallop height would simply be the elevation between the highest point of the remaining material and the desired surface plane.

In this study the surface roughness concerns only the selection of cutting conditions. Since in most cutting conditions the scallop height is much larger than the surface roughness, the polishing time is calculated only by considering the scallop height.

MACHINING PROCESS

Ball or flat end mills can be used to create complex surfaces. Both approaches have their own advantages depending on the situation they are used. Flexibility in manufacturing processes has derived new examinations to determine the advantages and disadvantages of both types of mills when cutting curved surfaces.[1] Extra degrees of freedom in milling machines, and computer assisted programming allows this flexibility.

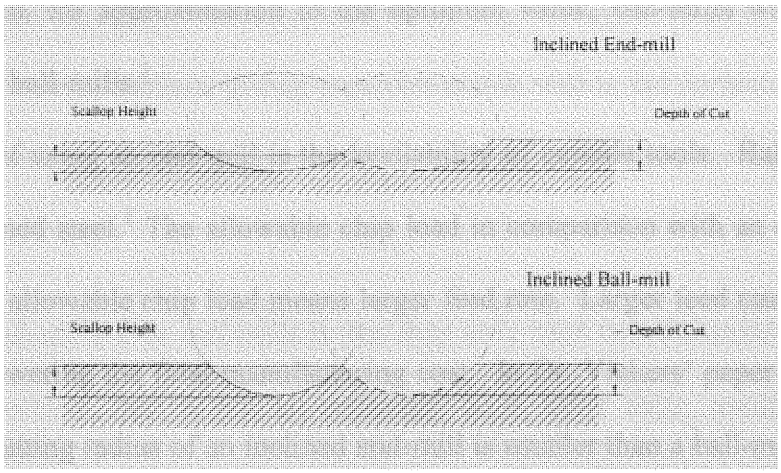

Figure 2: Difference in Scallop Height between End-mills and Ball-mills

The main goal in machining a part is to get the best surface finish in the least amount of time to reduce its cost.[2] Therefore, a comparison is necessary to know when each mill is best to be used. A paper written by G. W. Vickers, at the University of Victoria, B.C. Canada, came to the following conclusion:

> "The use of end-mills for machining low curvature surfaces…give a better match to the required surface geometry, and hence reduce the number of surface passes required. They also have a much better efficiency of material removal and longer tool life...the use of end-mills for curved surface work can typically reduce the overall cutting time by a factor of twenty-four."[1]

In this statement the geometry of both tool types is considered. They also note that the number of available sizes for end-mills is larger for than ball-mills.[1]

One can conclude from this documentation that the use of end-mills is more suitable to cut acrylic lenses. The fact that there is larger availability for end-mills than there is for ball-mills allows to provide a better tool diameter recommendation that best approximates the value calculated by the program. If the program calculates a tool diameter of 0.198 inches, there probably would be an available 0.2" end-mill, but only a

¼" ball-mill. Then, for approximation to the optimum value, end-mills will have an advantage over the ball-mills.[1]

A second benefit of using an end-mill is the availability of carbide tools. Ball-mills are made out of hardened steel.[1] The allowable chip load in comparison with an end-mill is much less. Larger allowable chip load means larger feed rate. Larger feed rates decrease machining time, lowering the final manufacturing cost. In Dr. Vickers' paper it is stated that the effective cutting radius of an inclined end-mill is smaller than a ball-mill, making the contact area larger, and reducing the scallop height.[1] Therefore, using an end-mill will not only save time in machining, but also in polishing the final lens. Even though the tool is not being inclined against the surface, the surface is relatively inclined against the tool. For that reason, instead of a square cutting area, the end-mill will have an elliptical cutting area.

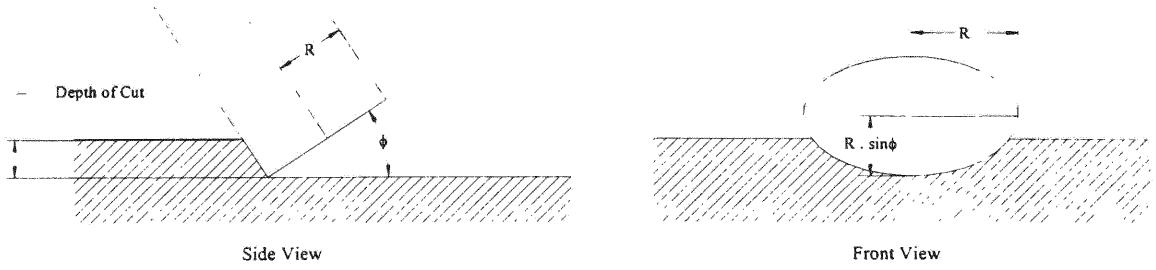

Figure 3: End-mill Cutting a Relative Inclined Plane

Note in Figure 3 that the effective cutting angle changes with the relative angle between the tool and the plane. If the angle decreases, so does the effective cutting angle,

increasing the contact surface. When the inclination angle is equal to zero, the tool is completely flat against the surface, giving a perfect match with a flat surface. Figure 2 shows how an end-mill provides a better cutting match than a ball-mill, leaving a smaller scallop height.

An advantage of ball-mills over end-mills is that they only need two-dimensional cutting compensation.[7] However, that is not an important issue these days, considering the modern CNC controllers and the current trend of computer-assisted programming that mange these calculations without affecting the production of the part.

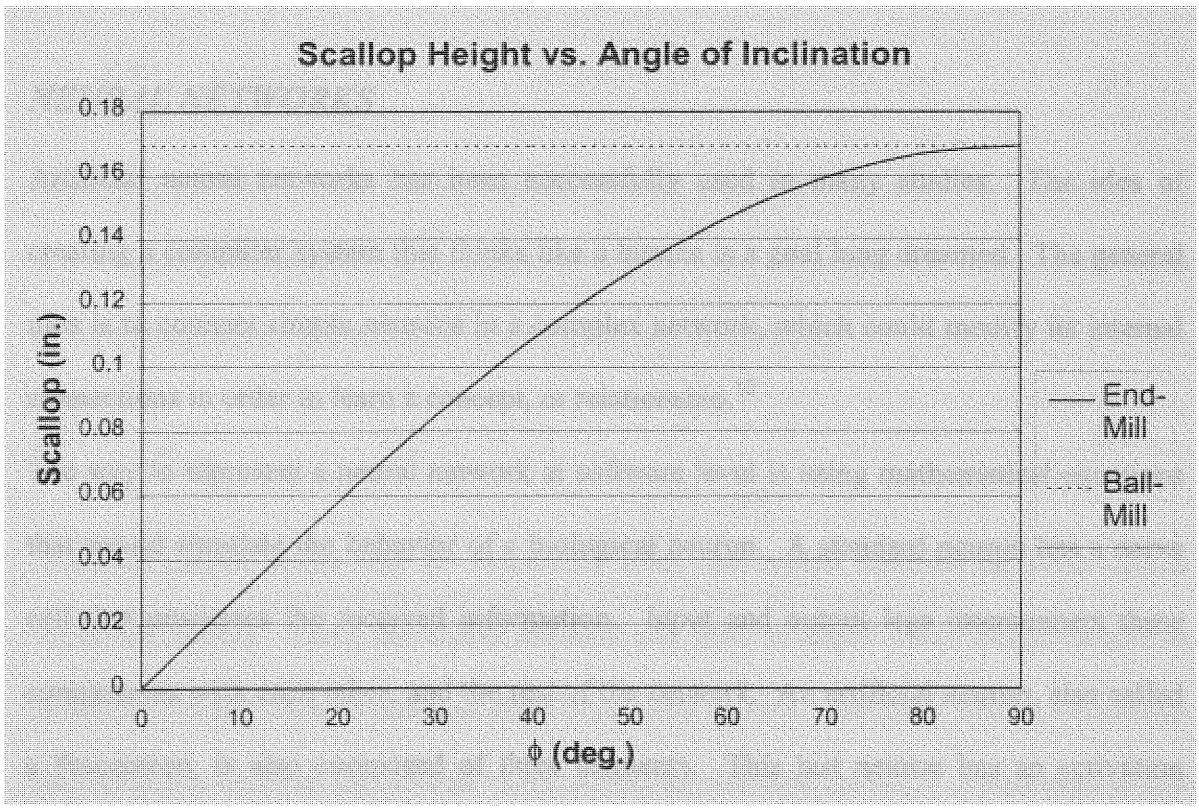

Figure 4: Scallop Height vs. Angle of Inclination Between the Tool and the Material

As one can see in this graph, the scallop height using a ball-mill is always larger, except when the material is being cut at 90 degrees, where both scallop height match (Taken into consideration that all the calculations made for scallop height in previous work only consideration when the tool is fed in the same direction than it is inclined). [1] Equations to calculate the scallop height in any situation are derived in the following chapters. These equations specify certain cases where the scallop height made with an end-mill may be larger than the one made with a ball-mill. Besides these cases, end-mills are usually recommended because of the variety of sizes, their strength, and the greater allowed chip load.

NEURAL NETWORKS

Artificial neural networks has been successfully used in many studies. The idea of creating a computer system that thinks like a human is a goal long dreamed. The general idea is to connect simple neurons in a complex network, which could modify its internal connections in order to learn a concept, or relationship.[12]

The way to represent a neural network at software level is using mathematical equations that would simulate the behavior of a biological neuron. A standard neuron has a nerve cell that processes the received information. Input and output legs interconnect these components with each other, and the rest of the network. An artificial neuron, also called a Perceptron, is also composed of three elements. This last neuron has an activation function that generates the output, weighted input and a summation function that receives all the inputs.[12]

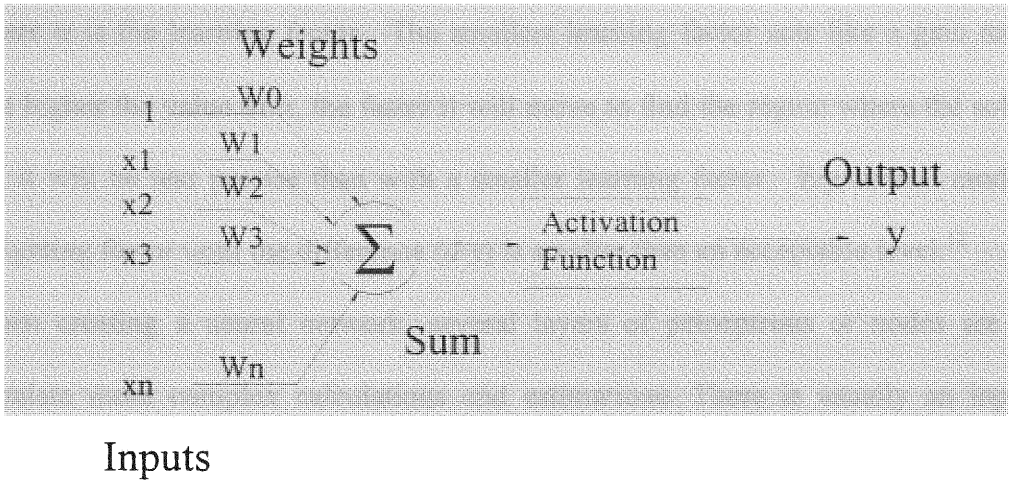

Figure 5: Representation of a Perceptron

If Figure 5 is represented as a mathematical equation it will appear like this:

$$y = f\left(\sum_{0}^{n} x_i \cdot W_i\right) \tag{1}$$

where y is the output from the Perceptron, f is the activation function, , x_i is an input, W_i is a weighting factor, and i is a number from 0 to n that represents the index of the input or weighting factor. An example of a function can be given as

$$y = \frac{1}{1 + e^{-\sum_{0}^{n} x_i \cdot W_i}} \tag{2}$$

Now, an output value can be calculated from the inputs considering some initial weighting factors. Then, the neural network will need to adjust those weight factors to reduce the output error as much as possible, ideally to zero. The error will be defined by the difference between the desired value, d, and the network output, y. Subsequently, the weighting factors are modified by the following equation:

$$Wi = Wi \cdot xi \cdot \sigma \cdot (d - y) \tag{3}$$

where σ is the learning factor. The smallest learning factor acts like a gain multiplier. The higher the value of it, the faster it will move to find the region where the solution is, but it will be less precise than with a smaller learning factor. Equation 3 modifies the weighting factor until the difference (d-y) is smaller than an acceptable level.[12]

When creating a neural network, several layers of perceptrons or nodes are used to simulate more complex calculations and estimations. There is usually an input layer composed by several input nodes, some hidden layers, and an output layer. They can be connected between them with different weighting factors creating a network called "multi-layer perceptrons."[12]

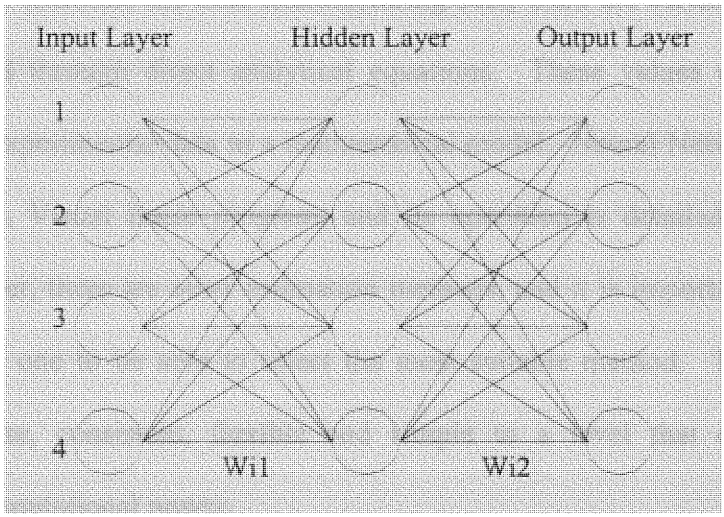

Figure 6: Neural Network with 4 Input, Hidden and Output Nodes.

In the case of a complete network, more complex cases can be learned, but also it would require longer computational time. The main problem with multi-layer networks is that the user does not know the desired output value for the hidden nodes. Then, the process to find the weighting factors on a single node could not be applied, unless desired output of each hidden node was found. In order to do that, a Back Propagation training

algorithm is used. This algorithm finds the hidden nodes outputs by obtaining the output nodes weighting factors. This would eventually generate the desired output.[12]

Then, the process of teaching the neural network will consist of providing a set of inputs and desired outputs. The network processes the information as a forward pass and a backward pass, finding the weighting factors of the hidden nodes. Once the final output error is minimized, it is assumed that the hidden nodes outputs are acceptable, and the learning process is stopped.[12]

OTHER ESTIMATION TECHNIQUES

In the process to calculate the total manufacturing cost, there are certain parameters that cannot be found through direct analytical equations. These values can be estimated approximating experimental data to a mathematical model. Neural Networks are utilized in the NPLSGA Visual Basic program, and its operation is detailed in the previous section. However, there are other estimation methods, and it is important to review and understand each one to be able to select the most suitable method. The ideal process most be accurate, consistent, flexible and simple to a level that does not require unnecessary computational power.

In this thesis, the most important parameter that cannot be found analytically is the polishing time per area. It is assumed that this value is only dependable on the scallop height. Surface roughness caused by machine vibrations and other factors is considered very small, compared to the scallop height, to create a difference in the polishing time. Therefore, even though more independent variables could be added later to the program estimation, polishing time could be easily found by curve fitting data obtained from

experimentation. The result will be a one-dimensional equation in which the scallop height is entered to obtain the estimate time.

CURVE FITTING METHODS

They are many techniques that allow obtaining mathematical functions to fit best experimentally measured data. There are two main approaches to fit given data to a curve: one is to find a function that passes through all known points, and the other is to find an approximated function that will graph as a smooth curve. Some of these approaches are: Method of the Least Squares, Curve Fitting with Polynomials, Curve Fitting with Fourier Series, Curve Fitting with Exponential Functions, Linear Interpolation, Cubic Interpolation, etc.[21]

If the given data is considered to be very accurate, probably the best method to use is the Least Squares.[21] This technique finds a function in which the sum of the squared errors between the real values and the estimated values in minimal. If the function has only one variable the logic could be as follows.

$$y = c_1 + c_2 x \tag{4}$$

$$r_i = y_{estimated} - y_{real} \tag{5}$$

$$\sum_{i=1}^{n} (r_i)^2 = \text{a minimum} \tag{6}$$

$$\sum_{i=1}^{n} r_i \frac{\partial r_i}{\partial c_k} = 0, \quad \text{k} = 1,2. \tag{7}$$

where c_1 and c_2 and constants to be found, r_i is the error between the ith experimental value and the corresponding curve-fitted value, and n is the number of given points. First a generic function is selected (Equation 4), and its constants are found so the square error

between the fitted value and the real experimental data is minimum. In order to do that, the sum of the error function is derived and equal to zero to find the constants and obtain the curve-fitted equation.[21]

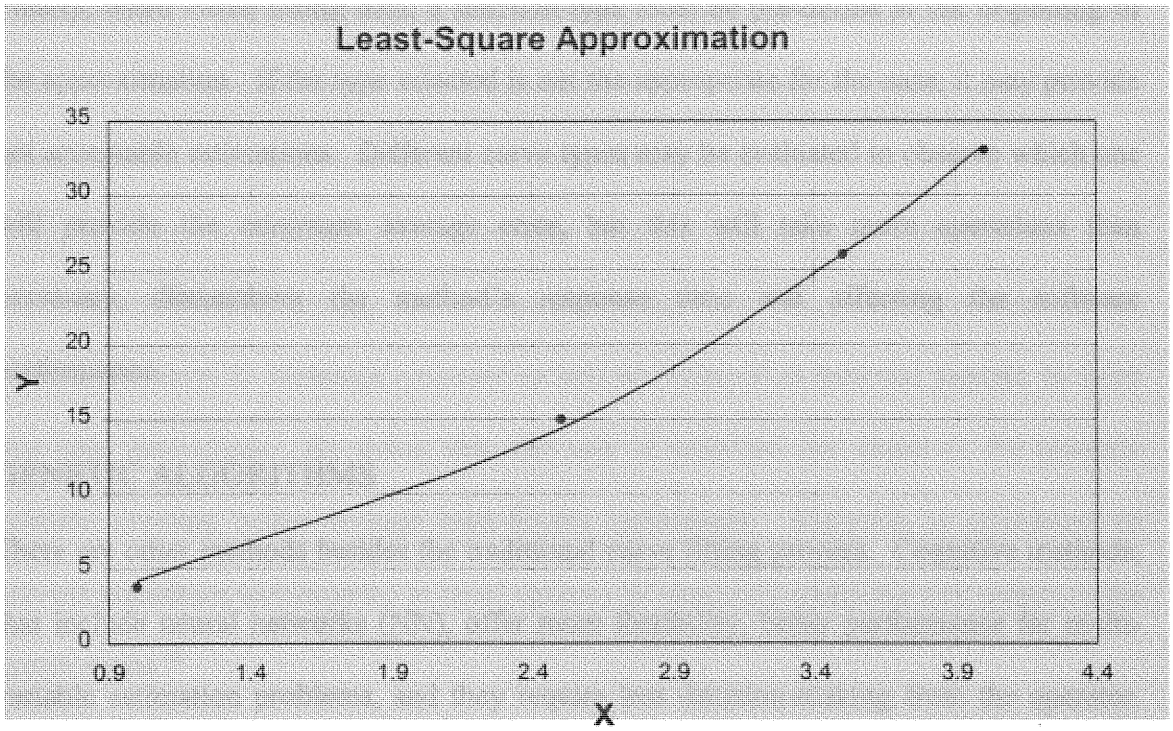

Figure 7: Least-Square Curve Fitting Example

Other numerical curve fitting methods could be used for curve fitting, using similar concepts for approximating a predefined generic curve to the given data. Some use polynomial type curves, or exponential functions to fit the data, and others use more complex approximations for better results like Fourier Series representations.[21]

Even though curve fitting would be the method that would provide the fastest response, it is the one with less flexibility. If more experimental data is found or more variables are used to estimate the polishing time, the curve-fitted equation is no longer valid, and the

curve approximation process must be repeated in order to find a new relationship. If more input variables are entered, the complexity level to approximate a curve is increase making the method not worthwhile anymore due to its computational requirements. Finally, when curve fitting data, a function type must be pre-define at the beginning of the approximation. If the type selected is not the appropriate for the case, it may provide not acceptable estimations. Different curve types may be evaluated to observe which one will provide the minimum average error, but this will take more operations than necessary, diminishing the method's response time and affecting the software performance.

GENETIC ALGORITHMS

There are other methods besides the traditional curve fitting process. Cognitive methods, that include neural networks (NN), offer more flexibility, better performance for multi-variable estimation problems, and they require less decision-making from the operator. One of these popular methods are Genetic Algorithms, which mimic the process of natural selection with the effect of creating a number of potentially optimal solutions to some complex search problem. It is an iterative procedure that consists of a constant-size population of individuals that are evaluated, crossover and, in a small scale, mutated in order to search a given problem space. Each individual contains a possible solution to the problem, and it is represented with a finite array of information stored in different cells or "genomes." Individual's information is evaluated and combined in a randomly fashion in order to obtain new groups on individuals (or new generations) that may contain the near-global optimum solution.[25]

The standard genetic algorithm first generates an initial population of individuals at random. Then, they are evaluated according to predefined quality criteria, or fitness function. Depending on the evaluation results, individuals could be selected for "reproduction." Individuals' selection probabilities increase as their fitness evaluation improves. Hence, high-fitness individuals stand a better chance of reproducing than low-fitness ones.[25]

After selecting the "parent" individuals they could be crossover or mutated based on a "crossover probability" (P_{cross}) and a "mutation probability" (P_{mut}, usually a small value). Crossover allows to exchange genomes or sets of information between parents, to form two new individuals. By doing this, the sets of solutions get closer to the possible solution creating better average high-fitness individuals. Mutation is introduced to prevent premature convergence to local optima, by randomly sampling new points in the search space.[26]

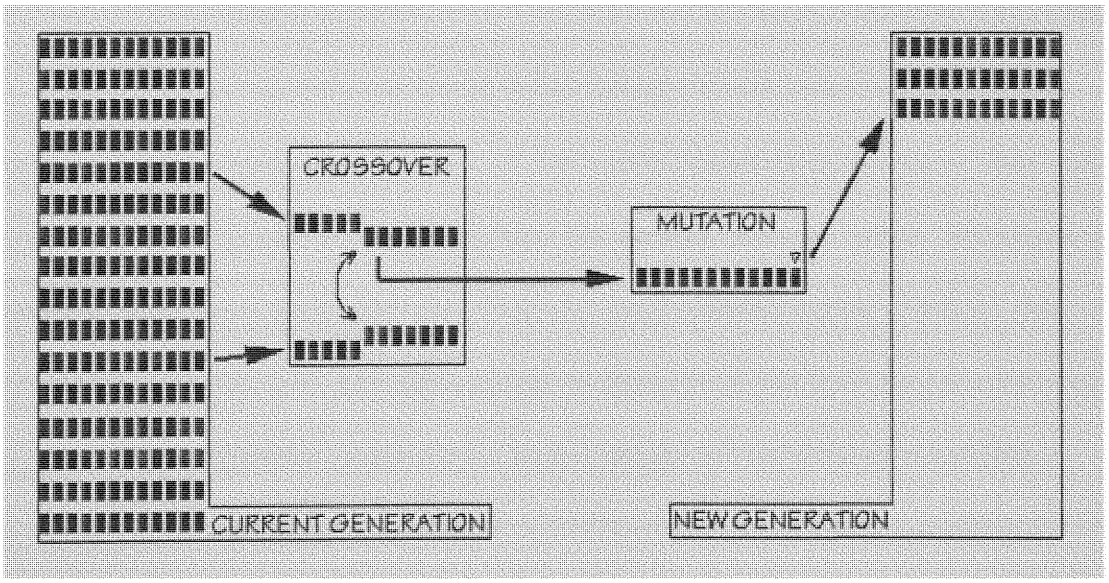

Figure 8: Genetic Algorithm Graphical Example

The main drawback for this method is that for each generation, several completely different solutions need to be verified and stored but just one will be the final one. This creates a big consume of computation resources making the algorithm very slow. Also, They are non-deterministic iterative processes that do not guarantee convergence to a valid solution. Most people uses the ability of genetic algorithms to search large, complex space to prepare data for neural networks, find the initial sets of parameters for training networks, and use genetic and the newer evolutionary techniques to evolve neural network architectures.

CHAPTER III

DERIVATION OF EQUATIONS AND OPTIMIZATION

As stated at the beginning, the goal of this project is to use a series of equations and estimations in order to provide a design engineer the best possible manufacturing parameters to reduce the total manufacturing cost of a curved acrylic lens. The overall process and the necessary parameters to find the total cost will be carefully studied. This will result in the finding of the minimal possible value by modifying the step size.

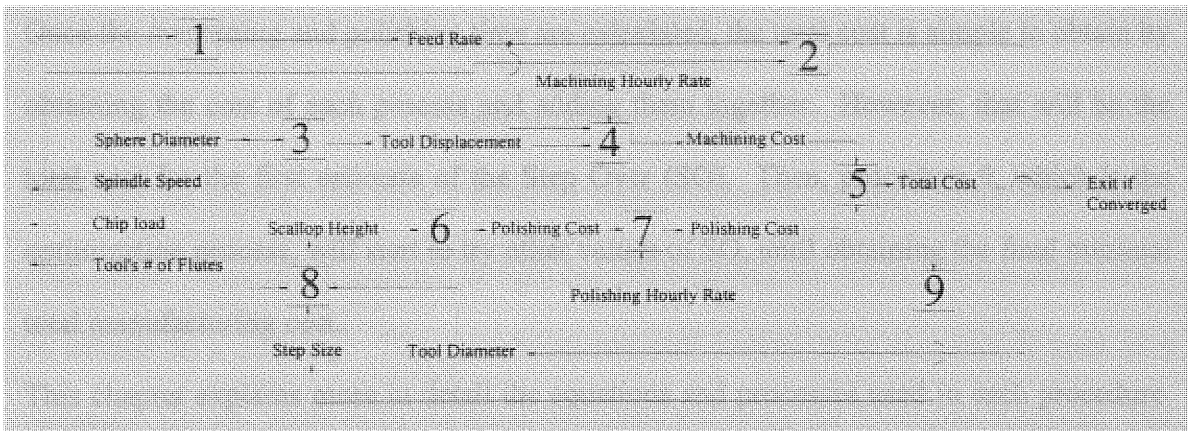

Figure 9: Optimization Process to Find Optimum Manufacturing Settings

Figure 9 shows a general flow chart of the entire optimization process. The initial variables are the sphere or lens diameter, the spindle rotation speed, the maximum allowable chip load, the cost per hour of each manufacturing method, and the characteristics of the tool, like the number of flutes or if end-mill or ball-mill. Each numbered square represents a subroutine used to obtain an internally used variable.

The first procedure, represented as the square labeled "1", is the method in which the best suitable feed rate is chosen based on the allowable chip load, the spindle's revolution per minutes, and the tool's characteristics. This feed rate is used along with the work-piece

geometric characteristics, in the second subroutine to find the optimum tool diameter. Tool diameter would be essential to calculate the scallop height (process 8). The scallop height calculation will also require the sphere diameter, and an initial guess for the step size (distance between parallel passes). The scallop height is later fed into a neural network. The result will provide an estimated polishing time. By knowing the speed of the tool and its total displacement, the machining time may be calculated (shown as process 4). This depends on the lens's geometry.

Once the machining and polishing time are calculated, the total cost is simple to obtain by obtaining the cost of each process and adding them together. However, that will be the total cost for the first estimate of the Step Over (SO). Then, procedure 9, which is the optimization process, will start to try other values of SO, until obtaining the final minimal total manufacturing cost.

This chapter explains more in detail the equations and processes used in each procedure. This is achieved by fully explaining the derivation of equations and estimation techniques.

SCALLOP HEIGHT ESTIMATION

The base of all the calculation in the Near-Perfect Lens Surface Generation Advisor (NPLSGA) Visual Basic software is the calculation of the scallop height. The scallop height, as mentioned in the previous chapter, depends on the geometry of the surface to be cut, the direction of the cutting tool, the tool diameter, and the step over. To begin with an easy example, it is assumed that a part of a wage form with a 45-degree angle is being cut (See Figure 10).

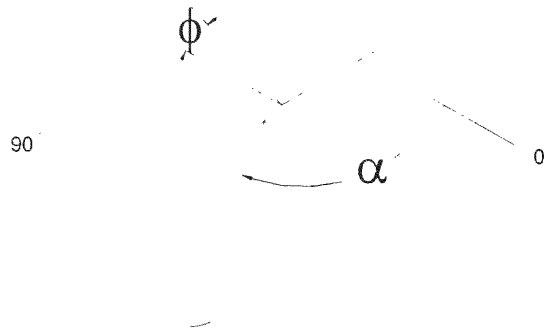

Figure 10: 45-degree Wedge

Note that the scallop will have a different shape depending on the cutting angle of the tool. If the wedge is cut from left to right ($\alpha = 90$ degrees), the shape of the scallop will be similar to a stairway. The cylindrical end-mill looked from one side is seen as a rectangle, and the shape of its cut looks like steps following the 45-degree diagonal. The equation to calculate the scallop height in this case is very simple:

$$Sh = SO \cdot Sin(\phi) \tag{8}$$

where Sh is the scallop height, SO is the step over, and ϕ is the angle of the inclined surface (in this case 45 degrees). Now, if the part is cut from front to back ($\alpha = 0$ degree), as shown in Figure 11, the shape changes, reflecting the curvature of the tool. The scallop height in the horizontal plane will be calculated using the following equation:

$$Sh_{horizontal} = R - \sqrt{R^2 - \left(\frac{SO}{2}\right)^2} \tag{9}$$

where R is the Tool Radius. The real scallop height is found by getting its projection in the cutting plane (see Figure 12):

$$Sh = R - \sqrt{R^2 - \left(\frac{SO}{2}\right)^2} \cdot Sin(\phi) \tag{10}$$

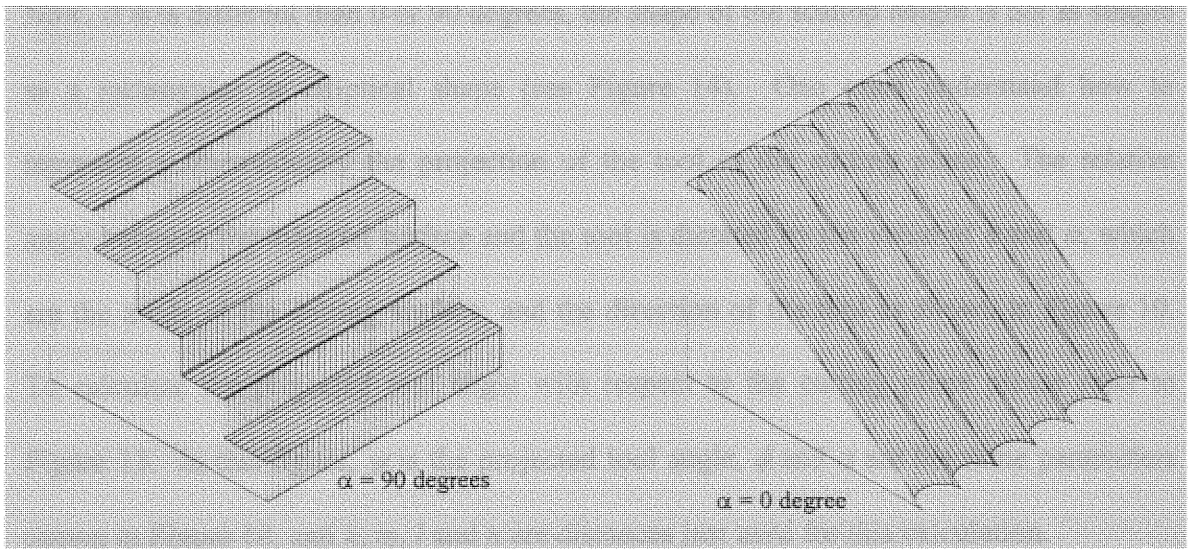

Figure 11: Shape of Scallop Depending on Cutting Direction

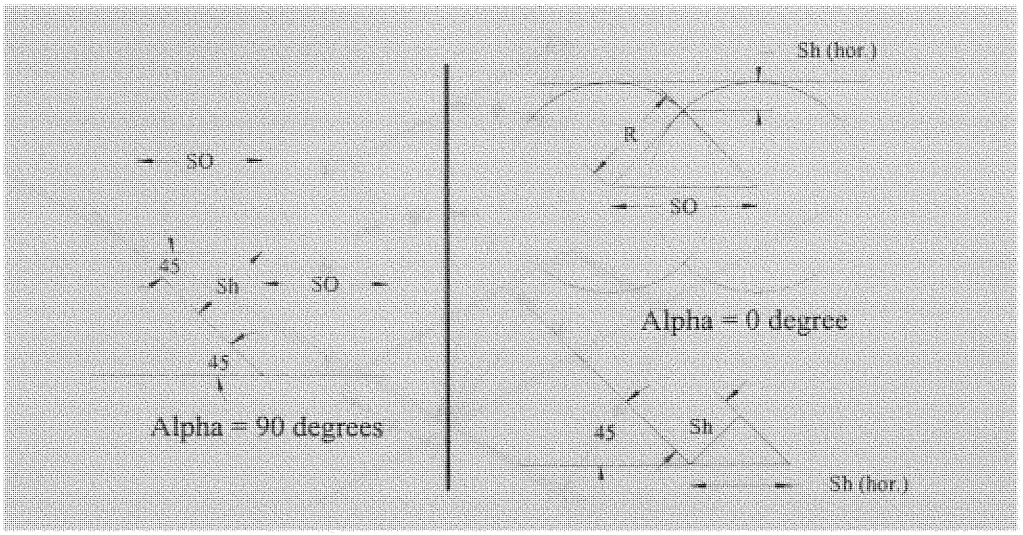

Figure 12: Calculation of Scallop Height in a Wedge

Now, if α is different than 0 or 90 degrees, the shape of the scallop height is not as simple as a rectangular or elliptical shape (see Figure 14). One must understand how to represent mathematically the projection of the tool in the cutting plane. The relative angle between the material surface and the tool is dictated by the value of ϕ and α, which are the angle of the cutting surface, and the direction of the tool as shown in Figure 10. To calculate the projected tool angle with respect to the cutting plane please refer to Figure 13. In this figure, ψ is the projected tool angle, ϕ and α are the same parameters shown in Figure 10, and h, m, l and n are distances just used for this calculation.

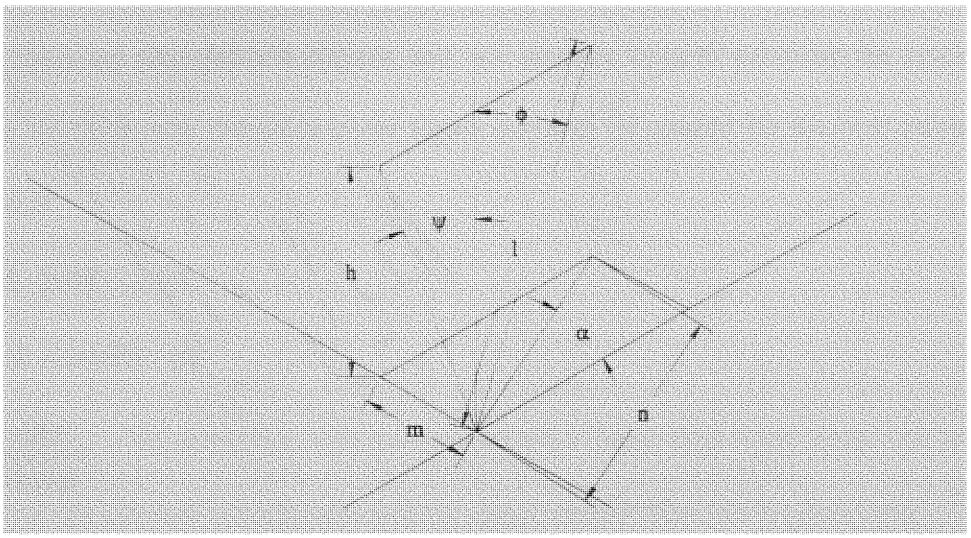

Figure 13: Calculating Projected Angle of Cutting Tool

$$\tan\psi = \frac{m}{n}$$

$$l \cdot \cos\phi = h$$

$$l \cdot \cos\phi = \frac{m}{\tan\psi}$$

$$l \cdot \cos\phi = \frac{n \cdot \sin\alpha}{\tan\psi}$$

$$l \cdot \cos\phi = \frac{l \cdot \sin\phi \cdot \sin\alpha}{\tan\psi}$$

$$\psi = \arctan(\tan\phi \cdot \sin\alpha) \qquad (11)$$

being ψ the value of the projected tool angle is very important for the following calculations.

To simplify calculations, it is assumed that the material surface is horizontal, and the tool is inclined. The same relative angle is kept in order to find the correct scallop height (see Figure 14).

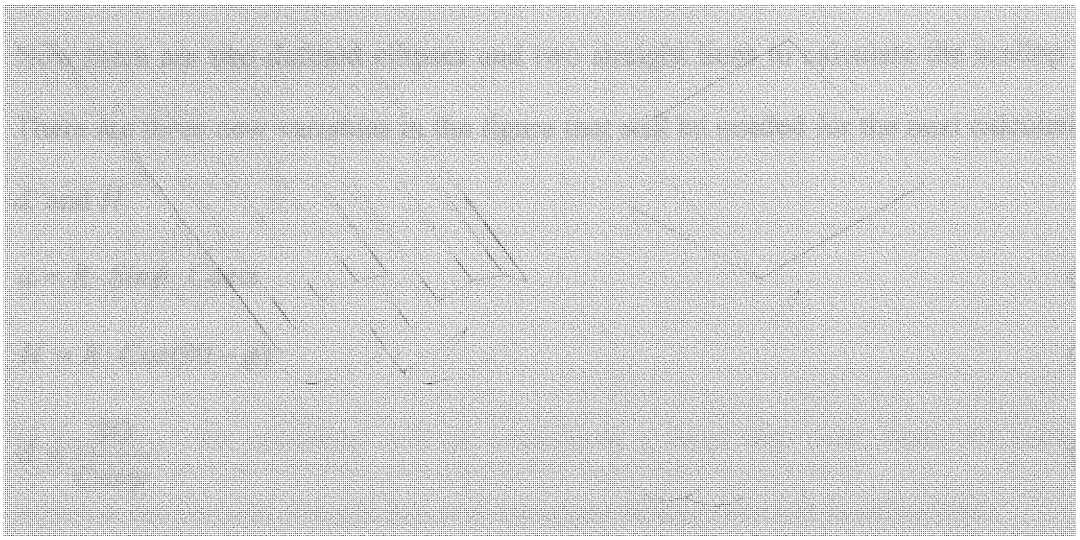

Figure 14: Representation of End-Mill Tool Cutting in $\alpha \neq 0$ or 90 degrees

Notice that the projected shape of the end mill is an ellipse drawn in an angle. That angle will be the same ψ calculated in Equation 11. Therefore, to simplify further the equations, the axes are rotated by ψ. After doing that, the tool projection with the scallop shape would look as Figure 15:

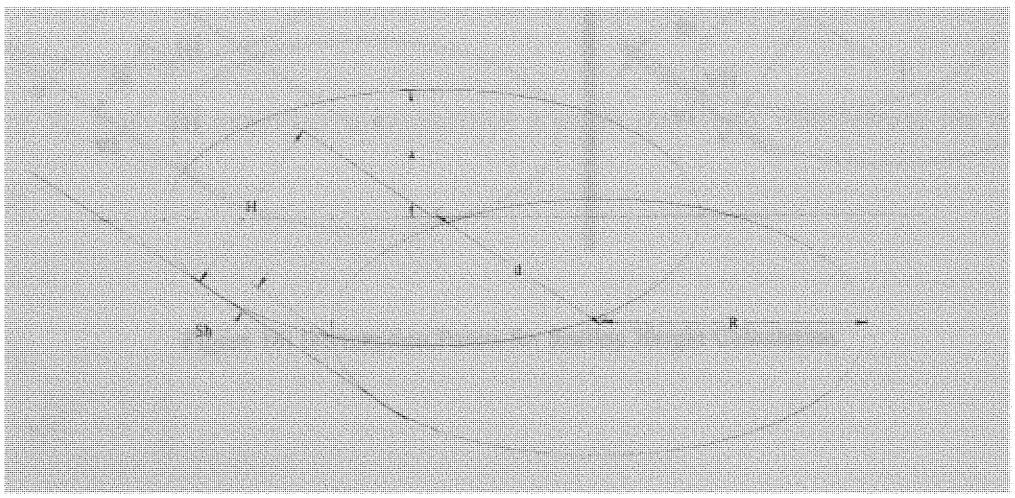

Figure 15: Ellipses representing the interception of two tool passes

Now there are two defined ellipses and two straight lines to calculate the scallop height. There are three new variables in the figure that will be needed for the next calculations: a, d, and H.

$$a = R \cdot Sin\phi \cdot Cos\alpha \tag{12}$$

$$H = R \cdot Cos(90 - \phi) \tag{13}$$

$$d = \frac{SO}{Cos\,\psi} \tag{14}$$

If the step over is small, the interception will occur between the two ellipses. Else, the interception will occur between the straight line going up, and the upper ellipse. The scallop height will be the perpendicular distance from that interception to the diagonal line (representing the material surface).

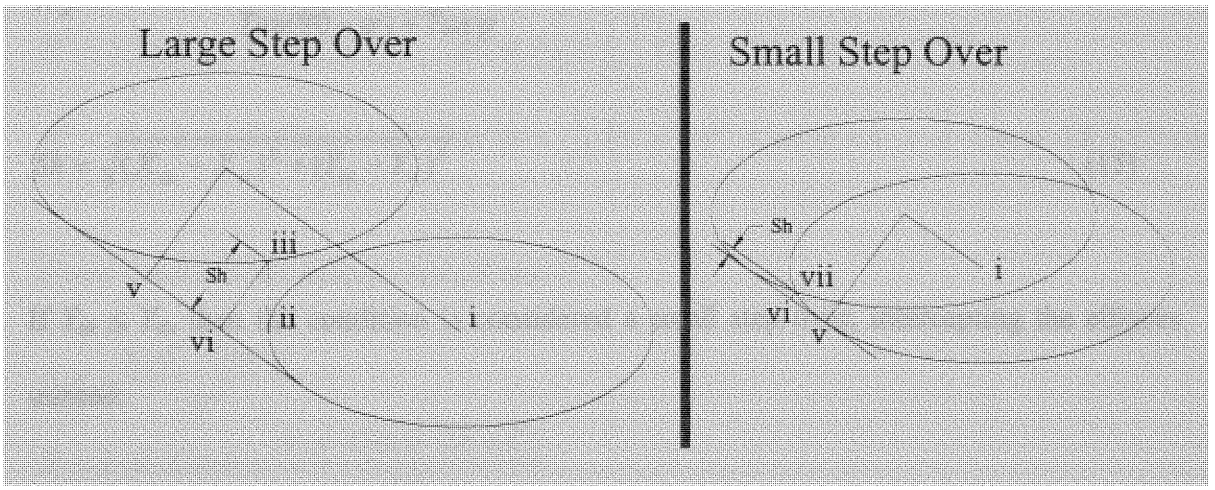

Figure 16: Interception Points for Scallop Height Calculation

$$X_{ii} = X_{iii} = d \cdot Cos\psi - R$$
$$Y_{ii} = -d \cdot Sin\psi$$
$$Y_{iii} = -\sqrt{R^2 - X_{iii}^2} \cdot \frac{a}{R}$$
$$X_{v} = -H \cdot Sin\psi$$
$$Y_{v} = -H \cdot Cos\psi$$

If $Y_{iii} > Y_{ii}$, the condition will be considered a large step over, then, the interception between the vertical straight line and the upper ellipse function is calculated:

$$Tan(90-\psi) \cdot (X_{vi} - X_{iii}) + Y_{iii} = Y_{vi}$$
$$-Tan\psi \cdot (X_{vi} - X_{v}) + Y_{v} = Y_{vi}$$
$$Tan(90-\psi) \cdot X_{vi} - Tan(90-\psi) \cdot X_{iii} + Y_{iii} = -Tan\psi \cdot X_{vi} + Tan\psi \cdot X_{v} + Y_{v}$$
$$X_{vi} = \frac{Y_{v} - Y_{iii} + Tan\psi \cdot X_{v} + Tan(90-\psi) \cdot X_{iii}}{Tan(90-\psi) + Tan\psi}$$

$$Sh = \sqrt{(X_{iii} - X_{vi})^2 + (Y_{iii} - Y_{vi})^2} \qquad (15)$$

If $Y_{iii} \leq Y_{ii}$, then the condition is considered a small step over, intercepting the ellipses instead:

$$x = X_{vii}$$
$$Y_{vii} = -\sqrt{R^2 - x^2}\,\frac{a}{R}$$

$$Y_{vii} = -\sqrt{R^2 - (x - d \cdot Cos\psi)^2}\,\frac{a}{R} - d \cdot Sin\psi$$

$$\sqrt{R^2 - x^2}\,\frac{a}{R} = \frac{a}{R}\sqrt{R^2 - (x - d \cdot Cos\psi)^2} + d \cdot Sin\psi$$

If $C = \left(\frac{R}{a}\right)^2 \cdot d^2 \cdot Sin^2\psi$, then

$$x^2 - (x^2 - 2 \cdot x \cdot d \cdot Cos\psi + d^2 \cdot Cos^2\psi) + \left(\frac{2 \cdot R \cdot d}{a}\right) \cdot Sin\psi \cdot \sqrt{R^2 - (x - d \cdot Cos\psi)^2} + C = 0$$

and if we constant numbers are substituted by E and D, defined as

$$E = -d^2 \cdot Cos^2\psi$$

$$D = \frac{2 \cdot R \cdot d}{a} \cdot Sin\psi$$

then,

$$2 \cdot x \cdot d \cdot Cos\psi + E + D \cdot \sqrt{R^2 + x^2 + 2 \cdot x \cdot d \cdot Cos\psi - d^2 \cdot Cos^2\psi} + C = 0$$

$$R^2 - x^2 + 2 \cdot x \cdot d \cdot Cos\psi - d^2 \cdot Cos^2\psi = \left(-\frac{1}{D}\right)^2 (2 \cdot x \cdot d \cdot Cos\psi + E + C)^2$$

$$R^2 - x^2 + 2 \cdot x \cdot d \cdot Cos\psi - d^2 \cdot Cos^2\psi = \frac{1}{D^2}\left(4 \cdot x^2 \cdot d^2 \cdot Cos^2\psi + 4 \cdot x \cdot d \cdot Cos\psi \cdot (E + C) + (E + C)^2\right)$$

If new constants are defined as Λ,Φ and Ω

$$\Lambda = -\frac{4 \cdot d^2 \cdot Cos^2\psi}{D^2} - 1$$

$$\Phi = 2 \cdot d \cdot Cos\psi - \frac{4 \cdot d \cdot Cos\psi \cdot (E + C)}{D^2}$$

$$\Omega = R^2 - d^2 \cdot Cos^2\psi - \left(\frac{E + C}{D}\right)^2$$

a final interception equation of second degree is obtained:

$$x^2 \cdot \Lambda + x \cdot \Phi + \Omega = 0$$

In the solution of this equation, a negative value for x (or X_{vii}) must be found. Then the solution will be:

$$X_{vii} = \frac{-\Phi + \sqrt{\Phi^2 - 4 \cdot \Lambda \cdot \Omega}}{2 \cdot \Lambda}$$

$$Y_{vii} = -\sqrt{R^2 - X_{vii}^2}\left(\frac{a}{R}\right)$$

$$X_{vi} = \frac{Y_v - Y_{vii} + Tan\psi \cdot X_v + Tan(90-\psi) \cdot X_{vii}}{Tan(90-\psi) + Tan\psi}$$

$$Y_{vi} = -Tan\psi \cdot (X_{vi} - X_v) + Y_v$$

By finding the interception points, the scallop height (Sh) may be known by calculating the distance between point (X_{vi},Y_{vi}) and (X_{vii},Y_{vii})

$$Sh = \sqrt{(X_{vii} - X_{vi})^2 + (Y_{vii} - Y_{vi})^2} \qquad (16)$$

The scallop height for any cutting tool diameter, direction and material inclination for flat surfaces is then established. Now, a factor must be added to the surface curvature.

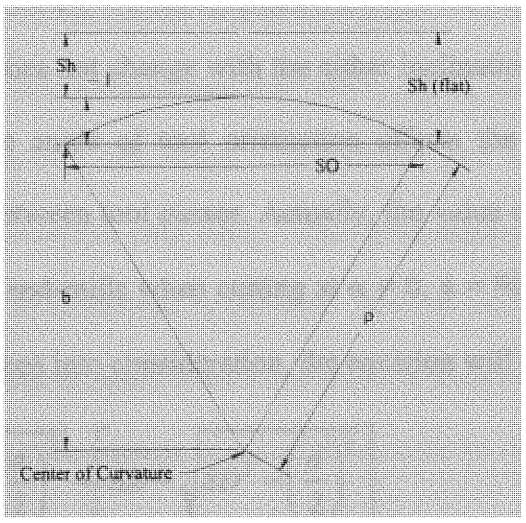

Figure 17: Calculating Curvature Factor

$$Sh = Sh(flat) - l$$
$$l = \rho - b$$
$$b = \sqrt{\rho^2 - \left(\frac{d}{2}\right)^2}$$

$$Sh = Sh(flat) - \left[\rho - \sqrt{\rho^2 - \left(\frac{d}{2}\right)^2}\right] \tag{17}$$

This is the final value of the scallop height calculation for curved surfaces. This value is used to estimate the polishing time explained in the incoming sections.

All these calculations are based under the assumption of using an end-mill. However, the program has the option of selecting a ball-mill to cut the material. In that case, calculations for the scallop height will be different.

Fortunately, calculation of the scallop height for ball-mills is much simpler than for end-mills. The semi-spherical shape that this mill type has makes the derivation of equations easy. The scallop shape does not change with the either inclination or cutting angle. The only matter in which the angle of tool orientation affects the scallop height in this situation is the distance between tool passes. Actually, the value of the scallop height for ball-mills is the same for end-mills when cutting at $\alpha = 0$, $\phi = 90$ degrees. Therefore, if the surface curvature is taken into consideration, the equation will be:

$$Sh = \left(\frac{TD}{2}\right) - \sqrt{\left(\frac{TD}{2}\right)^2 - \left(\frac{d}{2}\right)^2} - \left[SD - \sqrt{SD^2 - \left(\frac{d}{2}\right)^2}\right] \tag{18}$$

where TD is the tool diameter, SD is the lens diameter, and d is the distance between the tool's centers from one pass to the other. The variable d will depend on the step over and the relative angle between the tool and the part (See Equation 14).

FEED RATE AND STEP OVER RELATIONSHIP

Finding the optimal feed rate is a very important parameter in the machining process. As previously mentioned before, increasing the feed rate will reduce the time of cutting a part, and therefore save money. However, if the process is run too fast, the cutting forces, vibrations, or increased temperatures may cause damage to the tool or the work piece. Damaging the tool will cause losing as much or more money than first being saved.

Some CNC programmers consider that it is too risky to assign an optimum feed rate in a program because unexpected circumstances may threaten the cutting tool. It can be either a material defect, or a chip binding up when making an orifice, or vibrations at the tool due to tool wear. Therefore, even though the optimal value is the maximum speed the process can handle without failure, many machine operators just give a conservative value to the feed rate to keep the system in safe operation.[2]

In the NPLSGA program the optimum feed rate values will be calculated, monitoring that they do not exceed the defined safe range. The variable used to monitor the maximum feed rate would be the allowable chip load. This value depends on the tool and work piece material, and it has to be entered by the user. Most of this thesis work will be used for cutting acrylic with carbide tools. Then this value will be given as a default.

The chip load is defined as the amount of material removed in each cut. If the tool has 4 teeth, then it will make 4 cuts per revolution, for example.

$$ChipLoad = \frac{(1 - Overlap) \cdot FR}{NT \cdot SP} \tag{19}$$

where *ChipLoad* is the allowable chip load, *FR* is the feed rate, *NT* is the number, *SP* is the spindle speed, and the Overlap is the percentage of overlap between one pass and the next (see Figure 18).

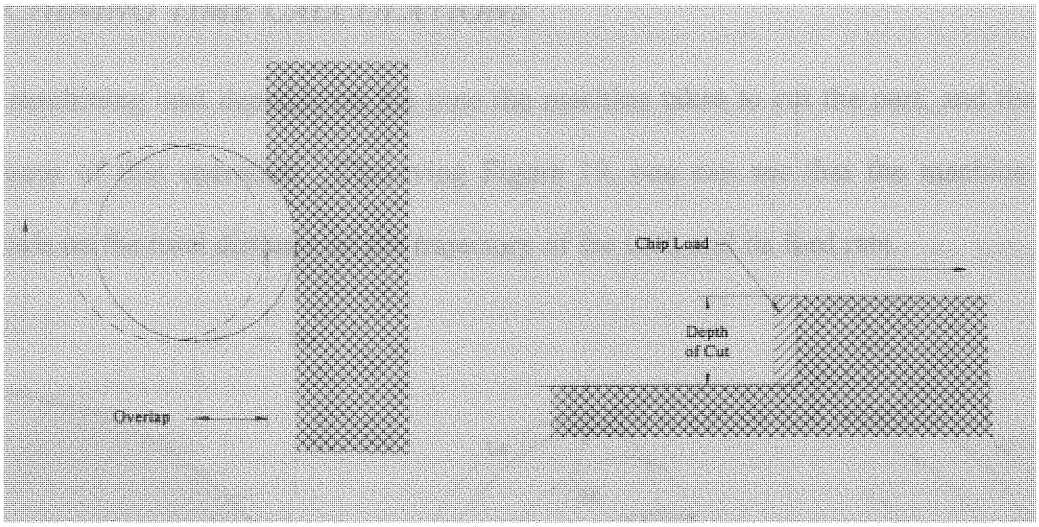

Figure 18: Chip Load and Overlap

If Equation 19 for feed fate is solved:

$$FR = \frac{ChipLoad \cdot NT \cdot SP}{(1 - Overlap)} \tag{20}$$

or if it is solved in terms of step over (*SO*) and tool diameter (*TD*),

$$FR = \frac{ChipLoad \cdot NT \cdot SP}{\left(\frac{SO}{TD/2}\right)} \tag{21}$$

A direct relationship between the feed rate and the step over can be established by knowing all the values in this equation.

Recapitulating, the feed rate, step over, and tool diameter are independent variables when calculating total manufacturing cost. A relationship between feed rate and step over has been stated. All feed rate values in the calculations can be substituted with Equation 21.

Hence, the equation system is reduced from three independent variables to two (Tool Diameter and Step Over), simplifying the optimization process.

MACHINING TIME CALCULATIONS

The machining cost depends on the time the machine takes to cut the part, and the cost per hour of this operation. Observing Figure 19, one can see that the machine time depends on the total distance on the trajectory of the tool, and its feed rate.

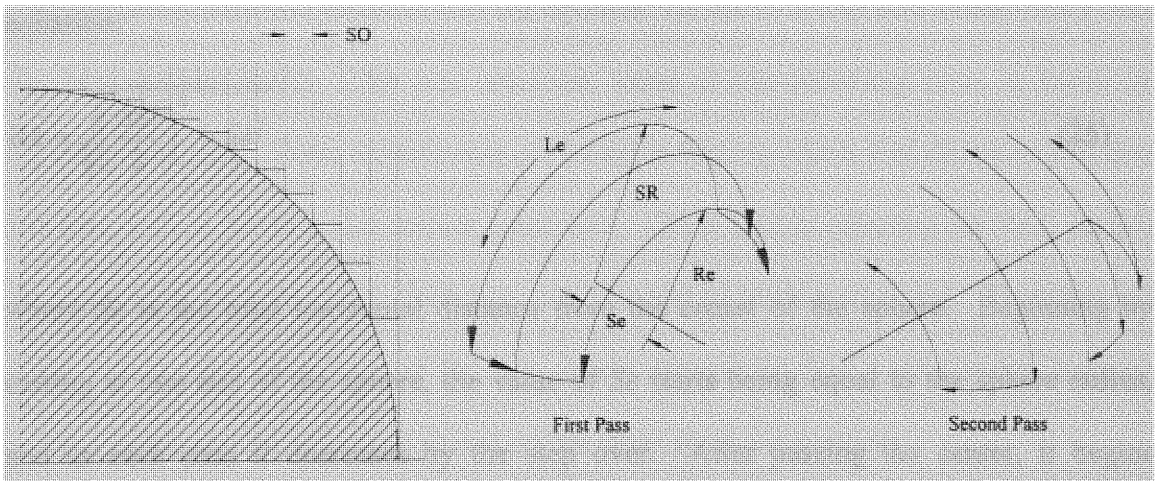

Figure 19: Number of Tool Passes in a Quarter Sphere

The normal operation to machine lenses is to give two perpendicular passes as shown. First the tool cuts in one direction, and then makes a finishing pass, perpendicular to the previous one. In this case it is assumed that a sphere is cut to half with a given diameter. To find the number of passes is very simple. Dividing the sphere diameter by the step over will yield to the number of passes in one direction. To find the total number of passes the number is multiplied by two.

$$\#\,passes_{FirstPass} = roundup\left(\frac{SD}{SO}\right) \tag{22}$$

$$\#\,passes_{SecondPass} = roundup\left(\frac{SD}{SO}\right) \tag{23}$$

$$\#\,passes_{Total} = \#\,passes_{FirstPass} + \#\,passes_{SecondPass} \tag{24}$$

To find the distance covered by the tool, the distance over each "arc" has to be calculated (see Figure 19). It is assumed that the tool is not cutting when it moves from one pass to the next. Then, the time spent to move from one arc, or pass to the next, will be depreciable. The radius of each arc as shown in the figure is found by the following equation:

$$\mathrm{Re} = \sqrt{\left(\frac{SD}{2}\right)^2 - Se^2} \tag{25}$$

where *Se* is the total horizontal distance from the center of the sphere. Beginning by cutting at one end of the sphere, the value of *Se* starts being equal to the sphere radius, and in each pass it decreases by the step over. After passing the center, it begins increasing its value again, until finishing the entire surface. If *Le* is the distance the tool moves in one pass, the value may be found by

$$Le = \pi \cdot \mathrm{Re} \tag{26}$$

then, the total distance that the tool moves cutting the surface is equal to:

$$Distance_{machining} = 2 \cdot \sum_{0}^{\#\,passes_{parallel}} Le \tag{27}$$

and the time in hours will be,

$$Time_{machining} = \frac{Distance_{machining} \cdot FR}{60} \tag{28}$$

where FR is the feed rate. Then, with these values calculated, the machining cost equation will be

$$Cost_{machining} = HMC \cdot Time_{machining}$$

where *HMC* is the hourly cost of machining per hour.

TOOL DIAMETER SUGGESTION

The NPLSGA software recommends the operator the use of a tool that would result with the best surface finish. The tool selection, same as the polishing time, is based on experimental data. These experiments are based in finding which tool diameter will give the best surface finish under certain conditions. Other factors affecting the final surface are chip load, spindle speed, feed rate, and number of teeth in the tool. Therefore, all these values were recorded and fed to a neural network to find a relationship between surface finish and tool diameter, based on the other given parameters.

The neural network chosen to find the tool diameter had to have three input nodes and one output node. The input nodes represent the spindle's revolutions per minute, the allowable chip load, and the machine's feed rate. The output node would represent the tool diameter. To find the right number of hidden nodes different numbers had to be tried until finding the most suitable system, which happens to be four hidden nodes. Configuration of the network was made under following parameters:

- Output Threshold = 0.5
- Learning Threshold = 0.00001
- Hidden Nodes = 4

- Learning Rate = 0.6
- Momentum = 0.9
- Random selection of cases for learning

The output and learning threshold are constants that affect the activation functions in each individual node. Depending on these values, the reaction of each node to a defined input would be different. The number of hidden nodes helps to increase the complexity of the neural network. The more hidden nodes there are, the best learning and estimation the network would have for certain situations. However, sometimes the network can be too complex for the problems it is solving, taking unnecessary extra time and giving the same result as a simpler multi-layer network.

The learning rate is a gain factor that the network uses to modify its weighting factors, after the error from the desired value is calculated. The larger the learning factor is, the faster it will change the weighting factor on each node. If the global error is large, a large learning rate is recommended, and decreases as the error does.[12] See the Theoretical Background Section for more details of each of these variables.

POLISHING TIME CALCULATIONS

Polishing time is calculated based only on the scallop height. Because it is a single function dependable of one independent variable, its estimation can be done in different procedures. The simpler manner to estimate polishing time, based on scallop height values, is to curve-fit the values obtained through experimentation. The procedure to obtain these pairs of values is explained in the following chapter called "Experimental Setup."

All the obtained cases will be considered as points on a x,y plot, so that a function y = (x) can be found. For example there can be ten different experimental cases as shown in Table 1.

Scallop Heights	Polishing time per unit area	P(x,y)
Sh1	Pt1	(Sh1,Pt1)
Sh2	Pt2	(Sh2,Pt2)
Sh3	Pt3	(Sh3,Pt3)
Sh4	Pt4	(Sh4,Pt4)
Sh5	Pt5	(Sh5,Pt5)
Sh6	Pt6	(Sh6,Pt6)
Sh7	Pt7	(Sh7,Pt7)
Sh8	Pt8	(Sh8,Pt8)
Sh9	Pt9	(Sh9,Pt9)
Sh10	Pt10	(Sh10,Pt10)

Table 1: Polishing Time Experimental Values to be Curve-fitted

Points P(x,y) will be used to obtain an approximated function y = f(x), that will represent the polishing time function:

$$\frac{Time_{Polishing}}{Area} = f_{Curve-fitted}(Sh) \tag{29}$$

A curve-fitted function will be easier to work with, and will provide a faster response time. The use of a neural network instead has other advantages. For example, if the finish tool is fed faster than recommended, it could leave some extra material, chips, and burr that will require extra polishing. Even though these extra parameters are not being considered as factors in this thesis work, feed rate, chip load, and other machining factors may affect the surface finish of a part. At the moment the main factor being considered is the scallop height. However, if a neural network is used instead of a curve-fitted equation, it could be modified to accept some extra variables to provide a more precise estimation of polishing time.

Hence, estimations for polishing time were made to observe the accuracy and response time of each method to calculate polishing time per square area.

OPTIMIZATION

After selecting the appropriate feed rate and tool diameter through estimations and mathematical relationships, the process of optimization simplifies significantly. A multi-dimensional problem to minimize the total cost function dependent of three variables (feed rate, tool diameter and step over) is now reduced to a one-dimensional optimization problem. Figure 20 shows the sequence of procedures that must be done to calculate and minimize the total manufacturing cost by numerically locating the optimal step over value.

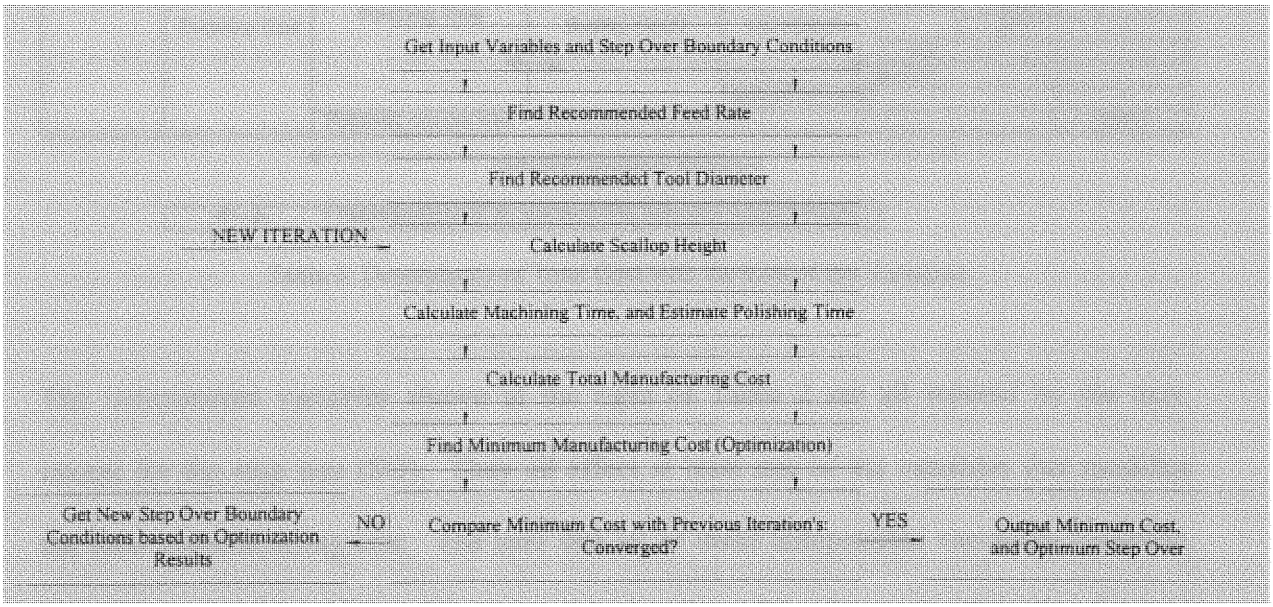

Figure 20: Simplified Flow Chart to Calculate and Minimize Total Cost

First, the tool diameter and the feed rate are fixed to recommended and feasible values as explained in previous sections. Then the scallop height is calculated based on an initial

step over value. From there, the total cost can be estimated, and later minimize by iterating the scallop height until the polishing cost and the machining cost will meet in a trade off point.

All the neural networks and the series of mathematical expressions make possible to represent the total manufacturing cost of the acrylic lens as a series of functions that are all dependent on the one independent variable: the step size. Refer to this following figure to observe the relationship between the total cost and the step over.

Dependency of Variables in order to find the Total Manufacturing Cost from Independent Parameters

Variable to Minimize							Independent Variables
Total Cost	Machining Cost	Machining time	Feed Rate	Chip Load			Chipload
				# teeth			NT
				max RPM			rpm
			Displacement of Tool	Sphere Diameter			SD
				Step size			SO
		Machining rate per hour					M$/hr
	Polishing Cost	Polishing Time	Surface Area	Sphere Diameter			-
			Scallop Height	Step size			-
				tool diameter	feedrate	Chip Load	-
						# teeth	-
						rpm	-
					SD		-
					rpm		-
				Sphere Diameter			-
		Polishing rate per hour					P$/hr.

Variables Calculation Sequence

= Representation of an operation that estimates output variable.

Table 2: Dependency of Total Cost with the Rest of the Variables

The independent variables shown at the right of Table 2 are all entered by the user, except for the step size (SO). They all affect at least one of the variables that modifies the total manufacturing cost. The user sets all of these independent variables; therefore, the software has to optimize the total cost in function of the step over. Note that the total cost of manufacturing is the sum of the machining cost plus the polishing cost (see the

same figure from left to right). Machining and polishing costs depend on the time taken to execute the process, and its hourly cost.

The machining time can be calculated from the speed that the tool is cutting, and the distance that the tool displaces around the work-piece. The speed of the tool is represented by the feed rate, and the total distance the tool moves can be calculated using the lens diameter. The feed rate would depend on the allowable chip load, the tool's number of teeth and the speed the spindle is operating.

The polishing time depends on the surface roughness and the total area that needs to be polished. The surface roughness is quantified by the scallop height, and its relation with the polishing time has to be estimated through a neural network. This neural network would learn several sample cases found through experimentation. Note that all relationships found through estimation, like a neural network, are represented in Table 2 as a double arrow ($\leftrightarrow$). The scallop height depends subsequently on the tool diameter, the step size, and the lens curvature. An estimation of the most appropriate tool diameter is described in the previous chapter, and expressed as a non-explicit function of the chip load, feed rate and tool's revolutions per minute. Therefore, the total cost will vary only with changes in the step size. If the step size increases, the scallop height will be larger and the surface will be rougher. Even though the machining process would take less time, the polishing time will take longer. The time of one process with respect to the other gets weighted in order to find the minimal manufacturing cost. If instead of minimizing the cost, the software user would like minimize the manufacturing time, regardless of its cost, he or she would need to se the machining rate per hour equal to the polishing rate per hour. Then, the optimization process by itself is simplified to the level

of minimizing a function with one independent variable. There are several different methods that may be used to minimize the studied function. The technique used in this work is similar to the bisection method with some modifications. See the next figure for a graphical representation of this technique.

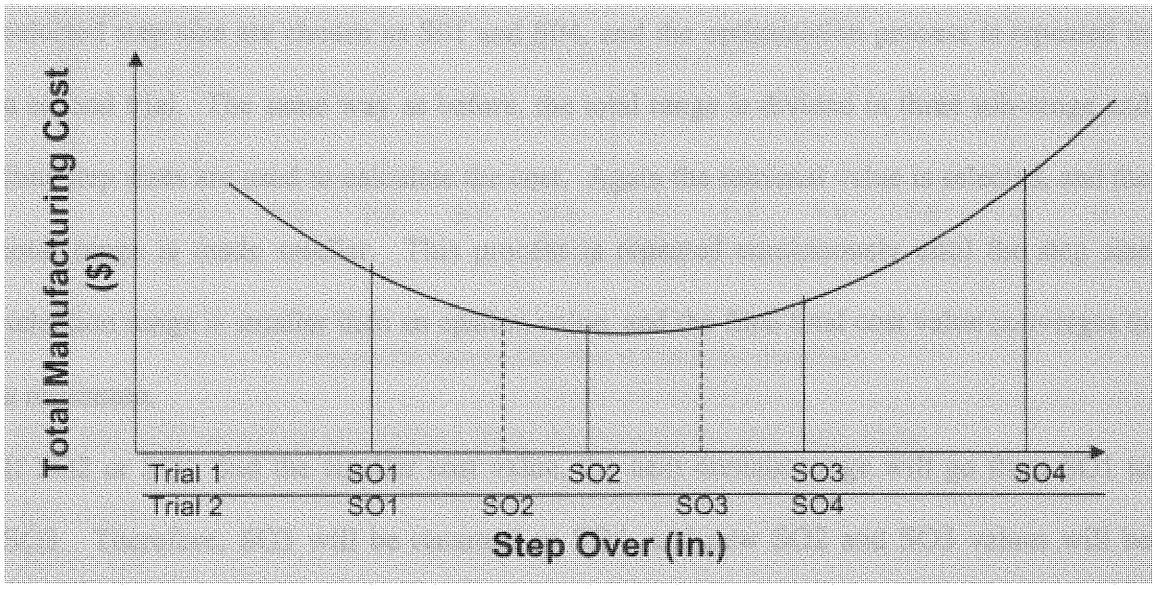

Figure 21: Total Manufacturing Cost vs. Machine Step Size

Figure 21 represents the function cost vs. step over, where its minimal value is at an unknown location. It is assumed that the cost in function of the step over has only one single global minimum, and there are not local minimums. Because of the fact the tool diameter and the feed rate are fixed to real possible values, it is less likely to find local minimum in the function due to non-linearity.

In order to find this minimal value, the outer boundaries, in which the step over value is physically feasible, must be set. The step over has to be smaller than the tool diameter and larger than the minimal possibly done by the machine used. Those boundaries are represented as SO1 and SO4 in the last figure. Next, the range from these values gets

divided in three sub-ranges. That is how SO2 and SO3 are found at the interception of each sub-range. After that, the total manufacturing cost is calculated for each of these values, and the smaller total cost is identified. In this example, SO2 will find the smaller total cost between all four step overs. Then, the closest left and right boundary to the selected step size are chosen as S01 and S04, and the optimization process is repeated in a second trial. The same way as before, the total range is divided in three sub-range and four step sizes are used to calculate the cost. Again the minimal value is selected, and the total range is made smaller. This process is repeated continuously until the step size selection range is smaller than an epsilon value, and the final total cost value converges to a minimum.

If SO2 or SO1 find the smaller cost value, the next domain range will go from SO1 to SO3. Therefore, SO3 will be renamed as SO4, and new SO2 and SO3 values will be found. Instead, if SO3 or SO4 find the minimal function point, the new range will go from SO2 to SO4. In the same way, SO2 will be renamed to SO1, and new step sizes for SO2 and SO3 will be found.

Example:

Assuming that the function to optimize would look as:

$$Cost_{Total} = (SO - 1)^2 + 0.5 \tag{30}$$

this function has a minimum at SO = 1. Assuming that a total cost difference of 1 cent (ε = 0.01) will be enough to converge the optimization. Then, the next step to optimize numerically this function is to set the boundaries as shown in Figure 22, to SO1 = 0.4 and SO4 = 1.8. These boundaries are set by known physical constrains.

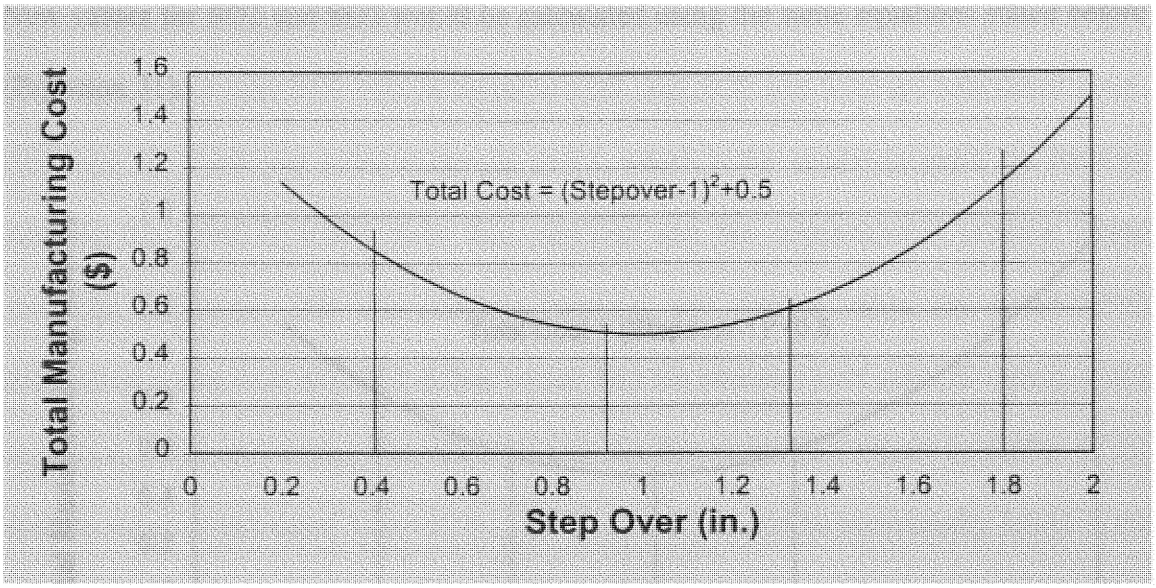

Figure 22: Sample Function to Optimize, Iteration 1

SO3 and SO2 would be equidistant from the boundaries and each other.

$$SO1 = 0.4$$
$$SO4 = 1.8$$
$$SO2 = SO1 + \left(\frac{SO4 - SO1}{3}\right) = 0.867 \tag{31}$$
$$SO3 = SO1 + 2 \cdot \left(\frac{SO4 - SO1}{3}\right) = 1.333$$

Afterwards, the total cost for each point is calculated and the minimal value is used to reduce the searching region.

$$Iteration = 1:$$
$$Cost^1_{Total} = (SO1 - 1)^2 + 0.5 = 0.86$$
$$Cost^2_{Total} = (SO2 - 1)^2 + 0.5 = 0.518$$
$$Cost^3_{Total} = (SO3 - 1)^2 + 0.5 = 0.611$$
$$Cost^4_{Total} = (SO4 - 1)^2 + 0.5 = 1.14$$

In this example the minimum cost is the one calculated from SO2, as seen in Figure 22. Now the search region will be from SO1 to SO3, and the SO3 value will be the new SO4.

New values for SO2 and SO3 would be calculated using the same equations stated before (Equation 31).

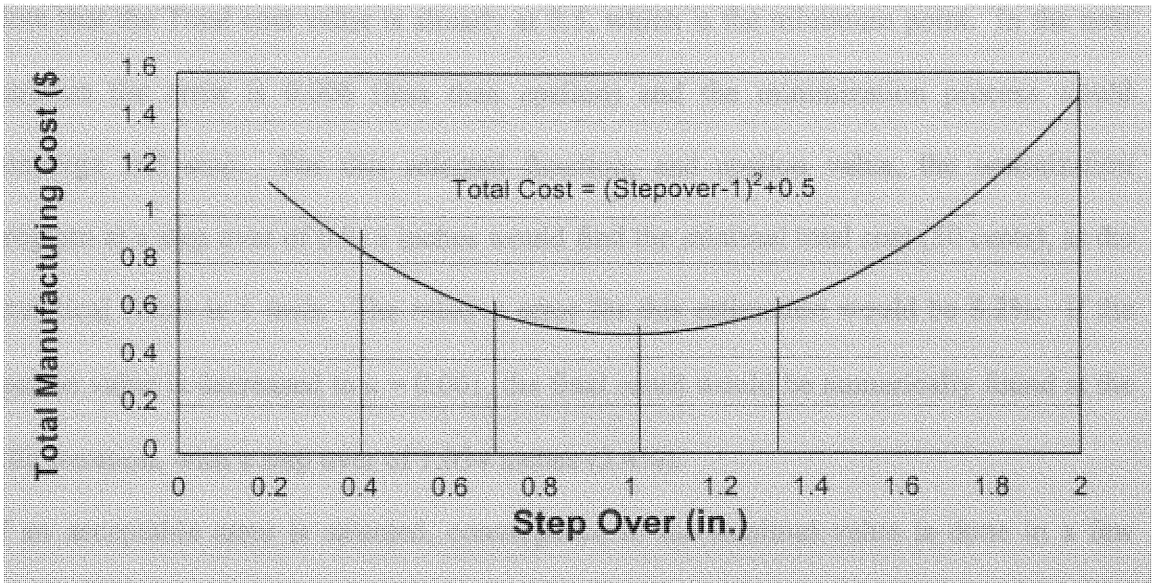

Figure 23: Sample Function to Optimize, Iteration 2

$Iteration = 2:$

$SO1 = 0.4; Cost^{1}_{Total} = 0.86$

$SO2 = 0.711; Cost^{2}_{Total} = 0.583$

$SO3 = 1.022; Cost^{3}_{Total} = 0.500$

$SO4 = 1.333; Cost^{4}_{Total} = 0.611$

The minimum on this iteration is located at SO3. Then, before continuing with a third iteration the optimization process has to be checked for convergence. To check for convergence, the difference between the minimum of this trial and the previous one has to be less than epsilon.

$$\left|Cost^{3}_{i=2} - Cost^{2}_{i=1}\right| < \varepsilon$$

$$\left|0.500 - 0.518\right| < 0.01$$

$$0.018 < 0.01 \leftarrow false$$

In this case the difference is larger than epsilon, the boundaries values have to be changed once more. The new SO1 will be equal to SO2, and SO2 and SO3 are calculated again. Note that the Total cost value is already at 0.5, which is the minimum point. In the next iteration the error is 0.006 (less than epsilon) and the minimization process would converge and stop. For this selected function, and epsilon value, the optimization program at a step over of 0.918 inches would find the minimal total cost (50 cents), as the analytical result is 1 inch. The relative error for this estimation would be 8.2%. If the epsilon value is decreased to $\varepsilon = 0.001$, the final result would be numerically found at the fifth iteration, with a step over of 1.034 inches (3.4%).

If an acceptable error is selected, convergence to the minimal value is done in a few iterations. This is a desirable factor when implementing the algorithm to the program, because it would require less computer resources, giving a better response performance.

CHAPTER IV

COMPUTER PROGRAM DEVELOPMENT

The Visual Basic code development is targeted to help the programmer select the cutting parameters consciously to create the desired lens at the first trial.

The software is designed to occupy the minimal space on the screen, and to use minimal hardware resources, so that it can always be running at the same time with other CAD/CAM applications. This will offer the user fast reference when selecting manufacturing configurations like the tool diameter, step size and feed rate, that are usually parameters estimated by the operator's experience. The new software creates a mathematical tool to select these parameters to optimize manufacturing costs for lenses.

The program is created in a modular style, so it can be easily modified to be used in other manufacturing processes. Nevertheless, the main optimization process would remain the same. Operational costs of any manufacturing procedure will be calculated using a combination of mathematical models and neural network estimations when necessary, or when direct analytical models would get too complicated. Then, those costs will be added together. These costs would fall into the optimization process to find the most suitable independent variables to minimize the total manufacturing operational cost.

In this specific case, NPLSGA is targeted to manufacture near-perfect prototype lenses. The procedures are reduced only to end milling and polishing, however, if necessary, the software is designed in a way that it could be expanded to include a large quantity of manufacturing processes. It will then calculate the operational cost using the most efficient tool, and would optimize the required variables, minimizing total expenditure.

To find the correct way to calculate one manufacturing process cost, one has to evaluate the following:

1. Is there a direct analytical relationship between the variables to be optimized and the operation cost?
2. If there is one: Is it complicated in a level that would affect the software performance?
3. Is there available substantial data that relates the independent variables with the final operation cost?
4. If not: Is it possible to build an experimental to find such data, to train a neural network?

Depending on the answers to these questions it would be decided weather the cost would be calculated with analytical equations or with estimations given by the neural network.

USER INTERFACE

As mentioned in the section before, the user interface is designed to be as simple as possible to the user, so the application could be referenced at any time during the design process. It shows only the required information, including inputs and outputs, giving default values in optional entries, like chip load, or machining and polishing costs.

In the main window the geometry of the lens is defined by the sphere diameter. Also, it is required to know the spindle revolution speed, and the number of flutes in the tool.

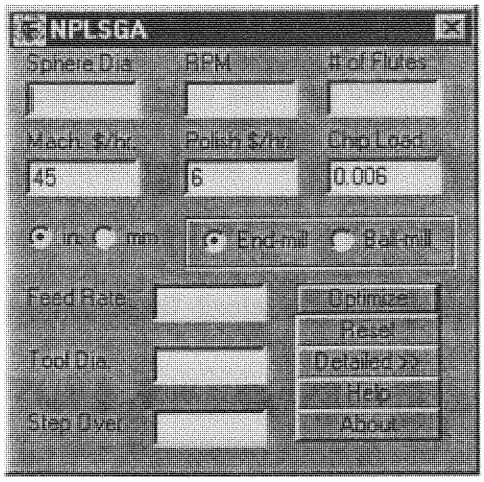

Figure 24: Main Window of the Near-Perfect Lens Surface Generation Advisor (NPLSGA)

From the point of view of the user, the process of optimization is as simple as pressing the "Optimize" button. All calculations discussed in the last chapter are used to obtain the best suitable values for feed rate, tool diameter, and step size.

The values used are in the imperial system, distances being measured in inches, and feed rate in inches per minute. If the option button of millimeters is selected, all shown numbers will be translated into the international measuring system, and then all values will be treated as millimeters or millimeters per minute. The user can change the measuring system at anytime, but every time the software starts it will default to the imperial system.

If any of the values entered is not physically possible or real, an error message box will let the user know that value is illegal. No calculations will be done if there is an error in the input values. To clear the error, the user may just change the problem value, or click on the command button "Reset", and all existing values will be cleared or returned to default. If all values are correct the operator may click on the "Optimize" button to get

the minimal operational cost parameters. The new parameter will overwrite the previous ones, showing the user all parameters he or she must use for the specific lens design.

If the operator wants to know more detailed information about the optimization process, the "Detailed >>" button may be clicked and the total cost of the part will show, as well as scallop height, distance that the tool displaces, polished surface area, total machining and polishing time, and cost.

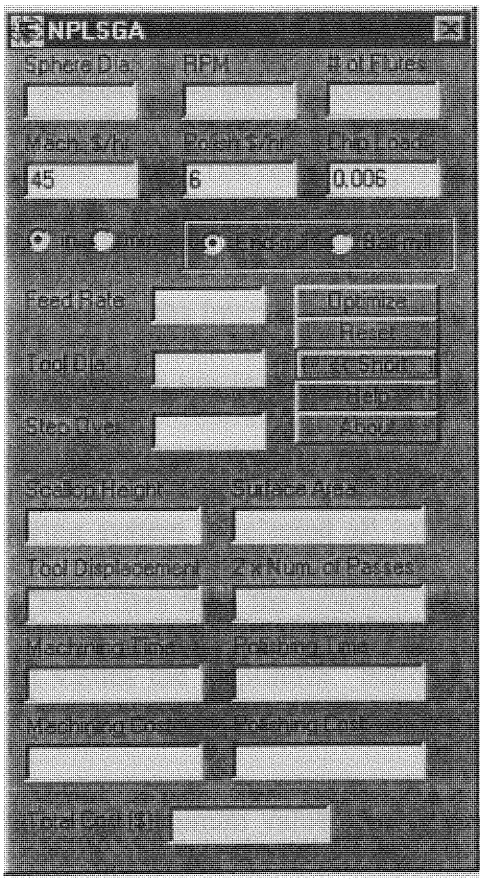

Figure 25: Detailed View Showing Extra Optimization Parameters

To go back into the simplified window the user just needs to click on the "<< Short" button and it will come back to its original display.

The operator is allowed to make changes only in the first six text boxes, which are: Sphere diameter (referencing to the lens diameter), RPM (spindle speed), # flutes (tool's

number of teeth), Machining hourly rate, Polishing hourly rate, and allowable chip load in inches per tooth. Sphere diameter and chip load can be entered as fractions and the software will understand the input. However, if fractions are entered in the RPM box, any of the rates, or number of flutes box, the program will display an error.

Another variable that the operator is allowed to change is the machining processes, if it uses an end-mill or a ball-mill. The default value will be end-mill because it gives a better match to the surface area, reducing the machining time. However, if the operator wishes to use a ball-mill, the scallop height will be calculated for this type of tool, and the calculations will follow using this value. The rest of the parameters, from the feed rate to the total cost, are given by the software and cannot be edited.

The user may also select to either use the international or imperial system when selecting either "in" or "mm". The default is to use inches at the start of the program. It can be easily changed to millimeters at the code level if the user requests it to the software developer. If another measuring system is selected all values in inches are changed to millimeters automatically. That differs from the selection of either end-mill or ball-mill, whereas to update the result values the "Optimize" button has to be pressed after the change has been done.

Feed rate text box will display the optimum value after the optimization process. The presented value will be greater than zero and under the maximum CNC and work-holder capacity. The same concept is used for the tool diameter and the step size. The step size will be smaller than the tool diameter, and larger than the minimum possible CNC displacement. The tool diameter will be the next available tool size from the calculated

recommended value. For example, if the program calculates a tool diameter of 0.3745325 inch, the software will select a 3/8" mill.

Additional information is displayed when selecting a detailed view. This is helpful to see how the entered values modify the final cost of the work-piece. The operator can obtain from the program the maximum scallop height, the total surface area, the number of passes and the distance the tool moved through those passes, and most important; the machining and polishing time and expense.

Other buttons are available, like "Help", and "About". The "About" button will display the version of the program, the authors, a brief introduction to its purpose, and system information about the computer. From the system information the user can obtain information about the computer's hardware resources, components, software environment and applications.

The "Help" button calls a Help Window that provides full support to the user in order to use and troubleshoot the program. The help window also guides the user in how to use the software, the algorithms used in the program, and what are the most common causes to simple problems and their solutions.

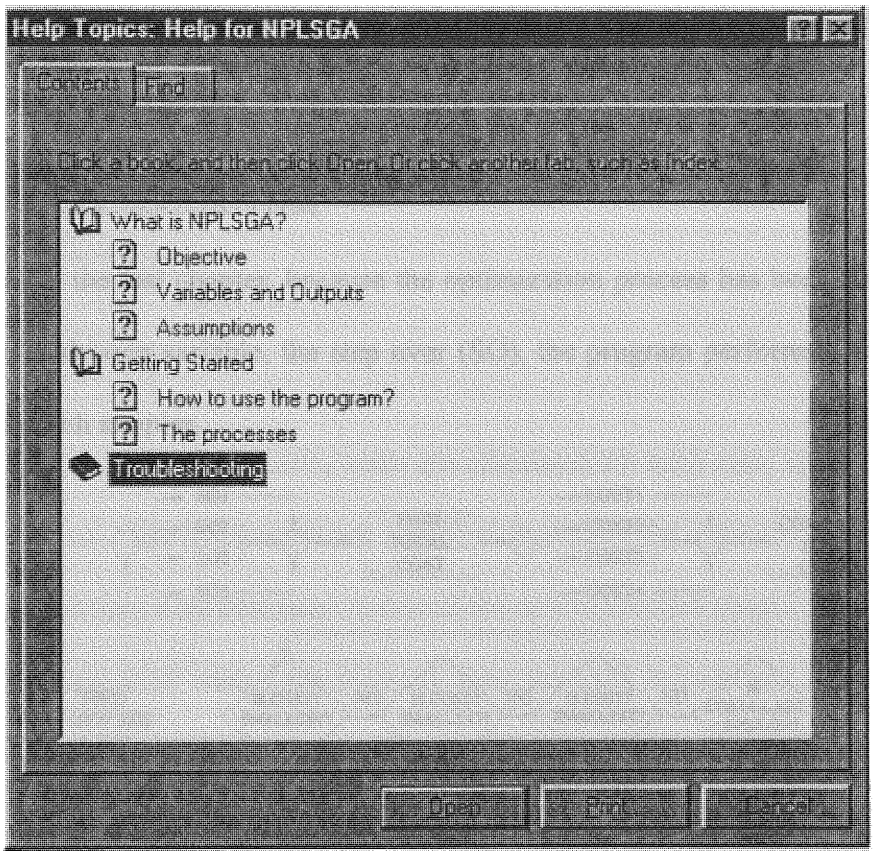

Figure 26: Contents in the NPLSHA Help Window

As any good windows program the Help Window provides real-time information of how to use the program in an appropriate manner. If the answer to the operator questions does not show in the standard menu contents items, the user can click in the Find Tab and search by keyword instead of content title. Most people that have used Windows operating system are familiar with this type of Help Windows.

ALGORITHMS

The algorithms and processes used in this program have been explained in the previous chapters when deriving the equations. However, in this section the final algorithms will

be explained in detail. First, the optimization algorithm will be reviewed, and the process used to minimize the total cost of manufacturing as well.

OPTIMIZATION ALGORITHM

As explained in the last chapter, because the optimization process has been simplified to a function of only one variable, the step over (SO), the program performs faster and the algorithm is much simpler.

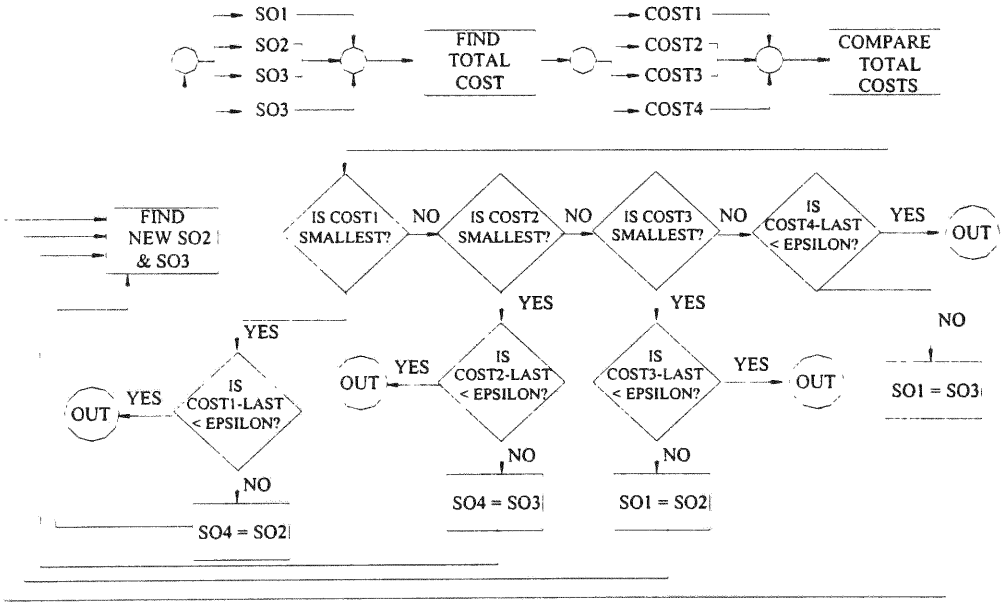

Figure 27: Optimization Algorithm

Following the flow chart in Figure 27, the first step that the software executes is finding the boundaries for the step over (SO1 and SO4). The minimal value for step over would be the smallest distance that the CNC can precisely move. The software has its setup for a value of SO1 = 0.001, which it can be changed if the operator requests smaller step sizes. The maximum step size would be the tool diameter itself, SO4 = TD. It is

assumed that there will always be an overlap between tool passes, therefore, the step over has to be less than the tool diameter.

Set that SO2 and SO3 are calculated as follows:

$$SO2 = SO1 + \left(\frac{SO4 - SO1}{3} \right)$$

$$SO3 = SO1 + 2 \cdot \left(\frac{SO4 - SO1}{3} \right)$$

Then, individual total manufacturing costs are calculated for each case, and the minimum value is identified. The boundaries are modified to reduce the selection domain, always keeping the minimum total cost point inside of the new optimized range.

Once the four step over values are selected, they are used to calculate the total cost at each condition. Basically, the all derived and approximated equations are run four times at each situation, and the algorithm will calculate the total costs. These costs are compared with each other to find the step over that will originate the minimum condition between the four values selected.

Once the minimum total cost is found, its value is compared with the minimum cost found in the previous iteration. If the difference between the two values is less than an accepted error ε, then the optimization process has converged, the total manufacturing cost and the optimal step size are assumed to be the last values found.

i.e., if ε = 0.001, then if:

$$\left| Min.Cost_{TOTAL}^{I} - Min.Cost_{TOTAL}^{i-1} \right| < \varepsilon \qquad (32)$$

The optimization process has converged. Thereafter, a new range for step sizes needs to be selected, repeating the processes in a new iteration. Refer to Figure 27 to see how a new range is selected.

Supposing that the process is in the ith iteration, and values for $SO1^i$ to $SO4^i$ have been selected as $SO1^i = 1$, $SO2^i = 2$, $SO3^i = 3$, and $SO4^i = 4$. Afterwards, total costs are calculated and they are compared to previous values to check if have converged as shown in Equation 32. If the error is still not considered small enough, and it calls for a new iteration, then the new range would be selected as the following:

- If $SO1^i$ finds the minimum cost, then the new range is from 1 to 2:

  $SO1^{i+1} = 1$, $SO2^{i+1} = 1.33$, $SO3^{i+1} = 167$, and $SO4^{i+1} = 2$.

- If $SO2^i$ finds the minimum cost, then the new range is from 1 to 3:

  $SO1^{i+1} = 1$, $SO2^{i+1} = 1.67$, $SO3^{i+1} = 2.33$, and $SO4^{i+1} = 3$.

- If $SO3^i$ finds the minimum cost, then the new range is from 2 to 4:

  $SO1^{i+1} = 2$, $SO2^{i+1} = 2.67$, $SO3^{i+1} .= 3.33$, and $SO4^{i+1} = 4$.

- If $SO4^i$ finds the minimum cost, then the new range is from 3 to 4:

  $SO1^{i+1} = 3$, $SO2^{i+1} = 3.33$, $SO3^{i+1} = 3.67$, and $SO4^{i+1} = 4$.

Once the new range is selected, total costs are calculated once again to continue with the iteration process.

SCALLOP HEIGHT ALGORITHM

The scallop height algorithm is a straightforward procedure, which is based in the equations given in previous chapters. The scallop height is calculated through a series of geometric equations. The scallop is considered the projected area between two cylinders

(if using end-mills) of radius R, separated by a distance d, in an angle ψ relative to a surface that has a radius of curvature ρ. Each value depends on the information given by the user, and they are calculated as follows:

$$R = \frac{TD}{2} \tag{33}$$

$$\psi = \arctan(\tan\phi \cdot \sin\alpha) \tag{34}$$

$$d = \frac{SO}{Cos\,\psi} \tag{35}$$

where TD is the cutting tool diameter, ϕ is the slope angle of the surface, α is the cutting angle, and SO is the step over. These values are then used into the following equations to find the scallop height:

$a = R * Sin(\phi) * Cos(\alpha)$

at this point, if the value of ϕ is zero, the scallop height is already estimated to be zero. Notice that is $\phi = 0$, and an end-mill will have a perfect match with the surface, leaving no scallop.

$\psi = \arctan((Tan(\phi) * Sin(\alpha)))$

$\theta = \arctan(Tan((\pi / 2) - \phi) * Sin(\alpha))$

$H = R * Cos((\pi / 2) - \phi)$

$Xiii = d * Cos(\psi) - R$

$Yii = -d * Sin(\psi)$

$Yiii = -Sqr(R * R - Xiii * Xiii) * (a / R)$

$Xv = -H * Sin(\psi)$

$Yv = -H * Cos(\psi)$

Now, the projected area for each cylinder's face will be more an ellipse than a circle, because there is a relative angle between the surface and the cutting tool. To check if the ellipses are intercepting or not a comparison like this may be made.

$Yiii > Yii$ (see Figure 15 in previous chapter).

If it is true, then the elliptical projections would not be intercepting and the scallop height would be calculated as follows:

$$Xvi = (Yv - Yiii + Tan(\psi) * Xv + Tan((\pi / 2) - \psi) * Xiii) / (Tan((\pi / 2) - \psi) + Tan(\psi))$$

$$Yvi = -Tan(\psi) * (Xvi - Xv) + Yv$$

$$\text{Scallop height} = Sqr((Xiii - Xvi) ^ 2 + (Yiii - Yvi) ^ 2) - \rho$$

Else, if the two ellipses are intercepting, the scallop height is calculated as:

$$CC = (R / a) ^ 2 * d ^ 2 * (Sin(\psi)) ^ 2$$

$$DD = (2 * R * d * Sin(\psi)) / a$$

$$EE = -(d ^ 2 * (Cos(\psi)) ^ 2)$$

$$AAA = (-(4 * d ^ 2 * (Cos(\psi)) ^ 2) / (DD ^ 2)) - 1$$

$$BBB = 2 * d * Cos(\psi) - (4 * d * Cos(\psi) * (EE + CC)) / (DD ^ 2)$$

$$CCC = R ^ 2 - d ^ 2 * (Cos(\psi)) ^ 2 - ((EE + CC) / DD) ^ 2$$

$$Xvii = (-BBB + Sqr((BBB ^ 2) - 4 * AAA * CCC)) / (2 * AAA)$$

$$Yvii = -Sqr(R ^ 2 - Xvii ^ 2) * (a / R)$$

$$Xvi = (Yv - Yvii + Tan(\psi) * Xv + Tan((\pi / 2) - \psi) * Xvii) / (Tan((\pi / 2) - \psi) + Tan(\psi\psi))$$

$$Yvi = -Tan(\psi) * (Xvi - Xv) + Yv$$

$$\text{Scallop height} = Sqr((Xvii - Xvi) ^ 2 + (Yvii - Yvi) ^ 2) - \rho$$

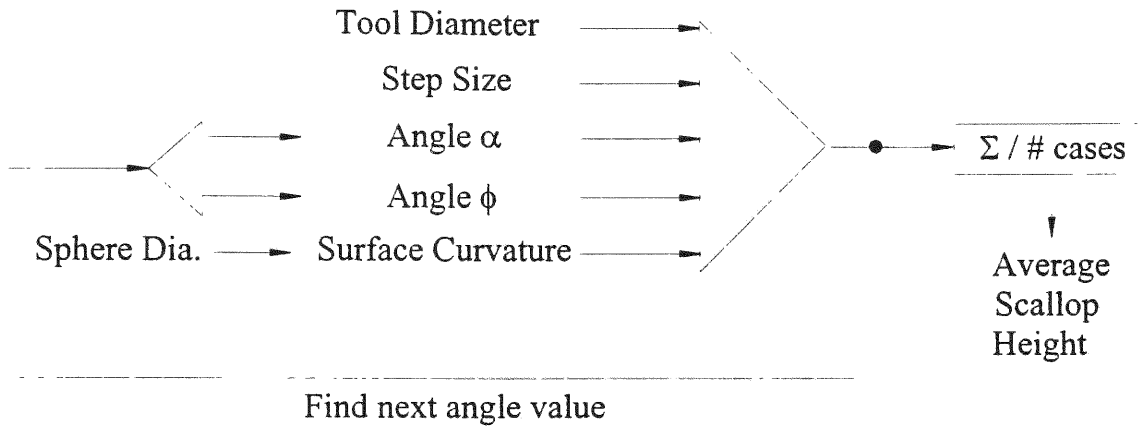

Figure 28: Average Scallop Height Calculation Flow Chart.

These series of equations would provide the scallop height at a determined point on a curved lens surface. The software repeats this process several times at different points throughout the work-piece surface, and takes the average as the representative scallop height.

MACHINING TIME ALGORITHM

Machining time depends on the distance the tool moves, and also its speed. When working with curved surfaces the machining displacement will consist on a series of arcs with different diameter (see Figure 19). The variables to deal in these calculations are the radius of each arc, the circumferences and their sum.

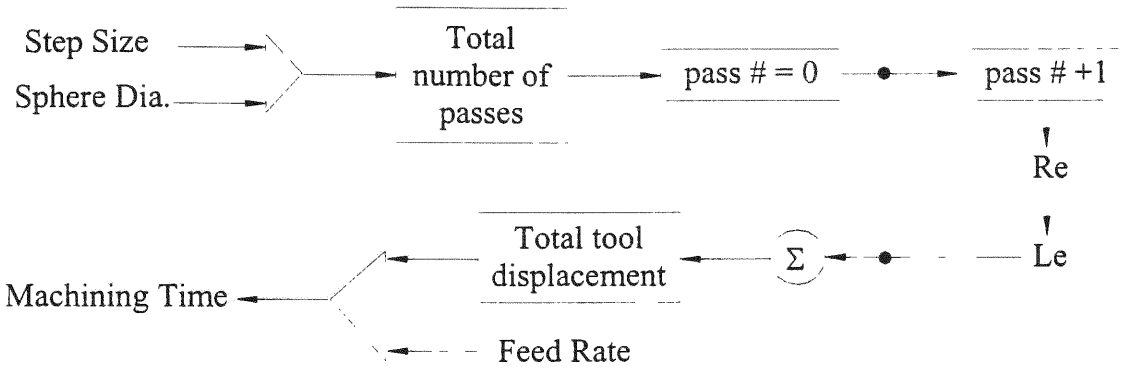

Figure 29: Machining Time Calculation Flow Chart

The finishing process cuts the lens surfaces in two perpendicular directions, as shown in Figure 19 in previous chapters. To find the total number of passes in one direction, the program divides the sphere diameter by the step size, and uses the rounded up approximation.

Then, for each pass, the arc's radius (Re) and later the circumference (Le) is calculated, to find the tool displacement per pass:

$$Re = Sqr((SD / 2) \wedge 2 - Se \wedge 2)$$

$$Le = \pi * Re$$

where SD is the sphere diameter, and Se is the horizontal distance the tool has moved, which would be equal to:

$$Se = Pass\# * Step\ over$$

The total displacement is the sum of all values of Le.

$$Displacement = \Sigma\ Le$$

Finally, the total machining cost is found dividing the tool displacement by the feed rate, and multiplying the result by the machining cost per hour current rate.

POLISHING TIME ALGORITHM

Total hand polishing is dependable of the scallop height and the surface area to refine. Two techniques can be used in order to estimate time of operations. The experimental data can be either curve-fitted or used to train a multi-layer neural network.

A curve-fit of the experimental data will provide a direct equation that will relate the scallop height and the polishing time. This direct relationship will be better at the software level, because it would reduce the calculations time, and utilize less computer resources. However, this approximated equation will only work for the conditions assumed at the beginning of the program. This creates a problem if the user wants to add extra factors that would modify the polishing time. For example, if the user decides that he or she wants to add the feed rate as a factor that affects the polishing procedure, the function of polishing time would depend on two independent variables. This would inevitably increase the difficulty of the curve fitting procedure. As more factors are evaluated to be important in the polishing time estimation equation, to find an approximated empiric equation is harder to obtain.

A neural network, when approximating a relationship between one input and one output, does not provide better results than a curve-fitted equation, but it does provide flexibility if more independent variables are needed. Due to its flexibility, the NPLSGA software uses neural network's algorithms to find the relation between scallop height and polishing time. It takes more computer computation time than the other process, but the difference in time is almost unnoticeable by the user.

Thus, the program has the flexibility of estimating the polishing time as a function of one, two, or many variables. The neural network can be re-trained to assimilate new experimental data, extra variables, or to change the threshold error value.

The neural network algorithm consists basically in Forth and Back Propagation subroutines, used in training and testing modes. The forth propagation (FP) subroutine is as follows:

```
Loop P through the number of Patterns
    Loop H trough the number of Hidden Nodes
        Estimate Hidden Data(H, P) = In Hidden Weights(Input Nodes, H) *
                                     BiasFactor
        Loop I through the number of Inputs Nodes
            Estimate Hidden Data(H, P) = Estimate Hidden Data(H, P) +
                                         In Hidden Weights(I, H) *
                                         Input Data(I, P)
Estimate Data = Estimate Hidden Data()
Hidden Result Data = ExponentTransform(Estimate Data, Hidden Nodes)

Loop P through the number of Patterns
    Loop J trough the number of Output Nodes
        Estimate Output Data(J, P) = Hi Output Weights(Hidden Nodes, J) *
                                     BiasFactor
        Loop H trough the number of Hidden Nodes
            Estimate Output Data(J, P) = Estimate Output Data(J, P) +
```

```
                                        Hi Output Weights(H, J) *
                                        Hi Result Data(H, P)
  Estimate Data = Estimate Output Data()
  Result Data = ExponentTransform(Estimate Data, Output Nodes)
```

This Forward Loop is used for training the neural network and to test new cases. It first loops between the input and the hidden layer; afterwards, it loops between the hidden and the output layer. On the other hand, The Back Propagation (BP) subroutine loops back, as the term self explains it, loops back from the output layer, to the hidden layer, and then finally to the input layer, making possible the finding of the weighting factors in the hidden layer. This subroutine is only used in the training mode. It would be the only situation where the network is given an output to find input. The subroutine for Back Propagation is shown:

```
Loop P from 1 to number of Patterns
    Loop J from 1 to number of Output Nodes
      Hi Output D(J, P) = (Output Data(J, P) – Result Data(J, P)) * Result Data(J, P) *
                                    (1 - ResultData(J, P))
    Loop H form 1 to number of Hidden Nodes
      Sum = 0#
      Loop J 1 to number of Output Nodes
        Sum = Sum + Hi Output D(J, P) * Hi Output Weights(H, J)
      In Hidden D(H, P) = Sum * Hi Result Data(H, P) * (1 – Hi Result Data(H, P))
  Loop J from 1 to number of Output Nodes
    Sum = 0#
```

Loop P from 1 to number of Patterns

Sum = Sum + Hi Output D(J, P)

Hi Output DWeights(Hidden Nodes, J) = Learning Rate * Sum +

Momentum Factor *

Hi Output Delta(Hidden Nodes, J)

Hi Output weights(Hidden Nodes, J) = Hi Output Weights(Hidden Nodes, J) +

Hi Output DWeights(Hidden Nodes, J)

Loop H from 1 to number of Hidden Nodes

Sum = 0#

Loop P from 1 to number of Patterns

Sum = Sum + Hi Output D(J, P) * Hi Result Data(H, P)

Hi Output DWeight(H, J) = Learning Rate * Sum + Momentum Factor *

Hi Output Delta(H, J)

Hi Out weights(H, J) = Hi Output Weights(H, J) + Hi Output DWeights(H, J)

For security reasons, the program comes with an already trained network as default, and with the re-training option disabled. However, by changing the value of a toggle-switch-constant in the program's code, the user may retrain the neural network characteristics.

ASSUMPTIONS

The interface architecture of the NPLSGA allows the user to enter any value into the given fields. For instance, the user may enter negative values for tool diameter, or even text instead of values. Therefore, the program needs to filter unwanted value to avoid erroneous calculations. If the user enters a value that is assumed to be physically unreal,

it will display an error message box. Error is usually display when the "Optimize" button is pressed. See the following figure:

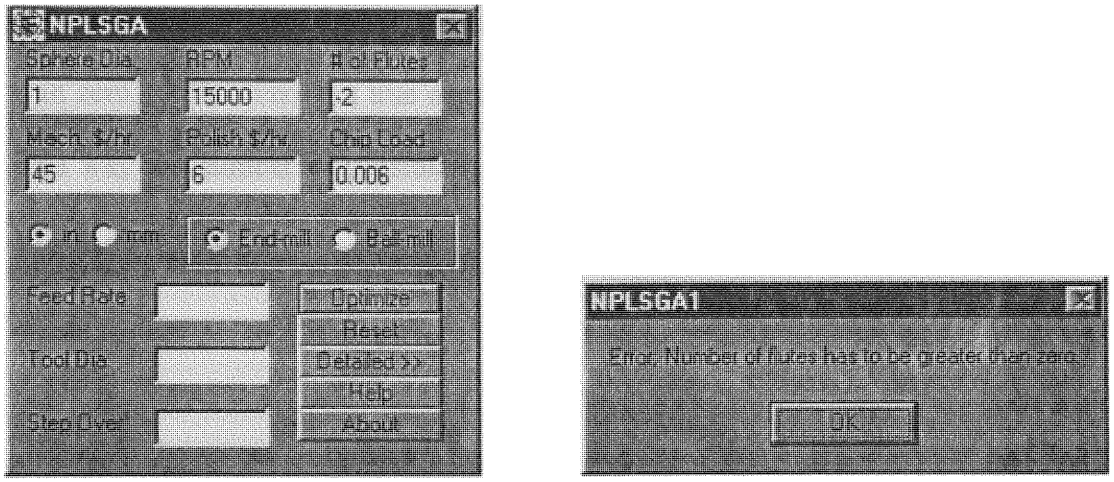

Figure 30: Error Message displayed if number of Flutes is entered as a Negative Number

The following assumptions are made in order to avoid problems in the program calculations. Without these assumptions, the program would incur into divisions by zero, or square roots of negative values. If this would happen the program would give an error message an abort all operations. To avoid this inconvenience, any invalid input variable will be filtered any real program error occurs.

Sphere Diameter > 0

Spindle Speed > 0

Tool's Number of Flutes > 0

Chip load > 0

Machining Hourly Rate > 0

Polishing Hourly Rate > 0

Tool Diameter > 0, and an available size

0.0001 < Step Over < TD

0 < Feed Rate < Maximum Allowable Feed Rate

Non of the following may be equal to zero: the sphere diameter, the step over, the tool diameter, nor the feed rate. It is assumed that these values are positive numbers greater than zero. If this were not true negative values or divisions by zero would be the result. This way the number of possible running errors, and so-called bugs, are limited in the NPLSGA software.

CHAPTER V

EXPERIMENTAL SET-UP

To calculate the polishing time of a lens, experimental data is necessary. To obtain the experimental polishing data three samples were prepared with different stopovers on a Fadal milling machine. All the samples were manually polished and the time was recorded.

Several acrylic samples were prepared in a 45-degrees-wedge shape. Their base area was 1 in. by 1 in. as shown in the figure:

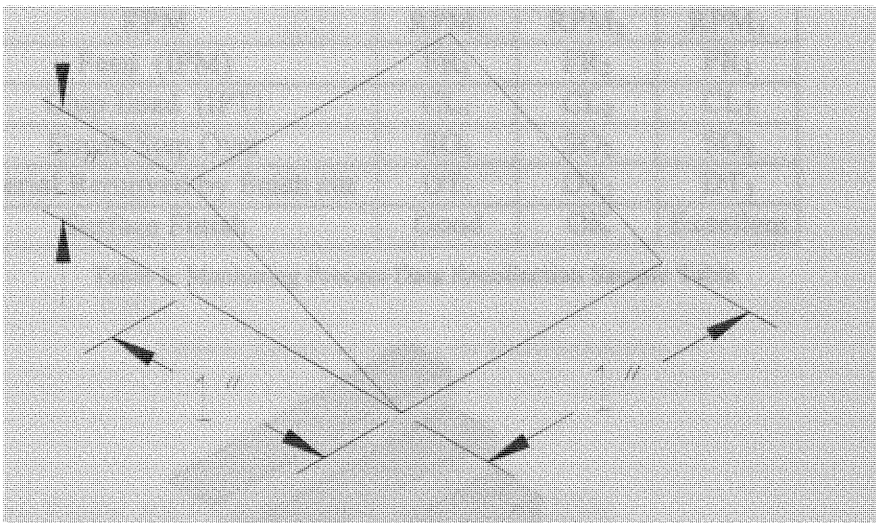

Figure 31: Acrylic 45-degrees Wedge Sample

After the wedge is prepared, it is machined across the inclined face with an end-mill. At this stage the CNC spindle speed, feed rate, tool diameter and number of tool teeth are recorded. Once the whole inclined faced is machined, the sample is examined and classified in one of the following categories depending on the surface finish:

- Rough
- OK
- Good
- Excellent

Excellent sample will require none or little polishing after the machining process is complete, to have the adequate lens surface finish. The ones labeled Good, are the ones that with a reasonable polishing time could obtain the acceptable surface finish. Table 3 shows how the machining results were recorded:

Sample #	1	2	3
Machining Time (min:sec)	Mt_1	Mt_2	Mt_3
Finish Tool Diameter (HSS)	TD_1	TD_2	TD_3
Finish Tool, Number of Flutes	Nt_1	Nt_2	Nt_3
RPM	RPM_1	RPM_2	RPM_3
Feed (IPM)	FR_1	FR_2	FR_3
Chip Load (IPT)	CL_1	CL_2	CL_3
Finish Step Over	SO_1	SO_2	SO_3
Material Removed by finish cut	DT_1	DT_2	DT_3
Surface Finish	Good	OK	Excellent

Table 3: Machining Process Data Recollection Sample Table

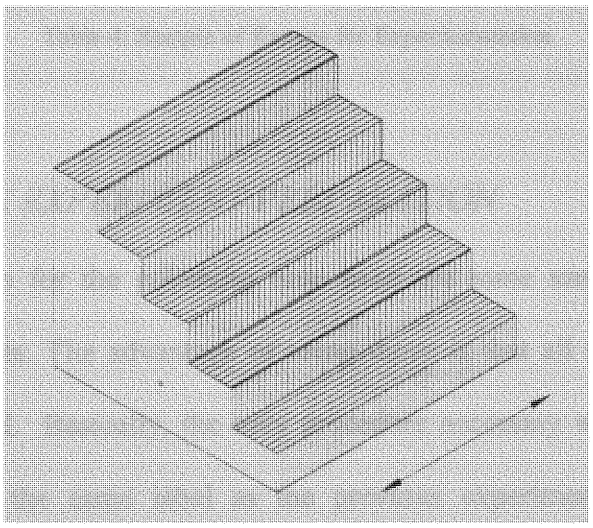

Figure 32: Wedge Sample After Machining

After all samples were run, they were grouped by surface finish. Then, the tool diameter, rpm, chip load, and feed rate are all extracted from the group of "Good" surface finish. Those were the values used to teach a neural network. Feed rate, chip load and spindle

speed were set as the input values, and the finish tool diameter as the output variable. Then the neural network found a relationship between these values.

Different tool sizes, feed rates, step over, and rpm were used. Table 4 shows the ranges in which each variable was tested. A broad number of combinations within these ranges is possible, so enough data was recollected to train the neural network.

Tool Size	RPM	Feed Rate	Step Over
3/8	5000	10	0.005
¼	.	.	.
1/5	.	.	.
1/8	.	.	.
1/16	60000	200	0.1
0.05			
0.04			
1/32			
0.02			

Table 4: Sample of Ranges for Experimentation

POLISHING TIME EXPERIMENT PREPARATION

The experiment setup for the polishing time used the same samples prepared in the machining experiments. The advantage of using those is that all machining parameters were already recorded. Basically, the only procedure necessary would be to polish the same sample, using the same label as the previous experiment, and recording the polishing time.

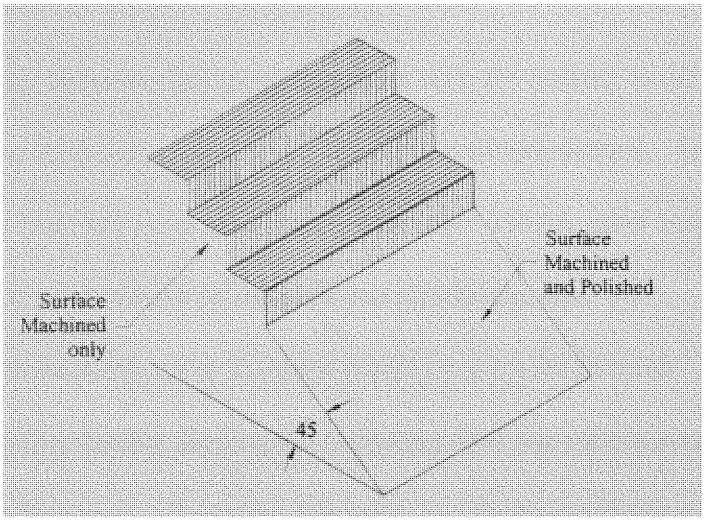

Figure 33: Wedge Sample after Machining and Polishing

To visually check the difference between the surface after machining, and after polishing, only a section of the inclined surface was polished. The surface area polished was approximately one square inch. After the process, the smoothed surface area was measured in order to get a more exact polishing time per square area.

CHAPTER VI

RESULTS AND DISCUSSION

To find a relationship between the input variables and the output, neural networks are used to learn several different cases found through experimentation. Once the neural networks are taught, they are used to estimate from new inputs, other output values that will be used later in the calculation of the total manufacturing cost.

There are two relationships that have to be estimated by a neural network. They are:

- Entering a scallop height: finding the polishing time per square area.
- Entering spindle speed, allowable chip load, and feed rate: finding the cutting tool diameter.

Hence, a sample data had to be created. This data showed several points of scallop height and its polishing time, points with spindle speed, allowable chip load, feed rate, and tool diameter. The structure of the neural network is presented in Figure 34.

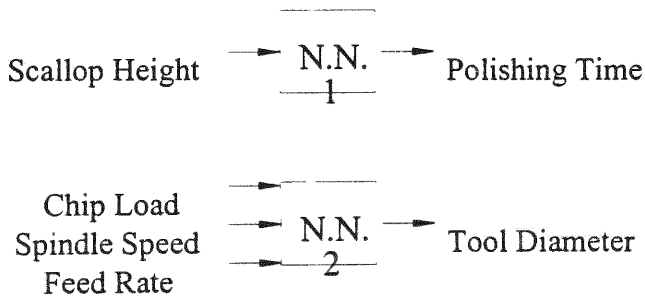

Figure 34: Neural Networks Representations to Estimate Polishing Time and Tool Diameter

The scallop height before polishing is calculated from the tool size, the step size. All other factors are fixed, like the surface slope and curvature (all flat surfaces), and cutting angle (cutting at α = 90 degrees). Then, similar to Table 3, the recollection table for polishing time looks like the following:

Sample #	1	2	...	N
Finish Tool Diameter (HSS)	TD_1	TD_2	...	TD_N
Finish Step Over	SO_1	SO_2	...	SO_N
Material Removed by finish cut	DT_1	DT_2	...	DT_N
Surface Finish	Good	OK	...	Excellent
Scallop Height	Sh_1	Sh_2	...	Sh_N
Polishing Time	Pt_1	Pt_2	...	Pt_N

Table 5: Hand Polishing Data Recollection

If the depth of cut is less than the calculated scallop height, the material removed will be less than the calculated scallop height. That happens when the tool is cutting from the sides of the wedge. In that case, the scallop height is set to a maximum physical value that is set by the depth of the cut and material removed.

To help in the calculation of the scallop height for, this specific case, another support software was developed to calculate scallop height in wedge shapes.

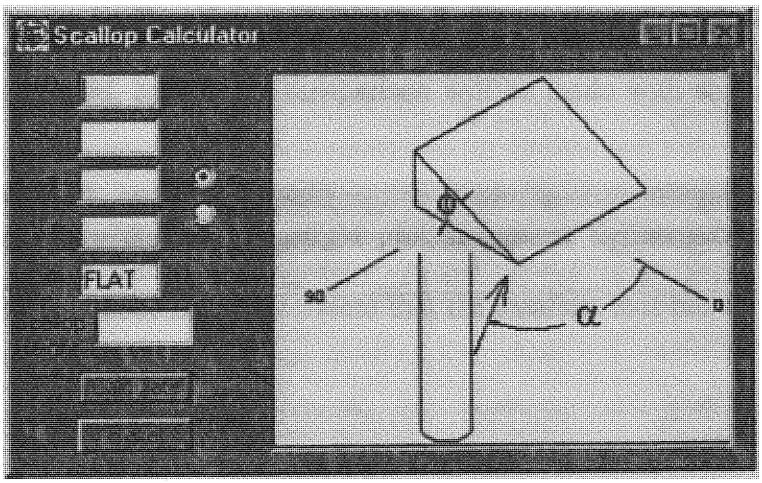

Figure 35: Scallop Calculator Software

The variables for this program are explained in the graphical representation of the wedge sample. The variables to calculate the scallop height are the tool diameter (TD), the step over (SO), the slope (ϕ), the cutting angle (α), and the radius of surface curvature (ρ). If

the surface were flat, instead of radius of curvature, the operator would write the word "FLAT". Then, all the curvature factors are ignored in the calculations and the right scallop height is calculated. This simplified program helps to get the calculation of the scallop height for the specific cases, without having to use the more complex program that optimizes the variables.

SCALLOP HEIGHT OF MACHINED SURFACES

In this section the scallop height equations are analyzed to see their reactions to changes in the different variables. The variable that makes changes in the scallop height more evident is the step size. However, we have to study how the scallop height is affected by variation in other variables, like tool diameter, surface slope and curvature, and direction of cut. Scallop height is calculated using the equations explained in previous chapters, and supported by the Scallop Height Calculator Software. First the values obtained from End-mills are analyzed, and then a comparison will be made with scallop height using ball-mills under the same conditions.

The first chart shows the scallop height plotted against the step over. Each curve represents a different Tool Diameter. Note that the smallest tool has the largest scallop height for the same step size.

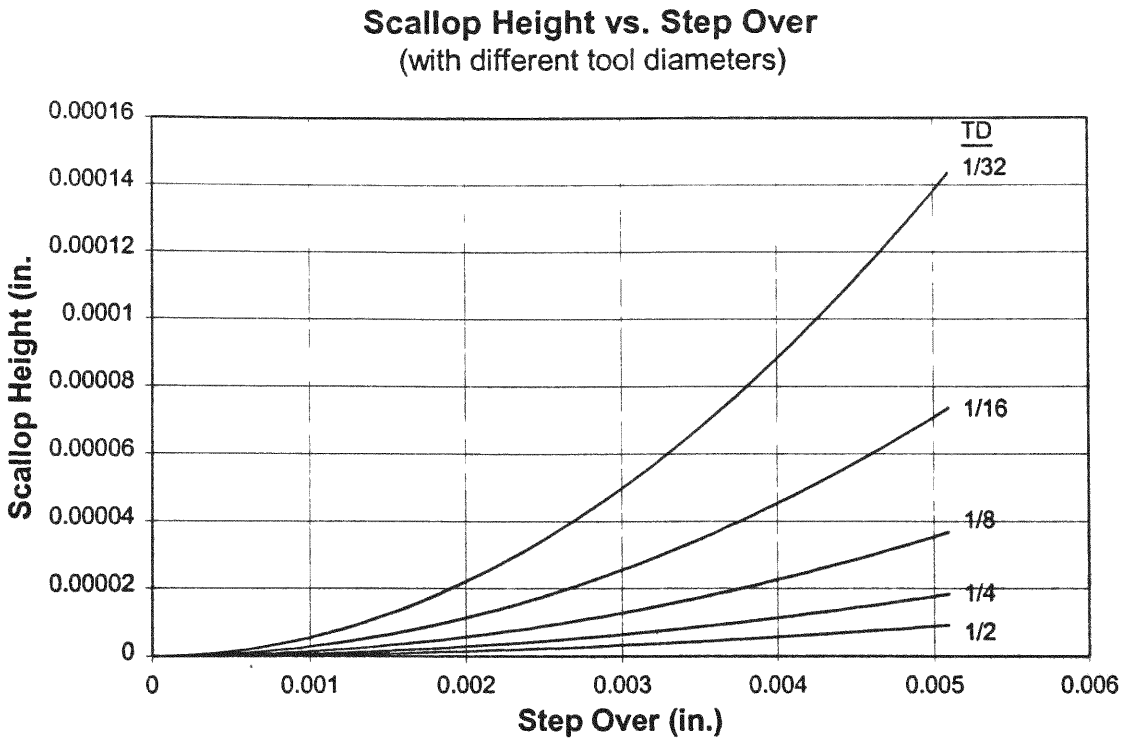

Figure 36: Scallop Height plotted against Step over with different Tool Diameters

The second chart shows the same combination but instead of having different curves for different tool sizes, now the curves are classified by surface angle ϕ. In this case, if all other variables are kept constant except ϕ and SO, as the slope increases so does the scallop height. When $\phi = 0$ (horizontal surface), then scallop height will be zero if using an end-mill. This tool has a flat bottom surface that makes a perfect match with horizontal surfaces.

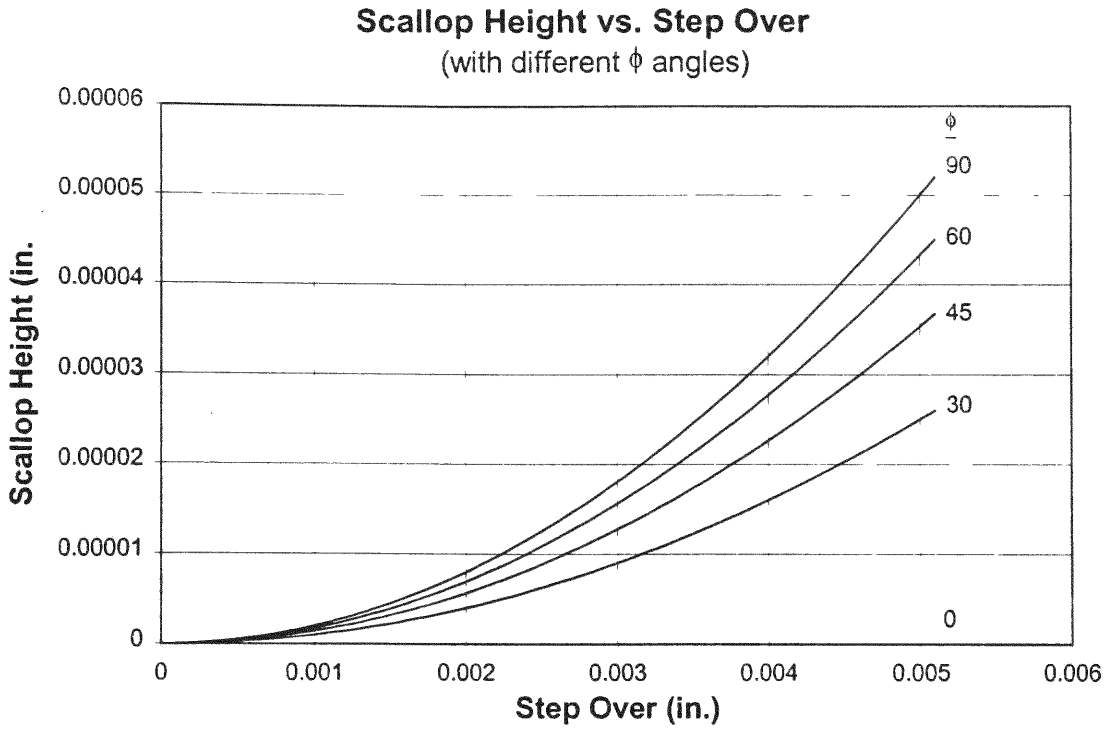

Figure 37: Scallop Height plotted against Step over with different Slope Angles

Following there is a chart describing the changes with the α angle, or direction of cut. As the angles increases, so does the scallop height. However, because of geometrical reasons, when the angle α gets closer to 90 degrees, the scallop shape changes from the interception of two ellipses to the interception of two rectangles (refer to Scallop Height Calculation Section). When that happens, especially with inclined surfaces, the scallop height depends linearly of the step size, and is much larger than if α would be close to zero.

Surface curvature also affects the scallop height. The maximum scallop is reached when it is a flat surface. If the curvature radius is increase, the scallop height will decrease as shown in the following charts.

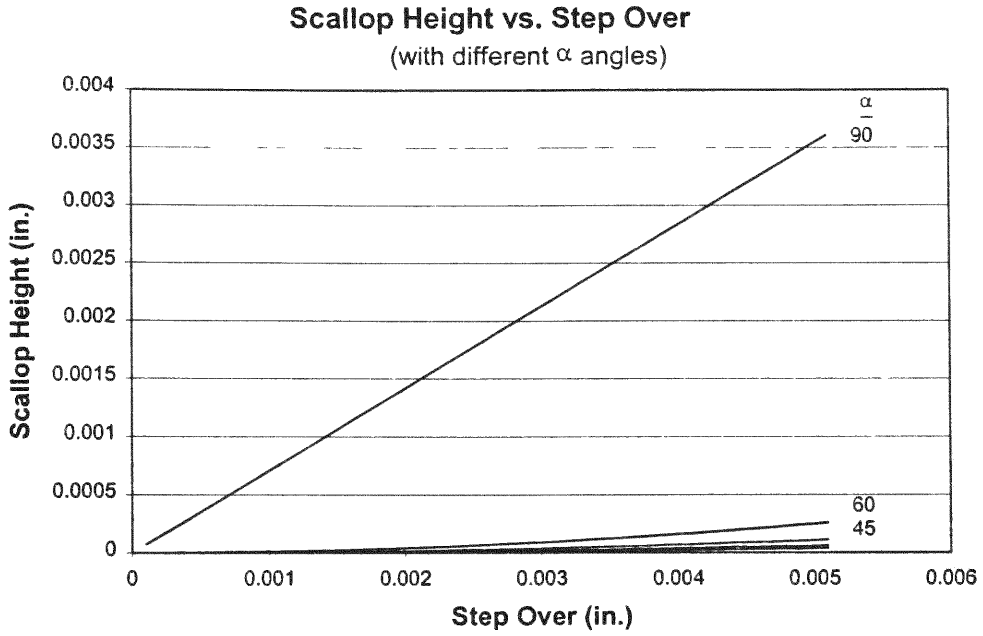

Figure 38: Scallop Height plotted against Step over with different Cutting Angles

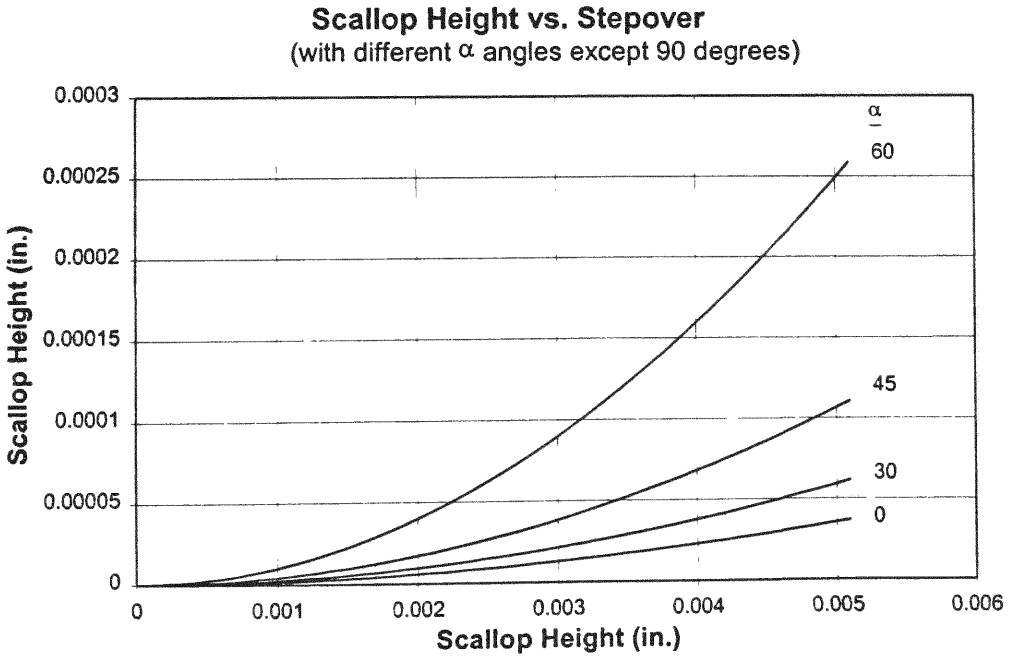

Figure 39: Scallop Height plotted against Step over with different Cutting Angles except $\alpha = 90°$

Considering the machining of a semi-sphere, the radius of curvature ρ is equal to the radius of the sphere or lens. Using that as an input variable, the scallop height can be found at any point through out the lens surface. In the next section, it is explained how the calculation of scallop height at different points through out the lens surface is used to estimate the polishing time per unit area.

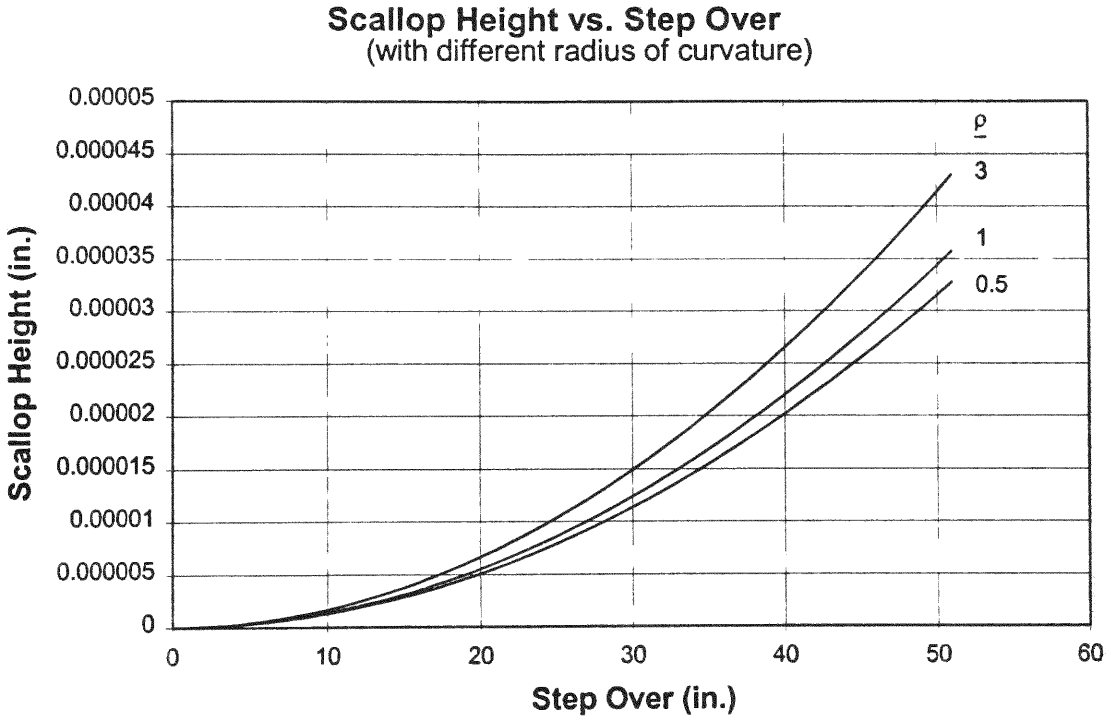

Figure 40: Scallop Height plotted against Step over with different Radius of Curvature

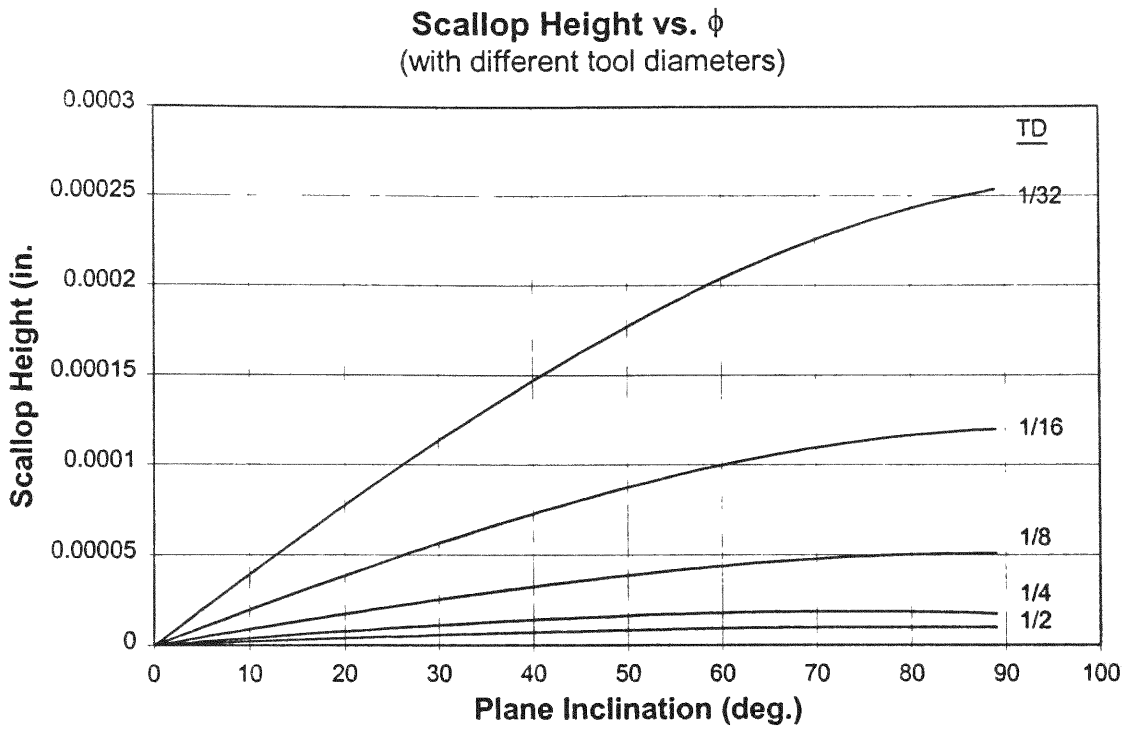

Figure 41: Scallop Height plotted against the Plane Inclination with different Tool Diameters

The estimation of hand-polishing time is dependant only on the scallop height, it is important that the mathematical model for the scallop be as accurate as possible.

The following pictures are screen shots from the NPLSGA program, and the Scallop Height Calculator (SHC) that show calculations done for a specific lens surface.

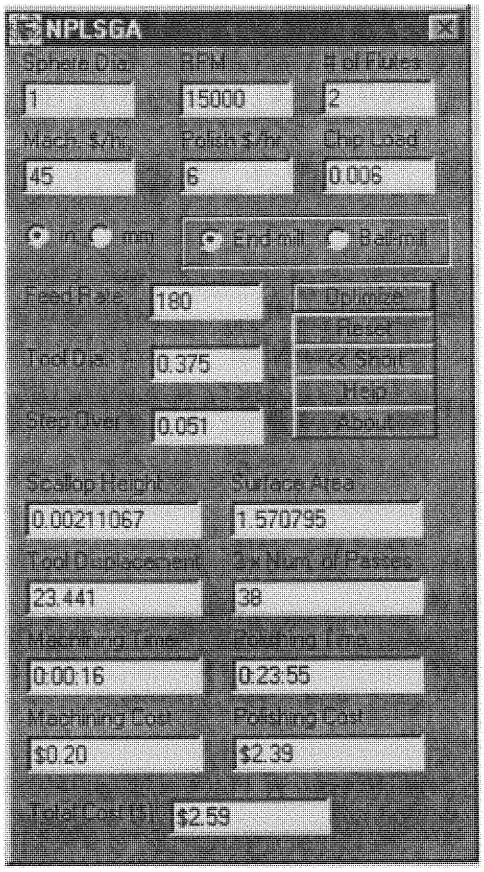

Figure 42: NPLSGA Program Displaying Average Scallop Height in a Lens

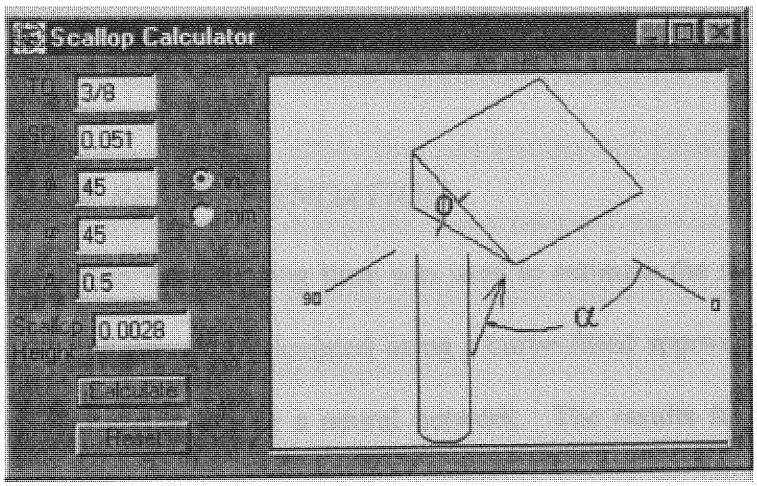

Figure 43: SHC Displaying the Scallop Height at One Point on the Same Lens

pass #	1	2	3	4	5	6	7	8	9	10
TD	0.375	0.375	0.375	0.375	0.375	0.375	0.375	0.375	0.375	0.375
SO	0.051	0.051	0.051	0.051	0.051	0.051	0.051	0.051	0.051	0.051
ϕ	67	55	46	38	31	24	18	12	6	0
ρ	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
S. H.	0.0093	0.004	0.003	0.002	0.001	9E-04	4E-04	1E-04	0	0
Average	0.0021									

Table 6: Scallop Height Calculated per Pass and Total Average

When the user enters a sphere diameter into the NPLSGA program, it automatically calculates scallop height across the whole surface area, and takes the average. The result is displayed in the Scallop Height text box. This value is ready to be entered either in a curved-fitted function or a neural network to find the estimated polish time.

In the other hand, the Scallop Height Calculator program only displays one of the calculations at a time. This support software is meant to be a more general application that will calculate the scallop height with any parameters, and not only for spherical surfaces. If the same NPLSGA's average height value wants to be found, it would be necessary to calculate the scallop height at each individual point, and then take the average solution.

TOOL DIAMETER RECOMMENDATION

The same approach than the polishing time was used to estimate the recommended tool diameter. First the solution was calculated using a curve-fitted approximation, and then the same value was obtained using a neural network. The results for the relationship between the chip load, the spindle speed, the feed rate and the tool diameter were compared using both methods. Figure 44 shows how each method approaches to the real value:

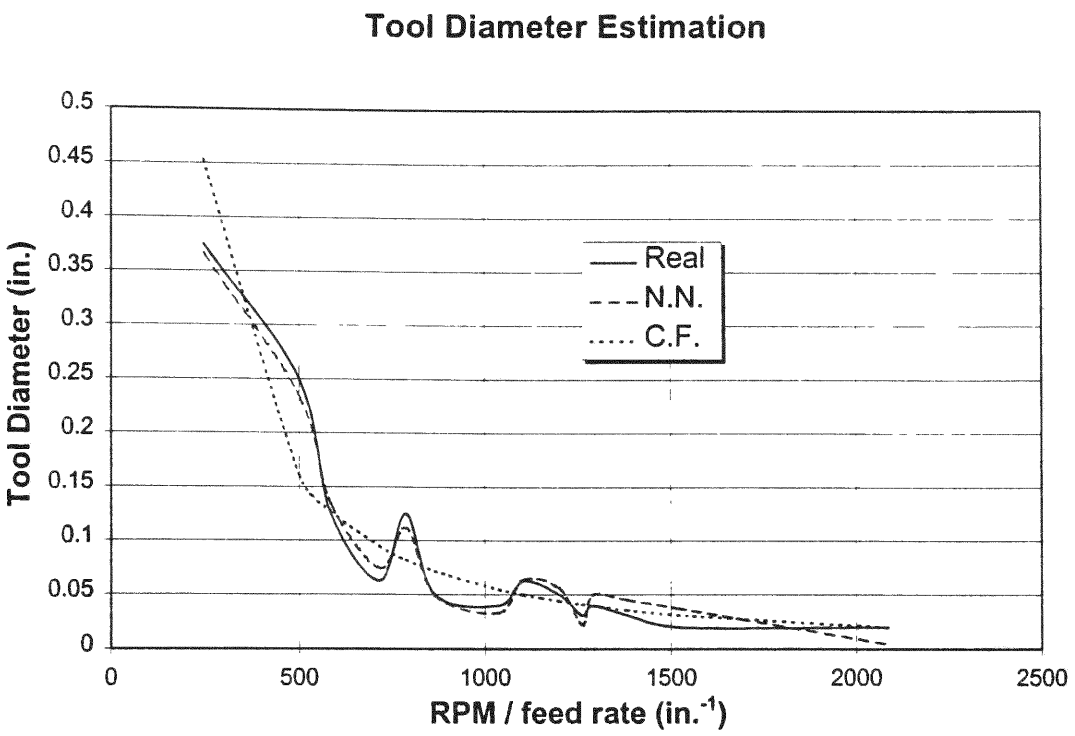

Figure 44: Tool Diameter Estimation vs. RPM / Feed Rate

The neural network (N.N.) approximation is a better match to the real experimental data than the curve-fitted (C.F.) function. However, considering that the real data may have experimental errors, the curve-fitted equation may be a better solution. This function has a smoother relationship between the RPM / feed rate and the tool diameter, and it would be easier to enter in computer software than the neural network approximation. Considering other input variables to observe the tool diameters selection, the results would change as seen in the following Figure 45.

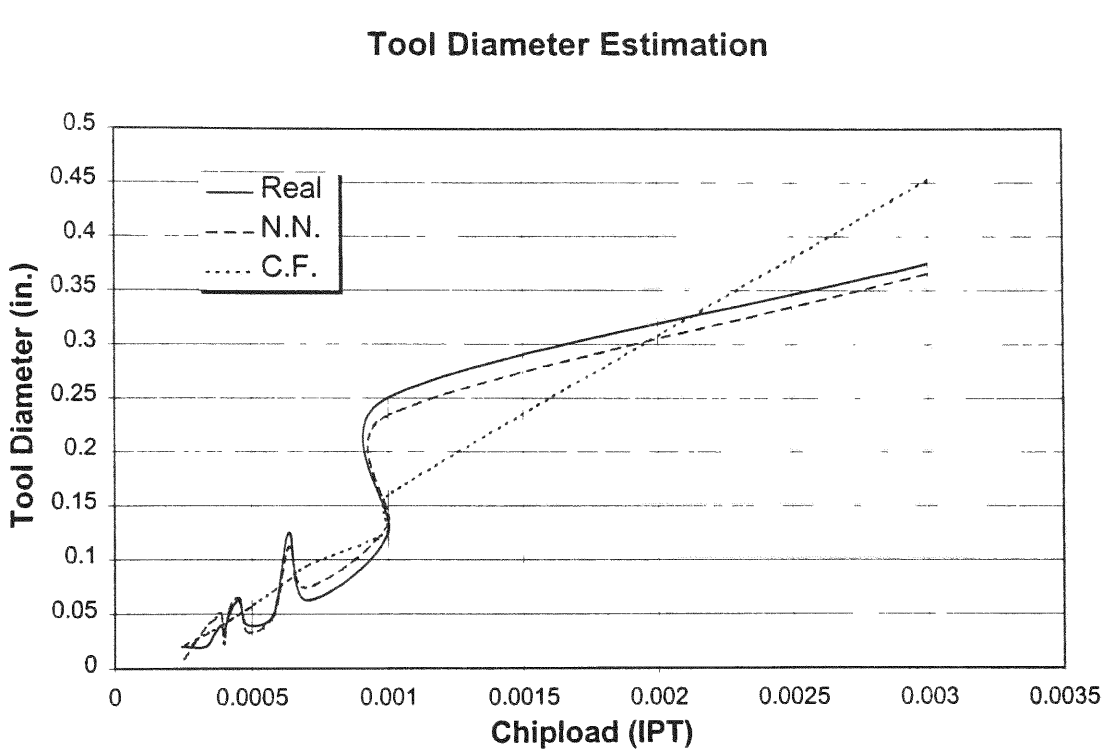

Figure 45: Tool Diameter Estimation vs. Chip Load

Even tough the error for a curve-fitted equation is larger, the function is easier to implement in computer code, producing results faster than a neural network. However, the neural network algorithm was selected to be used in the NPLSGA because it provides a modular flexibility to accept more input factors, if necessary. Besides that, using today's high-speed micro-processing power, the time difference between a neural network and a curve-fitted equation is less than 5 seconds.

Once a tool diameter is estimated, the program will select the tool size available closest to the calculated value. For example, if the approximated value is a tool diameter of 0.1178", the program will recommend using a 1/8" cutting tool.

EXPERIMENTAL POLISHING TIME

It was shown in the last section how different factors affected the scallop height and shape. Hand-polishing time is dependant of the scallop height, therefore all these factors affect indirectly the final total manufacturing cost of the lens.

When calculating the scallop height on a lens, a number of scallop height estimations are made at different points throughout the lens surface. If there are two finishing passes, one perpendicular to each other, it is assumed that large scallop heights formed, due to a cutting direction angle α equal to 90 degrees (see charts in previous section), are machined down.

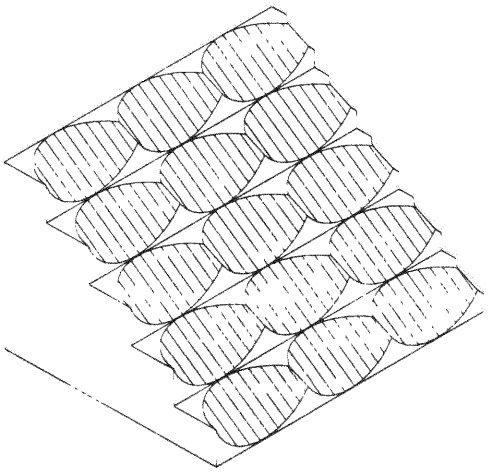

Figure 46: Surface that has had Two Perpendicular Finish Passes

All these estimations for scallop heights are averaged, and a neural network classifies the resultant. To evaluate if the usage of a neural network is necessary, the scallop height average is also entered in a curve-fitted function that finds the hand-polishing time.

Following the experimental setup explained in the previous chapter, different samples were machined and polished to find numerical approximation of polishing time and machining parameters. Figure 47 shows a picture of one of the samples machined, and

polished, to obtain training data. More pictures of these samples are shown in the Appendices under Figures 50, 51 and 52.

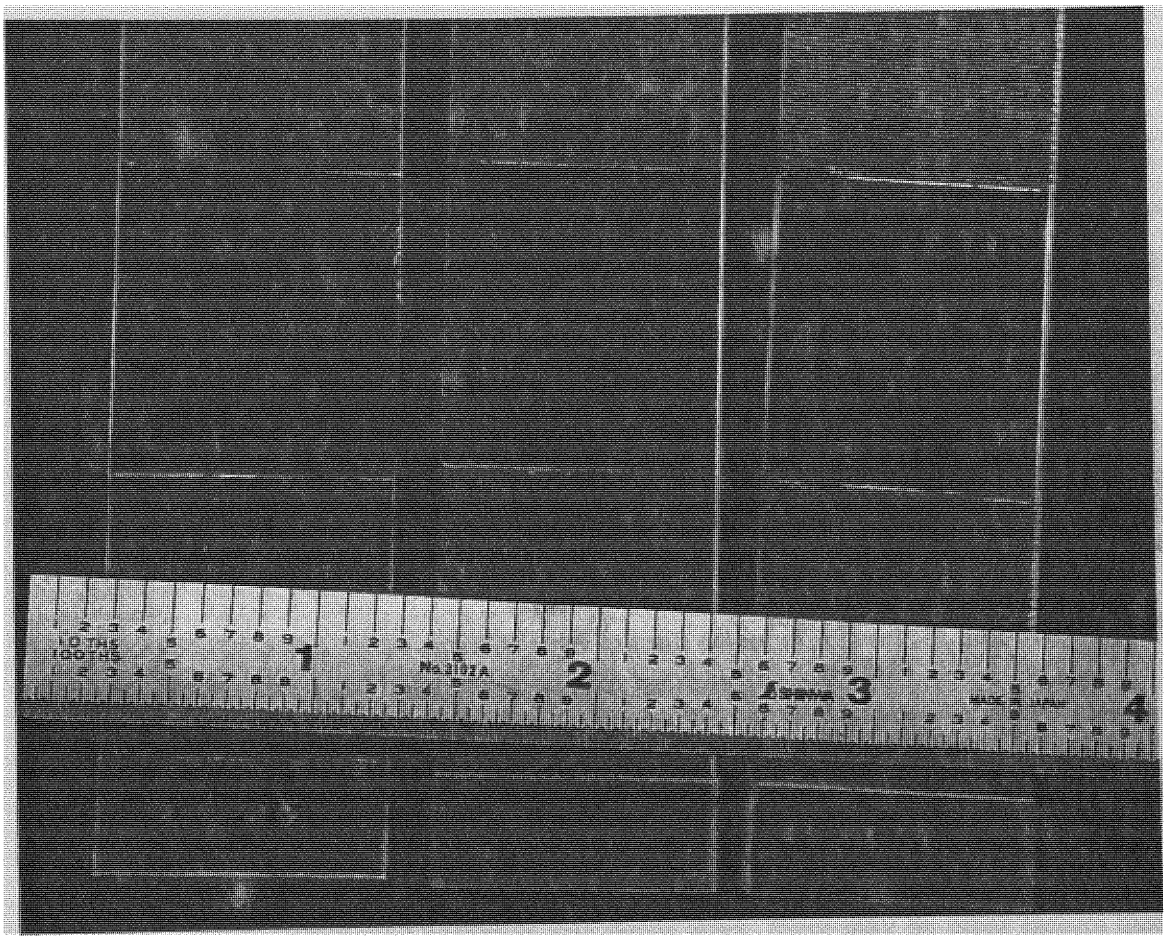

Figure 47: Experimental Samples

Machining parameters, and polishing time were recorded as specified, and the results are displayed in the following charts.

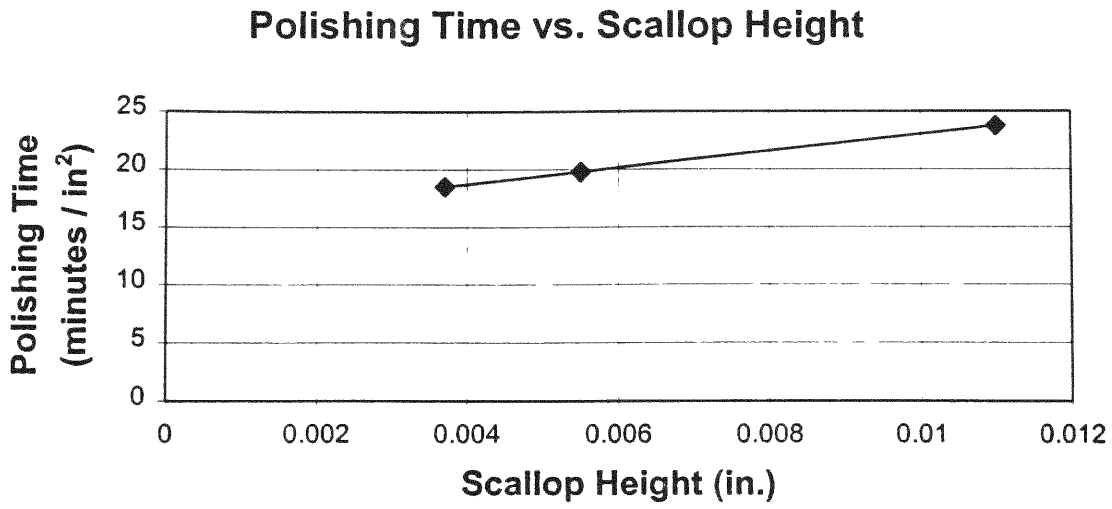

Figure 48: Results from Experimental Setup

This data could be either curve-fitted or used to train a neural network. Curve fitting will produce a faster response on computer software, would use less code and computer resources. If performance is a determining factor on the program design, an approximated function would be the ideal solution to find the estimated polishing time. However, as it was mentioned before in this thesis work, there may be other factors that would affect the polishing time, even though the scallop height would be the main factor. Other factors could be material defects and irregularities, burr left after machining, etc. If these extra factors want to be accounted for the final estimation, then a neural network would be better used because of its adaptability to any type of input. The network could be trained with new data, and its internal weighting factors would be modified to fit all given conditions and classify new cases.

In both cases, curve fitting and neural network classification, a good estimation of polishing time is achieved. The following charts show a comparison of hand-polishing time values estimated from each method.

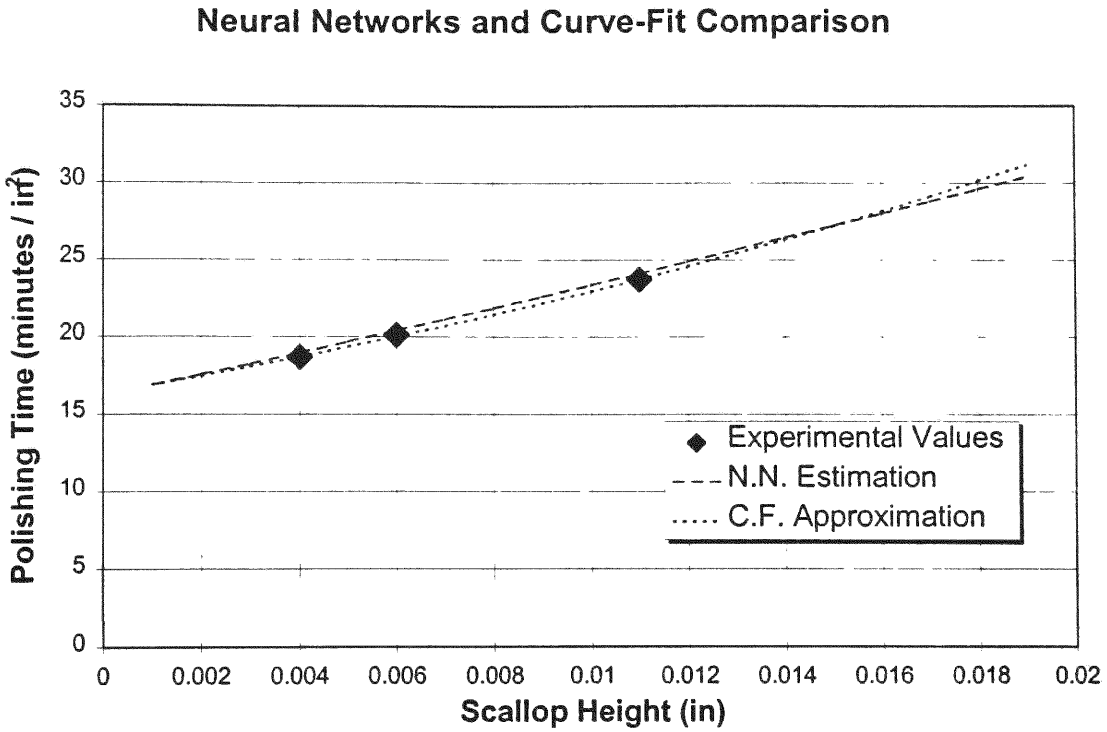

Figure 49: Approximation of Polishing Time Given by Neural Networks and Curve-Fitted equations

ACCURACY ESTIMATION OF THE MODEL

As observed in the previous chart (Figure 49) the approximations given by the curve-fitted equation, and the neural network are very close. The average relative error using the multi-layer network is 1.59%, and its maximum relative error is 1.68%. The curve-fitted approximation errors are smaller, in the magnitude of 0.1% average relative error. The curve-fit method is faster and more precise than the neural network to approximate the polishing time per square area.. However the NPLSGA uses neural network

approximations. The reason for method is that the relative error is still small using neural networks, and it provides extra advantages over the curve-fitted function method. A neural network provides flexibility in design, making possible to simply add input or outputs to the software in future versions.

PERFORMANCE OF THE PROGRAM

The NPLSGA overall performance was above expectations. Results of the optimization process were displayed with a maximum of 5 seconds. The displaying time would be the same, either if the software showed "Detailed" or "Summarized" information. Results were very precise, showing less than 2% relative total error.

The program was designed with few simple options that provided a user-friendly interface. Entering input values was as simple as typing inside the correspondent text boxes, and output was displayed in the same manner, correspondent to text boxes. The software size is less than half a Mega –byte, making it easy to store in a standard 3-1/2" floppy diskette.

CHAPTER VII

CONCLUSION

In this study a program is developed to help engineers to select the best cutting parameters for acrylic lens manufacturing. A 470Kb is prepared to perform this task in less than 5 seconds. Influence of different parameters to scallop height is also studied here.

Smooth acrylic lens prototypes were produced by machining the rough material in a CNC machine, and then polishing it by hand until the desired surface finish is obtained. Machining parameter selection was important to reduce the total part's manufacturing cost. This thesis work explained the procedures to estimate the optimal feed rate, tool diameter and cutting step size. Once these procedures were established the added cost between machining the part, and then hand polishing was minimized. The development of a Near-Perfect Lens Surface Generation Advisor (NPLSGA) software allowed different procedures. These estimations were based in given parameters set by a user. They included allowable chip load, machine's spindle speed, lens diameter, tool type, and current cost per hour rates for polishing and machining.

Tool diameter and feed rates were estimated based on formulas and experimental data that followed a defined pattern. Their optimal values were directly related to the given variables, like the chip load, tool's type, and its revolutions per minute. After the two recommended values for tool diameter and feed rate were calculated, the step over was found through the optimization process. The total manufacturing cost, dependable of several factors, could be represented as a value calculated through a series of functions, all dependable of one independent variable: the step size. As the distance between tool passes increased, the height of material not removed increased; therefore, increasing the

hand polishing time to get the adequate surface finish. When the step over was reduced, less polishing time was required, but extra time in the CNC machine increased the machining costs of operation. Thus, an optimal step over could be found using optimization techniques to minimize the total cost of machining plus hand polishing the curved acrylic lens.

Machining operation costs were calculated knowing the feed rate, the distance the tool moved across the work-piece surface, and the machine operation's cost per hour. Tool displacement was simply calculated from the lens geometry, and the selected step size. On the other end, the hand polishing expenses were dependable of the scallop height left on the material after machining. The hand polishing cost was harder to estimate. There was no analytical equation that could be derived from these two parameters.

Experimental data was used to train a neural network in order to create a pattern to estimate hand-polishing time per square area, based in the scallop height. Although cost and marketable computational capability concerns of Neural Networks discourages its use in the program, it was found very helpful because of its flexibility. The neural network allowed adding modular components to the program, including extra variables, operations and outputs. The results obtained from the NPLSGA showed a small relative error and were provided without any considerable waiting period.

The material left between passes was found using geometrical equations that depend on the tool type and size, the lens diameter, and the step size. The smaller the step size, the smaller the scallop height would be. Therefore, the time required to smooth the surface to an acceptable surface finish was also less. Once the polishing time per area was

estimated from the neural network, the total area to be polished, and the labor rate per hour were multiplied.

The processes to calculate the recommended tool diameter and feed rate, the total manufacturing operation costs, and the optimum step size were integrated in the NPLSGA program. This software allowed the use of these complicated series of calculations through a simple user-interface. A Design Engineer can refer to this software at any moment to obtain optimal parameters that will provide minimal manufacturing costs when the design goes to production. The program also displays the calculated machining and polishing cost, average scallop height, tool displacement, surface area, and number of passes.

The estimations given by the developed program were very close to real values. The program output, compared with collected experimental data, and the NPLSGA was consistent in providing results with a relative error less than of 2%.

The application of this thesis work will help Design Engineers in the proper selection of machining parameters to reduce the manufacturing cost of curved smooth acrylic lenses. Furthermore, the optimization processes explained herein can be applied in other manufacturing procedures, and can help reduce part costs in today's market. Function minimization and estimations using neural networks may be used virtually in any manufacturing process to estimate other processes' optimal operation conditions.

LIST OF REFERENCES

1. G.W. Vickers, K.W. Quan, "Ball-Mills Versus End-Mills for Curved Surface Machining" Journal of Engineering for Industry, Transactions of the ASME, February 1989, Vol. 111, pages: 22-26

2. Peter Zelinski, Special Projects Editor: "Find Your Feed Rate On The Fly", http://www.mmsonline.com/articles/079802.html.

3. Riadh Azouzi and Michael Guillot, "Study of an inverse process neurocontroller for machining optimization" International Journal of Smart Engineering System Design v 2 n 1 1999, pages: 57-68.

4. E. Brinksmeier, H.K. Toenshoff, C. Czenkusch and C.Heinzel, "Modelling and optimization of grinding processes" Journal of Intelligent Manufacturing v 9 n 4 Aug 1998, pages: 303-314.

5. Riadh Azouzi and Michael Guillot, "On-line optimization of the turning process using an inverse process neurocontroller" Journal of Manufacturing Science and Engineering, Transactions of the ASME v 120 n 1 Feb 1998, pages 101-108.

6. D.C.H. Yang and Z. Han, "Interference detection and optimal tool selection in 3-axis NC machining of free-form surfaces" CAD Computer Aided Design v 31 n 5 1999, pages 303-315.

7. R. Sarma and D. Dutta, "Geometry and generation of NC tool paths" Journal of Mechanical Design, Transactions Of the ASME v 119 n 2 June 1997, pages 253-258.

8. S. Bedi, F. Ismail, M.J. Mahjoob and Y. Chen, "Toroidal versus ball nose and flat bottom end mills" International Journal of Advanced Manufacturing Technology v 13 n 5 1997, pages 326-332 .

9. K. Suresh and D.C.H. Yang, "Constant scallop-height machining of free-form surfaces" Journal of Engineering for Industry, Transactions of the ASME v 116 n 2 May 1994, pages 253-259.

10. Jean-Pierre Kruth and Paul Klewais, "Optimization and dynamic adaptation of the cutter inclination during five-axis milling of sculptured surfaces" CIRP Annals v 43 n 1 1994, pages 443-448.

11. D. Kiritsis, K. Neuendorf and P Xirouchakis, "Petri net techniques for process planning cost estimation" Avances in Engineering Software v 30 n 6 Jun 1999, pages 375-387 .

12. P.C. Bressloff and D.J. Weir, "Neural networks" GEC J Res v 8 n 3 1991, pages 151-169.

13. Trevor Bailey, Derek M. Jenkins, Allan D. Spence and M.A. Elbestawi, "Integrated modeling for metal removal operations" Proceedings of the ASME Dynamic Systems and Control Division American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC v 58 1996. pages 191-198.

14. David A. Rehbein, and Nick Miller, "Visual basic as an application development tool for the process industry" Advances in Instrumentation and Control : International Conference and Exhibition v 50 n pt 3 1995. Instrument Society of America, Research Triangle Park, NC, USA. 1995 pages 1225-1234.

15. Y. Zhou and B. Malakooti, "An adaptive feedforward artificial neural network with the applications to multiple criteria decision making" Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 1990, pages 164-169.

16. J.C. Lin and C.C. Tai, "Accuracy optimization for mould surface profile milling" International Journal of Advanced Manufacturing Technology v 15 n 1 1999, pages 15-25.

17. X.P. Li, Kiynkaran and A.Y.C. Nee, "Hybrid machining simulator based on predictive machining theory and neural network modelling" Journal of Materials Processing Technology v 89-90 May 19 1999, pages 224-230.

18. Hu-Hsuan, Tsai, Joseph C. Chen and Shi-Jer Lou, "In-process surface recognition system based on neural networks in end milling cutting operations" International Journal of Machine Tools & Manufacture v 39 n 4 Apr 1999, pages 583-605.

19. S. Lou and J.C. Chen, "In-process surface roughness recognition (ISRR) system in end-milling operations" International Journal of Advanced Manufacturing Technology v 15 n 3 1999, pages 200-209.

20. John Clark Craig and Jeff Webb, (1998) Visual Basic 6.0 Developer's Workshop. Washington: Microsoft Press.

21. M.L. James, G.M. Smith and J.C. Wolford, (1993) Applied Numerical Methods for Digital Computation. New York: HarperCollins College Publishers.

22. Howard Anton, (1992) Calculus. Canada: John Wiley & Sons.

23. Symposium on Ceramic Machining and Surface Finishing, National Bureau of Standards, (1978) "The science of ceramic machining and surface finishing II" proceedings of a symposium held at the National Bureau of Standards.

24. ASM Handbook Committee, (1989) Metals handbook. Metals Park, Ohio: American Society for Metals.

25. E.D. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning" Addison Wesley Publishing Company, Inc, 1989.

26. K. Krishnakumar, S. N. Melkote, "Machining Fixture Layout Optimization Using the Genetic Algorithm" International Journal of Machine Tools & Manufacture, July 1999, pages 579-598.

27. A. Warkentin, F. Ismail, S. Bedi, "Comparison Between Multi-Point and Other 5-Axis Tool Positioning Strategies" International Journal of Machine Tools & Manufacture, June 1999, pages 185-208.

APPENDICES

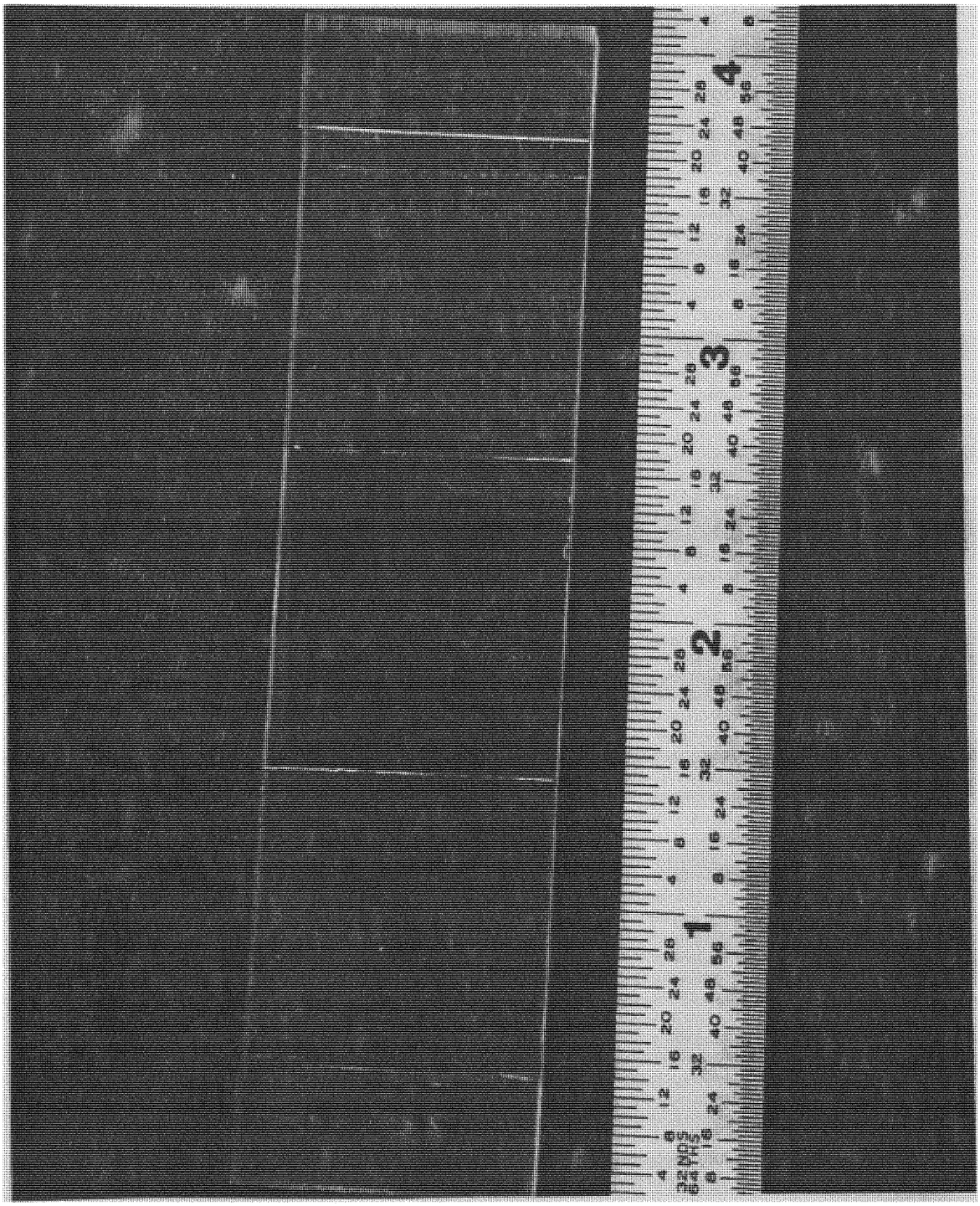

Figure 50: Acrylic Sample 1, Step Size = 0.052

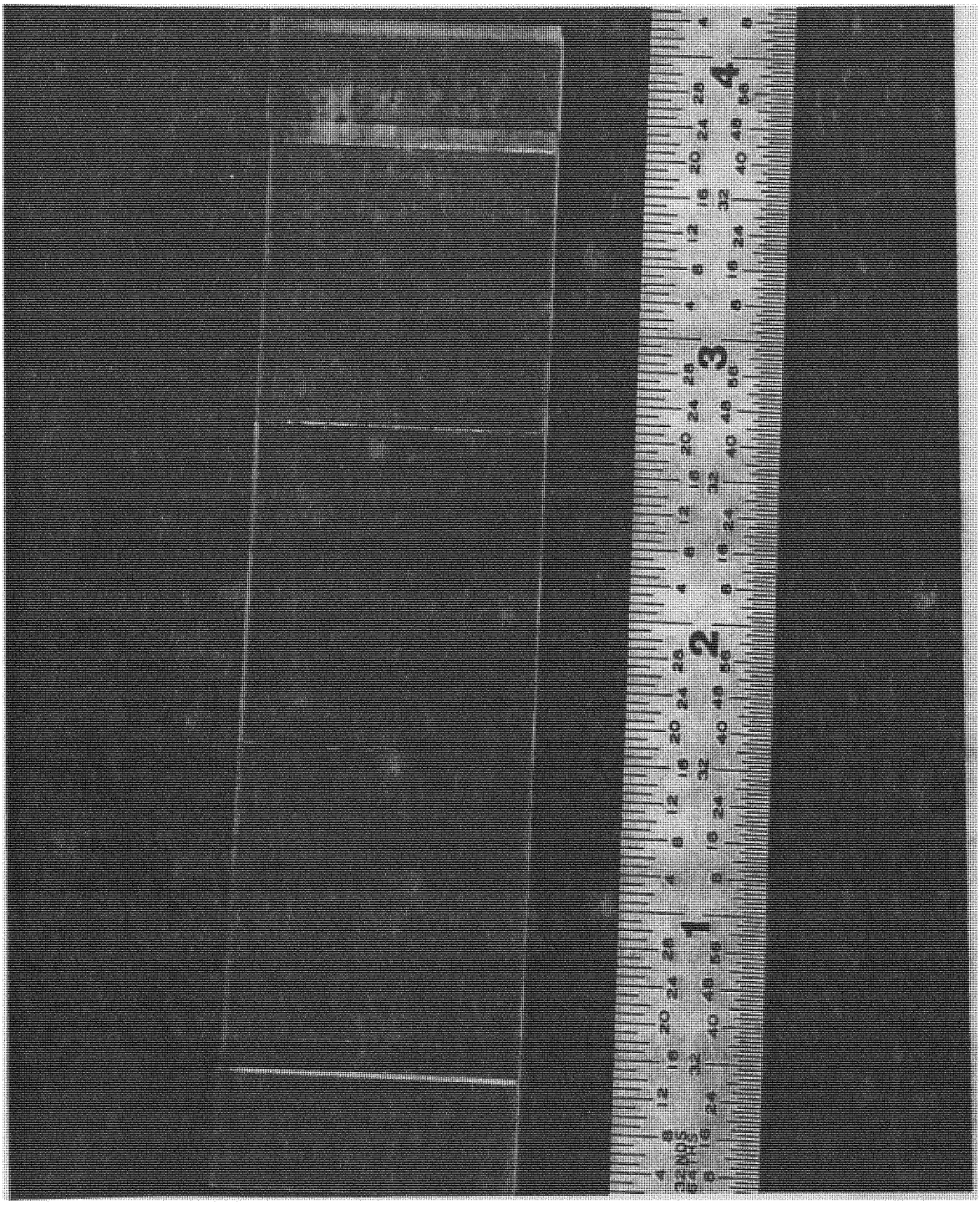

Figure 51: Acrylic Sample 2, Step Size = 0.078

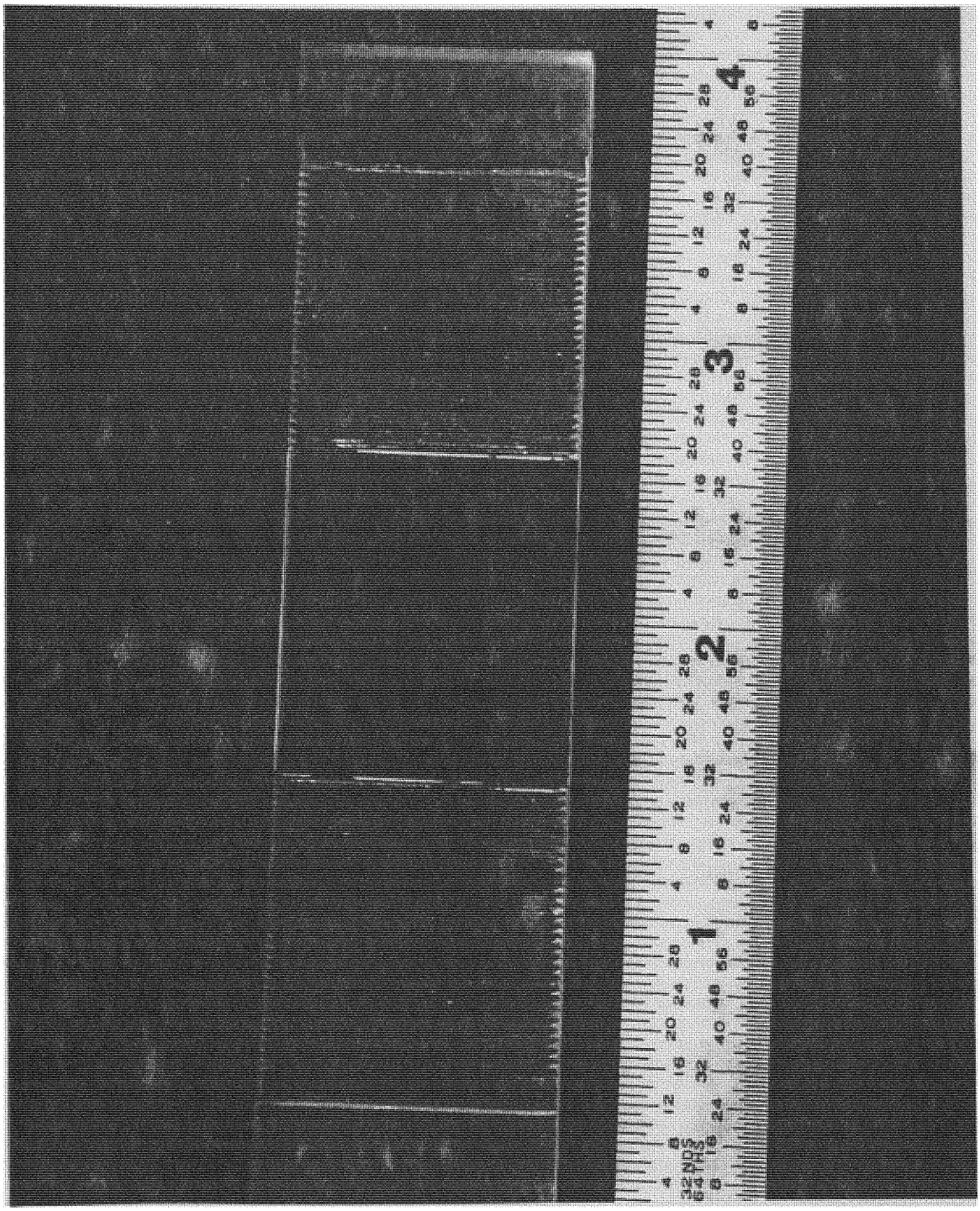

Figure 52: Acrylic Sample 3, Step Size = 0.156