Int J Adv Manuf Technol (2002) 20:727–734
Ownership and Copyright

The International Journal of
Advanced Manufacturing Technology

Optimisation of Multiple Tool CNC Rough Machining of a Hemisphere as a Genetic Algorithm Paradigm Application

A. Krimpenis and G.-C. Vosniakos

School of Mechanical Engineering, Manufacturing Technology Division, National Technical University of Athens, Iroon Polytexneiou, Athens, Greece

This work presents the development of a method to achieve optimal roughing of a hemisphere in terms of least machining time and maximum material removal from the original material block. This is considered as a problem formulation case study in the field of genetic algorithm applications – a preamble to more complex and generic geometry. A genetic algorithm is used off-the-shelf as an optimisation tool. Three different process parameters are used to model the problem : the number of scallops, the height of each scallop, and the tools used. Multiple tools are considered, in particular their diameters, maximum depths of cut and maximum feed allowed, as defined in a tool database, whereas the strategy of the tool path on each slice (but not the path details) is taken as known. Fitness functions that can be used independently or combined are the cutting time and the remaining material. The proposed method results in a concentrated result table containing the sequence of the tools employed and the corresponding scallop heights.

Keywords: CNC tool paths; Genetic algorithm; Rough machining; Tool selection

1. Introduction

Computer numerically controlled (CNC) machining of sculpted surfaces is supported by a multitude of computer-aided manufacturing (CAM) systems in three to five axes. However, optimality of the tool paths produced by those systems is still an issue. Most systems require the user to mentally split the surface to be machined into regions and apply some strategy (e.g. constant z-height, zigzagging, etc.) to each region. In roughing, in particular, where tools can be changed at will in order to accelerate the cutting process, much improvement could be achieved by the choice of the correct combination of cutting strategy and tools. This is a planning task and, apart from experience (which in the strict sense of optimality is of questionable value), few formal methods could prove helpful in tackling it successfully. Genetic algorithms, as a recent optimisation tool [1] can be one these methods, but before exploring this further, past approaches will be reviewed briefly.

In 1993, Dong et al. proposed an optimal rough machining of sculptured parts on a CNC milling machine in terms of least machining time [2]. The contour map cutting method is used to generate CNC tool paths based on the CAD model of sculptured parts. The part and stock geometry related parameters, including the number of cutting layers and the distributions of cutting depth, and the process parameters of feedrate and depth of cut, are optimised. The problem is presented as a nested mixed discrete constrained nonlinear programming formulation, and the optimisation tool used was a mixture of Powell's conjugate direction method, the mixed penalty approach, and the branch and bound strategy for integer programming, with no further details given.

A fuzzy basis material removal optimisation approach, albeit with a control flavour, to compensate for the variation of cutting speed owing to the change of gradient on a sculptured surface in the machining process is reported in [3]. A constant cutting force is maintained by adjusting the cutting feedrate for each cutting point, taking into account other machining parameters, such as tool life and surface gradient.

In [4], a maximum effective cutting radius approach is presented to solve the cutter selection problem for multi-axis machining. This is based on geometric evaluation.

Lim and Menq [5] propose two advanced strategies for machining planning, namely a cutting path/adaptive feedrate strategy and a control surface strategy in order to achieve higher productivity and product quality simultaneously for sculptured surface productions. In the former, machining time is reduced by cutting along low force/low error machining directions and by maximising feedrates. In the control surface strategy, machining errors are minimised by using a compensated control surface based on predicted machining errors.

The method developed in [6] uses an efficient volume model to simulate the cutting process, whereas the calculation of the optimal feedrate takes the main technological aspects of 3-axis milling into account.

Correspondence and offprint requests to: Dr G. Vosniakos, Department of Mechanical Engineering, Manufacturing Technology Division, National Technical University of Athens, Iron Polytexneiou, 157 80 Zografou, Athens, Greece. E-mail: vosniak@central.ntua.gr

Lin and Lee [7] proposed an adaptive CNC tool-path generation algorithm for precision machining of parts with freeform surfaces. From the current reference point of the cutting tool, the method can predict the next reference point of the tool for a given feed. If the predicted reference point does not satisfy a specified dimension tolerance, then the method will adjust itself according to the adaptive rules.

Yazar et al. [8] aim to develop a method for estimating the cutting forces in 3-axis milling so that the NC programmer can optimise the machining parameters, and to establish the "best" rough milling strategy to reduce machining time and cost. The paper focuses on optimising the feedrate to improve machining efficiency in end milling.

Lin and Gian [9] sought to generate machining instructions directly from the 3D point data derived from either contact or non-contact measuring devices, considering issues such as use of ordered data points to fit a smooth surface, automatic selection of multiple cutting tools, and generation of cutter paths for rough machining. The selection and optimisation of cutting tools are based on the criteria of maximum material removal and minimum tool changes, whereas the generated NC cutter path ensures fast machining and the required surface quality.

Five-axis CAM software is introduced in [10], varying the tool inclination during tool path generation, in order to achieve the best combination of scallop height, workpiece accuracy, surface roughness, and machining cost.

Lee and Chang [11] explore a methodology for applying computational geometry techniques to extract machining information of geometric constraints from a given complex surface design to support the process planning activity.

Chen and Woo [12] deal with the question of whether 3- or 4-axis machines with a different cutter could achieve the same task. Based on observations made on the geometry of the cutting tools and the degrees of freedom in 3-, 4-, and 5-axis NC machines, a new class of geometric algorithms is induced on the unit sphere.

A systematic tool-path generation methodology is presented in [13], which incorporates interference detection and optimal tool selection for machining freeform surfaces on 3-axis CNC machines using ball-end cutters. The machining time of each available tool is estimated by considering tool size, scallop height, and accessible surface area. The combination of tools, which possesses the minimum overall machining time, is selected as the optimal tool sizes.

Tool positioning in end milling of freeform surfaces is achieved in [14] based on evaluating the interference for a set of test points distributed around the circumference of the tip of the tool. Distinction is made between the cases of semi-finishing and finishing, which can be performed using a large diameter flat-end tool and a toroidal tool, respectively.

A feedrate optimisation program was developed in [15] from the relationship between optimal feedrate and local shape features, the latter being recognised by comparing NC codes of neighbouring points.

New methods of offset surface generation and milling tool selection are given in [16]. First, the workpiece area cut by a tool is calculated by detecting contact points between the tool and the workpiece in a lattice space model. A tool set is coded as a binary bit string. It is assumed the tools would be used in the reverse order of radius value and cut their corresponding area as entirely as possibly.

The work presented in the following sections started after realisation of the lack of optimality considerations in commercially available CAM software. Cutting simulation of a number of different surfaces using all the available roughing strategies, as offered by a leading CAM system, led to the observation that the necessary time for the rough machining of a part is one to two orders greater than the respective time for the finishing of the same part – following any roughing strategy, except, perhaps, for high-precision surfaces, where the minimum surface roughness is crucial, requiring small feedrates and therefore long tool paths.

Optimisation is taken to mean the minimisation of the machining time, which is mainly due to the rough machining of a surface. In addition, the finishing pass, which is always required after roughing, sets the requirement of "least material" remaining after roughing, in order to achieve higher finishing feedrates and lower tool wear, and also to achieve the desired surface roughness.

The application of optimisation methods, especially genetic algorithms, in the area of sculptured surface milling is completely new. Therefore, work would be expected to start modestly by tackling a known shape, experimenting with modelling parameters and gaining confidence, before constructing full optimisation models of generic sculptured surface milling. For this reason, the work presented focuses on a hemisphere as a milling artefact and examines modelling parameters for optimal roughing of this shape with multiple tools.

2. Genetic Algorithms

A genetic algorithm (GA) is a stochastic optimisation procedure, which can solve complex problems by imitating Darwinian theories of evolution on a computer. The concept behind the creation of genetic algorithms is the global optimisation of an objective function in a complex multi-modal search space [17]. This happens irrespectively of the nature of the studied phenomenon, meaning that the physical parameters of the problem are considered either as variables or as constants in its computational representation. Thus, a genetic algorithm can be used as an optimisation tool for any model, regardless of its physical substance [18].

Given that there are many off-the-shelf genetic algorithms available, the main step in the optimisation procedure is to develop a numerical model of the physical phenomenon and to define the problem variables. Variables belonging to the numerical domain may be combined and encoded into a series of binary strings (rows of ones and zeros) to form "numerical chromosomes". Using a random function, the genetic algorithm creates a group of chromosomes, referred to as the population. The members of the population, namely the chromosomes, are ranked according to the fitness function of the GA. The latter is a function that evaluates the objective function solution by the given chromosomes. The "fittest" chromosomes are allowed to survive and reproduce with each other, through continuously cycling genetic operators, in order to obtain a new "generation" of the population that is comprised of supposedly better chromosomes.

Although a number of genetic operators have been defined in published works, they are based mostly on the classic ones, namely inversion, mutation, and crossover. Inversion is applied to a single chromosome and reverses the order of the genes contained between two randomly selected points. Mutation is also applied to a single chromosome, and changes the allele of a randomly selected single gene to another possible value. Crossover is applied to two "mating" chromosomes. A crossover point is randomly selected for both of them and the second parts of each chromosome are swapped (simple crossover). This has also been the subject of many variations, e.g. partially matched or linear order crossover.

A genetic algorithm specifies how genetic operators are applied to a population to generate offspring, and how these replace members of the population and when the process stops.

Depending on the design of the GA, a population can have a constant or variable number of members. A powerful search engine is thus available which inherently preserves the critical balance needed in any search: the balance between exploitation (taking advantage of information already obtained) and exploration (searching new areas) [17]. The new population is then decoded back from the binary form to the variable form to reveal the actual solution. This procedure iterates in every generation until the GA converges.

A genetic algorithm's output – the value of the control variables and corresponding value of the fitness function – is usually a straightforward result. The fitness function is mainly responsible for the choice of the fittest chromosomes, and it should be designed with special attention [18].

In some cases, neural networks or other artificial intelligence methods are used, so as to lead to a better solution more quickly, by ensuring that those chromosomes with the best characteristics are in the solution [19].

3. Problem Modelling Concepts

3.1 General

First, a particular strategy for cutting hemispheres on 3-axis machining centres should be adopted. In this work, this is simply a "machining cylindrical slices" strategy. Each slice has a particular height and is machined by a particular tool. A slice can be machined by a circular tool path if the stock is not large compared to the tool diameter (simplest case) or by an additional zigzag tool path if there is much more material to clear before the cylindrical stock itself can be cleared.

Therefore, the complete solution to the problem consists of the set of slices (and corresponding scallop) heights, which in the general case are not equal, together with corresponding tools and the toolpath (circular plus any zigzagging necessary).

The quality of rough milling relies on criteria such as the least machining time and the least remaining material at the end of the roughing process. An important factor is the correct tool choice according to the geometric characteristics (diameter, depth of cut, engagement, etc.) and the applied feedrates. Note that owing to the cutting strategy selected only end-mill cutters need be, and indeed are, considered here.

The developed algorithm automatically exports the optimal tool combination for the roughing of the given surface, assuming a set of the available tools on a CNC machine.

The rapid movements of the cutter are considered negligible, because their feedrates are one to two classes of magnitude higher than those of the cutting movements. This, of course, also depends on the tool type.

For the implementation of the method, a genetic algorithm is used. For the purposes of this work, this optimisation algorithm is considered to be a "black box" which outputs values for different parameters within a prefixed range; the value of the objective function is fed back until its convergence is achieved. Details of the genetic algorithm tools can be found in [19].

3.2 Definition of the Original Block

Figure 1 shows the original block of material and the hemisphere to be machined. Let R be the radius of the hemisphere and K_1, K_2 and K_3 be the distances from the hemisphere to the borders of the block along the X, Y and Z axes, respectively. Minimisation of machining time does not depend on K_3, therefore K_3 can be set to zero. The size of the original block is $(2R + 2K_1) \times (2R + 2K_2) \times R$. The dimension A_i, see Fig. 2, is defined as $A_i = \sqrt{(R + K_1)^2 + (R + K_2)^2} - R_i$, where R_i is the radius of the circle on a layer parallel to the base of the hemisphere at a Z-height where the base of the ith scallop lies.

3.3 Optimisation Criteria and Fitness Evaluation Functions

The distribution of the roughing steps from top to bottom of the hemisphere can be assessed according to the least remaining volume criterion.

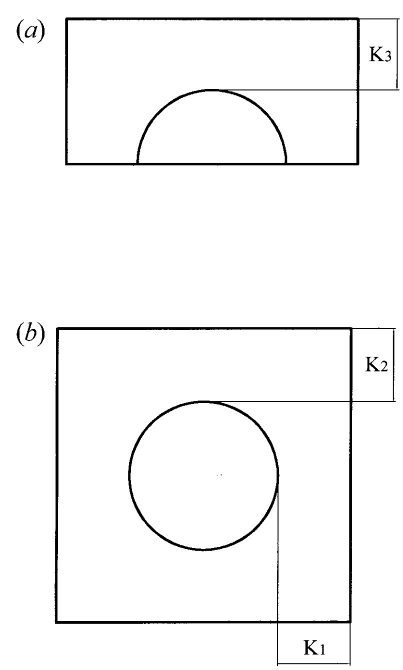

Fig. 1. Original material block definition. (*a*) *XZ* view. (*b*) *XY* view.

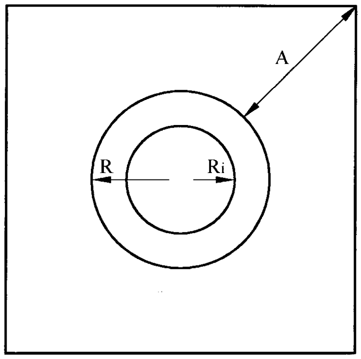

Fig. 2. Definition of A_i and R_i.

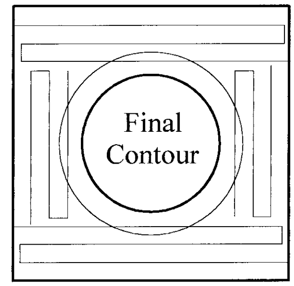

Fig. 4. The toolpath used for the method along with the original block limits and the final contour in a *XY* view for the *i*th scallop.

Optimality on each slice, as well as in the set of all slices can be judged according to the least machining time criterion.

3.3.1 Remaining Volume

Owing to the symmetry of the considered surface (hemisphere), instead of the volume of each scallop, the area in the *XZ*-plane of the first quadrant, which lies under the scallops and over the semicircle of the $Y = 0$ plane, is used for the volume in the objective function, see Fig. 3. In this way the calculations are simplified.

The area under each scallop and over the semi-circle is different in each scallop and it is defined by an iterative relation:

$$E_s = Zscallop \ (Xprev - X)$$

where *Zscallop* is the height of each scallop, *X* is the *x*-coordinate of the top intersection point of the scallop and the semicircle and *Xprev* is the *X* value for the previous scallop. Evidently, the initial and final values for the calculations must be given by the user.

The total remaining area is then calculated simply as follows:

$$E = \sum_{i=1}^{N} E_s - \frac{\pi R^2}{4} \tag{1}$$

3.3.2 Cutting Time

For the calculation of the cutting time, the tool-path length must first be calculated. It is assumed that the shape of the tool path is as in Fig. 4. The time Eq. is as follows:

$$T = \sum_i t_i = \sum_i \left\{ C_{ii} \frac{TraceX(d_{ii})}{u_{ii}} + C_{ij} \frac{TraceY(d_{ij})}{u_{ij}} + \frac{TraceZ(d_{ik})}{u_{ik}} + Penalty \right\} \tag{2}$$

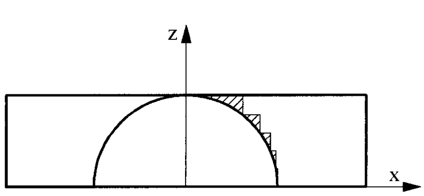

Fig. 3. The hatched area represents the remaining volume of material after the machining.

where,

the first index i is the serial number of the current step/scallop;

the second index i, j, k indicates the cutting direction of the tool with diameter d (i is for linear passes in the direction of x-axis, j is for linear passes in the direction of y-axis and k is for circular passes around the z-axis);

C_{ii}, C_{ij} are coefficients that have the value 0 or 1 depending on whether or not the specific pass is cut;

$TraceX(d_{ii})$ and $TraceY(d_i)$ denote the length of the passes realised by the d_{ii} and d_{ij} tools along the x and y direction, respectively, if C_{ii} and C_{ij} are different from 0;

$TraceZ(d_{ik})$ denotes the length of the circular pass that is always carried out by the d_{ik} tool around the z-axis;

Penalty stands for a time penalty for tool change;

u_{ii}, u_i and u_{ik} denote the feedrates of the tools d_{ii}, d_{ij}, and d_{ik}, respectively.

3.3.3 Combined Objective Function

A combination of the above two criteria may be built by using relative weights. Typically, this is

$$O = \frac{\lambda_1 T + \lambda_2 E}{\lambda_1 + \lambda_2} \tag{3}$$

where λ_1 and λ_2 are weighing coefficients (real non-negative numbers) which satisfy the equation $\lambda_1 + \lambda_2 = 1$.

4. Implementation Details

For a demonstration of the optimisation possibilities, a number of tools were considered, see Table 1, for machining medium alloy steel castings taken from a major tool manufacturer's catalogue [20], (T25M inserts).

In order for the algorithm to begin, the number of scallops to be generated is given first with upper and lower limits set by the relation $N = R/MaxDepth$ (see also Table 1). For example, for $R = 50$ mm and $MaxDepth = 5$–16 mm, it is calculated that $3 < N \leq 10$. The number of scallops is determined by the user.

Table 1. Tool Database of used tools and relevant measures.

Number	Type	Diameter (mm)	Max. depth of cut (mm)	Max feed (mm min^{-1})
1	End mill/drill	12	5	464.4
2	End mill/drill	15	5	371.5
3	End mill/drill	18	5	309.6
4	End mill/drill	20	7	356.7
5	End mill/drill	25	8	315.9
6	End mill/drill	32	11	246.8
7	End mill/drill	40	11	197.5
8	End mill/drill	50	11	158.0
9	End mill/drill	25	7	947.8
10	End mill/drill	32	8	740.4
11	End mill/drill	40	12	592.4
12	End mill/drill	50	12	473.9
13	End Mill	12	5	658.2
14	End Mill	16	5	987.3
15	End Mill	20	7	789.8
16	End Mill	25	8	631.8
17	End Mill	32	8	740.4
18	Deep End Mill	25	16	662.4
19	Deep End Mill	32	16	776.3
20	Deep End Mill	40	16	828.0
21	Deep End Mill	50	16	828.0

4.1 Multiple Slice Optimisation

A genetic algorithm, which is termed the "outer GA", provides (optimises) the heights $H(i)$ of the $i = 1, \ldots, N\text{-}1$ scallops. The height of the last scallop is calculated by the equation $\sum_i H(i) = R$.

The upper limits for $H(i)$ are given by the maximum of the variable *MaxDepth* of the tools. The lower limits are set according to logical constraints, so that the time required for the algorithm is not excessive. The height of each scallop and its base diameter together determine the geometric characteristic of the remaining material, R_{lim}, which is measured along the radius of the semicircle leading to the farthest scallop vertex, see Fig. 5. The maximum value for R_{lim}, is set by the user. Here, it was set to 12 mm, which is a value connected to the maximum depth of cut of a tool for a finishing operation.

Once a scallop height distribution (chromosome) is proposed, a geometric check is run in order to establish the group of tools that can cut each scallop, as far as their *Diameter* and *Max Depth of Cut* are concerned. The main criterion for a tool to be admissible is that its *Max Depth of Cut* should be greater than the scallop height. In that way, the validity of the chromosome is ensured.

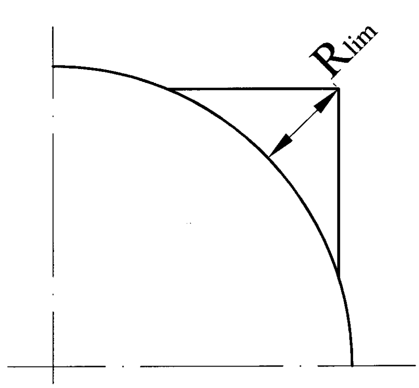

Fig. 5. Definition of R_{lim}.

The value of the optimisation function, i.e. the area under the scallops and above the semicircle, is calculated as:

$$E = \sum_{i=1}^{N} [R_i \cdot H_i] - \frac{\pi R^2}{4}$$

where $R_1 = R$ (the radius of the hemisphere) and $R_{N+1} = 0$ and N is the number of scallops.

Instead of the above, the total cutting time minimisation could be used. This is the sum of the best cutting time result for each slice optimised for the distribution of scallop height.

A third possibility is the consideration of total cutting time together with remaining scallop area as the evaluation tool by assigning non-zero relative weights λ_1 and λ_2 according to Eq. (3). The previous two possibilities correspond to marginal cases with $\lambda_1 = 1$, $\lambda_2 = 0$ and $\lambda_1 = 0$, $\lambda_2 = 1$, respectively.

4.2 Single Slice Optimisation

The other optimisation criterion is cutting time per slice, which is calculated by Eq. (2), and refers to the use of a particular tool diameter and tool path selection strategy.

Admissible tools whose diameter is greater than A_i, will be considered for a circular path strategy around the Z-axis, otherwise linear passes should be cut along the X and Y directions and then around the Z-axis.

In this second case, it is necessary to define the geometry of the linear passes and thus the block of material remaining, as controlled by angle f, see Fig. 6.

This is the task of the "inner GA", which generates the optimum value of angle f. The limits of angle f are given from the relations :

$$f_1 = \arctan\left(\frac{R}{R + K_1}\right), \quad f_2 = \arctan\left(\frac{R + K_2}{R}\right)$$

A_{ik}, is defined similarly to A_i, but for the internal block.

A_{ik} is set equal to the diameter of each tool of the group.

Depending on the comparison between K_1 and K_2, the side of the block to be cut first is decided. Experience has proved that it is best to cut the wider side first. This decision leads to the definition of the number of passes cut along the x and

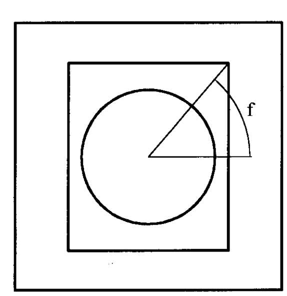

Fig. 6. Definition of angle f that is handled by the inner genetic algorithm.

y directions as a function of the diameter d_{ik} and d_{ii} or d_{ij} of the tool used. Therefore, two matrices containing values of the linear pass lengths that will be cut along these two directions are created accordingly. The expressions for calculating the number of passes along the x and y axis, the length of tool path, and the time penalty are given in Table 2.

A partial case of the method is when $K_1 = K_2 = K$, meaning that the initial block is square in the xy-plane. In this case, the angle $f_o = 45°$ and it is supposed that the internal block is also square on the same plane. Therefore, there is no reason to call the inner GA. Because of symmetry, it is considered that the same tool is used along the two cutting directions x and y, i.e. $d_{ii} = d_{ij}$.

5. Results and Discussion

To test the method developed, some software was developed in C++ in order to program the application that calls the GA tool. The implementation was targeted at proving the method's feasibility and obtaining estimates of execution time, convergence behaviour and a judgement on the degree of realism of the results and, as a consequence, also of the model constructed.

The target was the minimisation of rough cutting time of a hemisphere with $R = 50$ mm from an original block with the following dimensions $130 \times 150 \times 50$ along the x, y, and z directions, respectively. The number of scallops was taken as $N = 8$. The range of the values for angle f managed by the slave genetic algorithm was [0.61, 0.832], as defined by the original block geometry. The weights used in the evaluation function were $\lambda_1 = 0$ and $\lambda_2 = 1$.

The program outputs two data files. The first (Solution.log) gives the value of cutting time for each scallop, the minimum value of the objective function and the number of the generation of the genetic algorithm in which it appeared, see Table 3. The other (Scallops.log) presents the tools used with their IDs (idx is the serial number of the tool that cuts along the x-axis, idy is the serial number of the tool that cuts along the y-axis and idz is the serial number of the tool that cuts around the z-axis) in the tool database for each scallop, and the time for the rough cutting of each scallop, see Table 4.

The results of Tables 3 and 4 are presented in Figs 7 and 8, respectively. As can be seen in Fig. 7, the scallop heights are not constantly decreasing; there appears to be a minimum at points 2 to 4. This can be explained by keeping in mind that there are no geometrical constraints for the dimension R_{lim}, which can take any value ($\lambda_1 = 0$). This is the reason why the upper scallops are larger.

The program required several hours to run on a PC Pentium 550 MHz, 128 MB RAM with 3000 evaluations of each of the genetic algorithms used. An "evaluation" is a user-defined parameter of the genetic algorithm that refers to the number

Table 3. Results in the output file Solution.log.

```
# Objective(s): 1 401.1392
# at generation : 53
Total variables (incl. frozen): 7
5.3999 4.0370 4.0534 4.2331 5.2504   5.7116 6.0862 (15.2284)
```

Table 4. Results in the output file Scallops.log.

Scallop Number	idx	idy	idR	Time (s)
1	0	0	21	37.79
2	0	0	21	37.62
3	20	20	20	68.24
4	9	9	9	57.67
5	20	20	20	68.00
6	9	9	21	63.80
7	9	9	21	65.91
8	21	21	21	71.63

Table 2. Basic expressions for calculating scallop and pass parameters.

	$K_2 > K_1$	$K_1 > K_2$	$K_2 = K_1$
Number of passes			
N_x	$\mathrm{int}\left(\frac{(A+R)\sin f_0 - (d_{ik}+R_i)\sin f}{d_{ii}}\right)+1$	$\mathrm{int}\left(\frac{(A+R)\sin f_0 - (d_{ik}+R_i)\sin f}{d_{ii}}\right)+1$	$\mathrm{int}\left(\frac{(A+R)\sin 45° - (d_{ik}+R_i)\sin 45°}{d_{ii}}\right)+1$
N_y	$\mathrm{int}\left(\frac{(A+R)\cos f_0 - (d_{ik}+R_i)\cos f}{d_{ij}}\right)+1$	$\mathrm{int}\left(\frac{(A+R)\cos f_0 - (d_{ik}+R_i)\cos f}{d_{ij}}\right)+1$	$\mathrm{int}\left(\frac{(A+R)\sin 45° - (d_{ik}+R_i)\sin 45°}{d_{ii}}\right)+1$
Tool path length			
$TraceX(d_{ii})$	$4N_x((A+R)\sin f_0 - (d_{ik}+R_i)\sin f) + 2d_{ii}$	$4N_x(R+K_1) + 2d_{ii}$	$4N(R+K) + 2d_i i$
$TraceY(d_{ij})$	$4N_y(R+K_2) + 2d_{ij}$	$4N_y((A+R)\cos f_o - (d_{ik}+R_i)\cos f)+2d_{ij}$	$4N((A+R)\cos 45° -$ $(d_{ik}+R_i)cos\ 45°) + 2d_{ii}$
Angle f_0	$f_0 = \arctan\left(\frac{\mathrm{R}+\mathrm{K}_2}{\mathrm{R}+\mathrm{K}_1}\right)$		45°
Time penalty =	$4P$ if $d_{ii} \neq d_{ij} \neq d_{ik} \neq d_{ik-1}$ $3P$ if $d_{ii} \neq d_{ij} \neq d_{ik}$ and $d_{ij} = d_{ik-1}$ $2P$ if $d_{ii} = d_{ij} \neq d_{ik}$ and $d_{ij} \neq d_{ik-1}$ P if $d_{ii} = d_{ij} = d_{ik-1} \neq d_{ik}$ 0 if $d_{ii} = d_{ij} = d_{ik} = d_{ik-1}$	$4P$ if $d_{ii} \neq d_{ij} \neq d_{ik} \neq d_{ik-1}$ $3P$ if $d_{ii} \neq d_{ij} \neq d_{ik}$ and $d_{ii} = d_{ik-1}$ $2P$ if $d_{ii} = d_{ij} \neq d_{ik}$ and $d_{ii} \neq d_{ik-1}$ P if $d_{ii} = d_{ij} = d_{ik-1} \neq d_{ik}$ 0 if $d_{ii} = d_{ij} = d_{ik} = d_{ik-1}$	$2P$ if $d_{ii} \neq d_{ik} \neq d_{ik-1}$ P if $d_{ii} \neq d_{ik}$ and $d_{ii} = d_{ik-1}$ P if $d_{ii} = d_{ik} \neq d_{ik-1}$ 0 if $d_{ii} = d_{ik} = d_{ik-1}$

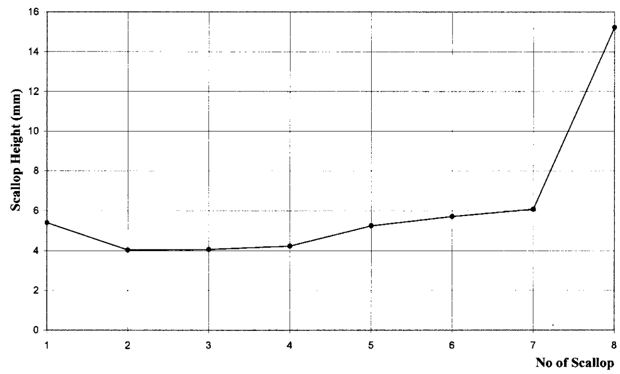

Fig. 7. Scallop height distribution.

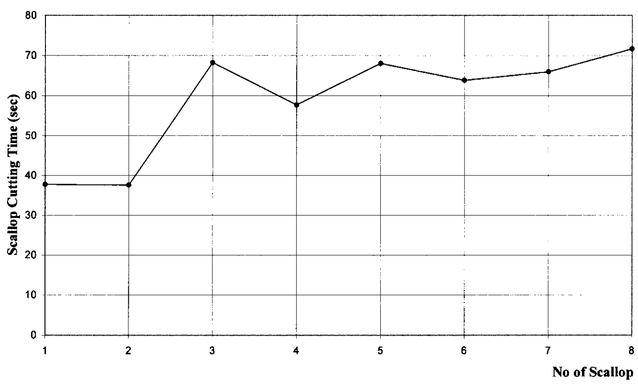

Fig. 8. Scallop cutting time.

of the objective function calls. The result is optimum for this specific number of evaluations. The higher the number of evaluations allowed, the more dependable the optimisation result is, but the higher the computational cost.

The high execution time is attributed to the continuous domain of the problem (scallop height values) combined with the large number of tools (21) considered in the tool pool. Using a commercially available GA optimisation tool the computation time would normally be much higher. This is because special convergence acceleration techniques were employed in the software used in this work [19]. However, high execution time should not misguide the reader, because the work does not claim to offer a practical industrial tool, but just to introduce a new approach that through further improvement might become practical. Note that in [19] the analogous problem encountered in shape optimisation of airfoils was connected to the need to run CFD computations within the evaluation function. It was overcome by using a neural network to replace most of the computations after training conducted with a set of initial CFD results.

The genetic algorithms used offer the possibility of viewing the value of the objective function on an appropriate data file (Conv.log). By creating the respective diagram, see Fig. 9, it can be seen that the solution obtained is satisfactory and that the value of the objective function will not be improved by more than 0.5% in the present evaluation as the number of the generations increases.

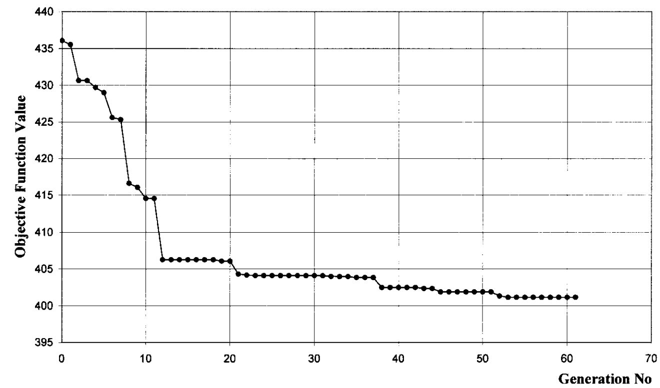

Fig. 9. Data in the Conv.log file.

According to the results, the decreasing distribution of the scallop heights does not necessarily offer the minimum total cutting time and the minimum remaining material on the desired surface, as at first thought. This specific solution means that for the given criteria there is a scallop height distribution in which the intermediate heights are minimum.

Table 4 gives as optimum tools those with ID numbers 9, 20, and 21. These tools, however, are not those with the maximum feedrates, see Table 1, but those which when combined give the minimum cutting time for each scallop and for the whole process.

During the current study, the rapid movements of the tools were neglected on purpose. The reason for this is that the rapid feedrates are between ten and a hundred times greater (~10000 mm min^{-1} on a typical CNC machine) than for the cutting movements and that the length of the rapid movements is at least ten times smaller than for the cutting movements. For this reason, a tool change penalty could absorb the rapid move time realistically.

The output of the implemented part of the algorithm is the value of the cutting time, but in general it is the value of the objective function that includes time, area, etc. Hence, the output value should be considered relatively, not absolutely. This means that this value is not the real cutting time for the given surface, even if it can give a very good approximation of it. We need to be reminded here of the initial objective of the exercise, which was to find the sequence of tools among the available ones and the scallop height distribution that produces the minimum cutting time among the possible combinations.

If calculation of the absolute cutting time of the specific toolpath is desired, it suffices to take into consideration the data related to height distribution and tool usage and to run a simulation on either a CAM system or on a CNC machine.

6. Conclusions and Recommendations for Further Work

The value of the method presented is that, for a typical tool path, the optimal tool usage is found and the height distribution of the created scallops is calculated. The calculations are performed off-line within the part programming process. A relative drawback is execution time, if a workstation or a parallel computer system is not available.

Further generalisation of the method may also involve different tool types. In such a case, problems of modelling of the different types of tool would be confronted, whereas criteria for the correct choice of tool type should be developed and the effects on the surface using a certain tool type should be fully defined (scallop pattern in 3 dimensions).

In addition, if the surface to be cut were more complex, e.g. if it had pockets, islands, etc., then the rapid movement time would not be negligible and it would be necessary to take it into account.

The number of scallops given by the user could be determined–optimised within the same procedure. The simple way would be to include all the above in a loop with a changing number of scallops (certainly within prescribed limits), which involves many computations. The elegant thing to do, however, would be to include the number of scallops within the problem representation chromosome, and allow the GA to change its values according to the results of the same evaluation function presented in this paper (time plus remaining volume).

The above are extension directions within a wider plan, which includes generic optimisation of the rough machining and finishing of freeform surfaces of different types, using a variety of tool types and cutting strategies on CNC machines.

Acknowledgements

The authors would like to acknowledge the kind offering of the genetic algorithm software developed by the NTUA optimisation group of Dr K. Giannakoglou, with the main contribution by A. P. Giotis. In addition, the help offered by Mr M. Karakassis, with the familiarisation and programming of the software is gratefully acknowledged.

References

1. D. E. Goldberg, Genetic Algorithms in Search, Optimisation and Machine Learning, Addison Wesley, 1989.
2. Z. Dong, H. Li and G. W. Vickers, "Optimal rough machining of sculptured parts on a CNC milling machine", Transactions ASME Journal of Engineering for Industry, 115(4), pp. 424–431, 1993.
3. W. L. R. Ip, "A fuzzy basis material removal optimisation strategy for sculptured surface machining using ball-nosed cutters", International Journal of Production Research, 36(9), pp. 2553–2571, 1998.
4. Y.-S. Lee and T.-C. Chang, "Automatic cutter selection for 5-axis sculptured surface machining", International Journal of Production Research, 34(4), pp. 977–998, 1996.
5. E. E. M. Lim and C.-H. Menq, "Integrated planning for precision machining of complex surfaces. Part 1: cutting-path and feedrate optimisation", International Journal of Machine Tools and Manufacture, 37(1), 61–75, 1997.
6. K. Weinert, A. Enselmann and J. Friedhoff, "Milling simulation for process optimisation in the field of die and mould manufacturing", Annals CIRP, 46(1), pp. 325–328, 1997.
7. Y.-J. Lin and T. S. Lee, "An adaptive tool path generation algorithm for precision surface machining", Computer-Aided Design, 31(4), pp. 237–247, 1999.
8. Z. Yazar, K.-F. Koch, T. Merrick and T. Altan, "Feed rate optimization based on cutting force calculations in 3-axis milling of dies and moulds with sculptured surfaces", International Journal of Machine Tools and Manufacture, 34(3), pp. 365–377, 1994.
9. A. C. Lin and R. Gian, "A multiple-tool approach to rough machining of sculptured surfaces", International Journal of Advanced Manufacturing Technology, 15(6), pp. 387–398, 1999.
10. J.-P. Kruth and P. Klewais, "Optimisation and dynamic adaptation of the cutter inclination during five-axis milling of sculpture surfaces", Annals CIRP, 43(1), pp. 443–448, 1994.
11. Y.-S. Lee and T.-C. Chang, "Application of computational geometry in optimising 2.5-D and 3-D NC surface machining", Computers in Industry, 26(1), pp. 41–59, 1995.
12. L.-L. Chen, and T. C. Woo, "Computational geometry on the sphere with application to automated machining", Transactions ASME Journal of Mechanical Design, 114(2), pp. 288–295, 1992.
13. D. C. H. Yang and Z. Han, "Interference detection and optimal tool selection in 3-axis NC machining of free-form surfaces", Computer-Aided Design, 31(5), pp. 303–315, 1999.
14. J.-M. Redonnet, W. Rubio, F. Monies and G. Dessein, "Optimising tool positioning for end-mill machining of free-form surfaces on 5-axis machines for both semi-finishing and finishing", International Journal of Advanced Manufacturing Technology, 16(6), pp. 383–391, 2000.
15. C. N. Chu, S. Y. Kim, "Feed-rate optimisation of ball end milling considering local shape features", Annals CIRP, 46(1), pp. 433–436, 1997.
16. Y. Mizugaki, M. Hao, M. Sakamoto and H. Makino, "Optimal tool selection based on genetic algorithm in a geometric cutting simulation", Annals CIRP, 43(1), pp. 433–436, 1994.
17. D. T. Pham and P. T. N. Pham, "Artificial Intelligence in Engineering", International Journal of Machine Tools and Manufacture, 39, pp. 937–949, 1999.
18. R. Teti and S. R. T. Kumara, "Intelligent computing methods for manufacturing systems", Annals CIRP, 46(2), pp. 629–652, 1997.
19. K. C. Giannakoglou, A. P. Giotis and M. K. Karakasis, "Low-cost genetic optimization based on inexact pre-evaluations and the sensitivity analysis of design parameters", Inverse Problems in Engineering (accepted)..
20. Seco-Guide Milling, © Seco Tools AB, Sweden, 1999.