

UTILIZATION OF GA FOR OPTIMIZATION OF TOOL PATH ON A 2D SURFACE

Zlatan Car*, Tonci Mikac*, Ivica Veza**

*Department of industrial engineering and management

Faculty of Engineering, Rijeka University

Vukovarska 58, 51000 Rijeka, Croatia

car@riteh.hr

** Industrial Engineering Department

Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture
University of Split

ABSTRACT

Machining centers based on the milling principles are becoming more and more popular due to their ability to handle geometrically complex workpieces. Unfortunately, several physical phenomena that often affect the quality of the desired surface. The most common problem is definition of the tool-path movements which have strong influence on the kinematics and dynamics of the machining centers. To solve this problem we have proposed utilization of the GA based mostly on the geometrical parameters. From the obtain results we could conclude that here presented method is promising and future development is necessary, especially for introduction more complex technological parameters.

Key words: List 4-5 keywords. (10 pt)

1. INTRODUCTION

Nowadays, milling machines, guided by a computer system, are employed to produce free-shape surfaces to supply the need of mass manufacturing production. Machining centers based on the milling principles are becoming more and more popular due to their ability to handle geometrically complex workpieces. Unfortunately, several physical phenomena, such as machine kinematics, thermal effects, static and dynamic loading, etc. often affect the quality of the desired surface. If we discuss this in the effect of the high speed machining the most common problem is definition of the tool-path movements which have strong influence on the kinematics and dynamics of the machining centers, and production utilization. In this paper we will discuss the problem of the tool-path definition by utilization of genetic algorithm (GA).

Computer-aided manufacturing (CAM) refers to the software used to generate the instruction codes for a CNC machine in order for it to cut out a shape designed in a

computer-aided design (CAD) system. The output from the CAM software is usually stored in cutter location (CL) data file and after that postprocessed in a simple text file of G-code. While it has long been the goal to make the CAM software that can run automatically, it generally requires a highly skill human operator to define the necessary parameters and strategies that will generate an effective tool path.

Firstly we will introduce a typical configuration of the small size vertical machining center with fourth-axis with the rotary axes on the table depicted in Figure 1, which we have used to simulate in our experiment. The cutting tool is guided by axial commands carrying the 3 spatial (Cartesian) coordinates of the tool-tip in the machine coordinate system M and the one rotation angles (Figure 1).

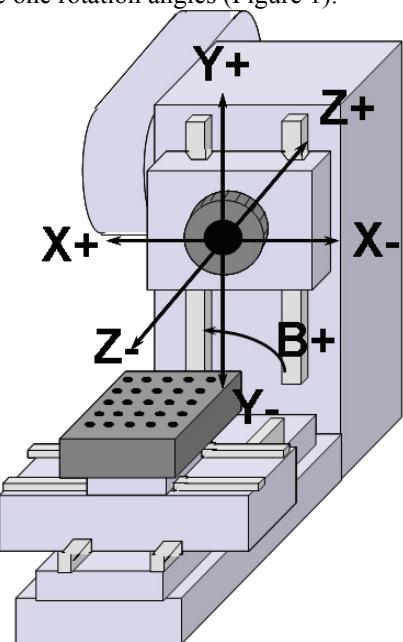


Fig. 1. Simulated four axes machining center

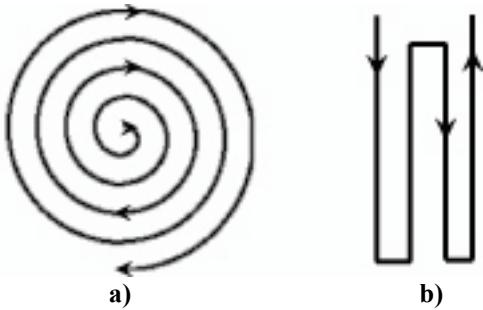


Fig. 2. Zigzag (a) and spiral (b) toll-path

Typically, the CAM software distributes the CL-points along a set of curves which constitutes the so called zigzag or spiral tool-path (Figure 2) [4]. An appropriate transformation into the M-system generates the set of the machine axial commands providing reference inputs for the controllers of the milling center.

In order to guide the tool between the prescribed CL-points, an axial command translates the centers of rotation (Figure 1.) and simultaneously rotates the P-coordinates. Consequently, the tool-tip (affected by the machine kinematics) travels in the P-system along a non-linear trajectory.

By making use of various software tools such as PRO/Engineer (Parametric Technology Corporation), CATIA (Dassault Systemes), etc. it is possible for a particular model to generate the tool motion for CNC machines. As it is not known by which method the tool path is generated, it cannot be determined with certainty whether such an obtained path is the good solution, most optimal one. The aim of using a genetic algorithm is to obtain as optimal tool-path as possible for a pre-set model, and as such compare it to a pre-set path.

The genetic algorithm was suggested by John H. Holland in the seventies of the last century, as a computer process which imitates the natural cycles and applies them to abstract genes (Golub, 1996; Haupt and Haupt, 1998; Hyun-Chul, et al., 2005, Pang, 2002). With time the genetic algorithms have proved to be an effective tool in the engineering practice, especially in the field of optimization.

In this paper we have used CAM tool PRO/Engineer (Parametric Technology Corporation) for automatic definition of the pre-set tool-path, which has been latter compared with results obtained from GA. The GA which was utilized it was based on the simple GA model with introduction of the some additional operators.

2. PROBLEM DEFINITION

Researchers have discussed a number of sophisticated methods to optimize a zigzag and spiral pattern (Srivastava et al., 1995; Chou and Yang, 1992) often combined with techniques dealing with the geometric complexity of the workpiece (Kim and Jeong, 1996). Moreover, a different

0	1	2	9
10	11	12	19
20							29
30				34			
...				44			...
...							...
...							...
90	91	92	n

- [■] initial point of machining
- [▨] cells which are not machined
- [■] final point of machining

Fig. 3. Approximation of the 2Dsurface

off-line methods are available to generate a suitable non-uniform tool-path, for instance: the neural network modeling approach (Suh and Shin, 1996), the Voronoi diagram technique (Jeong and Kim, 1999a), the monotone chain method (Park and Choi, 2000), etc. However, a robust algorithm to generate such complicated patterns is still an open problem. On the other hand, the zigzag/spiral approach is simple and robust and therefore suitable to be embedded into conventional CAM applications. But using those methods it can be very difficult to optimize tool-path between the islands of the geometrically complex workpieces.

To simplify the complex problem we will discuss here problem of the optimization of toll-path on the 2D surface, as approximation of 2.5 D tool movement on the machining center. To manipulate the genetic algorithm, as well as the surface parameters, 2D surface will be approximated as a set of cells of equal sizes as is shown on the Figure 3. Also we need to say that in this paper, an initial model is established by identifying on the machine the acceleration and the jerk of each axis. The following hypotheses are used:

- feedrate is considered as continuous,
- acceleration and deceleration present rectangular profiles,
- limited jerk is taken into account to limit feedrate at the crossing of each tool path curvature discontinuity.

2.1 Fitness function

Fitness function was defined only by geometrical requirements here presented paper, influence of the technological parameters have been mostly eliminated. Only one technological parameter was used, machining time. This was done to simplified problem. On a 2D surface, approximated with cells, the tool-path will be better, the more cells it has covered between the initial and final cell. The aim of the tool-path is to cover all cells in a few steps as possible. Fitness function needs to cover three requirements:

- all needed cells to be machined,

- minimum number of turnings, and
- to reduce number of tool-path crossovers.

The total individual fitness will be equivalent to the portions and significances of particular individual requirements.

$$f(x) = 0,55 \cdot f_{cov} + 0,25 \cdot f_{speed} + 0,15 \cdot f_{rep} + 0,05 \cdot f_t \quad (1)$$

The requirement by which the quality of tool path is described in terms of the coverage of the cells is:

$$f_{cov} = \frac{\frac{n_{necessary}}{(1 + Abs(n_{necessary} - n_{machined}))}}{n_{necessary}} \quad (2)$$

As technological parameter we have introduced a time needed to machine defined surface. The federate we have consider continuous this parameter can be presented as:

$$f_{speed} = \frac{t_m}{n_{necessary}} \quad (3)$$

Where t_m is defined as:

$$t_m = \frac{\pi \cdot L \cdot D}{1000 \cdot v \cdot Z \cdot S_z} \left(\frac{W}{b} \right) \quad (4)$$

L = length of workpiece [mm]

D = cutter diameter [mm]

W = width of workpiece [mm]

n = cutting speed [m/min]

Z = number of teeth/flutes

S_z = feed per tooth [mm/tooth], and

b = width of cut /radial depth of cut [mm].

Also the tool path should at best machine each cell only once. This cannot be defined as a marginal condition because the configuration of the surface may request of the tool-path that it move once again across an already machined cell for the machining of a yet unprocessed cell.

$$f_{rep} = \frac{\frac{n_{necessary}}{(1 + n_{doubled})}}{n_{necessary}} \quad (5)$$

Since it is a matter of tool path optimization, the total number of tool turnings affects substantially the processing performance, especially in a case of the HSM. This was taken in consideration by following requirement.

$$f_t = \frac{\frac{n_{necessary}}{(1 + n_{turns})}}{n_{necessary}} \quad (6)$$

3. DESCRIPTION OF GA IMPLEMENTATION

3.1 Individual representation

In this case the binary representation of a individual cannot be used. A set of real numbers will be used as description chromosome, in which every number represents a field on a 2D surface, and the order of numbers represents the course and the direction of the tool motion.

3.2 Evolutionary program

The evolutionary program is represented by pseudo-code:

```

Genetic_algorithm (StartCell, EndCell)
{
    Popualtion_Initialization (StartCell, EndCell)
    i = 1;
    Do While i < No_Generation
    {
        i = i + 1;
        Resolve_Doubled ()
        Fitness ()
        Sorting ()
        Selection (% Selection)
        Crossover (% Crossover)
        Mutation (% Selection, % Mutation, Int_Mut)
        Resolve_Loops ()
    }
    Solution_Representation ()
}
```

By the initialization of population the initial population from population size is generated. For every individual (path) the initial gene (cell) is given as an entry point of the tool. Every subsequent gene represents the adjoining cell and is randomly selected. In every evolutionary cycle the individuals go through an eliminatory selection, where the weaker individuals are dying out, and are replaced by newly individual's emergent crossover.

In the process of crossover a new crossover model is developed, which is not based on the former general methods of random selection of genes for crossover. An algorithm is set up which can define the gene i.e. the crossover cell. Model of the implemented crossover is given by the pseudo-code:

FindGenForCrossover (MyChrom)

```
{
    i = Random(1 to 3)
    if i = 1 then FindGenForCross =
        RandGenOfChro(MyChrom)
    else if i = 2 then FindGenForCross =
        GenWithTurning(MyChrom)
    else if i = 3 then FindGenForCross =
        DuplicatedGen(MyChrom)
}
```

In the crossover process, during the selection of genes for crossover, the evolutionary program itself find a gene from the chromosome by the selection from among three possible cases. The first may involves a random gene selection, which is based on the former familiar methods, whereas in the second case the algorithm will choose a gene after which the tool-path has turned unnecessarily from the previous course of the processing. In the third case the algorithm will find the gene which already exists in the path, and will offer it to the crossover operator.

3.2.1 The crossover gene on the unnecessary turning of the path

The algorithm finds a gene on which the tool-path changes while there is a possibility for a previous course of path to continue (Figure 4.).

Chromosome 1_i:

0-1-2-6-5-4-8-9-13-12,
 $Fitness1_i = 0,31428$

Chromosome 2_i:

0-1-2-3-7-11-15-14-10-6-5-9-13-12,
 $Fitness2_i = 0,44761$

1►	2►	3▼	
6▼	5◀	4◀	
7►	8▼		
10■	9◀		

a) Chromosome 1_i

1►	2►	3►	4▼
	11▼	10◀	5▼
	12▼	9▲	6▼
14■	13◀	8▲	7◀

b) Chromosome 2_i

Fig. 4. Example of individuals chosen for crossover, with presentation of crossing point

After the crossover new individuals are obtained (Figure 5.) with the characteristics as follows:

Chromosome 1_{i+1}:

0-1-2-6-5-9-13-12
 $Fitness1_{i+1} = 0,29777$

Chromosome 2_{i+1}:

0-1-2-3-7-11-15-14-10-6-5-4-8-9-13-12,
 $Fitness2_{i+1} = 0,91111$

1►	2►	3▼	
	5▼	4◀	
	6▼		
8■	7◀		

a) Chromosome 1_{i+1}

1►	2►	3►	4▼
12▼	11◀	10◀	5▼
13►	14▼	9▲	6▼
16■	15◀	8▲	7◀

b) Chromosome 2_{i+1}

Fig. 5. Example of individuals obtained after crossover

3.3.2 The crossover gene on the repeating cells

In this case the algorithm carries out the test on the chromosome with the aim of finding an occurrence of double (repeating) genes (Figure 6.).

Chromosome 1_i:

0-1-2-3-7-6-5-1-2-3-7-11-10-9-8-12,
 $Fitness1_i = 0,1925$

Chromosome 2_i:

0-1-5-4-8-9-10-11-15-14-13-12,
 $Fitness2_i = 0,35428$

1►	2►	8►	3►	9►	4▼	10▼
		7▲			6◀	5◀
15▼		14◀		13◀		12◀
16■		9◀				

a) Chromosome 1_i

1►	2▼		
4▼	3◀		
5►	6►	7►	8▼
■12	11◀	10◀	9◀

b) Chromosome 2_i

Fig. 6. Example of individuals chosen for crossover, with presentation of crossing point

After the crossover new individuals (Figure 7.) are obtained with the characteristics as follows:

Chromosome 1_{i+1}:

0-1-2-3-7-6-5-4-8-9-10-11-15-14-13-12,
 $Fitness1_{i+1} = 0,91428$

Chromosome 2_{i+1}:

0-1-5-1-2-3-7-11-10-9-8-12,
 $Fitness2_{i+1} = 0,23094$

1►	2►	3►	4▼
8▼	7◀	6◀	5◀
9►	10►	11►	12▼
16■	15◀	14◀	13◀

a) Chromosome 1_{i+1}

1►	2▼	4►	5►	6▼
	3▲			7▼
11▼	10◀	9◀	8◀	
12■				

b) Chromosome 2_{i+1}

Fig. 7. Example of individuals obtained after crossover

Mutation is performed in such a way, that an individual is selected from the population randomly, and with the same algorithm just like crossover, an adequate gene on an individual is found, and the path of that gene (cell) is muted. The mutation of a section of the path represents in fact a renewed creation of the path by random selection of neighbouring cells. The mutation intensity represents the number of repetitions of mutation procedure on the selected individual. In that way a permanent refreshment of the population with individuals is provided.

In population initialization, and in the procedure of chromosome crossover and mutation, it is possible for the tool path to perform the repeating motion in its smaller or greater part, in another worlds it performs a looping. In this case it is a matter of unnecessary motion so a genetic operator which recognizes such loops is created so it annuls them.

Chromosome_i:

$$0-1-2-3-7-6-5-9-10-6-5-4-8-9-10-14-13-12, \\ Fitness_{i_1} = 0,28242$$

By using a new operator on an individual, a new individual is produced with the characteristics as follows:

Chromosome_{i+1}:

$$0-1-2-3-7-6-5-4-8-9-10-14-13-12, \\ Fitness_{i+1} = 0,44761$$

1►	2►	3►	4▼
8▼	7◀	6◀	5◀
9►	10►	11▼	
14■	13◀	12◀	

a)

1►	2►	3►	4▼
12▼	7▼11◀	6◀10◀	5◀
13►	8►14►	9▲15▼	
18■	17◀	16◀	

b)

Fig. 8. Example of individuals before (a) and after (b) implementing loop operator

By population initialization, and in the procedure of chromosome crossover and mutation, there may be several identical paths. As they burden the algorithm unnecessarily, the algorithm discards individuals and replaces them by arbitrary creation of new individuals, much like in population initialization.

4. SIMULATION AND RESULTS

To test the quoted new algorithms a program is made using C++ as a tool. The tests have been done on a Pentium IV computer with the processing power of 3000 MHZ with 2 GB memory. The parameters of genetic algorithms used during experiments were as follows:

- population size = 50 x number of processed cells
- individual size (number of genes) = 4 x number of processed cells
- elimination selection = 20%
- mutation = 1%
- intensity of mutation = 1
- number of evolutionary cycles (generation) = 300

Firstly we have obtained tool-path by using CAM tool PRO/Engineer (Parametric Technology Corporation), and we have simulated the working condition of the small size horizontal machining center. Obtained tool-path from CAM software we have evaluated with same fitness function as we have described earlier. Then simulation was performed by algorithm which utilize GA presented in this paper. Results from one simulation run are presented on the Figure 9. From the obtained we can conclude even we have used only geometrical information for optimization of the tool-path, we have obtain better results for the optimization objective. In table 1. are presented obtained results from commercial package for defined surface.

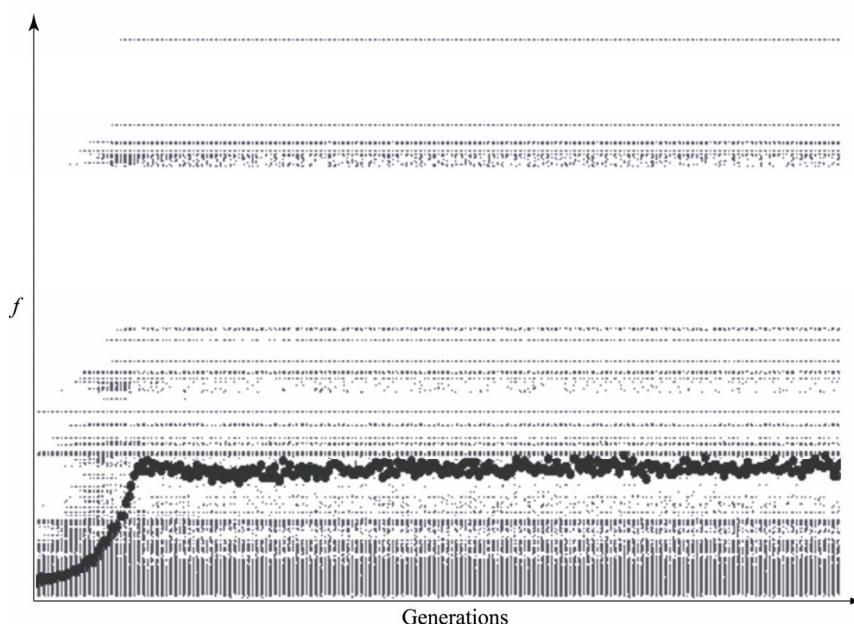


Fig. 9. Convergence of GA of one average simulation run

Table 1. Comparison of the results obtained from commercial and GA utilisation

Shape and description of surface	Commercial algorithm	Time of solution finding, fitness [seconds, fitness]	
		Genetic algorithm	
Square surface with 10 x 10 cells	t = 9 sec. f = 92,432	t = 25 sec., f = 91,25 t = 19 sec., f = 91,25 t = 10 sec., f = 91,43	Avg(t) = 15 sec. Avg(f) = 91,31
Square surface with 100 x 100 cells	t = 5022 sec. f = 91,15	t = 102 sec., f = 90,71 t = 85 sec., f = 91 t = 22 sec., f = 90,58	Avg(t) = 54,85 sec. Avg(f) = 90,78
Square surface with 100 x 75 cells	t = 1520 sec. f = 91,42	t = 267 sec., f = 90,58 t = 150 sec., f = 77,5 t = 19 sec., f = 91,11	Avg(t) = 146,33 sec. Avg(f) = 89,39
Square surface with 100 x 80 cells	t = 986 sec. f = 93	t = 89 sec., f = 90,83 t = 207 sec., f = 91 t = 202 sec., f = 90,9	Avg(t) = 156 sec. Avg(f) = 90,71

We can see that for simple geometry and small size surface commercial algorithm is giving better results. However, when geometry has become complex time needed to obtain good feasible solution was much lower then in a case of the using predefine algorithms in the commercial packages.

9. CONCLUSION

Nowadays, milling machines, guided by a computer system, are employed to produce free-shape surfaces to supply the need of mass manufacturing production. The most common problem is definition of the tool-path movements. In this paper it was introduced new approach for tool-path optimization. We have utilized GA for tool-optimization. In problem definition as main parameters we have used geometrical features, only one technological parameter was used (machining time). From obtained results we can conclude that for the complex geometry here proposed method is giving better results then a commercially used one. Feasible good solution is obtained earlier then in commercial one, and machining time is decreasing. As a next step in our research we will introduce more complex technological parameters. Also we are planning as next step to develop a modul for the commercial software for the online programming of the CNC machining centers.

REFERENCES

- Golub, M. (1996). An implementation of binary and floating point chromosome representation in genetic algorithm. *Proceedings of 18th International Conference ITI*, pp. 417-422., Pula, Croatia
- Haupt, R. L. and Haupt S. E. (1998). *Practical genetic algorithms*, A Wiley-Interscience publication, Canada.
- Hyun-Chul, K., Sung-Gun L. and Min-Yang Y. (2005). An Optimized Contour Parallel Tool Path for 2D Milling with Flat Endmill, *The International Journal of Advanced Manufacturing Technology*.

Pang, K. W. (2002). Tool path optimization in layered manufacturing. *IIE Transactions*, Vol.34., pp 335-347.

Srivastava, A. K., Veldhuis, S. and Elbestawit, A. (1995). Modeling geometric and thermal errors in a five-axis CNC machine tool. *International Journal of Machine Tools and Manufacturing*, Vol. 35, No9, pp. 1321-1337.

Chou, J. and Yang, D. (1992). On the generation of coordinate motion of five-axis CNC/CMM machines. *Journal of Engineering for Industry*, Vol. 114, pp. 15-22.

Kim, K., and Jeong, J. (1996). Finding feasible tool-approach directions for sculptured surface manufacture. *IIE Transactions*, Vol. 28, No.10, pp. 829-836.

Suh, S.-H. and Shin, Y.-S. (1996). Neural network modeling for tool path planning of the rough cut in complex pocket milling. *Journal of Manufacturing Systems*, Vol. 15, pp. 295-324.

Jeong, J. and Kim, K. (1999). Generating tool paths for free-form pocket machining using z-buffer-based Voronoi diagrams. *Advanced Manufacturing Technology*, Vol. 15, pp. 182-187.

Park, S. C. and Choi, B. K. (2000). Tool-path planning for directional parallel milling. *Computer Aided Design*, Vol. 32, pp. 17-25.