ACCELERATING TOOL PATH COMPUTING IN TURNING LATHE MACHINING

Antonio Jimeno, Sergio Cuenca, Antonio Martínez, Jose Luis Sánchez Romero

Computer Science Technology and Computation Department
University of Alicante
Apdo. Correos 99
03080 Alicante, Spain
email: {jimeno, sergio, amartinez}@dtic.ua.es

ABSTRACT

Tool path generation is one of the most complex problems in Computer Aided Manufacturing. Although some efficient strategies have been developed, most of them are only useful for standard machining. The algorithm called *Virtual Digitizing* avoids this problem by its own definition but its computing cost is high and make it difficult for being integrated in standard machining in order to adopt the new ISO standard 14649. Presented in the paper there is a Virtual Digitizing architecture that takes the advantages of Reconfigurable Computing (using Field Programmable Gate Arrays) in order to improve the algorithm efficiency. FPGAs are used as low cost and low frequency coprocessor to accelerate the calculation of tool path, meeting the actual restrictions of the Computer Numeric Controls (CNCs) at the same time. A prototype has been implemented to measure the real impact on the total computing time.

1. INTRODUCTION

In order to machine a surface by means of a cutting tool on a CNC machine tool, a series of 3D or 2D coordinates that define its motion must be supplied. These points are usually referred to as tool centre positions. In this way, the problem can be expressed as *obtaining a trajectory of tool centres that defines the desired object to be machined with a given precision,* in literature the problem is also known as *the tool compensation problem* [1].

With a given object and tool, a solution cannot always be found because of the curvature of the surfaces [2]. In these cases, the problem is redefined in order to obtain a trajectory that defines the closest surface that contains the desired object (that is, without collision). Figure 1 shows the trajectory (tool path) of a circle centre point in order to define a surface. In this case, for the sake of simplicity, the problem is presented in 2D. For 3D surfaces the problem becomes more complex.

Partial solutions to this problem use surface offsets generated by different methods [2,3,4]. However, these offset-surfaces are restricted to one-radius tools (i.e spherical, cylindrical and conical) and are not valid for more complex tools, such as toroidal ones with two radii. Moreover, in most cases, self-intersection problems arise according to the surface curvature. Thus, more sophisticated and higher cost computing techniques are needed to detect and solve these problems.

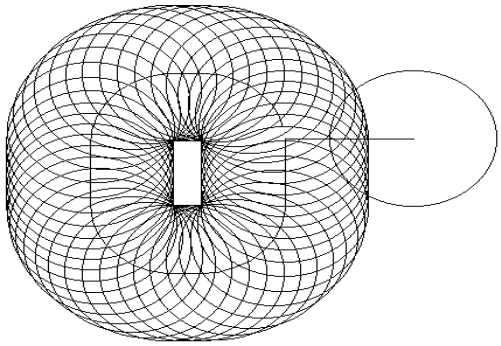

Fig.1. Circle trajectory in order to get a rectangle

The Virtual Digitizing algorithm [5] computes the tool path by means of a "virtually digitised" model of the surface and a geometry specification of the tool and its motion, so can be used even in non-standard machining (retrofitting). This algorithm was developed by one member of our research group and is included in commercial shoe last CAD/CAM software called Forma3D® (from the Spanish Footwear Research Institute, INESCOP). This software is currently a world leader in the CAD/CAM software for shoe lasts. The Virtual Digitalization is simple, robust and avoids the problem of tool-surface collision by its own definition, but its computational is higher than the others approaches.

On the other hand, the idea of integrating trajectory generation into the numerical control itself is now becoming more common. The new ISO standard 14649 (also called STEP-NC) remedies the shortcomings of ISO 6983 by specifying the machining processes rather than machine tool motion by means of machining tasks.

Unfortunately, in traditional industrial fields such as the footwear industry, there are no high-performance computers with a regard to design and manufacture. The use of low-performance computers and standard operating systems is therefore a restriction, since they share both the management tasks and those of the CAD/CAM. This is specially true in the CNC for shoe lasts turning lathe machines, where the clock frequency of the system must be keep low to improve the inmunity to the electromagnetic interferences generated by their electromechanical parts.

In [6, 7] some specific methods for shoe lasts turning lathe machines are proposed. However, their computing cost is high and no specific hardware approach is used for tool path computation.

In this work, we propose the use of specific hardware to accelerate the trajectory generation in the control itself and to facilitate the adoption of the new standard by the traditionals CNCs.

A study about the theoretical impact of Reconfigurable Computing on the Virtual Digitizing algorithm was performed in [8]. In the present work a real implementation has been carried out in order to evaluate the viability of the proposal. Using this approach, complex calculations can be made in real time without having to replace the numerical control computer and maintaining a low clock rate; only a FPGA-based coprocessing board would have to be added.

The paper is divided in the following parts: In section 2 the generic algorithm is explained. In Section 3 a FPGA-based architecture is proposed to implement it efficiently. Section 4 summarizes the experiments in order to probe the goodness of the architecture.

The architecture has been succesfully tested in the generation of helicoidal trajectories but, due to the algorithm implemented, it can be used in any other machining environment.

2. VIRTUAL DIGITIZING STRATEGY

With the virtual digitising approach the centre tool points are obtained by virtually touching the object to mechanise . This algorithm typically used to compute pencil curve tracing [9], internally works as mechanical copiers do: the copying arm touches the surface and a group of arms transmitted the movement to the cutting wheels which perform the same movement and finished the copied model.

Due to the fact that all the machining processes are simulated, this algorithm has no restrictions in tool or machine specifications, so the algorithm can be used even in non-standard machining (e.g. in retro-fitting machining).

The digitalization algorithm becomes simple once the surface and tool motion are well defined. Basically, the behavior can be described as follows: For each point of the trajectory the part surface is transformed in order to face the cutting tool. Then the minimum distance from every surface point to the tool is computed in the direction of tool attack axis. This distance determines the tool center point for the current step in the virtual digitalization process. Physically, we select the point that touches the tool surface in first place when the tool is moved along the attack axis.

The basic pseudo-code algorithm can be expressed as follows:

```
For every trajectory position Toolpos do
 Min_dist=∞
 For u in Surface_Rows do
   For v in Surface_Columns do
    p'(u,v) = p(u,v) * TR4x4
    Distance=D(p',Tool)
    If Distance<Min_dist
    then Min_dist=Dist
    Endif
   Endfor
 Endfor
Tool_centre=Centre_point(Min_dist,Toolpos,TR4x4)
Add_trajectory(Tool_centre)
EndFor
```

Fig.2. Basic virtual digitising algorithm

Analyzing algorithm, it is possible to observe up to three nested loops. One of them, the most internal one, is used to access to every surface point in the selected surface, that is, it consists into two loops, one for rows and the other for columns in fact. The most external loop goes through every trajectory position. In order to obtain a good finishing quality, it is necessary produce, at least, as many trajectory points as points the surface has.

Let assume n as the maximum number of surface points, and m as the number of trajectory positions, then the cost of the algorithm, is: $O(m \cdot n)$.

Values for n and m depend on the model size and the precision desired for the machining. Note that n value consists of a grid of Surface_Rows x Surface_Columns in size for the Algorthim 1. The more grid points used in each dimension to represent a surface, the finer the spatial resolution of our discretization and the more accurate our trajectory.

As a guide, usual values in shoe last machining can be 10 x 10 x 200 mm, and a grid of approximately 130 x 120 points is used, which implies a distance of 2 mm between points in each surface dimension. Fig 3 shows a traditional turning lathe, consisting of three different axes. All of them perform a spiral movement around the object to be machined. The tool selected is a 3D torus that simulates the cutting tool in movement.

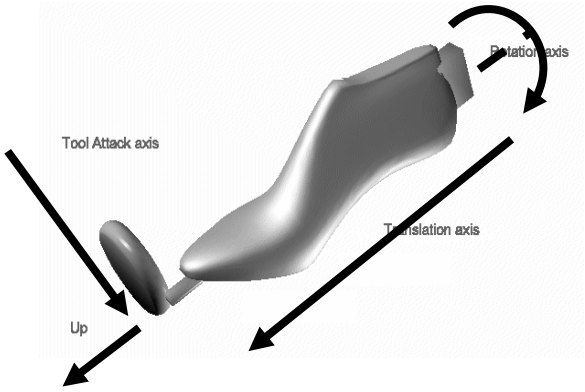

Fig. 3. Axis implied on a shoe last turning lathe machine

3. FPGA-BASED COPROCESSOR APPROACH

On observing the Basic Virtual Digitizing algorithm explained above, we notice that most of the complexity resides inside the third loop. By accelerating the functions called in this part, the total computation time can be significantly reduced.

There are three different operations inside the third loop:

Point transformation : $p'(u,v) = p(u,v) * TR_{4x4}$

A 3D transformation is applied to every surface point according to the selected strategy, so that the tool faces the surface. This operation is made by means of a 4x4 transformation matrix. From a generic standpoint, the process consists of a row * matrix post multiplying.

Distance computing: *D(p',Tool)*

This function computes the distance between a surface point and the tool in the tool attack direction. Depending on the complexity of the tool geometry - sphere, torus, cone, and so on - the function becomes more complex. For example, the distance function computes the distance between a 3D point and a 3D torus in the tool attack direction (Y axis) and can be expressed in equation 1.

$$D(x,y,z) = T_y - y - \sqrt{\left(R + \sqrt{r^2 - (x - T_x)^2}\right)^2 - z^2} \quad (1)$$

Where:

- T_x, T_y are the x,y coordinates of the torus centre
- x,y,z are the 3D point coordinates
- R, r are the major and minor torus radii

Comparison and assignment: *If Distance<Min_dist then Min_dist = Distance*

Finally, the third nested loop makes a comparison and an assignment if the computed distance is shorter than the current minimum distance.

These three different operations are carried out on every point of the tool trajectory and for every grid point on the original surface. Any optimization made at this level will significantly improve the total computation time.

As expressed above, distance computing implementation varies on tool geometry and point transformation depends on tool path strategy. So if we create hardware circuits (as ASICs) in order to speed up the algorithm for each function, we will need as many circuits as different strategies or tools we are going to use, that is, an expensive and complex architecture. A smart solution would be the use of reconfigurable circuits.

Figure 4 shows our proposed reconfigurable architecture used to perform tool trajectories with virtual digitizing. Different machines, tools and tool path strategies will imply different operation cores for point transformation and distance calculation functions. So GPU (General Purpose Unit) will choose the task involved at a time and reconfigure the RCU (Reconfigurable Co-processor Unit) with the appropriated configuration files stored in the Configuration Memory. Using the partial reconfiguration facilities provided by FPGA devices, only a few cores are needed to be maintained in the Conf. Memory while the framework of the algorithm keeps in the RCU.

Surface data and results of processing are stored in a shared memory in order to facilitate the data transactions between both units. An additional channel is used for controlling and configuring the RCU.

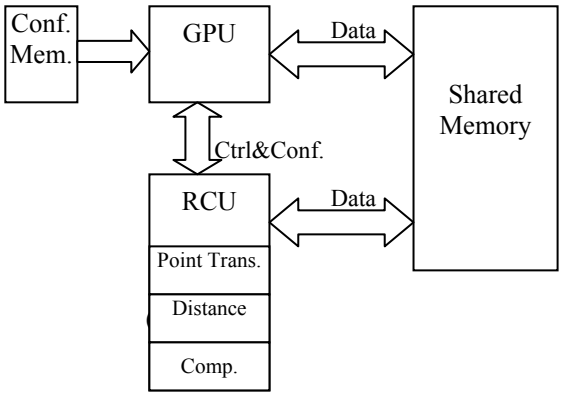

Fig. 4. Virtual Digitizing Architecture

Because the data dependencies between the three functions, the strategy adopted to get the best performance is to unroll the loop by means of a pipeline with three stages, one for every operation. At the same time, every stage will be segmented in several sub-stages to get a well balanced pipeline.

4. IMPLEMENTATION AND RESULTS

In order to validate the architecture a prototype has been implemented using the Celoxica RC1000PP PCI board and HandelC high level HDL. The board is populated with one Xilinx Virtex1000 FPGA and four 512Kx32 memory banks which can be accessed in parallel. The local memory of the board is shared with the host, and the transactions of large data blocks can be performed by means of high speed DMA channels. Also there are two additional ports to send commands to the FPGA (Control port) and receive status words in the host (Status port).

Following the segmentation strategy, all operations in the third loop could be included in one sufficiently large FPGA and executed in a pipeline fashion. However, the distance calculation, due to its floating point nature may be a difficult and resource consuming task for implementing on this version of the Virtex devices. On the other side, the point transformation operation can be easily migrated to fixed point arithmetic. For this reason we propose, as first approach, to use the FPGA to perform the point transformation and to leave the distance calculation in the charge of the host microprocessor. The algorithm partition is shown in figure 5.

```
//GPU task
For every trajectory position Toolpos do
 Receive Surface_TR
 For p in Surface_TR do
    Distance=D(p',Tool)
    If Distance<Min_dist
    then Min_dist=Dist
    Endif
 Endfor
 Tool_centre=Centre_point(Min_dist,
Toolpos,TR4x4)
 Add_trajectory(Tool_centre)
EndFor
```

```
//RCU task
For every trajectory position Toolpos do
 For u in Surface_Rows do
   For v in Surface_Columns do
    If p(u,v)∈Tool_Influence_Area
      p'(u,v) = p(u,v) * TR4x4
    Endif
   Endfor
 Endfor
 Send Surface_TR
EndFor
```

Fig. 5. Partition of the basic algorithm

Where:

- *Tool_Influence_Area* is the area of influence for the tool in every trajectory position, that is, the subset of the 3D surface points that the tool could reach in its actual position.
- *Surface_TR* is the set of transformed points in the *Tool_Influence_Area*.
- *Send* and *Receive* primitives include the mechanims and sinchronization methods to transfer blocks of data between the GPU and the RCU.

Two levels of segmentation were used to implement the architecture. The first level allows overlapping the tasks of both parts of the system. While RCU computes the Surface_TR for the iteration (i), the GPU works on the Surface_TR for the iteration (i-1).

```
//GPU task
Pack(Surface);
Do_DMA_Write(Surface,ram0,ram1)
PP1000WriteControl(START)
For every trajectory position Toolpos do
 PP1000ReadStatus() //block until FPGA write
 Do_DMA_Read(Surface_TR,ram2,ram3)
 PP1000WriteControl(ACK)
 Unpack (Surface_TR)
 Tool_centre=Process(Surface_TR)
 Add_trajectory(Tool_centre)
EndFor
```

```
//RCU task
PP100ReadControl() //block until host write
For every trajectory position Toolpos do
 Process(Surface,ram0,ram1,ram2,ram3)
 PP1000WriteStatus(DATA_VAL)
 PP1000ReadControl // block until host write
Endfor
```

Fig. 6. Synchronization of threads

The synchronization between the two threads (GPU task and RCU task) is performed by means of two RC1000 library functions that allow the access to the Control and Status ports. Both functions are blocking and only return when the read or write operation has completed.

As figure 6 shows, the GPU starts the algorithm writing the data surface to the memory banks 0 and 1 in the RC1000 board. Then it sends the command to the FPGA and keeps waiting the RCU response. Once the FPGA has received the command in the Control port, the pipeline starts its work reading the data surface and writing the transformed points to the banks 2 and 3. When the pipeline ends the loop iteration, the FPGA send the data valid code through the Status port. Next, the host read the data using a DMA, and acknowledges the transaction through the Control port. At this point the FGPA starts the second iteration while the host begins with the completion of the first one. A new synchronization will be place when both of them finish their work.

The second level of segmentation performs the unrolling of the internal loops to accelerate the execution within the RCU. In order to get the best performance, the points of the surface were packed into 64bit words, coding every coordinate in a 21bit fixed point number. This way was possible to access one point in a single clock cycle reading banks 0 and 1 in parallel. In a similar way, the transformed points can be written to the banks 2 and 3, in

only one cycle. To pack and unpack the data send and received to/from the RCU several software functions was designed and added to the GPU code. The resultant RCU pipeline is composed by 5 stages where the products of TR are perform by multipliers synthesized by the HandelC compiler. Its throughtput is 1point/cycle.

The error introduced by the 21bit fixed point approximation was analyzed resulting an absolute error, in the worst case, equal to 2^{-11}. This is lower than the maximum error allowed in the tool trajectory, which is 0,1mm.

For test purposes the RC1000 board was connected to a typical CNC platform: standard PC equipped with a Intel Pentium MMX @166MHz, 16MB RAM. The system was tested with a large data base of shoe last. The absolute error obtained for the tool trajectory points was always under 0,1mm as the previous analysis had predicted. Table 1 shows the average processing time for the trajectories calculated with a step of 100points/mm. As can be seen, working at very low frequencies the speed up obtained can reach the value of x4.

System	Time (s)	Speed-up
GPU	236,2	--
GPU+RCU (@16MHz)	92,3	2,6
GPU+RCU (@33Mhz)	57,6	4,1

Table 1. Speed-up of the system

The FPGA resources consumed by the core were 11% of the Slices and the maximum frequency was 53,073MHz.

5. CONCLUSIONS AND FUTURE WORKS

Because of the adoption of the new ISO standard 14649, nowadays is a tendency for numerical controls to incorporate advanced characteristics in trajectory generation.

The virtual digitising algorithm is simple to implement, offers good results and avoids the problem of tool collision by its own definition. However, the algorithm is not suitable for general-purpose machining, since it is too slow compared with other types of tool path generation algorithms. This is especially true for traditional manufacturing environments, such as in the manufacture of shoe lasts where is generalized the use of low performance computers.

A reconfigurable hardware approach has been proposed to solve this problem arising from the industrial CAD/CAM field. A prototype of the architecture has been implemented taken as baseline system a CNC for shoe last machining. The results show a significant increase of the compting speed, maintaining a relatively low frecuency at the same time. This fact confirm the feasibility of the proposal, the computational cost could be replaced by the insertion of specialized boards instead of changing the whole control system.

Further refinements can be adopted to improve the response time of the system, for instance the reduction in the number of RCU-GPU transactions. This can be made duplicating the pipeline to computing several loop iterations before the RCU send the transformed points to the GPU.

In addition, future studies aim to complete the hardware architecture so that it can support the distance calculation task in the virtual digitizing algorithm.

6. REFERENCES

[1] Farin, G. “Curves and Surfaces for Computer Aided Geometric Design. A Practical Guide”. Academic Press Inc., 1993.

[2] Wang, Y. “Intersection of offsets of parametric surfaces”. Computer Aided Geometric Design, vol. 13, pp.453-465, 1996.

[3] Choi, B. K. “Surface Modeling for CAD/CAM”. Elsevier Science Publishers, pp. 263-272, 1991.

[4] Held, M. “Voronoi Diagrams and Offset Curves of Curvilinear Polygons”. Computer-Aided Design, vol. 30, no. 4, pp. 287-300, 1998.

[5] Jimeno A, García J., Salas F. “Shoe lasts machining using Virtual Digitising”. International Journal of Advanced Manufacturing Technology, vol 17, no.10, pp. 744-750, 2001.

[6] Chen, J., Gong, Y., Jin, T., Tong, S, “Development of an integrated CAD/CAM system for shoe last”, 2005 IEEE International Conference on Mechatronics and Automation, ICMA 2005, pp. 1107-1111

[7] Denkena, B., Scherger, S., “A concept for shoe last manufacturing in mass customisation”. 2005 CIRP Annals - Manufacturing Technology 54 (1), pp. 341-344.

[8] Jimeno, A., Cuenca, S. “Reconfigurable Computing for Tool-Path Computation” International Journal of Advanced Manufacturing Technology, vol. 21, no 12, pp. 945-951, 2003.

[9] Jung W Park et al **“**Pencil Curve Tracing via Virtual Digitizing” Proc. of IFIP CAPE Conference, (1991), pp.97-104.