Advanced NC-programming of parallel machine tool structures using workspace models and NC-simulation systems

Bert Lauwers[1], Qiongyan Li[1], Francois Pierrot[2], André Crosnier[2], Olivier Company[2]

[1]*KULeuven, Division PMA, Celestijnenlaan 300B, B-3001 Heverlee, bert.lauwers@mech.kuleuven.ac.be*
[2]LIRMM, Montpellier, France

KEYWORDS

Parallel machine tools, CAD/CAM, NC-simulation, NC-postprocessors.

ABSTRACT

This paper describes new concepts and prototype software developments for the NC-programming of parallel machine tools. Important characteristics of the system are the integration of a machine workspace model with the CAM system and the integration of a NC-simulation system with the postprocessor. During tool path generation, each tool posture is checked for feasibility by a fast interrogation of the workspace model. The workspace model stores all possible tool postures (in terms of x,y,z,i,j,k) the parallel machine can reach. During postprocessing, each machine configuration is checked for collision and in case collisions occurs, the system applies an appropriate collision avoidance algorithm.

1 INTRODUCTION

The use of parallel structures is not new and the concept has been accepted as very beneficial for applications such as fast pick & move operations (Clavel, 1998). The application of these parallel structures for machine tools only started some years ago. Prototype machine tools have been built and many of them have been firstly shown during the EMO Hannover Fair of 1997.

Today, the introduction of these new machine tools is still limited and the impact is not that big as one had expected. The characteristics such as stiffness behavior still have to be improved in order to be better than classical machine tools. Further, the efficient NC-programming of these machines, taking into account the full workspace, is a complex task. The generation of an error free tool path (no collisions, no discontinuities,..) is even more complex then in case of the programming of classical multi-axis machine tools.

(Chrisp and Gindy, 1998; Satake et al. 1998; Falco; Sarma, 1998) describe research work related to the NC-programming of parallel machine tool structures. Most of them deal with simulation, but very often, the work is oriented to specific machine types. This paper describes the development of a general NC-programming system for parallel machine tools. The concept is applicable for both parallel machine tool structures and classical machine tools.

2 DEVELOPED CONCEPT

The NC-programming concept, as shown in Figure 1, has been developed in the frame of the European funded Brite-Euram research project ROBOTOOL (BE97-4177). The CAM system generates a multi-axis tool path and is output in a neutral format, called a CLDATA-file (CLDATA = Cutter Location DATA).

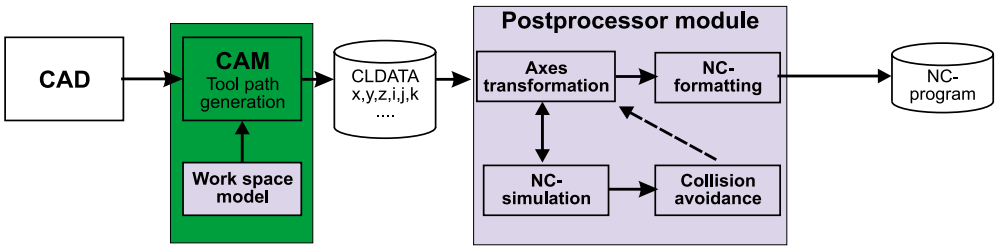

Figure 1: Developed concept

The postprocessor transforms the CLDATA-file to a machine specific NC-program. The developed postprocessor integrates machine simulation and the concept is based on a prototype developed earlier for classical multi-axis milling machines (Kruth and Lauwers, 1996). During axes transformation, each machine movement is checked for collision and in case a collision occurs, the system applies a collision avoidance rule such as searching for another machine axes configuration or a simple retract of the cutter. The search for another machine axes configuration is related to the degree of machine redundancy, meaning that several machine axes configurations are possible giving the same tool position and tool orientation.

Avoiding collisions by searching for other machine configurations (using the redundancy) can not solve all possible collisions. Part of the generated CAM tool path can be out of the workspace of the machine tool. To avoid this, a workspace model is linked to the CAM system. The workspace model stores all possible tool postures (defined in work piece co-ordinates, x,y,z,i,j,k) the parallel machine can reach. During tool path generation, each tool postures is checked for feasibility by a fast interrogation of the workspace model. For each machine, the appropriate workspace model can be loaded. The CAM system used in this research work is the PowerMill system (DELCAM).

The proposed concept still tries to be simple by splitting the CAM system (tool path generation) and the postprocessor module. This is often a request from many CAM developers. However, a full integration of a CAM system, a postprocessor and a simulation module allows to feed back occurring collisions to the CAM system as well. Other collision avoidance rules are then possible, such as changing the tool angle or the modification of a cutting strategy. This latter concept has been worked out in the frame of the European funded Brite-Euram research project OPTIMACH (BE96-3032) and is certainly beneficial in case the degree of redundancy is low (Lauwers, 1998; Saar et al, 1998).

3 INTEGRATION OF WORKSPACE MODELS WITHIN THE CAM THE SYSTEM

The workspace model of a classical three axis (X-Y-Z) machine is given by simple boundaries on each independent axis. In case of parallel machine tool structures, many CAM users take the workspace as composed by simple cubes or cylinders. However, such simplifications lead to a dramatic loss in the machine capabilities.

The workspace model should be as close as possible to the “real” workspace in terms of shape and size. It should be designed in such a way that the CAM system can perform a fast interrogation for feasibility of a generated tool posture.

For parallel machines, feasibility does not only mean accessibility. Of course, the user needs to know if the tool can access to a given point: this is a an issue related to the machine inverse kinematics, that is to say related to geometry of independent kinematic chains. But ultimately, the user needs to know if the machine is capable of performing correctly at this point. This is related to the behavior of the machine that can only be handled in a global sense, that is to say not “chain by chain”, and not only in a geometrical sense. Performance criterions must be defined such as: maximum possible velocity, minimum possible accuracy, minimum possible manipulability… All those criterions have to be computed for the machine (not chain by chain) and are very CPU consuming. Therefore, it has been decided to compute the workspace model in advance and to store it in some sort of database. This database should be kept as small as possible and must allow a very fast access by the CAM system.

3.1 Workspace model calculation

The calculation of the workspace model is done in different steps (Figure 3):

- Building a rough estimate of the accessibility workspace based on the inverse kinematics:
  An algorithm, called “flooding technique”, has been applied: a sphere defined by its Boundary Representation (B-Rep) is “expanded” from a point in the workspace and “deformed” until it reaches the boundaries. This gives on one hand a B-Rep of the workspace, and on the other hand, the limits on each axis.

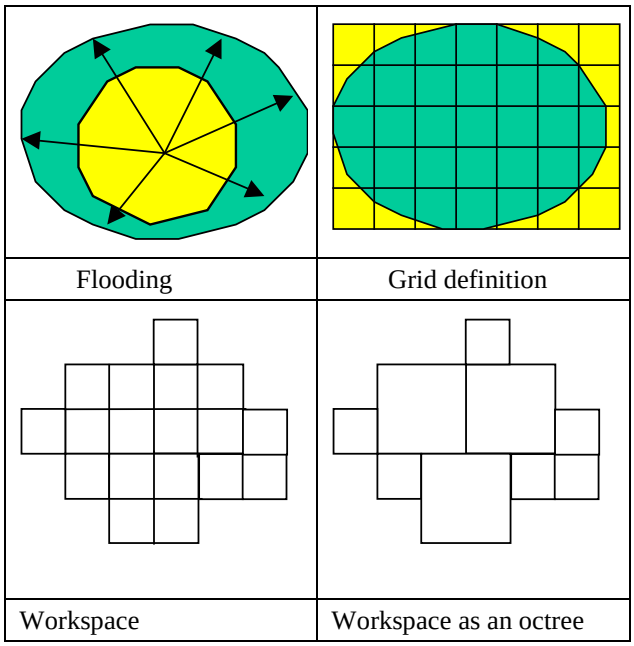

Figure 2: Building the workspace model

- According a user defined resolution, a grid is constructed whose resolutions are compatible with a 3-dimensional octree; every grid cell is checked for geometrical accessibility and for user defined performance criterions.
- The satisfying cells are grouped in an octree in order to dramatically reduce the amount of data;

Figure 3 shows workspace models for the Ingersoll machine without (a) and with (b) performance criterion. The performance criterion applied …..???

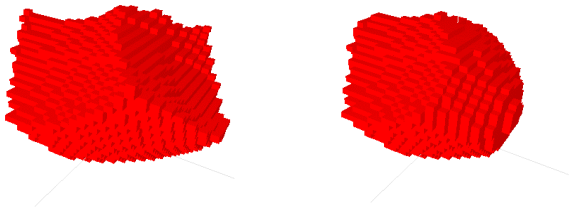

Figure 3: Example of a workspace evaluation for the Ingersoll machine (left)without, (right) with a performance criterion

3.2 "Workspace model – CAM system" interface

A robust and simple "workspace model – CAM system" interface has been developed. The interface supports two main functions that are used to interrogate the workspace model (Figure 4). These functions deal with the following questions asked by the CAM system:

- Is a given tool position (x,y,z,i,j,k) within the workspace
- Are two tool positions connected to each other. With the term "connected" is meant whether a straight line interpolation is possible between two given points.

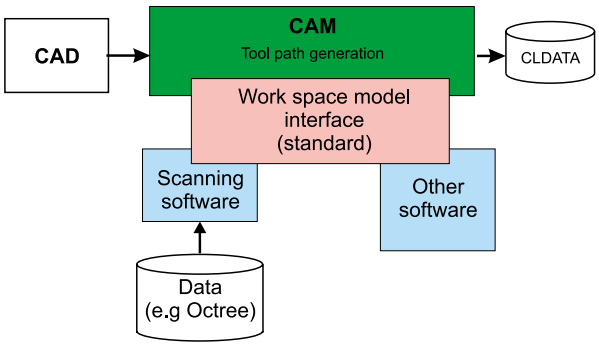

Figure 4: Integration of the workspace model within the CAM system

The workspace model, as described in previous paragraph, is stored in a file. The interrogation of the workspace model file is done by a scanning module. The connection of the workspace model via this simple interface even allows to connect workspace models of a different architecture. Any kind of software, that can be plugged into the interface, can be used as a kind of workspace model. The software only has to be able to give answers to the two main functions of the interface. In this sense, a machine simulation system, including an axes transformation module, could be used as a kind of workspace model. This latter is similar to the full integration that is worked out in the OPTIMACH project (BE96-3032).

If a tool path position (x,y,z,i,j,k) or a number of position are not reachable, the CAM system will take the necessary actions. At the moment, only a simple retract is applicable and the non-cut material is identified as rest material. Other strategies, such as a modification of the tool angle, are possible but will be subject of further research.

4 POSTPROCESSING WITH INTEGRATED NC-SIMULATION & COLLISION CONTROL

The postprocessor module reads the CLDATA-file and generates a machine specific NC-program. The postprocessor consists of four modules:

- Axes transformation module (kinematics engine): The kinematics engine performs the axes transformation form CLDATA co-ordinates (x,y,z,i,j,k) to specific machine co-ordinates.
- A machine simulation module: For each generated machine axes position, the simulation system checks the machine movement against collision.
- A collision avoidance module: The collision avoidance module proposes a collision avoidance rule in case a collision occurs.
- An NC-formatting module: This module formats the NC-program according to the syntax rules of the given machine.

4.1 Kinematics engine

The kinematics engine performs the axes transformation form CLDATA co-ordinates (x,y,z,i,j,k) to specific machine co-ordinates. For a hexapod type machine tool, the machine axes are the axes values L1, L2, L3, L4, L5 and L6 (Figure 5). These machines have normally redundant degrees of freedom. This means that there are different machine configurations possible yielding the same tool position and orientation. A rotation of the platform around the spindle axes yields many different machine configurations not affecting the cutting process.

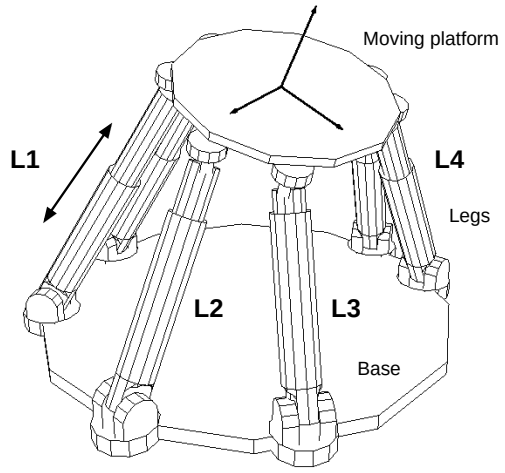

Figure 5: Schematic view of a hexapod type machine structure

Figure 6 shows another parallel machine structure (GeorgesV, IFW, Hannover). Three actuator systems (axes U1, U2, U3) are constructed in parallel while 2 serial axes A en B are mounted on the moving platform. This machine only has two possible machine configurations for a given tool position and orientation. The alternative position can be obtained as A' = A + 180 and B' = -B.

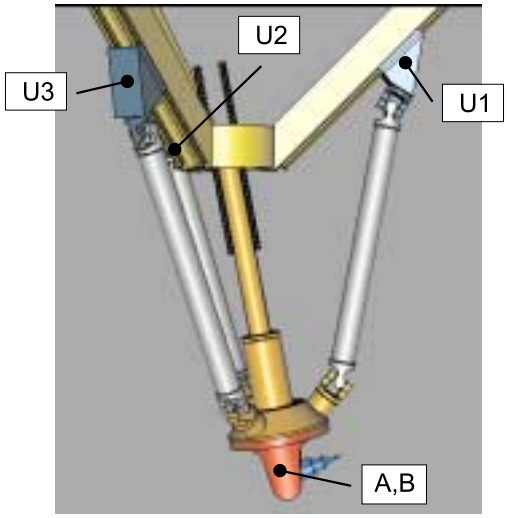

Figure 6: Axes configuration of the Georges V parallel machine tool

Initially, the kinematics engine chooses one machine configuration. Other configurations will be selected in case a collision is detected (see further).

4.2 Machine simulation & collision avoidance

For each generated machine axes position, the simulation system checks the machine movement against collision. Therefore, a commercial NC-simulation system has been integrated. Before the simulation can start, the geometrical models of the machine, the part and the used tools are loaded into the system. The postprocessor calls the simulation system in two modes:

- Discrete mode:
  A collision check is performed at each outputted machine configuration (e.g. a given L1,L2,L3,L4,L5,L6)
- Continous mode:
  The simulation system checks the full path from a given machine configuration to the next configuration.

After each check, the simulation system returns a collision messages, including the type of collision that occurred. Table 1 gives an overview of all the collision types that are of interest for collision avoidance purposes:

Collision between	***Tool***	***Check***	***Part***	***Machine head***	***Machin tool***	***Tool holder***
Tool		■	■	■	■	
Check				■	■	■
Part				■	■	■
Machine head					■	
Machine tool						■
Tool holder						

Table 1: Different type of collision checks

The NC-simulation system is normally driven by the different actuator values of the machine (L1,L2,L3,L4,L5,L6 for a hexapod type (Figure 5) and U1,U2,U3,A,B for the GeorgesV (Figure 6)). With this kind of co-ordinates, the simulation system has to apply inverse kinematics in order to control the machine model. Inverse kinematics for parallel machine tool is not easy and requires an iteration procedure. Therefore, it was decided to send another unique set of machine co-ordinates to the machine simulation system. For the GeorgesV, this is reference point of the moving platform (Xc,Yc,Zc) and the two serial axes A and B (Figure 7). The simulation system can directly position the machine model based on the co-ordinates Xc,Yc,Zc. The positioning of the additional serial axes is not a problem.

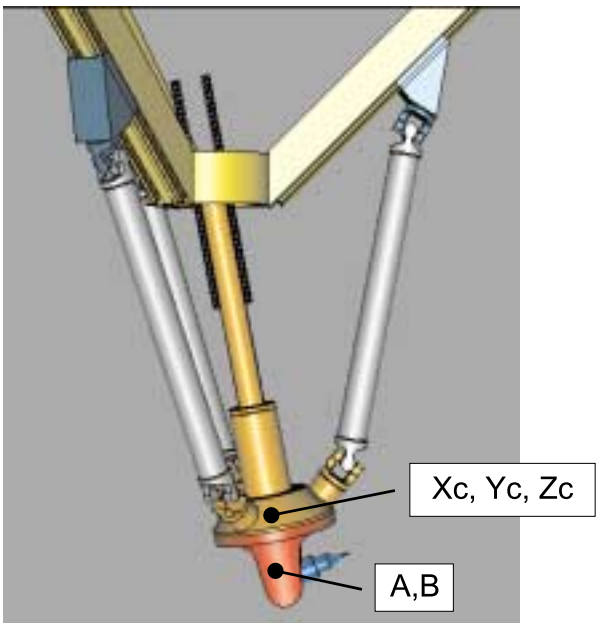

Figure 7: Charactherisation of a GeorgesV machine configuration by the co-ordinates Xc,Yc,Zc,A,B.

For a hexpod type machine (platform with 6 legs), the co-ordinates (Xc,Yc,Zc,α,β,γ) are used to drive the simulation system (Figure 8). Xc,Yc,Zc is reference point on the platform and the angles ,α,β and γ define the rotation of the platform.

This alternative set of machine co-ordinates are also generated by the kinematics engine. Figure 9 shows the different flows of co-ordinate values within the postprocessor module.

Based on the collision checked by the NC-simulation system, the collision avoidance algorithm will be applied. Currently, two collision avoidance rules are implemented: a simple retract of the cutter and a change of machine axes configuration. Other machine configurations are calculated by the kinematics engine.

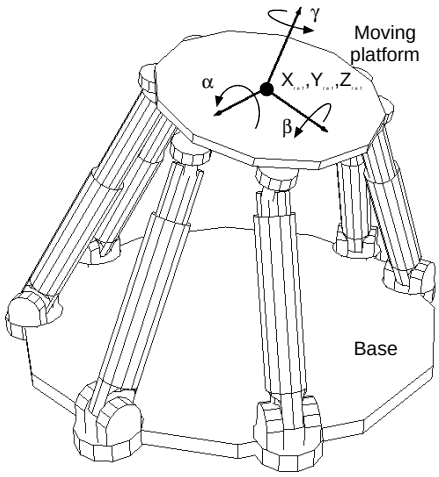

Figure 8: Positioning of a hexapod-type machine by the co-ordinates (Xc,Yc,Zc,α,β,γ).

5 IMPLEMENTATION ISSUES

The CAM system used in the frame of this research work is PowerMill (DELCAM). The "workspace model – CAM system" interface has been developed as a COM object which allow to link different work space models.

The postprocessor module has been developed on a NT-Windows platform. The simulation system has been provided by the company dCADE and is linked to the postprocessor module. During postprocessing, the simulation system runs in the background.

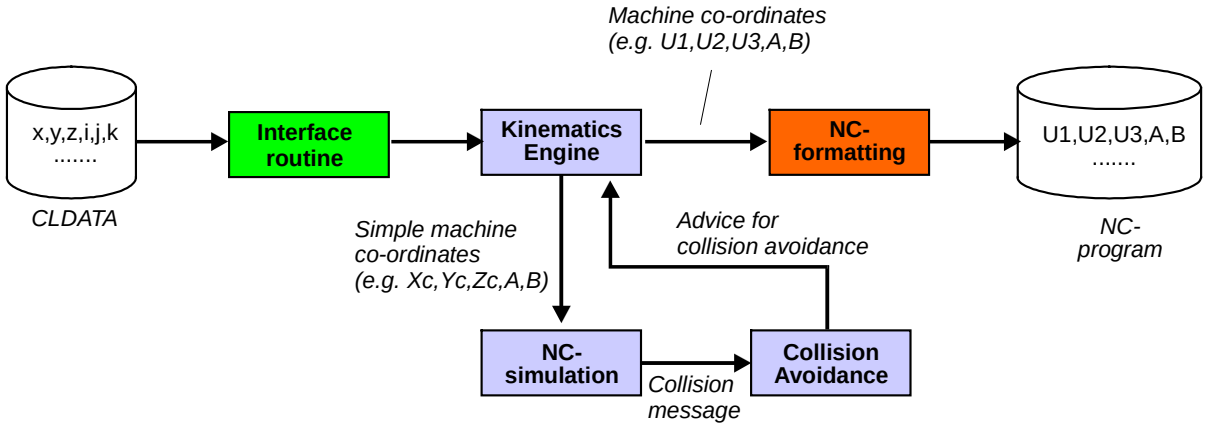

Figure 9: Flow of information in the postprocessor module

The addition of a new machine requires the following actions:

- Calculation of the workspace model to be connected to the CAM system. The file format is standard, so the scanning module (Figure 4) remains unchanged.
- The development of a geometrical machine model and virtual controller to be used by the NC-simulation system.
- Development of the axes transformation modules
- Selection of appropriate collision avoidance algorithms

Currently, workspace models have been generated for the GeorgesV (IFW-Hannover) and the Ingersoll (WZL, Aachen). The GeorgesV has also been completely implemented in the postprocessor module. The axes transformation modules have been developed for the Ingersoll (WZL, Aachen) and the NEOS tricept 805 (IVF, Sweden). Figure 10 shows a screenshot of the postprocessor module.

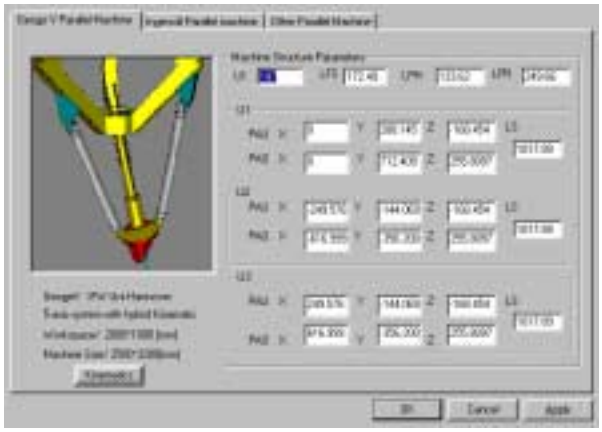

Figure 10: Screen shot of the postprocessor module (interface for the selection of the machine tool)

6 CONCLUSIONS

This paper presents a new NC-programming system that has been developed for the programming of parallel machine tool structures. A first innovative aspect is the integration of a workspace model into the CAM system. A fast interrogation of the workspace model by the CAM system is made possible by representing the workspace as an octree. A second innovative aspect is the integration of a NC-simulation system into the postprocessor module. Any kind of collision (machine-part, part-tool,..) is checked during postprocessing and a collision avoidance rule such as a search for another machine configuration can be applied in case a collision occurs.

Future research and development is still needed on collision avoidance methods for the CAM system as well as the postprocessor module.

ACKNOWLEDGEMENTS

This research has been funded through the European funded Brite-Euram research project ROBOTOOL (BE97-4177).

REFERENCES

Chrisp A. G. and N. Gindy. 1998 Parallel link machine tool: simulation, workspace analysis and component positioning, *First European-American Forum on PKM* (Milan, Italy, August 31, September 01).

Clavel R. 1998, DELTA, A fast robot with parallel geometry, *Proc. of ISIR*, 91-100.

Falco J. *Simulation tools for collaborative exploration of hexapod machine capabilities and applications.* Intelligent Systems Division, Manufacturing Engineering Laboratory, NIST, Gaithersburg, Maryland.

Kruth J.P., B. Lauwers amd P. Klewais. 1996. NC-postprocessing and NC-simulation for five-axis milling with automatic collision avoidance, *Proc. of International Manafacturing Engineering Conference, Manufacturing Engineering: 2000 and Beyond*, ISBN: 965-294-121-2. (Connecticut, USA). 56-58.

Lauwers B. 1998. Optimised tool path generation methods for economic and collision free multi-axis machining. *Unigraphics User Conference* (Frankfurt).

Saar A., B. Lauwers and P. Dejonghe., Optimised tool path generation methods for economic and collision free multi-axis machining. *Proc. of 31th ISATA Conference, Programme track on Automotive Mechatronics Design & Engineering*, (Dusseldorf). 477-486.

Sarma S. 1998. On the use and augmentation of hexapod machine tools. *First European-American Forum on PKM*. (Milan, Italy, August 31, September 01).

Satake T., A. Hayashi and H. Suzuki 1998. Development of CAM system for Machine Tool having Parallel Mechanism. *Mechantronics.* (Kitakyushu).