The Development of a Mechatronic Control System for Modular Reconfigurable Machine Tools

[1]J. Padayachee, [2]G. Bright, [3]L.J. Butler
Mechatronics and Robotics Research Group
Department of Mechanical Engineering
University of KwaZulu-Natal
Durban, South Africa
[1]203505399@ukzn.ac.za, [2]brightg@ukzn.ac.za , [3]207506509@ukzn.ac.za

Abstract

Modular Reconfigurable Machine tools have been developed to display a variation of degrees of freedom and processing functions on a single platform. The reconfigurability of the hardware is derived from the ability to alter the modular constituents of the mechanical architecture. The scalability of reconfigurable machinery required the development of a control system that is capable of managing a platform with a reconfigurable kinematic architecture and a varying number of sensors and actuators after each reconfiguration cycle. This paper presents the development of a suitable modular mechatronic control system. The focus of control development was the facilitation of seamless system integration between modular hardware and the controller at both hardware and software levels. A reference model for the control implementation is presented, including a script method for the reconfiguration of a modular machines physical representation in software. An appropriate force control method is identified and interpolation and servo control algorithms are presented and evaluated.

Keywords: Modular Machine Tools, Modular Mechatronic Control, Reconfigurable Numeric Control

1. Introduction

Global economic competition and rapid product development have forced a paradigm shift in the design of automated production systems. The current goal in automated machine design is the achievement of machine sensitivity in response to production changes in a dynamic manufacturing environment. [Mehrabi, 2000] proposed the development of Reconfigurable Machine Tools (RMTs), as novel class of machinery that will provide the desired solution. By initial design these machines must be rapidly reconfigurable in their mechanical hardware, electronic and software architectures in response to product or market changes [Koren, 2004].

This paper presents the development of a Mechatronic control system for Modular Reconfigurable Machine (MRM) tools. MRMs are a new class of production machinery, created by researchers at the University of KwaZulu- Natal (UKZN) to display a reconfigurable system architecture in response to dynamic production requirements [Padayachee, 2008a, 2008b]. The reconfigurability of the machine is achieved by the addition and deletion of modular hardware units on a platform. The modular structural configuration implied a variation in the number actuators, sensors and distributed control units present on the MRM platform after each reconfiguration cycle.

The scalability of the mechanical and electronic hardware required the design of a Mechatronic control system, distinctly different to the conventional control systems of Numerically Controlled (NC) machines. A Mechatronic design approach was adopted to ensure seamless system integration after each hardware reconfiguration cycle. The design of the control system at a hardware level focuses on the integration of mechanical modules and distributed control drive units with a PC based host controller.

A software reference model is presented for the management of a scalable configuration of distributed slave control units and mechanical modules. The reference model illustrates the essential functions and the interaction between function as necessary for the development of an optimized numerical control kernel. Functions within the reference model include routines for the software reconfiguration of a machines kinematic chain and collision detection parameters. A method for force control, optimized interpolation and servo control are also presented.

The control system developed, has been designed for a metal processing MRM based on numerical control; however the proceeding literature is applicable to the control of NC based robots, chip mounters and assembly machines with modular end effectors or other modular components.

2. Related Literature

2.1 Machine Design

Research in NC machine design, has in recent years, focused on technologies that increase a machines' structural and control reconfigurability. Researchers at the University of Michigan proposed the development of machinery from a precompiled library of machine modules [Tilbury, 1999; Landers, 2001]. The notion of designing machinery from a library of modules is critical in shortening the machine development process. Shinno and Ito [Shinno et al, 1981, 1984] have developed a methodology whereby machine tools may be structurally generated from simple geometric objects, thus facilitating the creation of a feasible library of modules. Moon and Kota [Moon et al, 2002a, 2002b] have developed a mathematical technique based on Screw Theory and Graph Theory which enables the necessary precompiled parameterized library modules to be identified for the structural reconfiguration of a modular machine. The literature survey conducted has indicated no previous attempt at exploiting these methods for the reconfiguration of machinery by a structurally modular design.

A notable development is design and control of the arch type RMT, discussed by [Katz et al, 2004]. The reconfiguration method employed in the arch type RMT involved the reorientation of machine axes in response to product changes. The method of reconfiguration employed in MRMs is distinctly different to those previously developed, necessitating the creation of a unique numerical control system.

2.2 Open Architecture Control

The ability of a machines control system to interface with factory wide networks and facilitate the integration of third party control algorithms has been identified as a necessity for dynamic manufacturing environments. Standards like Open Systems Architecture for Controls within Automation systems (OSACA) and Open Modular Architectures Controllers (OMAC) have been developed to eliminate the problems inherent in proprietary control architectures [Asato et al, 2004]. Such problems entail the alteration and adaptation of the controller rapidly and cost effectively. The controllers developed in reference to these standards are commonly termed Open Architecture Control (OAC) systems. OAC aims at the easy implementation and integration of customer specific controls by means of open interfaces and configuration methods in a vendor-neutral, standardized environment [Pritschow et al, 2001]. The advent of a global standard has however, yet to be realized.

3. MRM Mechanical Architecture

3.1 Mechanical Reconfiguration

MRMs are mechanically modular machines, being designed with the intention of manually reconfiguring the modular constituency of a machine to bring about a transformation in its architecture. The architectural change may either be kinematic, functional, or a combination of both. MRMs are assembled from a library of precompiled modules that are concatenated by means of a series of standardized mechanical interfaces. The standardization of mechanical interfaces permitted a variety of combinations in which modules could be joined. Figure 1 illustrates an MRM library; within the library are linear axes, rotary axes, cutting heads and auxiliary work support units that may be used to add degrees of freedom or processing functions to the MRM platform.
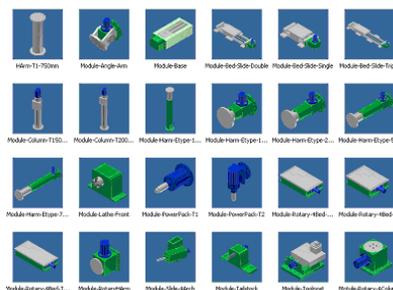

Figure 1: A library of mechanical modules for MRMs [Padayachee, 2008b]

The ability of an MRM to create different features (slots, grooves, thread, etc) on a part is dependant on its ability to provide the correct processing function. Conventional CNC machines are capable of a wide variety of processing functions; though the machine may yet not possess the specific function required. In this instance, the hardware of the CNC is not adaptable to the requirement. MRMs possess modular cutting heads that may be interchanged when processing requirements fall out of the machines current capability. The cutting heads are specialised for a dedicated set of functions, providing a more optimised solution than the use of a single generic spindle. Advantages include specialisation for vibration damping and high speed machining. Figure 2 illustrates the reconfiguration of machining functionality, by the change of a cutting head, as the end effector of the platform.
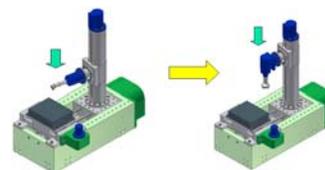

Figure 2: A reconfiguration of machining functionality [Padayachee 2008b]

The ability to provide a variation in tool orientation is facilitated by a kinematic reconfiguration of the MRM structure. Figure 3 illustrates the reconfiguration of a three axis mill to a four axis mill by the addition of a rotary DOF.
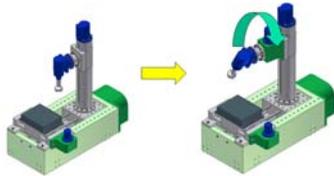

Figure 3: A reconfiguration of machine functionality [Padayachee, 2008b]

3.2 Sensor Systems and Integration

Mechanical modules formed self contained units, with motors and sensors contained within a defined space. The confinement of sensors and motors to an enclosed space required the creation of a standardized power and control interface, for a consistent style of integration with spindle and servo drive units. Servo or spindle control units map to axes and cutting heads on a 1:1 basis where each unit is dedicated to the hardware it controls.

Modules possessed a combination of optical encoders, load cells, accelerometers and temperature sensors depending on the topology of the module. Modules did not possess any embedded control hardware and all sensor data was externally analyzed.

4. Software Reference & Implementation

The primary stage in the development of a Mechatronic control system for MRMs involved the creation of a reference architecture for software development. The reference encapsulated features of software design and subroutines that enable the control of a scalable mechanical architecture. The reference defines the necessary software control functions and the interaction between these functions. Figure 4 illustrates a complete pictorial of the reference used in the development of the controller.

4.1 Open Control Systems

At the head of the control system was a desktop PC supporting the LINUX Fedora 7 operating system. A PC based control system was implemented, based on already existing support for Ethernet, USB and RS232 communication. The LINUX operating system was implemented due to the openness of the software thus enhancing the manipulability of the operating system. It should be noted that no specific Open Architecture standard has been strictly adhered to. The support of control openness has been promoted by implementation of standardized hardware interfaces for the system; used in conjunction with conventional communication protocols.

Software development in C++ further enabled the creation of well defined classes for integration into an Application Programming Interface (API). The API provided access to the various control routines in the multi tier system of Figure 4. The creation of an API enabled a standardized method for the integration of third party control algorithms into the system, thus opening the controller to user manipulation.
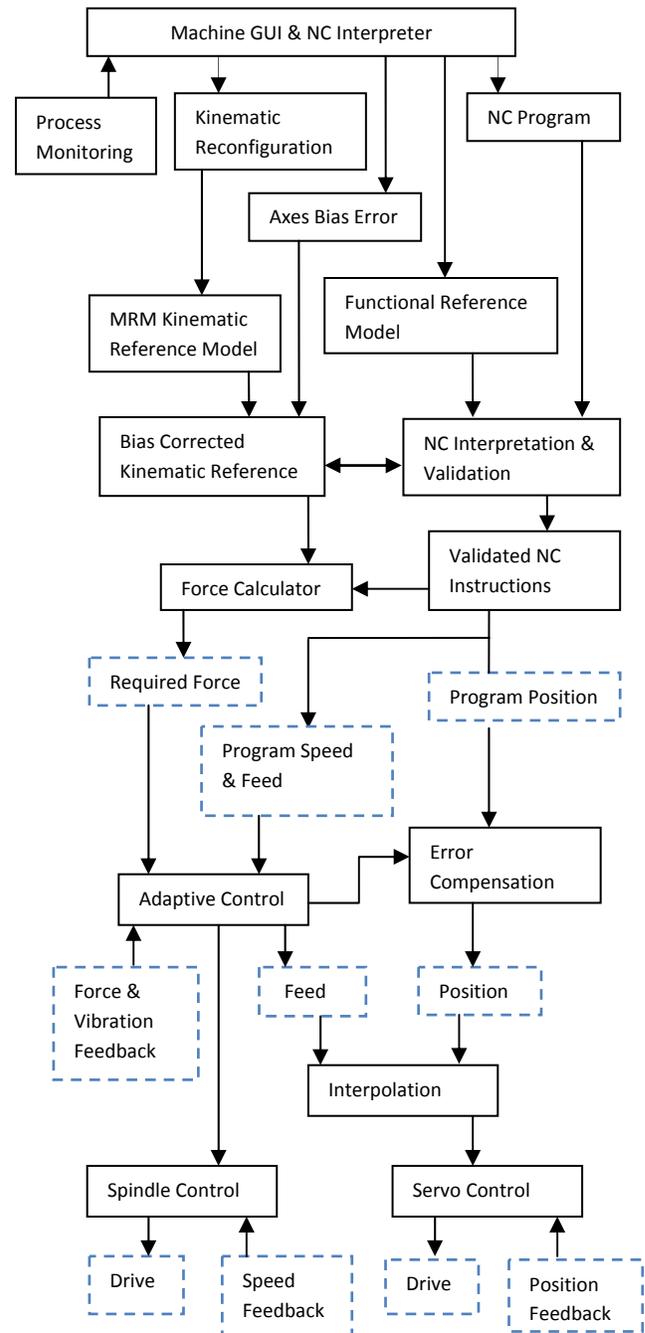

Figure 4: Software Reference architecture used in the Development of a Mechatronic Control System for MRMs'

4.2 Host Control System

All control functions were performed by the host PC, excluding direct sensor monitoring and servo and spindle control. At the highest tier of the PC based controller are the software routines concerned with the reconfiguration of the machines controller in accordance to a functional or kinematic reconfiguration of the mechanical hardware. Section Five discusses a script based reconfiguration method for the transformation of a machines physical representation in software. The host control system was responsible for the adaptive control and error compensation of programmed instruction based on sensor data received from subordinate control units. High level, error compensated instructions were used by the interpolator for the generation of axis/servo control commands. The host controller transmitted control commands to subordinate control units via USB and IIC.

4.3 Subordinate Control Units

The spindle control drive received instructions and relayed data back to the host control system via USB. The USB protocol was adopted since at any instant there is only one cutting head and one spindle control drive present on a MRM platform. The selection of the USB standard was also based on the higher data feedback revived from a spindle control drive as opposed to an axis control drive. A spindle control drive consisted of two Atmega32 microcontrollers. The first chip managed the speed control of the spindle and the second chip was responsible for sensor management.

Servo control drives received instructions via an IIC network. The selection of a network based protocol for communication was based on the variable number of servo control drives on a platform after each reconfiguration. The IIC protocol implements a seven bit slave address, permitting up to 128   modules to be connected to the network. The IIC protocol was implemented on a test basis and communication networks such as PROFIBUS, SERCOS and CAN are recommended for industrial grade communication with microcontrollers and DSP's used for low-level processing and I/O operations. Servo control drives consisted of an ATmega 32 chip for position and speed control, and an ATmega 8 chip for sensor management.

5. MRM Kinematic Reference Model

The programming language of choice, for the creation of the MRM host software architecture was C++. The object oriented design approach and the property of inheritance permitted the creation of generic software modules that described linear and rotary axes and cutting heads. The parametric description of each axis or cutting head was encapsulated in a data object that is instantiated through a script entered in the machine GUI.

The kinematic reference model was created in software through the concatenation of transformation matrices that describe the kinematic chain of the tool. Each mechanical module in the system is described by a single matrix. This matrix describes the relationship between two Cartesian reference frames placed at the two mechanical interfaces of a module (see Figure 5).
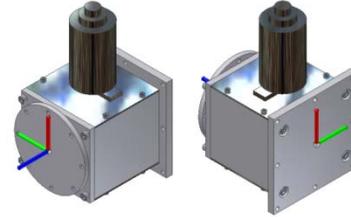

Figure 5: Placement of Reference Frames

The transformation matrix for any module is given by:

$$M_n = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma & x \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma & y \\ -s\beta & c\beta s\gamma & c\beta c\gamma & z \\ 0 & 0 & 0 & 1 \end{bmatrix} (1)$$

The rotation component of the transformation matrix was represented by X-Y-Z Euler angles. The X-Y-Z Euler angle convention enabled a generic script to be developed for the entry of a kinematic configuration into software. The details of the script are as follows:

Instantiating a kinematic link
RF01 LN (n) **XA** (α) **YA** (β)   **ZA** (γ) **X** (x offset) **Y** (y offset) **Z** (z offset)

Enforcing Joint Limits
RF02 LN (n) **UP** (limit) **LW** (limit)

Compensation of Mechanical Assembly Errors
RF03 LN (n) **XA** (α err) **YA** (β err)   **ZA** (γ err) **X** (x err) **Y**(y err) **Z** (z err)

The script is called Numerical Control Configuration (NCC) code, and has been formatted as a potential extension of conventional NC or G code (ISO 6983). The extension of existing machine tool programming languages was pursued as a goal in facilitating the integration of MRM technology with that of currently existing CNC based processing equipment. The "RF" word indicates to the machine tool that the proceeding script is a reconfiguration script. Three RF words were created to as flags to an underlying text interpretation routine, indicating three types of reconfiguration operations.

The 'RF01'command initializes the building of the MRM kinematic chain by instantiating a data object that is dedicated to the software parameterization of a specific link in the chain. The 'LN' word indicates the numeric name of

the link for future reference with other RF words. The link numeric name is of further importance as it is used to assemble the links in order of appearance in the kinematic chain. Link LN1, the fist link in the chain, is always descriptive of cutting head of the machine. The kinematic chain continues in a direction such that link LN(n), the last link in the chain, is always   descriptive of the machine work table or work clamp. Thus the position of the machine spindle is determined in relation to the work piece.

The 'RF02' command was created to set the upper and lower limits of an axis range. A separate command for the initialization of axes ranges enabled the reconfiguration of the limits after the kinematic chain has been constructed and stored in software.

The 'RF03'command enables post reconfiguration calibration of the machine to be performed simply, with no alteration to hardware required. Due to the modular nature of MRMs, the assembly procedure may introduce alignment errors between interfacing modules. These errors, called axes bias errors, are compensated for using the 'RF03'command. This command alters the software kinematic configuration to exactly reflect the current hardware status and also alters NC programs to produce the required tool trajectory, regardless of misalignment errors existing in the mechanical architecture.

Code Example

Consider the following sample code:

```
N001 RF01 LN1 XA0 YA90 ZA0 X-80 Y0 Z75
N002 RF01 LN2 XA0 YA0 ZA0 X0 Y0 Z-370
N003 RF01 LN3 XA0 YA0 ZA# X0 Y0 Z-140
N004 RF02 LN3 UP180 LW-180
N005 BUILD
```

This code initializes the kinematic chain of equation two (2), where lines 001 to 003 initialize the transformation matrices according to the Euler convention of equation (1).

$$T_1^3 = \begin{bmatrix} c\alpha_1 & -s\alpha_1 & 0 & 0 \\ s\alpha_1 & c\alpha_1 & 0 & 0 \\ 0 & 0 & 1 & -140 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -370 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 & -80 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 75 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

The link transformation matrices are concatenated in order of numeric precedence (numeric names). If an element of a matrix is to be instantiated as a variable, the variable must be declared with the '#'flag. The Z Euler angle of link three was declared as a variable using the flag: **ZA#**. Line 004 of the code enforced the axis limit $-180 \leq \alpha \leq 180$ on the rotary joint described by this link.

Consider the following bias correction code, which may be incorporated into the kinematic chain of equation (2).

```
N006 RF03 LN1 XA0 YA0 ZA0 X0 Y0 Z-5
N007 RF03 LN3 XA0 YA0 ZA10 X0 Y0 Z0
N008 BUILD
N009 G90 T1 F100…
```

The bias correction is reflected in the altered kinematic chain of equation (3) where bias compensatory elements are indicated in red text:

$$\begin{bmatrix} c(\alpha_1 + 10) & -s(\alpha_1 + 10) & 0 & 0 \\ s(\alpha_1 + 10) & c(\alpha_1 + 10) & 0 & 0 \\ 0 & 0 & 1 & -140 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -370 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 & -80 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 70 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

Compensatory values may be determined by laser measurement methods currently used in commercially available CNC calibration services or a quality control routine during production ramp-up. The **BUILD** command is always issued prior to proceeding with conventional NC programming as in line N009. The example code illustrates how a text approach enables a quick reconfiguration of machine software. An added advantage of this approach is reconfiguration of software via a factory wide control network, through the transmission of a text file.

6. Collision Avoidance and NC Validation

The primary objective of validating an NC program is the filtration of invalid program coordinated that are outside the domain of the operational workspace. The operational workspace of the MRM is determined by the axes limits instituted in the NCC code routine. These limits must be carefully selected by a programmer so as to avoid hardware collisions within the machine tool. The limits are virtual and do not necessarily represent the true range of an axis.

The definition of an operating workspace by virtual axes limits results in a reduced workspace in comparison to the reachable workspace achievable by the hardware.   The structural transformability of a modular machine tool makes obsolete the software filtration of an NC program through a fixed programmed collision avoidance model. The need for an optimized NCC programmable collision avoidance model exists. However, the creation of the RF02 command, as discussed in section five, allowed for the swift reconfiguration of the operational workspace in accordance to changing kinematic requirements. This command was used without reconstructing the kinematic chain of the machine. The work space, under this system, is customized to the current operation resulting in a simple, practical solution to collision avoidance. The institution of a customized work space as per operation is an acceptable solution as; MRMs are not intended for batch-of-one production.

7. Force Control

7.1 Force Regulation
Process monitoring is orientated towards increasing quality and production output, while reducing scrap and tool breakage. Aspects of process monitoring include chatter vibration, force and tool wear monitoring. Of primary concern are static cutting forces that lead to mechanical deflections on the machine tool. Modular machines are inherently light weight, being designed for easy reconfigurability. The light weigh machining structure may provide inadequate structural support under different configurations of the architecture. The effective limitation of cutting forces exerted on a MRMs mechanical hardware is essential feature of a Mechatronic control solution.

A simple model for the static cutting force exerted on a machine's tool is given by [Nwokah et al, 2002].

$$F = Kd^{\beta}V^{\gamma}f^{\alpha} \qquad (4)$$

Where F is the cutting force, K is the gain, d is the depth of cut, V is the cutting speed and α, β and γ are experimentally determined coefficients of the force relationship. Model parameters are unique to each material, tool and set of cutting conditions. Load cells incorporated into cutting modules provided data for tool force monitoring. Excessive torque and forces on the machine joints were reduced by a corresponding reduction in axis feed rates (i.e. f^{α}). In general the force-speed relationship of equation (4) is weak (i.e $\gamma \approx 0$) and spindle speed was not actively adjusted for force control. The research of [Otieno et al, 2008] concurs with the adjustment of axis feed rates, as an optimal control variable for cutting force regulation. Axis feeds were adjusted by the scaling of interpolation cycle times:

$$T_{IPO} = \frac{\sqrt{(x_{i+1}-x_i)^2-(y_{i+1}-y_i)^2}}{fK_{override}} \qquad (5)$$

Where $K_{override}$ is the adjusted variable for cycle time scaling.

7.2 Joint Actuation
The knowledge   of the force exerted on a tool tip was used to determine the force and torque propagation throughout the tool. The link-wise torque and force propagation is given by:
$$^{i}f_i = {}_{i+1}^{i}R^{i+1}f_{i+1} \qquad (6)$$

$$^{i}n_i = {}_{i+1}^{i}R^{i+1}n_{i+1} + {}^{i}P_{i+1} \times {}^{i}f_i \qquad (7)$$

Where f_i and n_i are the force and torque exerted on link i by link i+1. The position and rotation matrices were derived directly from the transformation matrices discussed in section five. A model on the link-wise torque and force propagation for a MRM was obtained from equations (6) and (7). For an expected cutting force the joint actuation torques and forces are obtainable. The actuation torque required by a rotary axis was calculated by:

$$\tau_i = {}^{i}n_i^T {}^{i}\hat{Z}_i \qquad (8)$$

The actuation force for a linear axis was calculated by:

$$\tau_i = {}^{i}f_i^T {}^{i}\hat{Z}_i \qquad (9)$$

$^{i}\hat{Z}_i$ is the joint axis unit vector which is derived from the NNC code discussed in section five.

8. Interpolation

8.1 Interpolation Model
A Sampled-Data interpolator was selected for implementation in MRMs. The Reference Word algorithm incorporating the Improved Tustin Method (ITM) was the favored combination. The ITM is generally preferred for software orientated numerical control designs, where floating point arithmetic is possible.   With the implementation of a PC based control solution, the magnitude of truncation errors in the interpolation algorithm are attributed to the limitation of a machine Basic Length Unit (BLU) per axis.

The ITM exhibits high accuracy and a relatively low number of iterations in comparison to Euler and Taylor methods [Shu et al, 2008]. The ITM/Reference Word algorithm involved the linear segmentation of both linear and non orthogonal trajectories. The implementation of the ITM for a linear tool path is simple and the accuracy of the trajectory depends primarily on the position control of the servo drive unit. The accuracy of the algorithm is better evaluated on its application of the achievement of a circular trajectory by multiple linear interpolations. The linear axial increments (Δx and Δy) per interpolation cycle time, for a trajectory along an arc of radius R, are given by equation (10).

$$\Delta x = x(i+1) - x(i) , \Delta y = y(i+1) - y(i) \quad (10)$$

The calculation of the appropriate linear increments is dependent on the current position of a machines tool tip after each interpolation cycle. This is given by:

$$x(i+1) = R(i)cos\big(\theta(i+1)\big) \qquad (11)$$

$$y(i+1) = R(i)\sin\big(\theta(i+1)\big) \qquad (12)$$

R(i) is the true radius of the interpolated arc from the previous interpolation cycle; this is given by equation (13).

$$R(i) = \sqrt{x(i)^2 + y(i)^2} \qquad (13)$$

The angular position of the tool tip is calculated by equation (14), where α is the angular increment in tool position, for each interpolation cycle.

$$\theta(i + i) = \theta(i) + \alpha \qquad (14)$$

$$\alpha = \sqrt{\frac{16}{R-1}} \cong \frac{4}{R} \qquad (15)$$

8.2 Interpolation Results

The chord height error is the difference in the true radius of an arc and the actual radial position of a tool at the middle of an interpolation step (0.5 α). Figure 6 illustrates the chord height error exhibited by the ITM, when interpolating an arc of 100 mm radius, through an angle of ninety degrees, on axial modules with a BLU of 0.01 mm. The interpolation was achieved with 39 interpolation steps. Euler and Taylor methods would have both required 56 interpolation steps to achieve the same trajectory with no significant improvement in performance. The maximum chord height error exhibited in the results of Figure 6 is 0.012 mm, with a mean error of 0.0051 mm
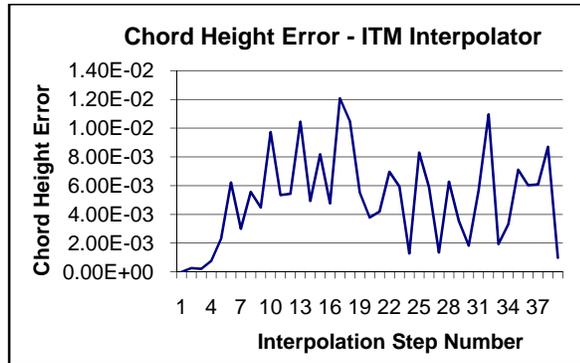

Figure 6: Chord Height Error

9. Servo Control

9.1 Servo Control Structure

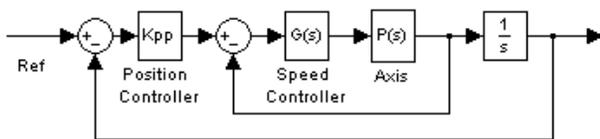The servo control of an NC machine may be achieved by the cascaded control system of Figure 7:

Figure 7: Cascade Structure for Closed Loop Servo Control

The efficiency of the cascaded structure for closed loop servo control depends primarily on the speed and stability of the inner control loop. A slow response on the inner loop reduces machining accuracy around circles and corners. A PID based speed controller was selected for implementation in individual axes, with a constant K_{pp} regulating the position control loop. The transfer function of the controller is defined by equation (16):

$$G(s) = K_p + \frac{1}{s}K_i + sK_d \qquad (16)$$

Where K_p, K_i and K_d are the gains of the controller. The algorithm was implemented in its discreet form according to equation (17):

$$x_n = K_p e_n + K_I \left(\frac{(e_n + e_{n-1})T_s}{2} + Int_{prev} \right) + K_d \frac{e_n - e_{n-1}}{T_s} \qquad (17)$$

Where e is the error, T_s is the signal sampling period and Int_{prev} is the error integral. The results of the control implementation are discussed below.
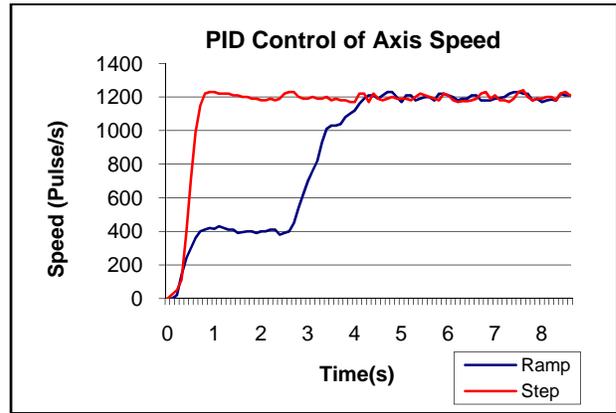
9.2 Speed Control Results

Figure 8: PID Control of Axis Speed

Figure 8 illustrates the PID response of an MRM linear actuation module. The controller is tracking a stepped servo speed reference of 1200 pulses/sec. The response exhibits a 400 ms rise time and less than 5% overshoot. The mean velocity of the response after initial rise time is 1206 pulses/sec, which implies a mean error of 0.5%.

The second response of Figure 8 illustrates a ramped velocity input into the PID controller. The velocity reference is ramped linearly to 400 pulses/sec in the first 700 ms, held constant and then ramped up to 1200 pulses/sec after 4500 ms. The PID controller displays a stable response at both speeds and less than 5% overshoot in both stages. The non-linearity of the velocity curve in the accelerated region is attributed to a ramped velocity reference used in the PID loop without current control. The inclusion of current control would result in a smoother acceleration profile for non-orthogonal tool trajectories.

10. Conclusion

Modular Reconfigurable Machines are mechanically transformable systems by means of the integration or removal of modular functional units on a platform. The scalable mechanical architecture of the system required the development of a modular Mechatronic control system. This paper has presented a software reference model for MRMs, which is capable of being reconfigured to control a system with a varying kinematic structure. The exploitation of the X-Y-Z Euler convention enabled the quick reconfiguration of a machines' physical representation in software.

An appropriate force control method has been identified and interpolation and servo control algorithms have been presented and evaluated. The Improved Tustin Method for interpolation exhibited high accuracy, and a low number for iterations for a circular trajectory. The low number of iterations was advantageous for the network orientated communication of instructions to low level servo control drives. The cascade structure for closed loop servo control exhibited acceptable results with a PID speed control algorithm implemented in the inner loop. Future work on MRM control will entail advancement of the software reconfiguration process and the improvement of system performance by manipulation of a real time kernel for numerical control.

Acknowledgements

References

[Mehrabi, 2000] M.G. Mehrabi, A.G. Ulsoy, Y.Koren; *Reconfigurable Manufacturing System and Their Enabling Technologies*, International Journal of Manufacturing Technology and Management; Vol. 1, 2000, Pg 113-130

[Koren, 2004] Y. Koren; *Reconfigurable Manufacturing Systems*; Proceedings: International Conference On Competitive Manufacturing; COMA '04, 4-6 February 2004; Stellenbosh, South Africa; Pg 69 – 79

[Padayachee, 2008a] J. Padayachee, G. Bright; *Modular Reconfigurable Machines for Reconfigurable Manufacturing Systems*; 24th ISPE International Conference on CAD/CAM, Robotics and Factories of the Future; 29-31 July 2008; Koriyama; Japan

[Padayachee, 2008b] J. Padayachee, I. Masekamela, G. Bright, C. Kumile, N.S. Tlale; *Modular Reconfigurable Machines Incorporating Open Architecture Control*; 15th International Conference on Mechatronics and Machine Vision in Practice (M2VIP'08); 2-4 December 2008; Auckland; New Zealand

[Landers, 2001] R.G. Landers, B.K. Min and Y. Koren; *Reconfigurable Machine Tools*; CIRP Annals - Manufacturing Technology; Volume 50; Issue 1; 2001; Pg 269-274.

[Tilbury, 1999] D.M. Tilbury, S. Kota; *Integrated Machine and Control Design for Reconfigurable Machine Tools*; Proceedings: Advanced Intelligent Mechatronics; 1999 IEEE/ASME International Conference on; Pg 629-634

[Shinno et al, 1981] H. Shinno and Y. Ito, *Structural Description of Machine Tools 1: Description Method and Application*; Bulletin of the JSME 24(187); January 1981; Pg 251-258

[Shinno et al, 1984] H. Shinno and Y. Ito; *A Proposed Generating Method For The Structural Configuration Of Machine Tools*; In ASME Winter Annual Meeting; 1984; ASME paper 84-WA/Prod-22.

[Moon et al, 2002a] Y.M Moon, S Kota; *Design of Reconfigurable Machine Tools*; Journal of Manufacturing Science and Engineering; May 2002; Volume 124; Issue 2; Pg 480-483

[Moon et al, 2002b] Y.M Moon, Sridhar Kota; Generalized *Kinematic Modeling of Reconfigurable Machine Tools*; Journal of Mechanical Design; March 2002, Vol. 124; Pg 47- 51

[Katz et al, 2004] R. Katz, J. Yook, Y. Koren; *Control of a Non-Orthogonal Reconfigurable Machine Tool*; Journal of Dynamic Systems, Measurement and Control; July 2004; Vol. 126; Pg 397-405

[Asato et al, 2002] O. L. Asato, E. R. R. Kato, R. Y. Inamasu; A. J. V. Porto; "*Analysis of Open CNC Architecture for Machine Tools*" Journal of the Brazilian. Society of Mechanical Sciences; July 2002; Vol. XXIV; Pg 208 - 212

[Pritschow et al, 2001] G. Pritschow, Y. Altintas, F. Jovane, Y. Koren, M. Mitsuishi, S. Takata, H. van Brussel, M. Weck, K. Yamazaki; *Open Controller Architecture Past, Present and Future*; CIRP Annals - Manufacturing Technology, Volume 50, Issue 2, 2001, Pg 463-470

[Nwokah et al, 2002] O.D.I Nwokah, Y. Hurmuzlu; *The Mechanical Systems Design Handbook: Modeling, Measurement and Control*; First edition; CRC Press; Boca Raton, Florida; 2002

[Shu et al, 2008] S.H Shu, S.K. Kang, D.H. Chung, I. Stroud; *Theory and Design of CNC Systems*; First edition; Springer -Verlang London Limited; London; 2008

[Otieno et al, 2008] A. Otieno, C. Mirman; *Finite Element Analysis of Cutting Forces and Temperatures on Microtools in the Micromachining of Aluminum Alloys;* Proceedings of the 2008 IAJC-IJME International Conference; 2008; Paper 191