Model Predictive Control for Reference Tracking on an Industrial Machine Tool Servo Drive

Michael A. Stephens, Chris Manzie, *Member, IEEE*, and Malcolm C. Good, *Member, IEEE*

***Abstract*—The benefits of model predictive control (MPC) have been well established; however its application to reference tracking on digital servo drives (DSDs), which typically have very fast update rates, is limited by the computational power of present-day processors. This paper presents a novel MPC formulation, which provides a mechanism to trade-off online computation effort with *tracking* performance, while maintaining stability. This is achieved by introducing a *trajectory* horizon, which is distinct from the prediction and control horizons typically encountered in MPC formulations. It is shown that increasing the trajectory horizon inherently leads to improved tracking; however larger horizon lengths also have the unwanted effect of increasing online computation. The proposed MPC formulation is compatible with recently developed explicit MPC solutions, and hence the burden of online optimization is avoided. The new approach is successfully implemented on an industrial machine tool DSD, and in terms of tracking accuracy, is shown to outperform the incumbent approach of cascaded PID control.**

***Index Terms*—Discrete-time systems, model predictive control (MPC), motion control, tracking systems, servomechanisms.**

I. Introduction

MACHINE tools are a fundamental component of modern manufacturing. Essentially every manufactured good is formed, in part, by a machining process. The demands on high precision machine tools are growing at a significant rate as the competing requirements of high accuracy and low cycle time become differentiators between manufacturers. At the heart of the modern machine tool are the motion control elements, which comprise a computer numerical control (CNC) and a number of digital servo drives (DSDs).

Historically, and still today, the majority of DSDs operate on a cascaded PID control architecture [1], [2]. While generally effective and simple to implement, the PID controller does have several limitations, especially as the performance requirements become more exacting. By its nature, it is reactive and thus requires a degree of positioning error to enact control effort. Secondly, in practice the process of tuning it with sufficient performance often requires a high level of skill and experience. Model predictive control (MPC) on the other hand is proactive [3], in that it predicts what control effort is required to ensure accurate tracking of the reference trajectory. MPC retains a feedback element to account for model uncertainty and is the only approach to explicitly account for state and actuator constraints. Furthermore, an integrator can be embedded into the formulation to account for unknown disturbances [4]. Moreover, given that a suitable dynamic model of the plant is available, tuning the controller is essentially a trade-off between tracking accuracy and control effort. Given the rich literature on system identification [5], developing a suitable plant model is generally a straightforward task.

The main limitation of MPC is that significant computation is required to perform the optimization for all but the most trivial of applications. The ability to use MPC in applications with fast dynamics and limited computational resources is a topic which has received significant attention recently [6], [7]. The focus has taken several paths, some of which include: tailoring the optimization technique to exploit the specific structure of the MPC problem [8], [9]; terminating the optimization early to ensure a result is obtained within the available calculation window [10], [11]; or using multiplex optimization to distribute the computational load [12], [13]. While these methods have the potential to reduce the online computation requirements, none are particularly suited to *trajectory tracking* MPC for servo drive applications. However, an alternative approach is to modify the structure of the original MPC problem to reduce the degrees of freedom [14]–[16]. The method proposed in this paper falls into this last category.

As it will be utilized in the implementation stages of this work, before proceeding it is necessary to introduce explicit MPC (EMPC) [17]. In this approach, an algorithm is developed to produce an equivalent closed-form (explicit) solution to the original MPC problem offline. The result is a gain-scheduling algorithm which is a function of a parameter vector that contains the current state of the system and the desired future output. The solution is shown to be piecewise affine in the parameter vector, where different controllers are defined for discrete polyhedral regions X_i within the parameter space. The online implementation then simplifies to a sequential search through the regions to locate the one to which the current parameter vector belongs. The controller associated with the identified region is then used to generate the input for the plant. The algorithm to construct the explicit solution is further refined in [18].

While this approach has the potential to reduce the online computation requirements, it does result in an increase in the memory used to store the program in the controller hardware. Moreover, as the MPC problem becomes more complex (higher order plant model, more input/output variables, more constraints, longer horizons, etc.) the number of regions in

Manuscript received October 29, 2011; revised February 03, 2012, March 23, 2012, May 03, 2012; accepted July 06, 2012. Date of publication October 09, 2012; date of current version January 09, 2013. Paper no. TII-11-692.

M. A. Stephens is with ANCA Motion, Bayswater North, Victoria 3153, Australia (e-mail: michael.stephens@ancamotion.com).

C. Manzie and M. C. Good are with the Department of Mechanical Engineering, The University of Melbourne, Victoria 3010, Australia (e-mail: manziec@unimelb.edu.au; mcgood@unimelb.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2012.2223222

the explicit solution grows. The computational advantage over online optimization is then somewhat diminished since it can take a long time to locate the appropriate region within the parameter space. In the worst case every region must be explored. In [19] an algorithm which takes the explicit solution and organizes it into a binary search tree is presented. Not only does this drastically reduce the online search time, but it also results in a reduction in the storage requirements.

While the problem formulation in [17] is adaptable to *trajectory* tracking problems, typically only *set-point* tracking is implemented. In [20], the authors successfully applied EMPC to a drive system with an update period of $T = 500$ μs. However only set-point tracking is considered and the prediction and control horizons are very limited. While commercial tools such as the Hybrid Toolbox for MATLAB [21] exist, they only allow for set-point tracking. Finally, in [22] it is claimed that even with EMPC formulated as a binary search tree, trajectory tracking is *impossible* from a computation standpoint.

This paper explores the application of MPC to trajectory tracking on an industrial machine tool DSD where update periods are in the order of 100 μs. The paper is organized as follows. Section 2 sets up the problem for trajectory tracking in a form suitable for EMPC. The general approach taken here is to modify the MPC problem to reduce the degrees of freedom. Section 3 presents simulation and experimental results from implementation on an industrial machine tool. Finally, in Section 4 some concluding remarks are made.

II. MPC Design for Trajectory Tracking

Consider a plant, with n states, m controlled outputs and l inputs, which is modelled as the following LTI discrete-time system

$$\mathbf{x}_p(k+1) = \mathbf{A}_p\mathbf{x}_p(k) + \mathbf{B}_p\mathbf{u}(k) \tag{1a}$$
$$\mathbf{y}(k) = \mathbf{C}_p\mathbf{x}_p(k) \tag{1b}$$

where $\mathbf{x}_p \in \mathbb{R}^{n\times 1}$ is the state vector, $\mathbf{u} \in \mathbb{R}^{l\times 1}$ is the input vector, $\mathbf{y} \in \mathbb{R}^{m\times 1}$ is the output vector, and $\mathbf{A}_p$, $\mathbf{B}_p$, and $\mathbf{C}_p$ are constant matrices of appropriate dimensions.

To ensure offset-free tracking in the face of model uncertainties and unknown constant disturbances, integral action in the controller is desirable. A method for incorporating an integrator within the MPC framework is to modify the plant model so that the input is the control update $\Delta\mathbf{u}(k)$, rather than control $\mathbf{u}(k)$ itself [23], [4]. This is achieved by taking the difference of both sides of (1a) to form

$$\Delta\mathbf{x}_p(k+1) = \mathbf{A}_p\Delta\mathbf{x}_p(k) + \mathbf{B}_p\Delta\mathbf{u}(k) \tag{2}$$

where $\Delta\mathbf{x}_p(k) = \mathbf{x}_p(k) - \mathbf{x}_p(k-1)$ and $\Delta\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1)$.

Next $\Delta\mathbf{x}_p(k)$ must be connected to the output $\mathbf{y}(k)$. Take the difference of both sides of (1b) to form

$$\begin{aligned}\mathbf{y}(k+1) - \mathbf{y}(k) &= \mathbf{C}_p(\mathbf{x}_p(k+1) - \mathbf{x}_p(k)) \\ &= \mathbf{C}_p\mathbf{A}_p\Delta\mathbf{x}_p(k) + \mathbf{C}_p\mathbf{B}_p\Delta\mathbf{u}(k).\end{aligned} \tag{3}$$

Finally, defining a new state vector as

$$\mathbf{x}(k) = \begin{bmatrix}\Delta\mathbf{x}_p(k) \\ \mathbf{y}(k)\end{bmatrix} \tag{4}$$

allows (2) and (3) to be combined to form

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\Delta\mathbf{u}(k) \tag{5a}$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) \tag{5b}$$

where

$$\mathbf{A} = \begin{bmatrix}\mathbf{A}_p & \mathbf{0}_{mn}^{\mathrm{T}} \\ \mathbf{C}_p\mathbf{A}_p & \mathbf{I}_m\end{bmatrix}$$
$$\mathbf{B} = \begin{bmatrix}\mathbf{B}_p \\ \mathbf{C}_p\mathbf{B}_p\end{bmatrix}$$
$$\mathbf{C} = [\mathbf{0}_{mn} \quad \mathbf{I}_m]$$

and $\mathbf{0}_{mn}$ is an $m \times n$ zero matrix and $\mathbf{I}_m$ is the identity matrix of size m.

Note that if the full state vector $\mathbf{x}_p$ is not available via direct measurement, as is typically the case, it can be estimated using, for example, a Kalman filter. The state vector $\mathbf{x}_p$, which is used to construct (4), is then replaced with its estimate $\hat{\mathbf{x}}_p$. The Kalman filter is an appropriate choice because the system under consideration is subject to both measurement and process noise.

For a trajectory tracking system, the optimization objective used in the MPC formulation typically includes a penalty on both predicted tracking error and predicted changes to the control input. It usually takes the form [3]

$$\min_{\Delta\mathcal{U}(k)} \left\{ \sum_{i=1}^{H_p} [\hat{\mathbf{y}}(i\,|\,k) - \mathbf{r}(i\,|\,k)]^{\mathrm{T}}\mathbf{Q}(i)[\hat{\mathbf{y}}(i\,|\,k) - \mathbf{r}(i\,|\,k)] + \sum_{i=0}^{H_u-1} \Delta\hat{\mathbf{u}}^{\mathrm{T}}(i\,|\,k)\mathbf{R}(i)\Delta\hat{\mathbf{u}}(i\,|\,k) \right\} \tag{6}$$

where $\hat{\mathbf{y}} \in \mathbb{R}^{m\times 1}$ is the predicted output, $\mathbf{r} \in \mathbb{R}^{m\times 1}$ is the reference trajectory, $\Delta\hat{\mathbf{u}} \in \mathbb{R}^{l\times 1}$ is the future change in the control input ($\Delta\hat{\mathbf{u}}(i\,|\,k) = 0$ for $i \geqslant H_u$), and $\Delta\mathcal{U}(k) \triangleq [\Delta\hat{\mathbf{u}}(0\,|\,k), \ldots, \Delta\hat{\mathbf{u}}(H_u - 1\,|\,k)]^{\mathrm{T}}$. Furthermore, H_p is the length of the prediction horizon, and $H_u \leqslant H_p$ is the length of the control horizon. In the interest of a simplified notation, the use of $(i\,|\,k)$ in the equation should be interpreted as $(k+i\,|\,k)$.

The optimization objective considered in this novel implementation of MPC is

$$\min_{\Delta\mathcal{U}(k)} \left\{ \sum_{i=1}^{H_t-2} [\hat{\mathbf{y}}(i\,|\,k) - \mathbf{r}(i\,|\,k)]^{\mathrm{T}}\mathbf{Q}(i)[\hat{\mathbf{y}}(i\,|\,k) - \mathbf{r}(i\,|\,k)] + \sum_{i=H_t-1}^{H_p} [\hat{\mathbf{y}}(i\,|\,k) - \hat{\mathbf{r}}(i\,|\,k)]^{\mathrm{T}}\mathbf{Q}(i)[\hat{\mathbf{y}}(i\,|\,k) - \hat{\mathbf{r}}(i\,|\,k)] + \sum_{i=0}^{H_u-1} \Delta\hat{\mathbf{u}}^{\mathrm{T}}(i\,|\,k)\mathbf{R}(i)\Delta\hat{\mathbf{u}}(i\,|\,k) \right\} \tag{7}$$

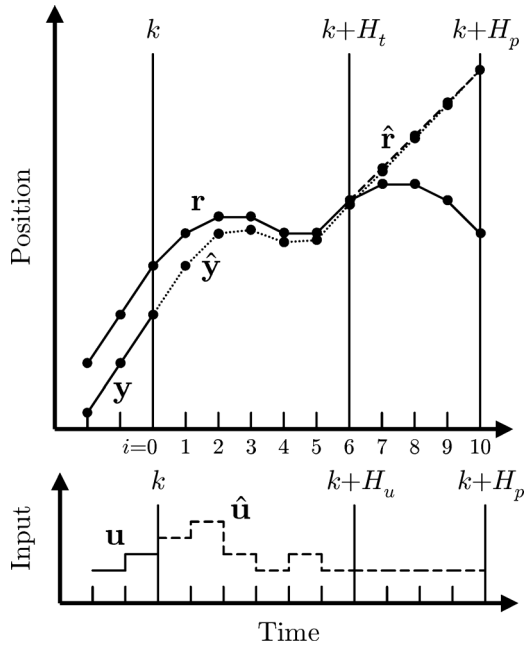

Fig. 1. Comparison of MPC horizons, where $H_t = H_u = 6$ and $H_p = 10$. The extended reference trajectory is modelled as a first-order hold (FOH).

where $\hat{\mathbf{r}} \in \mathbb{R}^{m\times 1}$ is the extended reference trajectory (ERT)—a full definition of which is given shortly—and $H_t \leqslant H_p$ is the length of the *trajectory* horizon. The introduction of H_t in the MPC formulation is believed to be a novel approach. Fig. 1 shows the relations between the different horizons.

When evaluating (7) the contribution of the predicted tracking error is weighted by the nonnegative diagonal matrix $\mathbf{Q} \in \mathbb{R}^{m\times m}$, while the magnitudes of changes to the control inputs are weighted by a strictly-positive diagonal matrix $\mathbf{R} \in \mathbb{R}^{l\times l}$. These weights need not be constant at each step in the horizon, hence the inclusion of the index i.

The predicted output is given by

$$\begin{bmatrix} \hat{\mathbf{y}}(1\,|\,k) \\ \vdots \\ \hat{\mathbf{y}}(H_p\,|\,k) \end{bmatrix} = \Psi\mathbf{x}(k) + \Theta\Delta\mathcal{U}(k) \tag{8}$$

where

$$\Psi = \begin{bmatrix} \mathbf{CA} \\ \vdots \\ \mathbf{CA}^{H_p} \end{bmatrix}$$

$$\Theta = \begin{bmatrix} \mathbf{CB} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{CAB} & \mathbf{CB} & & \mathbf{0} \\ \vdots & & \ddots & \\ \mathbf{CA}^{H_u-1}\mathbf{B} & \mathbf{CA}^{H_u-2}\mathbf{B} & \cdots & \mathbf{CB} \\ \vdots & \vdots & & \vdots \\ \mathbf{CA}^{H_p-1}\mathbf{B} & \mathbf{CA}^{H_p-2}\mathbf{B} & \cdots & \mathbf{CA}^{H_p-H_u}\mathbf{B} \end{bmatrix}.$$

To ensure stability, the prediction horizon H_p typically needs to be large [24], [3]. The requirement is normally for the plant to be stable and H_p to be large enough to include the transients of the dominant plant dynamics—*the plant settling time* (T_{settle}). Under this requirement, $H_p \geqslant (T_{\text{settle}})/(T)$, where T is the controller update period. Fortunately, the class of system considered in this work, machine tool servo drives, are nearly always open-loop stable, and extending H_p has only a minimal effect on the complexity of solving the MPC problem. On the other hand, extending H_t or H_u will increase the complexity significantly. The reason for this will become apparent shortly. For now it is enough to state that by adjusting the length of H_t, improved tracking precision (large H_t) can be traded-off against reduced online computation effort (small H_t). Furthermore, the length of both H_t and H_u has no influence on the closed-loop stability of the system. The inclusion of the trajectory horizon in the MPC formulation is especially suited to trajectory tracking where available computation time is limited; such is the case for controllers with fast update rates.

For modern motion control systems, the reference trajectory $\mathbf{r}$ is typically a set of discrete position commands and the input $\mathbf{u}$ is the motor torque, or equivalently, for a permanent magnet synchronous motor, the q-axis (torque producing) motor current [25]. The condition $\Delta\hat{\mathbf{u}}(i\,|\,k) = 0$ for $i \geqslant H_u$ implies constant torque over the same interval. Given the existence of mechanical friction, a nonzero constant torque will result in the prediction of constant velocity at steady state. Under this reasoning, for the class of ERTs defined below, it makes sense for $H_u = H_t$. In the general case this may not be true, hence the decision in the MPC formulation proposed here to introduce the trajectory horizon and distinguish it from the control horizon.

Clearly the ERT can be defined in several ways. The simplest is $\hat{\mathbf{r}}(i\,|\,k) = \mathbf{r}(H_t\,|\,k)$, $\forall i \in [H_t+1, H_p]$; effectively a zero-order hold (ZOH). With this choice, the predicted input $\hat{\mathbf{u}}$ at the end of the control horizon will be required to *reverse* in order to drive the axis velocity to zero. Since machine tools typically follow smooth contours, a better choice, which is proposed in this paper, is a first-order hold (FOH) as shown in Fig. 1. In this way, at the end of the control horizon the predicted input $\hat{\mathbf{u}}$ will approach a level that produces a velocity close to that of the reference trajectory at $k + H_t$. To implement a FOH-ERT, there is the additional requirement that $H_t \geqslant 2$. The FOH-ERT is given by

$$\hat{\mathbf{r}}(i\,|\,k) = \mathbf{r}(H_t - 1\,|\,k) + (i - H_t + 1)(\mathbf{r}(H_t\,|\,k) - \mathbf{r}(H_t - 1\,|\,k))$$

$\forall i \in [H_t - 1, H_p]$, or in matrix form

$$\begin{bmatrix} \hat{\mathbf{r}}(H_t - 1\,|\,k) \\ \vdots \\ \hat{\mathbf{r}}(H_p\,|\,k) \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{r}(H_t - 1\,|\,k) \\ \mathbf{r}(H_t\,|\,k) \end{bmatrix}^{\mathrm{T}} \Lambda$$

where

$$\Lambda = \begin{bmatrix} \mathbf{I}_m & \mathbf{0}_m & -\mathbf{I}_m & \cdots & (H_t - H_p)\mathbf{I}_m \\ \mathbf{0}_m & \mathbf{I}_m & 2\mathbf{I}_m & \cdots & (H_p - H_t + 1)\mathbf{I}_m \end{bmatrix}$$

and $\mathbf{I}_m$ and $\mathbf{0}_m$ are $m \times m$ identity and zero matrices, respectively.

With respect to

$$\begin{aligned} \mathcal{R} &\triangleq \operatorname{diag}(\mathbf{R}(0), \ldots, \mathbf{R}(H_u - 1)) \\ \mathcal{Q} &\triangleq \operatorname{diag}(\mathbf{Q}(1), \ldots, \mathbf{Q}(H_p)) \\ \mathcal{Q}_t &\triangleq \operatorname{diag}(\mathbf{Q}(1), \ldots, \mathbf{Q}(H_t - 2)) \\ \mathcal{Q}_p &\triangleq \operatorname{diag}(\mathbf{Q}(H_t - 1), \ldots, \mathbf{Q}(H_p)). \end{aligned}$$

Equation (7) may be rewritten as

$$\min_{\Delta\mathcal{U}(k)} \left\{ \frac{1}{2}\Delta\mathcal{U}^{\mathrm{T}}(k)\mathcal{H}\Delta\mathcal{U}(k) + \theta(k)\mathcal{F}\Delta\mathcal{U}(k) \right\}. \tag{9}$$

Here

$$\mathcal{H} = 2\mathcal{R} + 2\Theta^{\mathrm{T}}\mathcal{Q}\Theta$$

$$\mathcal{F} = \begin{bmatrix} 2\Psi^{\mathrm{T}}\mathcal{Q}\Theta \\ \mathbf{0}_u \\ -2\mathcal{Q}_t\Theta_t \\ -2\Lambda\mathcal{Q}_p\Theta_p \end{bmatrix}$$

$$\theta(k) = \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{u}(k-1) \\ \mathbf{r}(1\,|\,k) \\ \vdots \\ \mathbf{r}(H_t\,|\,k) \end{bmatrix}^{\mathrm{T}}$$

where Θ_t and Θ_p are respectively the first $m(H_t-2)$ rows and the last $m(H_p-H_t+2)$ rows of Θ, and $\mathbf{0}_u$ is a zero matrix of size $l \times lH_u$.

In addition, it is possible to define constraints on the input $\hat{\mathbf{u}}$, the input change $\Delta\hat{\mathbf{u}}$, and the output $\hat{\mathbf{y}}$ as

$$\hat{\mathbf{u}}_{\min} \leqslant \hat{\mathbf{u}}(i\,|\,k) \leqslant \hat{\mathbf{u}}_{\max} \quad \forall i \in [0, H_u-1] \tag{10}$$

$$\Delta\hat{\mathbf{u}}_{\min} \leqslant \Delta\hat{\mathbf{u}}(i\,|\,k) \leqslant \Delta\hat{\mathbf{u}}_{\max} \quad \forall i \in [0, H_u-1] \tag{11}$$

$$\hat{\mathbf{y}}_{\min} \leqslant \hat{\mathbf{y}}(i\,|\,k) \leqslant \hat{\mathbf{y}}_{\max} \quad \forall i \in [1, H_p]. \tag{12}$$

These constraints can be expressed in terms of $\Delta\mathcal{U}$ and combined (see [3]) to form

$$\mathcal{G}\Delta\mathcal{U}(k) \leqslant \mathcal{W} + \mathcal{X}\theta^{\mathrm{T}}(k). \tag{13}$$

Equation (9) in conjunction with (13) is in the standard form of a quadratic optimization problem, where $\theta(k)$ is a parameter vector and is known *a priori*. At each sample instant k, the optimization is solved and the input $\mathbf{u}(k) = \mathbf{u}(k-1) + \Delta\hat{\mathbf{u}}(0\,|\,k)$ is applied to the plant. At the next sample instant the optimization is repeated using the updated parameter vector and a shifted horizon.

The complexity of the optimization problem is in part a function of the length of $\theta(k)$. As the number of elements in $\theta(k)$ increases, so too does the effort required to complete the optimization. This is why the inclusion of the trajectory horizon H_t into the formulation benefits practical implementations of trajectory tracking MPC. If *classical* trajectory tracking MPC is used, then the parameter vector would be

$$\theta(k) = \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{u}(k-1) \\ \mathbf{r}(1\,|\,k) \\ \vdots \\ \mathbf{r}(H_p\,|\,k) \end{bmatrix}^{\mathrm{T}} \tag{14}$$

which is of length $n+m+l+H_p$. Typically $H_p \gg n+m+l$, hence replacing H_p with the shorter H_t has a significant impact on reducing computational complexity. As will be shown in the next section of the paper, H_t need only be a fraction of the length of H_p in order to achieve a stable system with a high level of tracking precision. Finally, $H_t = 1$ with a ERT-ZOH is in effect *set-point* tracking, as implemented in [21].

Fig. 2. Production 5-axis CNC tool and cutter grinding machine.

Fig. 3. Headstock axis loaded with a large workpiece.

To further address the issue arising from the limited computational power associated with modern industrial DSDs, the problem is formulated as an explicit MPC solution [17], [18], yielding solutions for $\Delta\hat{\mathbf{u}}$ of the form

$$\Delta\hat{\mathbf{u}}(\theta(k)) = \mathbf{M}_i\theta^{\mathrm{T}}(k) + \mathbf{m}_i, \quad \theta(k) \in X_i \tag{15}$$

where $\mathbf{M}_i \in \mathbb{R}^{l\times(n+m+l+mH_t)}$ and $\mathbf{m}_i \in \mathbb{R}^{l\times 1}$ represent the controller gains associated with polyhedral region X_i.

III. Results and Discussion

In this section, the performance of the proposed control methodology is demonstrated both in simulation and experimentation. The system chosen for testing is the headstock (servo) axis of the production 5-axis CNC tool and cutter grinding machine shown in Fig. 2. The headstock with a large workpiece mounted is shown in Fig. 3. This axis configuration (the combined dynamics of the motor, coupling and workpiece) has proven to be a challenge to control using the incumbent approach of cascaded PID [2]. To stabilize the plant, and at the same time provide adequate performance, *manual* placement of a current reference notch filter is required. Since the centre frequency of the filter is dependent on the physical characteristics of the workpiece, the need for an engineer or experienced technician to frequently retune the axis is a continuing problem. This provides the motivation for this work: design a high performance controller which is easily tuned.

The architecture of the complete motion control system for this machine is as follows. The CNC (an industrial PC running a real-time operating system) interprets a part program and generates a sequence of position commands for each of the axes at

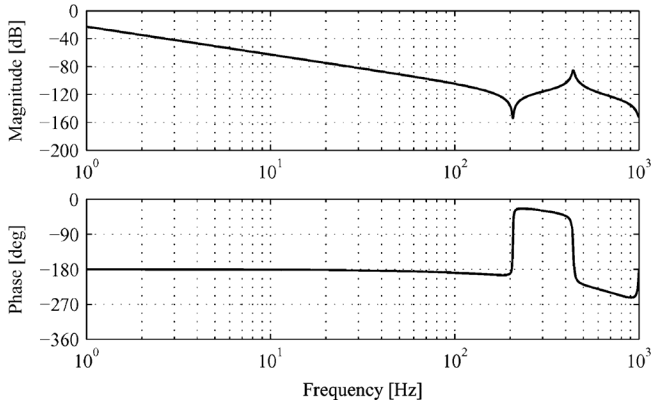

Fig. 4. Frequency response of the headstock axis plant model.

a rate of 250 Hz (or one every 4 ms). These position commands are determined so that constraints on the axes in terms of position, velocity, acceleration and jerk (the time derivative of acceleration) are met. The only constraint which must be maintained at the DSD level is the magnitude of the motor current; that is, the input to the mechanical plant $\mathbf{u}$. Hence only the constraint given by (10) must be included in the MPC formulation.

A. Plant Model

Derivation of the plant model for the headstock axis of the CNC grinding machine (including the large workpiece) and the methods used for identification, is presented in [2]. The paper also demonstrates the validity of the simulation models in reproducing the dynamics of the industrial plant. The model is given by

$$\mathbf{A}_p = \begin{bmatrix} 3.6\times10^{-1} & 6.4\times10^{-1} & -2.1\times10^{3} & 2.1\times10^{3} \\ 1.8\times10^{-1} & 8.2\times10^{-1} & 5.9\times10^{2} & -5.9\times10^{2} \\ 3.8\times10^{-4} & 1.2\times10^{-4} & 3.7\times10^{-1} & 6.3\times10^{-1} \\ 3.3\times10^{-5} & 4.7\times10^{-4} & 1.8\times10^{-1} & 8.2\times10^{-1} \end{bmatrix}$$

$$\mathbf{B}_p = [8.6\times10^{-5} \quad 7.5\times10^{-6} \quad 2.5\times10^{-8} \quad 9.9\times10^{-10}]^{\mathrm{T}}$$

$$\mathbf{C}_p = [0 \quad 0 \quad 5.7\times10^{1} \quad 0].$$

In this fourth order model ($n = 4$), the single input ($l = 1$) is the q-axis motor current in milliamps, and the single output ($m = 1$) is the motor position in degrees. The update period is $T = 500\ \mu$s.[1] As can be seen in the bode plot shown in Fig. 4, this plant contains a lightly damped resonance. This dynamic adds an extra challenge for controller design. The only measured state is the motor position; hence the other states must be estimated using a Kalman filter. In addition, it should be noted that this is a direct-drive rotary axis and operates in modulo 360°. This must be accounted for when calculating the plant states and also the ERT.

As the servo axis is a mechanical device supported by lubricated bearings, perhaps not surprisingly, it is affected by nonlinear friction. Since the MPC implementation here assumes a linear plant, there are really two options for dealing with this system property. One option is to identify a model of the friction and use a type of feedback linearization to explicitly account for the effect on the motion of the servo axis. Alternatively, treat the nonlinearity as an unmodeled disturbance and let the integral action of the controller reject it. So as not to complicate the implementation, the latter approach is used here. For this application, the experimental results do not show the telltale signs that nonlinear friction is a problem for this controller. However, if this were not the case, then the extra design effort and computational overhead to implement feedback linearization may be appropriate.

[1]B-spline interpolation is done in the DSD to convert from 4 ms to 500 μs.

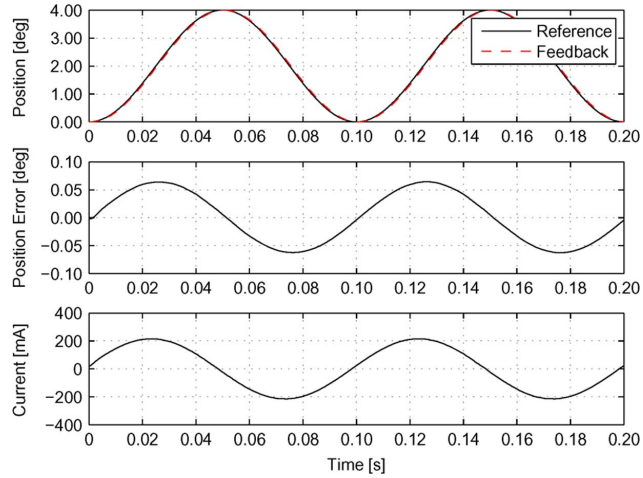

Fig. 5. Simulated tracking performance for MPC using a FOH-ERT, $H_t = H_u = 8$ and $H_p = 50$.

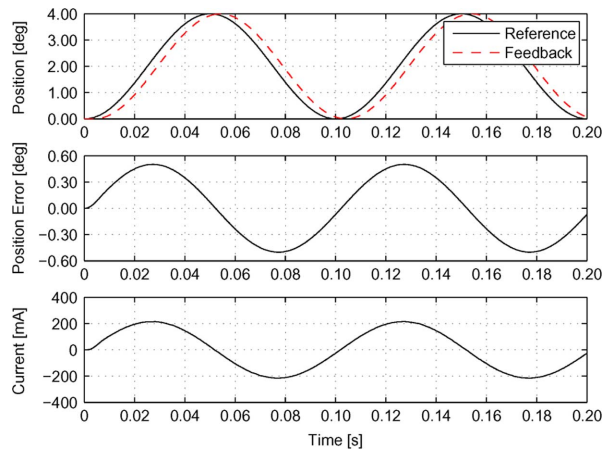

Fig. 6. Simulated tracking performance for a cascaded PID controller typically used in industry.

B. Simulation Results

Initially, results obtained via simulation are used to compare the different control approaches. The investigation begins with a comparison between cascaded PID control, MPC set-point tracking (where $H_t = 1$), and MPC trajectory tracking (where $H_t > 1$), using both ZOH-ERT and FOH-ERT schemes. The objective function weights for the MPC are $\mathbf{Q}(i) = 10^6$, $\forall i \in [0, H_p]$, and $\mathbf{R}(i) = 1$, $\forall i \in [1, H_u - 1]$, and the input constraints are ± 1000 mA. Since in simulation there is no requirement for real-time calculation, online optimization is used. The reference trajectory is a sinusoid with an amplitude of 2° and a frequency of 10 Hz. Fig. 5 shows the results for a FOH-ERT with $H_t = H_u = 8$ and $H_p = 50$. The maximum tracking error is found to be 0.0643° or 3.2% of the reference signal amplitude. This can be compared to the performance of a cascaded PID controller used in industry (Fig. 6). The maximum tracking error is found to be 0.5021° or 25.1%.

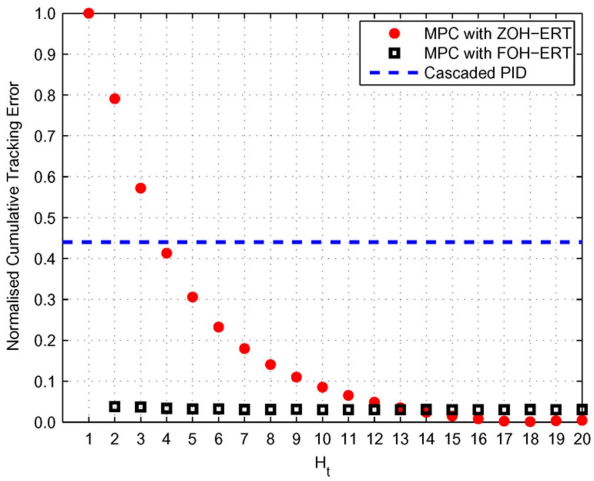

Fig. 7. Comparison of simulated tracking performance between cascaded PID control, MPC set-point tracking, and MPC trajectory tracking for both ZOH-ERT and FOH-ERT schemes, where $H_t = H_u$ and $H_p = 50$, for a 1 Hz sinusoidal reference.

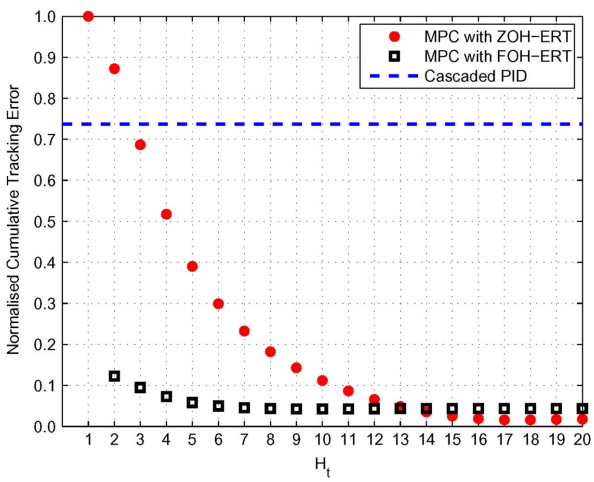

Fig. 8. Comparison of simulated tracking performance between cascaded PID control, MPC set-point tracking, and MPC trajectory tracking for both ZOH-ERT and FOH-ERT schemes, where $H_t = H_u$ and $H_p = 50$, for a 10 Hz sinusoidal reference.

Figs. 7 and 8 show the normalized cumulative tracking error for each of the control approaches for both 1 Hz and 10 Hz sinusoidal reference trajectories. These frequencies represent the typical operating range for this machine. The metric is calculated by summing up the magnitude of the tracking error at each control update over a complete cycle of the sinusoidal reference trajectory, and then normalizing to the result for MPC set-point tracking. From the graphs it can be seen that as the MPC trajectory horizon is increased, the tracking error decreases. And most importantly, for values of H_t which are able to be implemented on the available hardware ($H_t = H_u \leqslant 6$), FOH-ERT is better than ZOH-ERT. However, the true benefits of MPC can be seen when compared to the PID controller. For $H_t > 3$, the performance of MPC is superior to PID control, especially when tracking high frequency trajectories.

For insight into why FOH-ERT performs better than ZOH-ERT, Figs. 9 and 10 include waterfall graphs with the motor current. The thin lines show the predicted control input along the

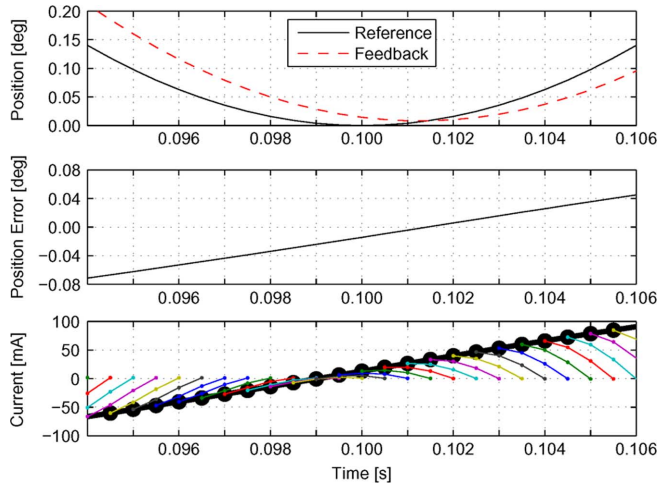

Fig. 9. Waterfall chart for MPC with ZOH-ERT, $H_t = H_u = 4$ and $H_p = 50$.

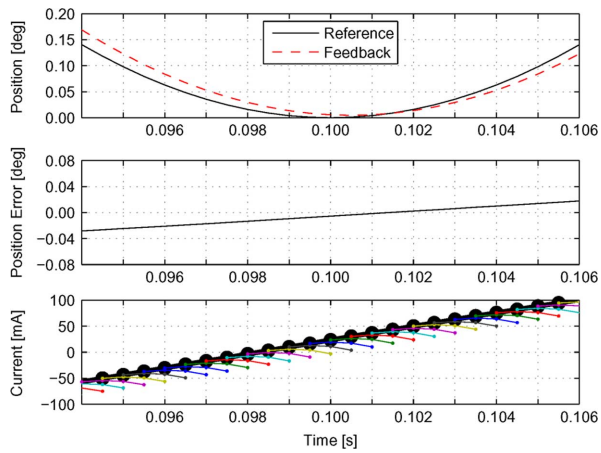

Fig. 10. Waterfall chart for MPC with FOH-ERT, $H_t = H_u = 4$ and $H_p = 50$.

Fig. 11. DSDs which drive the five servo axes in the machine tool depicted in Fig. 2. The sixth DSD is for the grinding wheel spindle. Each DSD has a DSP which implements the motion control algorithms.

entire control horizon.[2] In both simulations $H_t = H_u = 4$ and $H_p = 50$. Knowing that for a ZOH-ERT, the controller is expecting the trajectory reference to come to rest after H_t updates, it is not surprising that $\hat{\mathbf{u}}(H_u \,|\, k) \approx 0$, $\forall k$. Note that the plant model includes viscous friction, hence the motor current does not actually have to *reverse* to ensure that the motor velocity approaches zero along the remainder of the prediction horizon. Comparing this to FOH-ERT, it is seen that the predicted control inputs are a closer match to the actual control inputs; that

[2]Recall that only the first control input is actually applied to the plant.

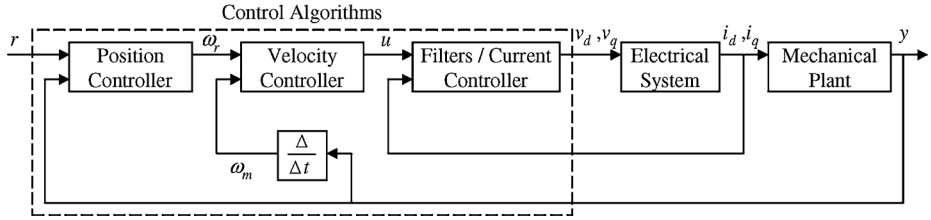

Fig. 12. Structure of the cascaded PID motion control algorithm implemented in the DSD of this production machine.

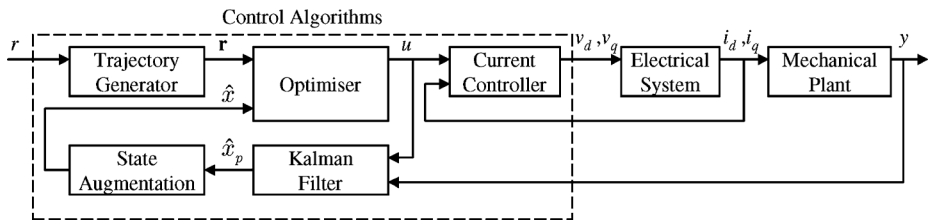

Fig. 13. Structure of the proposed MPC implementation.

TABLE I
PID CONTROLLER TUNING PARAMETERS

Parameter	Value	Unit
K_{pos}	3.6111	RPM/deg
K_{vel}	0.4444	A/RPM
T_i	2000	μs

is, the thin lines follow the thick line more closely. As a result, the tracking error is greatly reduced.

C. Control Implementation

As indicated earlier in the paper, the traditional control approach in industry is cascaded PID. The DSD hardware used in this machine and shown in Fig. 11 implements this approach on a Texas Instruments TMS320VC33 DSP. The structure of the PID controller architecture is shown in Fig. 12. Here the position, velocity and current controllers are all implementations of PID control. The Filters label included with the current controller refers to the notch filter used to manage any mechanical resonance, as is the case for the system under investigation. The controller gains for the position controller (Proportional) and the velocity controller (Proportional plus Integral) are shown in Table I, and the notch filter is an implementation of the following discrete transfer function

$$\frac{0.9352(z^2 - 1.9774\,z + 1)}{z^2 - 1.8492\,z + 0.8651}. \tag{16}$$

Implementation of the proposed MPC approach begins by removing the position and velocity controllers from the existing DSD source code and disabling the notch filter. The existing current controller is retained, which provides the interface for the output of the MPC controller. Fig. 13 shows the structure of the complete system.

The optimizer in Fig. 13 can be an online method or an explicit approach. Due to the limited computation power of the processor on the DSD, online optimization is not possible; even EMPC with sequential search is found to be unachievable for $H_t = H_u > 2$ (essentially when more than 15 regions, or approximately 50 hyperplanes, must be checked to locate to which region the parameter vector belongs). However, using the binary search tree approach [19], it is possible to implement MPC with longer horizons: as stated previously, up to $H_t = H_u = 6$.

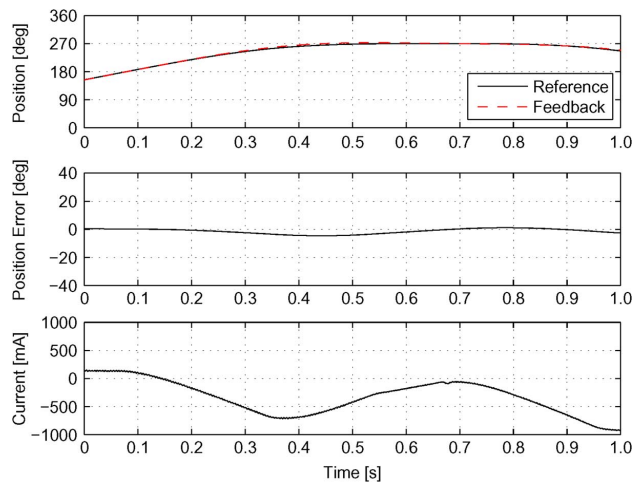

Fig. 14. Experimental results of the headstock axis of the CNC machine tool tracking a portion of a trapezoidal velocity profile using EMPC with FOH-ERT, $H_t = H_u = 3$, and $H_p = 50$.

Since the proposed MPC controller had already been developed and tested in the Simulink simulation environment, the use of Simulink Coder (formally known as Real-Time Workshop) was used to automatically generate the C-code required for the implementation.

D. Experimental Results

Finally, the experimental results of MPC on a production CNC machine tool (Fig. 2) are presented. The performance of MPC is compared to that of the PID controller. Again the input constraints are set to ± 1000 mA.[3] The data acquisition system on the DSD works by buffering data in local RAM, and when the buffer is full the data is streamed back to the CNC via non-cyclic data transfer, where it can be stored for later analysis. Due

[3]The motor is actually rated at a continuous 12 A. The setting of such a low value allows the authors to introduce a sizeable disturbance manually.

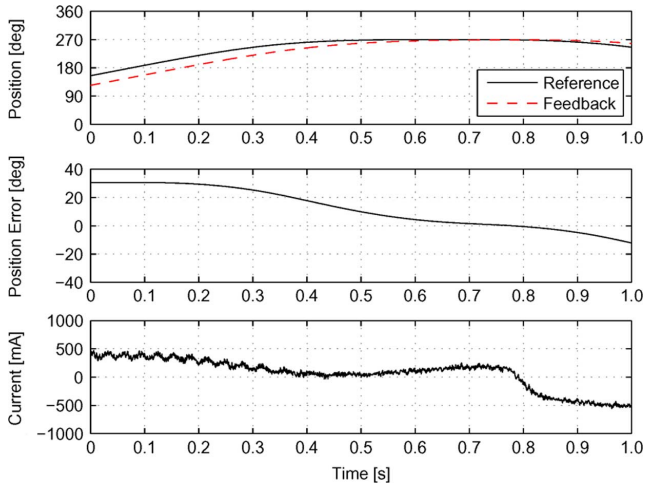

Fig. 15. Experimental results of the headstock axis of the CNC machine tool tracking a portion of a trapezoidal velocity profile using a cascaded PID controller used in industry.

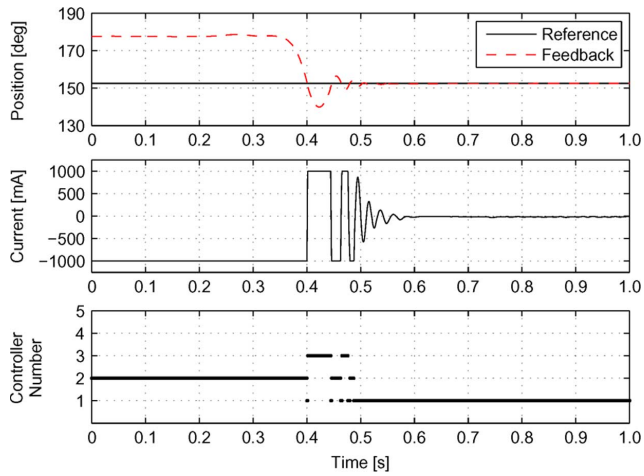

Fig. 16. Experimental results of the headstock axis of the CNC machine tool rejecting a large position disturbance using EMPC with FOH-ERT, $H_t = H_u = 3$, and $H_p = 50$.

to the limited memory on the DSD only four variables, each of 2048 data points, can be logged simultaneously in each experiment. Given a sample period of $T = 500$ μs, this means that only about 1 s of data capture is possible.

Fig. 14 shows how an EMPC controller with a FOH-ERT, $H_t = H_u = 3$, and $H_p = 50$ performs on the machine. The EMPC contains 11 controllers across 25 polyhedral regions, and is organized into a binary search tree with 127 nodes, where the maximum depth is 7. Each of the 11 controllers take the form of (15). The trajectory being tracked by the axis is essentially a trapezoidal velocity profile, with a stroke of 270° and a maximum velocity of 60 r/min. Since the current remains well within the input limits, the EMPC never leaves region 1, which corresponds to controller #1 and linear operation (no constraints are active over the entire control horizon). This can be compared to the PID controller (Fig. 15) where the magnitude of the tracking error is much larger. Fig. 16 shows how the MPC responds to a large position disturbance. As can be seen, shortly after the disturbance is removed at 0.35 s, the position feedback initially overshoots but eventually settles at the set-point. The graph on the bottom of Fig. 16 also shows how the controller transitions between different polyhedral regions (with different controllers) as the input constraints along the control horizon become active. Note that while the EMPC solution actually contains 11 controllers, only three (#1, #2, and #3) are required during the transient stage after the disturbance is removed. As a final note, with the current limits set to 12 A, the axis is found to be very stiff and the integrator within the MPC effectively rejects realistic disturbances.

IV. Conclusion

Where previously *trajectory* tracking MPC using standard industrial processors was only possible on systems with relatively slow updates, it has been demonstrated in this work that implementation is possible on industrial machine tool servo drives, where update periods are in the order 100 μs. By using the proposed MPC formulation, specifically the inclusion of the trajectory horizon (H_t) and the first-order hold extended reference trajectory (FOH-ERT), it has been demonstrated that the extra implementation effort is justified, as superior performance is achieved in comparison to both *set-point* tracking MPC and cascaded PID control.

The significance of this is that for the first time the benefits of MPC, such as optimal tracking performance with constraint management, and importantly, a straightforward method to tune the controller (given successful system identification is achieved), can be used to improve the tracking performance of machine tools, and differentiate a machine tool manufacturer in a global market worth around US$70 billion [26]. Furthermore, improving the performance of machine tools has a positive impact on both the quality of manufactured goods as well as throughput; both of these metrics have direct economic benefits for end-users of machine tools.

References

[1] M. A. Stephens, C. Manzie, and M. C. Good, "Control-oriented modeling requirements of a direct-drive machine tool axis," *J. Dyn. Syst. Meas. Control*, vol. 134, no. 5, p. 054503, Sep. 2012.

[2] M. A. Stephens, C. Manzie, and M. C. Good, "On the stability analysis and modelling of a multirate control direct-drive machine tool axis subject to large changes in load dynamics," in *Proc. Amer. Control Conf.*, 2010, pp. 1550–1555.

[3] J. M. Maciejowski, *Predictive Control With Constraints*. Harlow, U.K.: Prentice-Hall, 2002.

[4] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB*. London, U.K.: Springer-Verlog, 2009.

[5] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 1999.

[6] C. Lozoya, M. Velasco, and P. Martí, "The one-shot task model for robust real-time embedded control systems," *IEEE Trans. Ind. Inf.*, vol. 4, no. 3, pp. 164–174, Aug. 2008.

[7] R. Bautista-Quintero and M. J. Pont, "Implementation of h-infinity control algorithms for sensor-constrained mechatronic systems using low-cost microcontrollers," *IEEE Trans. Ind. Inf.*, vol. 4, no. 3, pp. 175–184, Aug. 2008.

[8] S. Boyd, *Convex Optimisation*. New York, NY: Cambridge Univ. Press, 2004.

[9] E. A. Yildirim and S. J. Wright, "Warm-start strategies in interior-point methods for linear programming," *SIAM J. Opt.*, vol. 12, no. 3, pp. 782–810, 2002.

[10] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Fast, large-scale model predictive control by partial enumeration," *Automatica*, vol. 43, no. 5, pp. 852–860, 2007.

[11] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, Mar. 2010.

[12] K. V. Ling, J. Maciejowski, and B. Wu, "Multiplexed model predictive control," in *Proc. IFAC WC*, Prague, 2005.

[13] K. V. Ling, W. K. Ho, Y. Feng, and B. Wu, "Integral-square-error performance of multiplexed model predictive control," *IEEE Trans. Ind. Inf.*, vol. 7, no. 2, pp. 196–203, May 2011.
[14] U. Halldorsson, M. Fikar, and H. Unbehauen, "Nonlinear predictive control with multirate optimisation step lengths," *IEE Proc.—Control Theory Appl.*, vol. 152, no. 3, pp. 273–284, May 2005.
[15] R. Cagienard, P. Grieder, E. C. Kerrigan, and M. Morari, "Move blocking strategies in receding horizon control," *J. Process Control*, vol. 17, no. 6, pp. 563–570, 2007.
[16] S. Thomsen, N. Hoffmann, and F. W. Fuchs, "PI control, PI-based state space control, and model-based predictive control for drive systems with elastically coupled loads—A comparative study," *IEEE Trans. Ind. Electron.*, vol. 58, no. 8, pp. 3647–3657, Aug. 2011.
[17] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 39, no. 10, pp. 1845–1846, 2002.
[18] P. Tøndel, T. A. Johansen, and A. Bemporad, "An algorithm for multiparametric quadratic programming and explicit MPC solutions," *Automatica*, vol. 39, no. 3, pp. 489–497, 2003.
[19] P. Tøndel, T. A. Johansen, and A. Bemporad, "Evaluation of piecewise affine control via binary search tree," *Automatica*, vol. 39, no. 3, pp. 945–950, 2003.
[20] M. Cychowski, K. Szabat, and T. Orlowska-Kowalska, "Constrained model predictive control of the drive system with mechanical elasticity," *IEEE Trans. Ind. Electron.*, vol. 56, no. 6, pp. 1963–1973, Jun. 2009.
[21] A. Bemporad, *Hybrid Toolbox for MATLAB—User's Guide*. [Online]. Available: cse.lab.imtlucca.it/~bemporad/hybrid/toolbox/ 2011
[22] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *Int. J. Robust Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.
[23] D. M. Prett and C. E. Garcia, *Fundamental Process Control*, H. Brenner, Ed. Stoneham, MA: Butterworth, 1988.
[24] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.
[25] P. C. Krause, O. Wasynczuk, and S. D. Sudhoff, *Analysis of Electric Machinery and Drive Systems*, 2nd ed. Piscataway, NJ: IEEE Press, 2002.
[26] Global Industry Analysts, "Machine tools—A global strategic business report, Tech. Rep., 2011.

Michael A. Stephens received the B.E.(Hons.) degree in mechatronics engineering and the B.C.S. degree from the University of Melbourne, Melbourne, Australia, both in 2004, and the Ph.D. degree from the same university in 2012.

Since 2006, he has been with ANCA Pty. Ltd., Melbourne, Australia, where he currently leads the Embedded Software Group of the subsidiary company, ANCA Motion Pty. Ltd. His focus in this role is on the development and implementation of motion control solutions for the machine tool and general automation industries. His current research interests include developing methods for practical application of model-based control techniques, and analysis of mechanical and electrical system dynamics.

Chris Manzie (M'06) received the B.S. degree in physics and the B.E.(Hons.) degree in electrical and electronic engineering from the University of Melbourne, Melbourne, Australia, both in 1996, and the Ph.D. degree from the same university in 2001.

Since 2003, he has been affiliated with the Department of Mechanical Engineering, University of Melbourne, where he is currently an Associate Professor and an Australian Research Council Future Fellow. He was a Visiting Scholar with the University of California, San Diego, in 2007, and a Visiteur Scientifique at IFP Energies Nouvelles, Paris, France, in 2012. He has industry collaborations with companies including Ford Australia, BAE Systems, ANCA Motion, and Virtual Sailing. His research interests lie in applications of model-based and extremum-seeking control in fields including mechatronics and energy systems.

Associate Prof. Manzie is a Member of the IFAC Technical Committees on Automotive Control.

Malcolm C. Good (M'00) received the Ph.D. degree in mechanical engineering from The University of Melbourne, Melbourne, Australia, in 1975.

Since 1989 he has been with the Mechanical and Manufacturing Engineering Department, University of Melbourne, serving as Head of Department from 1992 to 1996, and where he is currently an Emeritus Professor. Previously, he was Program Leader for Integrated Manufacture with the Division of Manufacturing Technology of the Commonwealth Scientific and Industrial Research Organization (CSIRO), Melbourne, Australia. He has held visiting appointments at the Institute for Sound and Vibration Research (ISVR) of the University of Southampton, Southampton, U.K.; the Highway Safety Research Institute (HSRI), University of Michigan at Ann Arbor; General Electric Corporate Research and Development, Schenectady; and Cambridge University, Cambridge, U.K. His research has been in the fields of fluid mechanics, highway geometrics, human factors of automobile and motorcycle control, vehicular impact with roadside structures, dynamics and control of machine tools and industrial robots, and automotive drive-by-wire technologies.

Prof. Good has been President of the Australian Robot Association, Australian Contact Person for the International Advanced Robotics Program, Interim Director of the Advanced Engineering Centre for Manufacturing, and a Program Leader and Board Member of the Research Centre for Advanced By-Wire Technologies, University of Melbourne.