MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Cascaded Reference Governor-MPC for Motion Control of Two-Stage Manufacturing Machines

Di Cairano, S.; Goldsmith, A.; Kalabić, U.; Bortoff, S.A.

TR2018-083 July 12, 2018

Abstract

Two-stage processing machines for manufacturing are equipped with two sets of actuators for motion control with different operating ranges and bandwidths. The processing time, product quality and flexibility of the manufacturing process can be optimized by coordinating these actuators and exploiting the entire actuator operating range. We propose a motion control strategy for two-stage processing machines based on model predictive control (MPC). By exploiting timescale separation, we formulate the problem as a single-stage motion control with reference-dependent constraints. Feasibility of the MPC problem is guaranteed by using a reference governor that adjusts the feed rate. The proposed method guarantees correct processing while satisfying the actuators' range and dynamics constraints, finite time processing of a given spatial pattern and real-time execution even with limited computational resources. Simulation and experimental results on a real processing pattern are shown for a scaled laboratory demonstration machine.

IEEE Transactions on Control Systems Technology

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Cascaded Reference Governor–MPC for Motion Control of Two-Stage Manufacturing Machines

S. Di Cairano *Senior Member, IEEE,*, A. Goldsmith *Member, IEEE,*, U.V. Kalabić *Member, IEEE,*, S.A. Bortoff *Senior Member, IEEE,*

***Abstract*—Two-stage processing machines for manufacturing are equipped with two sets of actuators for motion control with different operating ranges and bandwidths. The processing time, product quality and flexibility of the manufacturing process can be optimized by coordinating these actuators and exploiting the entire actuator operating range. We propose a motion control strategy for two-stage processing machines based on model predictive control (MPC). By exploiting timescale separation, we formulate the problem as a single-stage motion control with reference-dependent constraints. Feasibility of the MPC problem is guaranteed by using a reference governor that adjusts the feed rate. The proposed method guarantees correct processing while satisfying the actuators' range and dynamics constraints, finite time processing of a given spatial pattern and real-time execution even with limited computational resources. Simulation and experimental results on a real processing pattern are shown for a scaled laboratory demonstration machine.**

***Index Terms*—Precision Manufacturing, Motion Control, Model predictive control, Constrained Control**

I. Introduction

Precision manufacturing is a relevant application area of control systems [1], [2], due to the need for high quality processing of the machined parts and for high throughput to reduce manufacturing costs. High throughput requires high speed processing, while high quality requires high precision, and a trade-off must be achieved between these conflicting requirements by hardware and software design.

Manufacturing machines are composed of a worktool capable of performing operations such as milling, cutting, welding or drilling on a piece of raw material, and of a motion control system that moves the worktool. The worktool may have a considerably large mass, also due to the size of the support structure, and it may need to move over large ranges due to processing large parts or multiple parts in a single block of raw material. Moving a large mass over a long range with high accelerations and high precision results in conflicting objectives. The motion control system aims at achieving a desirable trade-off between such objectives.

Controlling a processing machine usually involves actuating the worktool to follow a geometric path that represents the machining pattern of the part being produced. The common performance requirements are the time to process the path, the precision of the obtained machined path, which is also affected by the vibrations induced by the machine moving parts, and the energy consumed during processing. Due to the high rates of operation and relatively low computational power of the microprocessors used in factory automation, the control methods used in manufacturing have been mostly limited to PID and loop shaping filters in the servomotors. In fact, for common designs, the axes are dynamically decoupled and each axis of the single-stage processing machines can be considered single-input single-output system. thus, the worktool path is computed off-line, based on the geometry of the parts to be machined [3], [4], and the worktool velocity, i.e., the path feed rate, is scheduled based on the desired precision. The obtained trajectories are then used in "open-loop", i.e., without modifications during execution, relying on the feedback loops of the servomotors to reject disturbances and to compensate for uncertainty. For repeated operation, supervisory strategies such as iterative learning control have been suggested in [2]. A real-time trajectory generation may be advantageous, for instance by allowing modifications to the processed part between different runs, or even within a run, without the need to stop the processing.

In order to increase the speed of processing and the precision over a large workspace, the motion control system of high performance manufacturing machines may exploit multiple actuation stages per processing axis, where actuators with different bandwidths and operating ranges are combined to process large workpieces at high rate. Two-stage actuation has been applied in different fields ([5]–[7]). In particular, in a two-stage machine each axis is equipped with: a "slow" stage with large operating range but small bandwidth and acceleration limits, and a "fast" stage with large bandwidth and acceleration limits but small operating range (see Figure 1). For each axis, the overall position of the worktool is the sum of the positions of the two stages. With this architecture the machine can rapidly process small features of the workpiece by actuating the fast stage, and still be able to process large features by superimposing the motion of the slow stage.

Trajectory generation and control for two-stage machines is more complicated than for single stage ones. Each axis of a two-stage machine is effectively modeled as a multiple input–single output system [8] subject to constraints on velocity, acceleration, and operating range. Classical methods based on frequency separation [5], [9], [10] may be significantly suboptimal and may require convoluted ad-hoc logic and iterations to handle the constraints, thus limiting the applicability for online trajectory generation. Instead, model predictive control (MPC) [11] has proven effective for controlling multivariable systems subject to constraints, and its application domain has been extended in the last few years from process and

The authors are with Mitsubishi Electric Research Laboratories, Cambridge, MA, email: dicairano,goldsmith,kalabic,bortoff@merl.com

chemical control to systems with faster dynamics and reduced computational resources, such as in automotive, aerospace, and mechatronics [12], [13]. Recently some MPC applications in precision manufacturing have been developed. In particular, [14]–[16] (see also references therein) propose path following MPC algorithms, and their applications to contouring control, that operate in the spatial domain, and hence require (explicitly or implicitly) the linearization of the spatial nonlinear dynamics. These approaches have been developed for single-stage machines where the algorithms operate on a linearized (i.e., approximate) model along a path known in advance, and the machining error is a performance measure, and hence not subjects to hard bounds.

However, application of path following MPC to two stage processing machines is not straightforward. The reasons include the timescale separation of the stages (up to 100x), the high control rates of the fast stage, (up to 200kHz), the accuracy being not a performance metric, but rather a constraint to be enforced on all points of the path, on the order of hundreds of thousands, and the lack of an a-priori defined path for linearization, as the system trajectory is not uniquely defined due to the overactuation. In addition, for real-time control, a simple algorithm implementation is needed to facilitate deployment in low computational power processors.

In this paper[1] we propose a strategy for controlling two-stage processing machines equipped with a small-range fast actuator (fast stage) and a large-range slow actuator (slow stage) per axis. An example of such machine is described and modeled in Section II. Rather than controlling both actuators in the same loop, in Section II we exploit the timescale separation between the stages to re-formulate the machine control problem into controlling the slow stage by a tracking MPC with constraints that depend on the reference trajectory, which is generated for an"ideal" single stage, that has the range of the slow stage and the bandwidth of the fast stage.

Due to the dependency of the constraints on the reference, the reference trajectory may need to be modified for ensuring feasibility of MPC, which is a recently studied problem, see, e.g., [18], [19]. Unfortunately, such methods cannot be directly applied to the problem considered here because the modification to the setpoint will in general cause a modification to the spatial path, which results in an incorrectly machined part. As described in Section III, we modify online the upcoming segment of the ideal trajectory to be feasible and to ensure future constraint satisfaction, in a way that maintains the spatial shape of the pattern to be processed by using a particular reference governor [20]. The modified trajectory is provided to an appropriately designed MPC, which is recursively feasible and achieves finite-time processing of the machining path.

As discussed in Section IV, a benefit of the proposed approach is that it only requires the evaluation of linear inequalities for the reference governor, and the solution of a convex quadratic program for MPC, which can be achieved by low complexity solvers. Furthermore, the control architecture allows for real-time operation where a recursively feasible solution is found within a pre-assigned computing time.

[1]Preliminary studies related to this research appeared in [17].

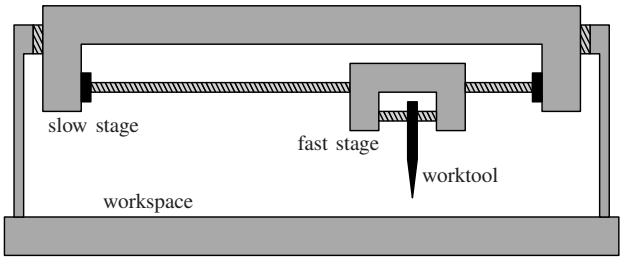

Fig. 1. Schematic of the front view of a two-stage machine.

In Section V we evaluate the algorithm on a laboratory experiment built with standard off-the-shelf components with scaled slow stage with respect to existing processing machines. We draw our conclusions in Section VI hinting at the effectiveness of MPC in manufacturing applications.

Notation: $\mathbb{R}$, $\mathbb{R}_{0+}$, $\mathbb{R}_+$ and $\mathbb{Z}$, $\mathbb{Z}_{0+}$, $\mathbb{Z}_+$ are the sets of real, nonnegative real, positive real, and integer, nonnegative integer, positive integer numbers, and we use notations like $\mathbb{Z}_{[a,b)} = \{z \in \mathbb{Z} : a \leq z < b\}$ to denote intervals. By $[a]_i$ we denote the i-th component of a, for $a \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $(a, b) = [a' \ b']' \in \mathbb{R}^{n+m}$ is the stacked vector, and I, $\mathbf{1}$, and 0 are the identity, the all-1, and the zero matrices of appropriate size, respectively. Relational operators between vectors are intended componentwise, while for matrices denote (semi)definiteness. Given a set $\mathcal{A}$ and $(a, b) \in \mathcal{A}$ we denote by $\mathcal{A}(b)$ the section of $\mathcal{A}$ in the coordinates of b at the values of b. We denote the Minkowski set sum by $\oplus$. $\mathcal{B}(\rho)$ denotes the norm-ball of radius $\rho \in \mathbb{R}_+$ centered at the origin. For a discrete-time signal $x \in \mathbb{R}^n$ with sampling period T, x_t is the value at sampling instant t, i.e., at time Tt, and $x_{k|t}$ denotes the predicted value of x at sample $t + k$, i.e., x_{t+k}, based on data at sample t, where $x_{0|t} = x_t$. We denote the convolution operator by $*$, and, with a little abuse of notation, $y(t) = G(t) * u(t) = \int_0^t G(t - \tau)u(\tau)\mathrm{d}\tau$.

II. Architecture, Model and Control Problem

We consider a two-stage dual-axis (i.e., 2D) processing machine as in the schematics in Figure 1. The objective of the machine is to process blocks of raw materials at a high rate and with high precision to manufacture parts that have small and large features. The two-stage machine aims at resolving some of the conflicting requirements in precision manufacturing. Due to small features, e.g., less than a millimeter, and the requirement to process at high rate, the worktool may need to sustain large accelerations, on the order of several g. However, due to large features, e.g., more than a meter, and possibly multiple parts being produced from a single block of raw material, the worktool must have a large operating range and as a consequence a large mass, up to several hundred kilograms. Large accelerations and large operating range (i.e., large mass) are obviously in conflict. Thus, two-stage machines combine slow stages and fast stages, see Figure 1, that are implemented by different actuators, such as motors, piezoelectric actuators, electromagnetic actuators, in closed-loop with their servocontrollers. The fast stages have smaller operating range, and as a consequence are smaller and less massive, and thus

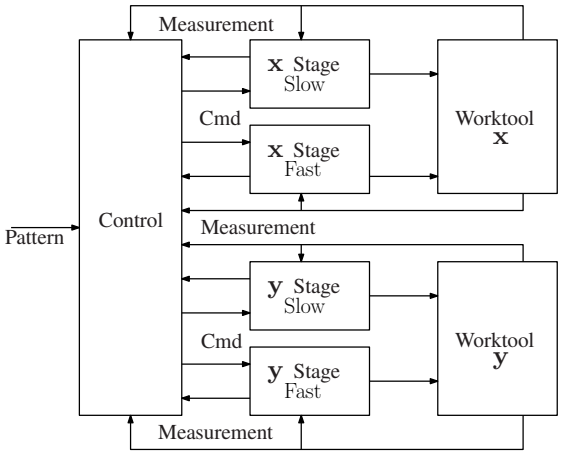

Fig. 2. Control architecture of a two-stage dual-axis processing machine.

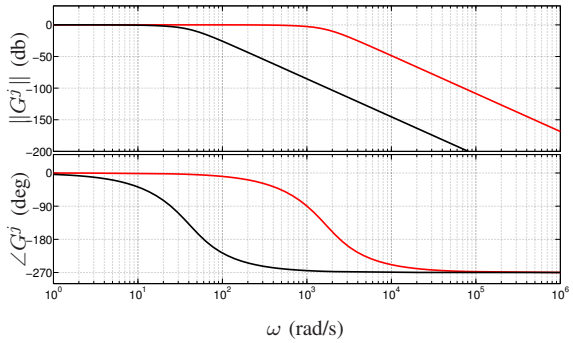

Fig. 3. Bode plot of the frequency responses of the slow stage (black) and fast stage (red), for both $\mathbf{x}$ and $\mathbf{y}$ axes, showing a 50x bandwidth difference.

can achieve larger accelerations. On the other hand the slow stages have larger operating ranges, and as a consequence, are more massive, and can achieve smaller accelerations. The rationale for such an architecture is that the small features of the machined part can be processed by high-acceleration movements of the fast stage, while large features can still be processed by superimposing the large range movements of the slow stage to the fast stage movements.

The objective of trajectory generation and control for a two-stage machine is to coordinate the motion of the slow and fast stages for each processing axis, to achieve the desired processing pattern as fast as possible within the given precision specifications. Additional secondary objectives are to reduce the acceleration for the massive slow stage, thus reducing power consumption and the vibrations induced by the forces to accelerate the large mass.

A. Motion models of two-stage axes

A schematic of the control architecture of the machine considered here is shown in Figure 2, where the control algorithm generates commands for the slow stage and the fast stage of two processing axes $(\mathbf{x}, \mathbf{y})$. The commands are then provided to the corresponding stage actuators, which follow them using conventional feedback loops. The combined effects of the slow stage result in the worktool achieving the desired processing pattern.

The motion model of the stage in closed-loop with the servocontroller is

$$y_j^i(t) = G_j^i(t) * u_j^i(t),\ j \in \{s, f\},\ i \in \{\mathbf{x}, \mathbf{y}\}, \qquad (1)$$

where y is the position, u is the position command, G is the transfer function with unitary dc-gain, $j \in \{s, f\}$ is the index of the stage (slow/fast) and $i \in \{\mathbf{x}, \mathbf{y}\}$ is the index of the processing axis (**x**/**y**).

For the considered architecture, the position of the worktool is the sum of the positions of the two stages in each axis,

$$\begin{aligned} y^i(t) &= y_f^i(t) + y_s^i(t) \\ &= G_f^i(t) * u_f^i(t) + G_s^i(t) * u_s^i(t),\ i \in \{\mathbf{x}, \mathbf{y}\}. \end{aligned} \qquad (2)$$

The stages are subject to constraints due to physical limitations and operating requirements. The stage operating ranges are constrained due to the actuator mechanical limitations,

$$-\overline{y}_j^i \leq \quad y_j^i \quad \leq \overline{y}_j^i, \qquad (3)$$

where $\overline{y}_j^i,\ j \in \{s, f\},\ i \in \{\mathbf{x}, \mathbf{y}\}$ determine (symmetric) upper and lower bounds. Velocities and accelerations are also constrained

$$-\overline{\dot{y}}_j^i \leq \quad \dot{y}_j^i \quad \leq \overline{\dot{y}}_j^i \qquad (4)$$

$$-\overline{\ddot{y}}_j^i \leq \quad \ddot{y}_j^i \quad \leq \overline{\ddot{y}}_j^i, \qquad (5)$$

where $\overline{\dot{y}}_j^i,\ \overline{\ddot{y}}_j^i,\ j \in \{s, f\},\ i \in \{\mathbf{x}, \mathbf{y}\}$.

The differences between the slow and fast stages reside in the bandwidth of the transfer functions (1), and in the constraints (3), (5). The fast stages have much larger bandwidths, due to the reduced actuation range,

$$\mathrm{BW}_3(G_f^i) \gg \mathrm{BW}_3(G_s^i),\ i \in \{\mathbf{x}, \mathbf{y}\},$$

where $\mathrm{BW}_3(\cdot)$ denotes the 3-db bandwidth. The slow and fast stage frequency responses for the machine considered here are showed in Figure 3, where $\mathrm{BW}_3(G_f^i) = 50\mathrm{BW}_3(G_s^i),\ i \in \{\mathbf{x}, \mathbf{y}\}$. The fast stage range is much smaller than that of the slow stage, but it allows for much larger accelerations,

$$\overline{y}_f^i \ll \overline{y}_s^i, \qquad \overline{\ddot{y}}_f^i \gg \overline{\ddot{y}}_s^i, i \in \{\mathbf{x}, \mathbf{y}\},$$

Due to the different bandwidths, the sampling periods for the discrete-time servocontrollers of the fast and slow stage, and hence for the models in (1), are different, i.e., $T_f \ll T_s$. Specifically here, $T_s = M \cdot T_f$, where $M \in \mathbb{Z}_+$ and $M \gg 1$.

B. Tracking control problem definition

The objective for the control system in Figure 2 is to make the worktool follow a geometric path representing the pattern to be processed within a desired spatial accuracy $\varepsilon \in \mathbb{R}_+$, e.g., on the order of microns, while satisfying all the constraints, and can be formulated as follows.

Problem 1: Given (2) subject to (3), (4), (5), and the regular geometric curve $p(\sigma) = [p^{\mathbf{x}}(\sigma)\ p^{\mathbf{y}}(\sigma)]'$, where $\sigma \in \mathbb{R}_{[0,1]}$ is the (normalized) path parameter [15] and $\varepsilon \in \mathbb{R}_+$ is the

accuracy tolerance, control the commands u^i_j, $i \in \{\mathbf{x}, \mathbf{y}\}$, $j \in \{s, f\}$ such that

$$\|(y^{\mathbf{x}}(\sigma), y^{\mathbf{y}}(\sigma)) - (p^{\mathbf{x}}(\sigma), p^{\mathbf{y}}(\sigma))\|_\infty \leq \varepsilon, \ \forall \sigma \in \mathbb{R}_{[0,1]}. \quad (6)$$

A holistic approach for solving Problem 1 is to formulate a path following MPC, (see, e.g., [14], [16], [21] and references therein). However, in the spatial domain the dynamics and the constraints are nonlinear, and hence the controller has to solve a non-convex optimization problem at every iteration. Thus, the command calculation in the path following approaches in [**?**], [16], [21], [22] is based on the repeated linearization or approximation of the nonlinear problem along the path, which is reasonable when the tracking error is a performance variable, i.e., there is no bound that the error must satisfy. On the other hand, due to the linearization and the usage in the cost function, a bound on the tracking error is difficult to guaranteed. In fact, [14], [16], [21] do not impose tracking error constraints which is however the case here due to (6). Besides this, there are others complicating factors in two-stage machines. The timescale separation of the stages imposes a prediction horizon related to slow stage bandwidth, and a sampling period based on the fast stage bandwidth, thus resulting in a large optimization problem with poor numerical conditioning. Furthermore, since the machine has multiple stages, the stage trajectory is not uniquely determined by the processing pattern, and hence it is not possible to determine a "good" nominal trajectory for linearization. Thus, a significant linearization error may occur, causing the violation of (6). For instance the path following MPC in [14] assumes the prediction model to be square and accordingly reduces the degrees of freedom in the application, while multi-stage machines have more inputs than outputs.

Instead, we formulate a modified problem whose solution also solves Problem 1 as described next. First, we consider the "ideal" single stage machine that has accuracy ε, the range of the slow stage and the dynamics, velocity, and acceleration limits of the fast stage. Being this a single stage machine, we determine a path feed rate by standard methods resulting in the trajectory as the sequence of points

$$\{q(hT_f)\}_h = \{(q^{\mathbf{x}}(hT_f), q^{\mathbf{y}}(hT_f))\}_h, \ h \in \mathbb{Z}_{[0,\bar{h}]}, \quad (7)$$

where h is the point index and $\bar{h} \in \mathbb{R}_+$ is the total number of points. The trajectory in (7) is such that $\tilde{y}^i(t) = G^i_f(t) * q^i(t)$, $i \in \{\mathbf{x}, \mathbf{y}\}$ is feasible for the ideal machine that has all the strengths of the two stages and none of the weaknesses, i.e., $\{q(hT_f)\}_h$ satisfies (5) for $j = f$, (3) for $j = s$, and (6).

In addition to those already enforced by (7), the machine needs only to satisfy the actual constraint (3) for $j = f$ by a motion of the slow stage that satisfies (3), (4), (5), for $j = s$. To this end we control (1), (3), (4), (5), for $j = s$ such that

$$-\overline{y}^i_f \leq y^i_s(t) - q^i(t) \leq \overline{y}^i_f, \ i \in \{\mathbf{x}, \mathbf{y}\}. \quad (8)$$

In an MPC context, this amounts to solving in receding horizon with sampling period T_s

$$\min_{U^i_{st}} \quad F(x^i_{N|t}, q^i_{N|t}) + \sum_{k=0}^{N-1} L(x^i_{sk|t}, q^i_{k|t}, u^i_{sk|t}) \quad (9a)$$

$$\text{s.t.} \quad x^i_{sk+1|t} = f^i_s(x^i_{sk|t}, u^i_{sk|t}) \quad (9b)$$

$$(3), \ (4), \ (5), \quad j = s \quad (9c)$$

$$-\overline{y}^i_f \leq y^i_{sk|t} - q^i_{k|t} \leq \overline{y}^i_f \quad (9d)$$

$$\mathcal{H}(x_{k|t}, q_{k|t}, u_{k|t}) \leq 0 \quad (9e)$$

for $i \in \{\mathbf{x}, \mathbf{y}\}$, where x^i_s, is the state that includes at least stage position, velocity and acceleration, f_s is the state update equation representing (1) for $j = s$, $N \in \mathbb{Z}_{0+}$ is the prediction horizon, $U^i_{st} = [u^i_{s0|t} \dots u^i_{sN-1|t}]$, F, L are the terminal and stage cost, respectively, $\mathcal{H}$ models additional constraints, and perfect preview of the reference q for N steps is available.

A complicating feature of (9) is that constraint (9e) depends on the reference trajectory. Thus, feasibility of (9) depends on an exogenous variable and cannot be guaranteed for an unmodifiable reference. Even if (9) is feasible at time t, there is no guarantee of recursive feasibility, i.e., that (9) is feasible for $\tau > t$. However, since the timing along the path is not fixed, there is some freedom in modifying the reference as long as such modifications do not change the spatial pattern. For instance, changing the feed rate, the rate at which the points of the path are processed, does not change the spatial path. Thus, we formulate the following alternative to Problem 1.

Problem 2: Given $\{q(hT_f)\}_h$, $h \in \mathbb{Z}_{[0,\bar{h}]}$, that satisfies (4), (5) for $j = f$, (3) for $j = s$, and such that $\tilde{y}^i(t) = G^i_f(t) * q^i(t)$ satisfies (6), design F, L, $\mathcal{H}$ in (9) and construct $\{r(tT_s)\}_t = \{(r^{\mathbf{x}}(tT_f), r^{\mathbf{y}}(tT_s))\}_t$ such that $y^i(t) = G^i_f(t) * r^i(t)$ satisfies (6), and (9) is strictly convex and recursively feasible when q is substituted by r. Also, when $\{q(hT_f)\}_{h=0}^{\bar{h}}$ is finite, the processing time is finite.

Problem 2 involves the simultaneous modification of the reference and generation of the command to track such a modified reference, and has attracted interest in recent years, see, e.g., [18], [19]. Unfortunately, the previously proposed methods modify the reference within the MPC problem in ways that do not guarantee that the geometry of the path is maintained, and hence (6) may be violated. Instead, we solve Problem 1 by cascading a tracking MPC to a "spatial" reference governor (SPRG), which does not change the spatial shape of the reference and hence does not affect the satisfaction of (6). In practice, the effect of the SPRG is to change the feed rate to ensure that the constraints can be satisfied. It is worth noting that the path tracking error of (1) is inversely proportional to the feed rate. Hence, if $\{q(hT_f)\}_h$ satisfies (6) and the feed rate is only reduced to generate $\{r(tT_s)\}_t$, then also the latter satisfies (6).

III. SPRG-MPC FOR TWO-STAGE PROCESSING MACHINES

We propose a control system design for solving Problem 2, and consequently Problem 1, with the block diagram shown in Figure 4. Using conventional methods, we generate offline a reference trajectory for an ideal single stage machine with dynamics, velocity and acceleration constraints of the fast stage, and the range of the slow stage, thus obtaining an

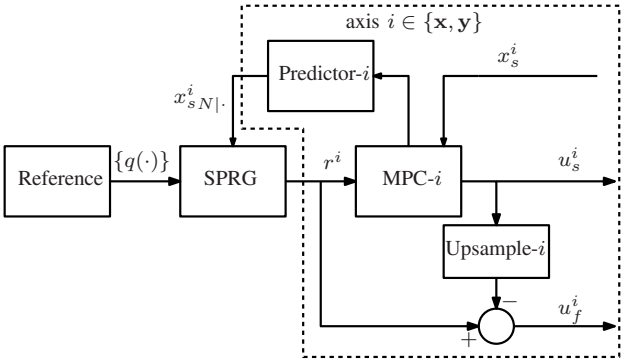

Fig. 4. Trajectory generation and control block diagram for the two-stage processing machine.

idealized, usually infeasible, trajectory. The spatial SPRG developed in Section III-A modifies the reference to obtain a feasible reference that does not change the spatial processing pattern. The feasible reference is provided to a MPC algorithm developed in Section III-B that optimizes the trajectory with guaranteed recursive feasibility. The MPC feeds the command to the slow stage actuator while the fast stage actuator commands are obtained as the difference between the slow stage trajectory and the feasible reference, with the implementation discussed in Section IV-C. The predicted terminal state is provided to the SPRG for continuing operation.

A. Spatial Reference Governor

In order to obtain a reference trajectory that guarantees the feasibility of the MPC problem (9) and ensures satisfaction of (6) we develop a reference governor that operates on the spatial points $\{q(h)\}_h$, represented by the SPRG block in Figure 4. First, we recall few useful notions.

Definition 1: Given $x(t+1) = f(x(t))$, $x \in \mathbb{R}^n$, with $z = h(x(t))$, $z \in \mathbb{R}^{n_z}$, such that $z \in \mathcal{Z} \subset \mathbb{R}^{n_z}$, a constraint admissible set $\mathcal{S}_\infty \subset \mathbb{R}^n$ is a set such that

$$x(t) \in \mathcal{S}_\infty \Rightarrow h(x(\tau)) \in \mathcal{Z},\ \forall \tau \geq t. \tag{10}$$

Given any constraint admissible set $\mathcal{S}_\infty$, $x \in \mathcal{S}_\infty$ implies that $f(x) \in \mathcal{S}_\infty$, i.e., $\mathcal{S}_\infty$ is positive invariant (PI) for $x(t+1) = f(x(t))$. The maximal constraint admissible set $\mathcal{O}_\infty$ is the union of all constraint admissible sets. □

Let the discrete-time state-space formulation of the slow stage motion, i.e., (1) for $j = s$, sampled with period T_s be

$$x^i(t+1) = A^i x^i(t) + B^i u_s^i(t),\ i \in \{\mathbf{x}, \mathbf{y}\} \tag{11}$$

where y_s^i, $\dot{y}_s^i$, $\ddot{y}_s^i$ are components of the state vector x , and define $z^i = C_x^i x^i + C_r r^i = [y_s^i\ \dot{y}_s^i\ \ddot{y}_s^i\ y_s^i - r^i]'$. Thus, (11) with $u_s^i(t) = r^i(t)$ and constant reference dynamics, and subject to (3), (4), (5), for $j = s$, and (8), results in

$$\begin{aligned} x^i(t+1) &= A^i x^i(t) + B^i r^i(t) && \text{(12a)}\\ r^i(t+1) &= r^i(t) && \text{(12b)}\\ z^i(t) &= C_x^i x^i(t) + C_r^i r^i(t) && \text{(12c)}\\ & H_z^i z^i(t) \leq K^i. && \text{(12d)} \end{aligned}$$

Given r^i, let $x_e^i(r^i)$ denote the corresponding steady state equilibrium. We use the following result (see, e.g., [20]).

Result 1: For $i \in \{\mathbf{x}, \mathbf{y}\}$, consider (12) where $C_z^i = [C_x^i\ C_r^i]$, $A_z^i = \left[\begin{smallmatrix} A^i & 0 \\ 0 & 1 \end{smallmatrix}\right]$, (C_z^i, A_z^i) is observable and $\mathcal{Z}^i = \{z : H_z^i z^i \leq K^i\}$ is a polytope, i.e., closed and bounded. Let $\mathcal{R}^i$ be the set of commands that are steady state admissible, that is, for all $r^i \in \mathcal{R}^i$ the corresponding equilibria $x_e^i(r^i)$ satisfy $C_x x_e^i(r^i) + C_r r^i \in \text{int}(\mathcal{Z}^i)$. Then, with arbitrary precision, the maximum constraint admissible set for $r \in \mathcal{R}^i$ is the polytope

$$\mathcal{O}_\infty^i = \{(x^i, r^i) : H_{x\infty}^i x^i + H_{r\infty}^i r^i \leq K_\infty^i\}. \tag{13}$$

defined by a finite number of constraints . ■

$\mathcal{O}_\infty^i$ can be computed by solving offline a sequence of linear programs [20]. We denote the section of $\mathcal{O}_\infty^i$ at reference value r^i by $\mathcal{O}_\infty^i(r^i)$, which is the set of states that are in the maximum constraint admissible set when the reference is r^i, i.e., $x^i \in \mathcal{O}_\infty(r^i)$ iff $(x_i, r_i) \in \mathcal{O}_\infty^i$.

Given x^i, $i \in \{\mathbf{x}, \mathbf{y}\}$, the SPRG uses Result 1 to choose r^i in the sequence of points to be processed, $\{q(hT_f)\}_h$. Let

$$\begin{aligned} &\kappa(x, \mu, \{q(h)\}_h) = \\ &\qquad \max_{\eta \in \mathbb{Z}_{[0,M]}} \quad \eta && \text{(14a)}\\ &\qquad \text{s.t.} \quad (x^i, q^i(\mu+\eta)) \in \mathcal{O}_\infty^i && \text{(14b)}\\ &\qquad\quad \min_{[0,\eta]} q^i(\mu+\eta) \geq [C_x^i x^i]_1 - \overline{y}_f^i && \text{(14c)}\\ &\qquad\quad \max_{[0,\eta]} q^i(\mu+\eta) \leq [C_x^i x^i]_1 + \overline{y}_f^i && \text{(14d)}\\ &\qquad\quad i \in \{\mathbf{x}, \mathbf{y}\}, \end{aligned}$$

and let $\mu(t) \in \mathbb{Z}_{0+}$ be the index of the last processed point within the t^{th} sampling interval, i.e., $r(t) = q(\mu(t))$, then

$$\begin{aligned} \mu(t) &= \kappa(x(t), \mu(t-1), \{q(h)\}_h) && \text{(15a)}\\ r^i(t) &= q^i(\mu(t)),\ i \in \{\mathbf{x}, \mathbf{y}\}. && \text{(15b)} \end{aligned}$$

The SPRG (15) constructs the reference by solving (14), searching how many points can be processed until the next sampling period without violating the constraints, and ensuring that the selected reference can be maintained as a target in the future without violating the constraints. Since constraint satisfaction is ensured only at the sampling instants, (14c), (14d) are included to enforce that all the points $\{q(\mu+j)\}_{j=0}^{\eta}$ to be processed by the fast stage during the sampling interval are within the fast stage range. Instead of enforcing $\|y_s^i - q(\mu+j)\| \leq y_f$ for all $j \in \mathbb{Z}_{[0,\eta]}$, (14c), (14d) are enforced on the worst case values to reduce the computations. Let $\underline{q}^i(t) = \min_{\mu \in [\mu(t-1), \mu(t))]} q^i(\mu)$, $\overline{q}^i(t) = \max_{\mu \in [\mu(t-1), \mu(t))]} q^i(\mu)$ be the worst case intersampling bounds that are fixed once the reference has been selected, since they depend only on the points that are processed in the current sampling interval. While (14) seems a difficult problem, it is solved by checking the inequalities while scanning the points in $\{q(h)\}_h$, and, in doing that, the computation of $\overline{q}^i$ and $\underline{q}^i$ is straightforward.

By using (15), the feasible reference is such that $r(t) \in \{q(h)\}_h$ for all $t \in \mathbb{Z}_{0+}$, and hence the spatial pattern is not deformed, but its traversal is slowed down due by reducing the feed rate. The maximum of M points in (14) is imposed due to the maximum number of points that can be processed by the

fast stage in a single sampling period of the slow stage. This is needed for satisfying (6), since starting from a trajectory $\{q(h)\}_h$ that enforces (6), the trajectory $\{r(t)\}_t$ should not be faster than $\{q(hT_f)\}_h$ to prevent the fast stage tracking error from increasing. Hence, a maximum of M points of $\{q(h)\}_h$ can be processed during a each slow stage sampling period, which is the ratio between T_s and T_f.

Theorem 1: Let $\{q(hT_f)\}_{h=0}^{\bar{h}}$ be a finite-time trajectory such that for all $h \in \mathbb{Z}_{[0,\bar{h}]}$, $(x_e^i(q^i(h)), q^i(h+1)) \in \text{int}(\mathcal{O}_\infty^i)$, and let $x(0)$ be such that $(x^i(0), q^i(0)) \in \mathcal{O}_\infty^i$, $i \in \{\mathbf{x}, \mathbf{y}\}$. For (12) in closed-loop with (15), (14) is recursively feasible, (12d) is satisfied and there exists $\bar{t} \in \mathbb{Z}_+$ such that $r(\bar{t}T_s) = q(\bar{h}T_f)$. □

For this paper, all the proofs are in the Appendix.

Remark 1: The reference governor (14), (15), preserves the geometry, i.e., (6) by: (i) selecting only points on the trajectory $\{q(h)\}_h$ rather than selecting points close to $\{q(h)\}_h$ but not necessarily on it, and (ii) computing the reference for both axis at the same time, even if the dynamics of the axes are decoupled, allowing the geometry of the path to couple them. The operation of the reference governor amounts to a (nonlinear) transformation on the curve parameter, i.e., in case of $\{q(h)\}_h$, the time, which does not change the curve. Thus, the reference governor "stretches" the time, to reduce the processing speed when needed to guarantee constraint satisfaction.

B. Tracking MPC for two-stage machines

Next, we construct the MPC block in Figure 4 by appropriately specifying the optimization problem (9) to operate with the SPRG. For the ease of notation, in what follows we omit the superscript $i \in \{\mathbf{x}, \mathbf{y}\}$, when clear from context, and the subscript s, since all the variables refer to the slow stage.

Given the reference trajectory $R_t = [r_{0|t} \dots r_{N|t}]$, and the corresponding intersampling worst case bounds $W_t^{\min} = [\underline{q}_{0|t} \cdots \underline{q}_{N|t}]$, $W_t^{\max} = [\overline{q}_{0|t} \dots \overline{q}_{N|t}]$, the MPC finite horizon optimal control problem is

$$\mathcal{V}(x(t)) = \tag{16a}$$

$$\min_{U_t} \quad F(x_{N|t}, r_{N|t}) + \sum_{k=0}^{N-1} L(x_{k|t}, r_{k|t}, u_{k|t}) \tag{16b}$$

$$\text{s.t.} \quad x_{k+1|t} = Ax_{k|t} + Bu_{k|t} \tag{16c}$$

$$HC_x x_{k|t} + HC_q r_{k|t} \le K \tag{16d}$$

$$[C_x x_{k|t}]_1 \ge \overline{q}_{k|t} - \overline{y}_f \tag{16e}$$

$$[C_x x_{k|t}]_1 \le \underline{q}_{k|t} + \overline{y}_f \tag{16f}$$

$$x_{N|t} \in \mathcal{O}_\infty(r_{N|t}) \tag{16g}$$

$$\bar{x}_{0|t} = \bar{x}(t). \tag{16h}$$

where $F(x,r) \ge 0$ for all x, r, and $F(x,r) = 0$ iff $x = (x_e(r), r)$, i.e., at the equilibrium for r where $y = r$, $L(x,r,u) \ge 0$ for all x, r, u, and $L(x,r,u) = 0$ iff $\bar{x} = x_e(r)$ and $u = r$, since the dc-gain is 1, $U_t = [u_{k|t} \dots u_{N-1|t}]$ is the command sequence to be optimized at time t and we denote the optimal command sequence at time t by U_t^*.

Next we prove some properties for the control strategy shown in Figure 4 where: (i) the SPRG implemented by (15) computes a new reference step, together with its worst case intersampling bounds, from the previous prediction of the MPC terminal state, (ii) the MPC shifts the previous reference by one time step and appends the new reference step provided by SPRG, (iii) solves (16), and (iv) provides the new terminal state to the SPRG.

Theorem 2: Consider the MPC that at any time $t \in \mathbb{Z}_{0+}$ solves (16), where $r_{k|t} = r_{k+1|t-1}$, $k \in \mathbb{Z}_{[0,N-1]}$, $r_{N|t} = q(\mu_{N|t})$, and $\mu_{N|t} = \kappa(x_{N|t-1}, \mu_{N|t-1}, \{q(h)\}_h)$. Let (16) be feasible at time $t \in \mathbb{Z}_{0+}$, then (16) is feasible for all $\tau \ge t$. □

Theorem 3: Let $\{q(hT_f)\}_{h=0}^{\bar{h}}$ be a finite-time trajectory such that for all h the equilibrium $x_e(q(h))$ for $q(h)$ satisfies $(x_e(q(h)), q(h+1)) \in \text{int}(\mathcal{O}_\infty)$, and let $(x(0), q(0)) \in \mathcal{O}_\infty$. Consider the MPC that at every iteration solves (16) where

$$r_{k|t} = r_{k+1|t-1}, \ k \in \mathbb{Z}_{[0,N-1]}, \quad r_{N|t} = q(\mu_{N|t}),$$

$\mu_{N|t} = \kappa(x_{N|t-1}, \mu_{N|t-1}, \{q(h)\}_h)$. Assume that

$$\forall (x,r) \in \mathcal{O}_\infty, \ \exists u \in \mathbb{R} : \\ F(Ax + Bu, r) + L(x, r, u) - F(x, r) \le 0. \tag{17}$$

Then, there exists a finite $t \in \mathbb{Z}_{0+}$ when $r_{N|t} = q(\bar{h})$ and $(x_{N|t}, r_{N|t}) \in \mathcal{O}_\infty$. □

The assumption in (17) is related to the existence of a control Lyapunov function [11] for the tracking problem. Since the SPRG keeps the reference constant until it can move to the next point and the prediction model is asymptotically stable, for linear-quadratic MPC a simple choice for F is based on solving the Lyapunov equation with respect to the origin as steady state equilibrium, and use such solution to construct F as function of x and r by translation of steady state state and input. Feasibility of this choice is ensured by using $\mathcal{O}_\infty$ as terminal set in (16).

IV. Control System Implementation

In this section we describe the implementation of the numerical algorithms for the control system designed in Section III, in particular the computation of the SPRG and the numerical solver for the MPC optimal control problem (16). Due to the real-time nature of the control system, the numerical algorithms must terminate in a given fixed time, which requires additional considerations. Finally, we briefly discuss the command generation for the fast stage .

A. Implementation of SPRG and MPC

The computation of the feasible reference by SPRG requires solving (14). Since r^i is a scalar the easiest way to solve (14) is by direct evaluation. Thus, solving (14) amounts to the following steps: (i) substitute x^i in the constraints, eliminate redundant constraints and set $\eta = M$; (ii) compute $\overline{q}$, $\underline{q}$ from μ and η; (iii) if $q(\mu + \eta)$ satisfies the constraints, both the maximum constraint admissible set and the worst case intersampling bounds, then $r = q(\mu + \eta)$, and terminate; otherwise $\eta = \eta - 1$ and go to (ii).

It can be noted that x^i is given, and hence constraint (14b) can be transformed into $H_{r_\infty}^i r^i \le K_\infty^i - H_{x_\infty}^i x^i$, which, after minor manipulations, results in all but two constraints to

be redundant, since r^i is scalar. Thus, a total of 4 constraints have to be verified: two for the maximum constraint admissible set and two for the worst case intersampling bounds. Finally, instead of the decremental search, different updates for η can be applied, such as bi-section. Decremental search is particularly favorable for the situations when it is expected η to be very close to its maximum value, i.e., the feed rate is only minimally reduced.

As for the MPC implementation, they key element is the solver for (16). When F, L are convex quadratic functions,

$$\begin{aligned} L(x,r,v) &= \begin{bmatrix} x' & r' \end{bmatrix} Q \begin{bmatrix} x \\ r \end{bmatrix} + \begin{bmatrix} u' & r' \end{bmatrix} R \begin{bmatrix} u \\ r \end{bmatrix} \\ &= \varrho_p (y-r)^2 + \varrho_v \dot{y}^2 + \varrho_a \ddot{y}^2 + \varrho_u (u-r)^2 \\ F(x,r) &= \begin{bmatrix} x' & r' \end{bmatrix} P \begin{bmatrix} x \\ r \end{bmatrix}, \end{aligned} \tag{18}$$

where ϱ_p, ϱ_v ϱ_a, ϱ_u are positive weights on slow stage position, velocity, acceleration and command, and $P \geq 0$ is designed to satisfy the assumptions in Theorem 3, (16) results in a convex parametric quadratic program

$$\min_{\zeta} \quad \frac{1}{2}\zeta' Q_p \zeta + \theta' C_p' \zeta + \frac{1}{2}\theta' \Omega_p \theta \tag{19a}$$

$$\text{s.t.} \quad G_p \zeta \leq S_p \theta + W_p \ . \tag{19b}$$

where at time t, $\zeta = U_t$ and $\theta \in \mathbb{R}^{n_\theta}$ is the parameter vector, $\theta = [x'\ R_t'\ W_t^{\max\prime}\ W_t^{\min\prime}]'$.

For solving (19) one can iterate [23]

$$[\xi_{(\ell+1)}]_i = \frac{[(Q_d^- + \phi)\xi_{(\ell)} + F_d^-]_i}{[(Q_d^+ + \phi)\xi_{(\ell)} + F_d^+]_i} [\xi_{(\ell)}]_i \tag{20}$$

where $F_d = S_d\theta + W_d$, and $Q_d = G_p Q_P^{-1} G_p'$, $S_d = (G_p Q_p^{-1} C_p + S_p)$, $W_d = W_p$, $\Omega_d = C_p' Q_p^{-1} C_p - \Omega_p$ are the matrices of the dual problem of (19), and A^+ and A^- are the positive and negative parts of matrix $A = A^+ - A^-$. The solution of (19) is constructed from the fixed point ξ^* of (20)

$$\zeta(\xi^*) = \Psi_{d2p}(\theta, \xi^*) = \Gamma_d \theta + \Xi_d \xi^*, \tag{21}$$

where $\Xi_d = -Q_p^{-1} G_p'$, $\Gamma_d = -Q_p^{-1} C_p$.

B. Real-time Implementation

Both SPRG and MPC can be implemented and verified even in embedded platforms with minimal capabilities, because SPRG only requires verifying satisfaction of linear inequalities, and MPC only requires computing the simple iteration (20).

For real-time operation, it is necessary that the number of worst case operations computed by the control algorithm is deterministic, and fits in the available computing time. The number of operations of the SPRG is simple to bound since given that $\eta \leq M$, at most M iterations will be performed, and in each iteration a known number of inequalities is evaluated.

For MPC, computing the number of operations in each control cycle is in general difficult. Some notable exceptions are explicit MPC, see, e.g., [12], and some recently developed low complexity solvers that allows for complexity certification, see, e.g., [24], [25]. These methods estimate an upper bound to the number of iterations (and operations) to compute the MPC solution. Then, a processor must be selected to fit the computational requirements. This often results in an over-dimensioned processor, even by one or more orders of magnitude, due to the estimate being an upper bound of the worst case. Here, we use a different approach that aims at not imposing constraints on the processor, but rather uses all the available computing time, and if that is not sufficient, it exploits a readily available backup solution. Such an approach is beneficial as it avoids over-dimensioning the processor for an unlikely worst case, and hence avoids unnecessary costs.

Due to the simplicity of the solver in Section III, we can compute the number of operations per iteration, so that, given any processor, the maximum number $\bar{\ell} \in \mathbb{Z}_+$ of iterations (20) per computing cycle can be determined. Due to hard real-time requirements, $\bar{\ell} \in \mathbb{Z}_+$ may not be enough to always solve to optimality. However, in case the solver does not return a solution within $\bar{\ell}$ iterations, the SPRG solution can be used to construct a feasible solution. Let $\xi^{(\bar{\ell})}$ be the candidate solution from (20), $\tilde{z}$ be the corresponding solution of (19) from (21), and $\tilde{U}_t$ be the corresponding optimal control sequence. If $\tilde{U}_t$ is feasible (possibly optimal), it will be used, because the constraints are satisfied and the terminal constraint guarantees recursive feasibility for any feasible solution. If $\tilde{U}_t$ is not feasible, we exploit the previous feasible solution and the current reference.

Corollary 1: Let U_{t-1} be a feasible solution for $t \in \mathbb{Z}_+$. The solution $\bar{U}_t$ where $\bar{u}_{k|t} = u_{k+1|t-1}$ and $u_{N|t} = r_{N|t}$ is feasible for (16). □

According to Corollary 1, (15) provides a backup strategy for when a feasible solution is not found within the available iterations. By Corollary 1, one can dimension the processor for a certain probability to terminate within $\bar{\ell}$ iterations, and then use (15) as backup strategy.

C. Fast Stage Command Generation

The fast stage receives as targets the points that have to be processed during a slow stage interval, i.e., for $t \in [(k-1)T_s, kT_s]$, these are $\{q_h\}_{h=\mu(k-1)}^{\mu(k)}$. The curve visiting those points is sampled to obtain a time reference $r_f^i(\tau)$, $i \in \{\mathbf{x}, \mathbf{y}\}$ with sampling frequency T_f. The fast stage command is generated while accounting for the motion of the slow stage as

$$u_f^i(\tau) = r_f^i(\tau) - \bar{G}_s^i * \hat{u}_s^i(\tau), \tag{22}$$

where $\bar{G}_s^i$ is a discrete-time representation of G_s^i with sampling period T_f, which is realized in the upsampling block shown in Figure 4, and $\hat{u}_s^i(\tau)$ is an up-sampling to period T_f of $u_s^i(t)$, so that $\hat{u}_s^i(\tau) = u_s^i(t)$ for all $\tau \in \mathbb{Z}_{[Mt,(M+1)t-1]}$. In (22) the fast-stage dynamics are infinitely fast or at least significantly faster than the slow stage ones. Alternatively, the motion of the slow stage can be compensated for by including in the upsampling block in Figure 4 an appropriate matching filter

$$u_f^i(\tau) = r_f^i(\tau) - W_f^i(\tau) * u_s^i(\tau), \quad W_f^i(\tau) = \frac{\bar{G}_s^i(\tau)}{\bar{G}_f^i(\tau)}. \tag{23}$$

Then, the overall worktool motion results in

$$y^i(\tau) = \bar{G}^i_f(\tau) * r^i_f(\tau) + \bar{G}^i_f(\tau) W^i_f(\tau) * u^i_s(\tau) - \bar{G}^i_f(\tau) * u^i_s(\tau),$$

and hence $y^i(\tau) = \bar{G}^i_f(\tau) * r^i_f(\tau)$ which shows that the motion of the worktool is exactly the same as the one of the ideal machine having the dynamics of the fast stage. While in general matching filters such as (23) are not recommended, due to the fact that models are imprecise, in precision manufacturing the models of the stage motion are of very high precision. Also, while the matching filter does not account for the plant constraints, here the velocity, acceleration, and range constraints are enforced by the ideal reference generation and the SPRG-MPC of Section III. While discussed here for completeness, if the stages are well separated in frequency, the impact of the matching filter (23) is negligible.

V. Simulation and Experimental Results

For evaluating the proposed control design, we have built a laboratory prototype two-stage dual-axis machine using Mitsubishi Electric components for manufacturing applications. The workspace is approximately one quarter of that of a real precision manufacturing machine, and in the prototype we have retained the ratios of relevant quantities such as motor power, stage inertia, workspace area, etc. The slow stage actuation is implemented by 3 servomotors, with 2kW power each, driven by servomotor amplifiers. In the frequency range of interest, the closed-loop slow stage response model from position command to position is the $3^{\rm rd}$ order function whose Bode plot is shown in Figure 3. The range of the slow stage is $\overline{y}^i_s = 0.4$m, $i \in \{\mathbf{x}, \mathbf{y}\}$. The maximal velocity and the maximal acceleration for the slow stage are $\overline{\dot{y}}^i_s = 0.7$m/s and $\overline{\ddot{y}}^i_s = 1.5$g, $i \in \{\mathbf{x}, \mathbf{y}\}$, respectively. The fast stage is implemented by a low inertia high speed servomotor that in the frequency range of interest has the $3^{\rm rd}$ order closed-loop response model shown in Figure 3. The range of the fast stage is $\overline{y}^i_f = 35$mm, $i \in \{\mathbf{x}, \mathbf{y}\}$. A high-end laser diode is used to mark the position of the worktool actuated by the two-stage machine. The microcontroller is based on the TMS320C6678 digital signal processor, for which we reserved only 20% of the computing time for each control loop.

For testing, we have designed using a CAD program (SolidWorks) a set of parts that needs to be machined on a single batch of material. The parts are composed of common shapes machined in manufacturing applications, and include both small and large features. Thus, processing the parts requires both, short range high acceleration movements, and long range smooth movements. The resulting worktool path is shown in Figure 6. The CAD design is then processed by a CAM program realizing its conversion into instructions for a standard numerical motion controller, that is appropriately configured and upsampled to T_f, to generate trajectories for the ideal machine with range of the slow stage and dynamics of the fast stage. This results in the ideal reference trajectory $\{q(h)\}_{h=1}^{\bar{h}}$, with $\bar{h} = 232100$ being the total number of processing points, and an ideal processing time of 46.42s. The accuracy bound for all the tests reported next is $\varepsilon = 50\mu$m, i.e., $|y^i_f(t) + y^i_s(t) - r^i(t)| \leq 50\mu$m for all $t \in \mathbb{R}_{0+}$. The

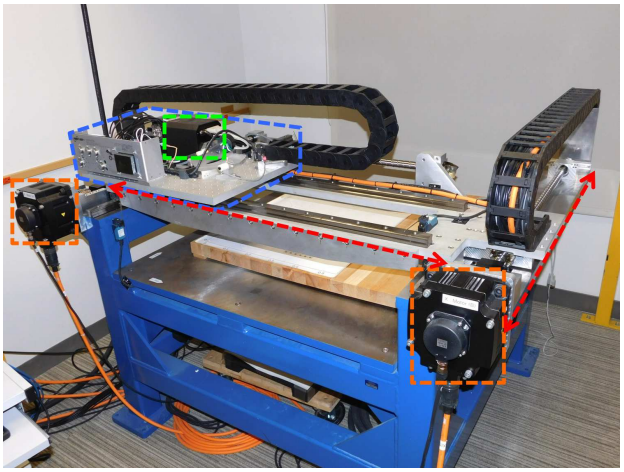

Fig. 5. Experimental two-stage machine used for experimental validation with key components highlighted. Slow stage servomotors, orange. Slow stage range, red. Slow stage moving base, blue. Fast stage base, actuator and worktool, green.

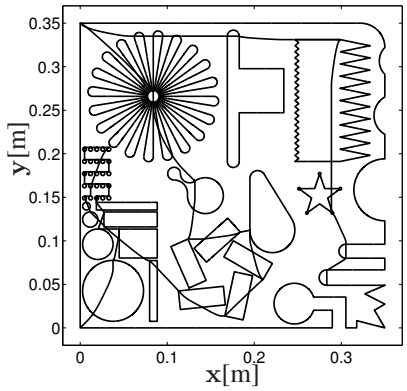

Fig. 6. Worktool path resulting from CAD design of the parts to be machined.

reference generation enforces such a precision using classical approaches for the ideal single stage machine, and the method proposed here only reduces the feed rate, which is inverse proportional to the tracking error, hence further increasing the precision. It shall be noted that for this application the accuracy is a hard bound, as opposed to being the performance objective as in path following MPC [14], [16], and all tests reported next satisfy such bound.

The controller is designed with the quadratic cost function (18) in (16), $M = 150$, $T_s = 30$ms, and a prediction horizon $N = 20$ for the base design. Based on the target processor and on the computational load of iteration (20), we have imposed a limit of $\bar{\ell} = 500$ iterations (20) per control cycle per axis, which can easily fit in the available computing time during the control cycle. In practice, such a limit is very conservative for the considered processor, but it has been chosen to demonstrate how Corollary 1 ensures real-time feasibility in the presence of limited computation.

A. Simulations

First we have performed an extensive simulation study, testing several controller calibrations in closed-loop with a high fidelity model of the two-stage machine, composed of the mechanical models and the servo models that result in the closed-loop axis stage frequency response shown in Figure 3. The results for our best calibration, i.e., the calibration that achieves a desirable trade off between fast processing of the machined part, and low acceleration of the slow stage, and hence limited vibrations, are reported in Figures 7–11.

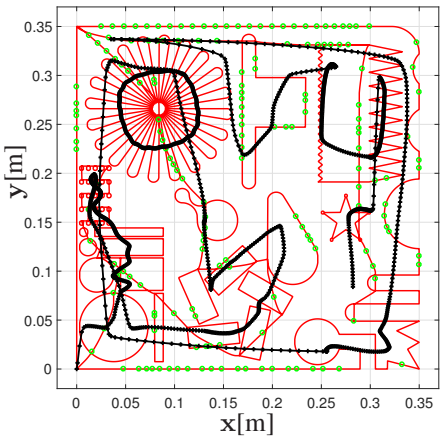

Fig. 7. Processed path (red) covering the desired path, slow stage motion (black), and points where the reference governor reduces the feed rate (green).

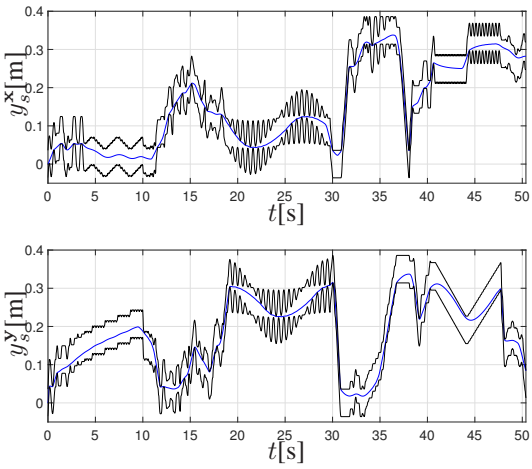

Fig. 8. Position of the slow stage $\mathbf{x}$, $\mathbf{y}$ axes (blue) and constraints (black) due to the reference and fast stage range.

Figure 7 shows the processed path, the slow stage motion, and the points where the reference governor reduces the number of points processed per sample. Most reductions occur on long straight segments when the reference governor needs to guarantee the capability of braking should the reference trajectory change direction suddenly, e.g., due to a corner. The slow stage position is shown in Figure 8 for $\mathbf{x}$ and $\mathbf{y}$ coordinates together with the constraints related to the maximal allowed distance from the reference. Notice how the controller exploits the fast stage range to reduce the motion of the slow stage. This reduces the power consumption and the machine vibrations, thus reducing unmodelled disturbances on the worktool, hence increasing machining precision.

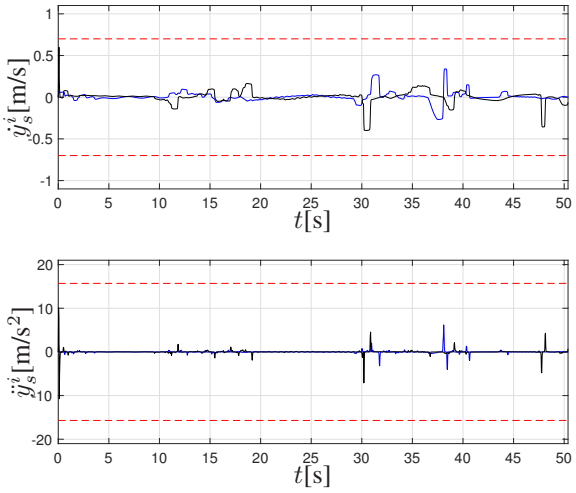

Fig. 9. Velocity and acceleration of the slow stage $\mathbf{x}$ axis (blue), $\mathbf{y}$ axis (black) and constraints (red).

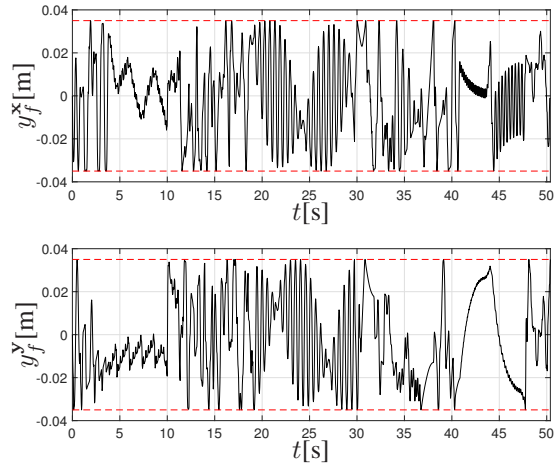

Fig. 10. Position of the fast stage for $\mathbf{x}$ and $\mathbf{y}$ axes and range limits (red).

The smoothness of the slow stage motion stage can be further noted in Figures 9, reporting the velocity and acceleration profiles. For this particular calibration the controller keeps the motion of the slow stage far from the velocity and acceleration constraints. Instead, the fast stage range constraints, i.e., the slow stage tracking error constraints, are often active, as seen in Figure 8, and more easily in Figure 10. Finally, Figure 11 shows for each step the ratio of processed points with respect to the maximum $M = 150$, and the number of iterations executed by the QP solver, where a value above 500 is shown

Method	Time [s]	Maximum accel. [m/s^2]	Mean accel. [m/s^2]	Mean vel. [cm/s]	Mean fast stage motion [mm]	Slow stage work per mass [W/kg]
LFS	55.98	$(6.18, 5.113)$	$(0.66, 0.54)$	$(7.71,\ 7.49)$	$(11.22, 10.74)$	$(3.02, 2.55)$
MPC-20-B	50.38	$(6.18, 10.73)$	$(0.08, 0.11)$	$(3.23,\ 4.13)$	$(15.35, 16.50)$	$(0.34, 0.82)$
MPC-20-H	48.25	$(7.10, 12.73)$	$(0.84, 0.70)$	$(8.59,\ 8.35)$	$(5.15,\ 4.24)$	$(4.35, 4.00)$
MPC-20-L	54.81	$(5.87, 10.11)$	$(0.07, 0.08)$	$(2.83,\ 3.66)$	$(17.28, 19.98)$	$(0.26, 0.43)$
MPC-03-B	51.35	$(14.66, 15.13)$	$(0.49, 0.59)$	$(3.02,\ 3.90)$	$(20.92, 24.89)$	$(5.32, 5.77)$
MPC-03-H	48.58	$(13.08, 12.94)$	$(1.21, 1.18)$	$(7.86,\ 7.70)$	$(9.88,\ 9.55)$	$(9.80, 9.55)$
MPC-03-L	55.71	$(12.59, 12.57)$	$(0.45, 0.54)$	$(2.68,\ 3.50)$	$(22.21, 25.53)$	$(4.93, 5.26)$
SPRG	50.75	$(15.69, 15.69)$	$(1.99, 1.66)$	$(12.53, 11.02)$	$(3.51,\ 3.07)$	$(16.57, 14.26)$

TABLE I

SUMMARY OF THE SIMULATION RESULTS: LINEAR FREQUENCY SEPARATION (LFS), SPATIAL REFERENCE GOVERNOR (SPRG), MPC WITH DIFFERENT HORIZONS AND CALIBRATIONS (MPC-N-C, $N \in \{3, 20\}$, $C \in \{B, H, L\}$). RESULTS PER EACH AXIS REPORTED AS $(\mathbf{x}, \mathbf{y})$.

to indicate that the algorithm did not find a feasible solution within the allowed 500 iterations and the feasible solution was generated by Corollary 1. In these cases, the constraints are satisfied but the trajectory tends to be less smooth, see for instance immediately after $t = 30$s where the iteration limit is exceeded multiple times first on the $\mathbf{y}$ and then on the $\mathbf{x}$ axis, which results in larger accelerations and velocities.

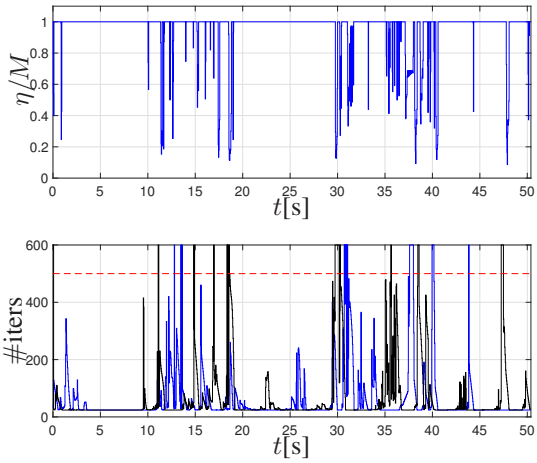

Fig. 11. Top: Feed rate ratio with respect to maximum (η/M). Bottom: MPC iterations ($\mathbf{x}$ axis blue, y axis black), where 600 indicates that a feasible solution is not found within $\bar{\ell} = 500$ iterations and Corollary 1 is used.

In addition to the results in Figures 7–11, representing the most desirable trade off, we have evaluated the proposed algorithm for different controller calibrations to analyze how the performance is affected by changes in the calibration of the cost function weights (, specifically, Q in (18)) and the prediction horizon (N in (16)), and to ensure that the controller operates appropriately under such conditions, that may cause different set of constraints to be active. Table I reports the summary of the results, when using 3 calibrations for the MPC cost function: Best (B, whose results are shown in Figures 7-11) representing a trade off between aggressive tracking and smooth motion, and High (H) and Low (L) aggressiveness, favoring aggressive reference tracking by increasing the tracking weight, and low acceleration by increasing the acceleration

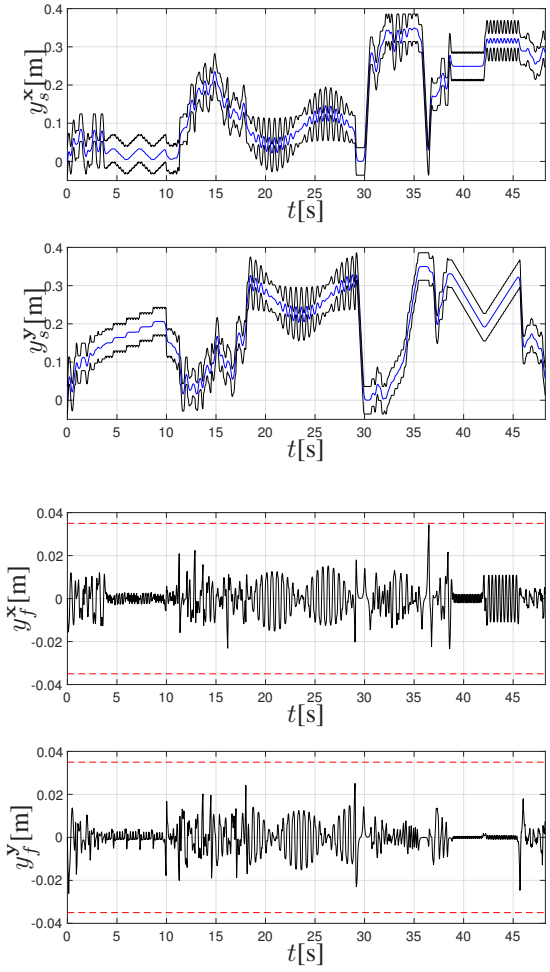

Fig. 12. Results for MPC-20-H. Top: Position of the slow stage $\mathbf{x}$, $\mathbf{y}$ axes (blue) and constraints (black) due to the reference and fast stage range. Bottom: Position of the fast stage for $\mathbf{x}$ and $\mathbf{y}$ axes, and range limits (red).

weight, respectively, also with a shorter horizon $N = 3$. For completeness we have also included the results obtained by running the SPRG alone, i.e., $u_s^i(t) = r^i(t)$, $i \in \{\mathbf{x}, \mathbf{y}\}$, which is feasible by Theorem 1, and the results obtained by applying

a linear frequency separation (LFS) method, along the lines of conventional methods used in production machines. In the LFS approach, the command for the slow stage is obtained by low pass filtering the reference trajectory(see, e.g., [9]), and the command of the fast stage is obtained as the difference between the slow stage position and the reference trajectory. The filter is tuned offline to avoid violation of the acceleration and velocity constraints and the amount of processed points at each step is selected as the largest value, not larger than M, that can be reached by the fast stage range.

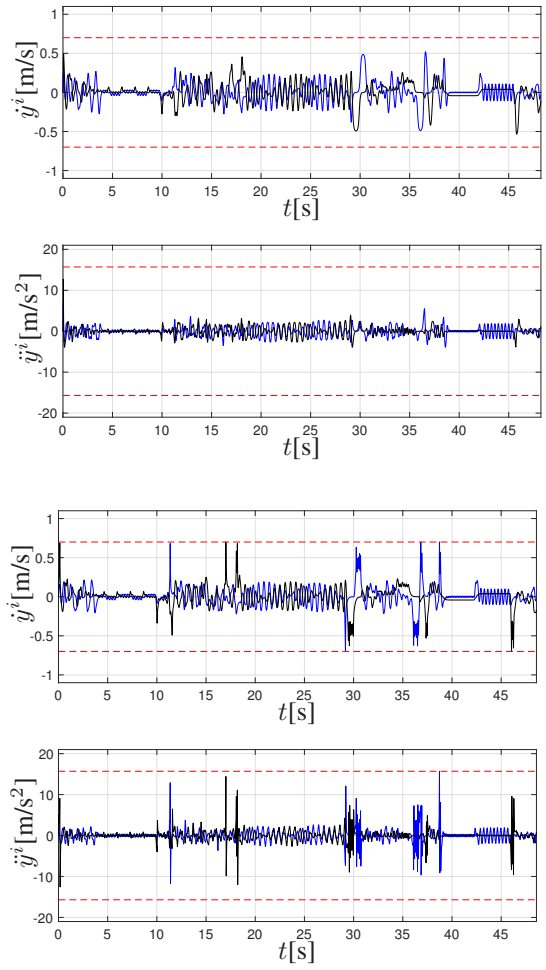

Fig. 13. Velocity and acceleration of the slow stage $\mathbf{x}$ axis (blue), $\mathbf{y}$ axis (black) and constraints (red).Top: MPC-20-H. Bottom: MPC-3-H.

The results in Table I compare the algorithms in terms of machining completion time, mean fast stage motion during the slow stage period, and acceleration, velocity, mechanical work per unit mass, i.e., the integral of the product of velocity and acceleration, of the slow stage. The base MPC shown in Figures 7–11 improves the processing time by more than 10% with respect to the frequency separation method, and still results in lower mean acceleration, velocity, and energy of the slow stage. When compared to using the SPRG alone, the base MPC provides a slightly better processing time, but above all a much smoother motion of the slow stage.

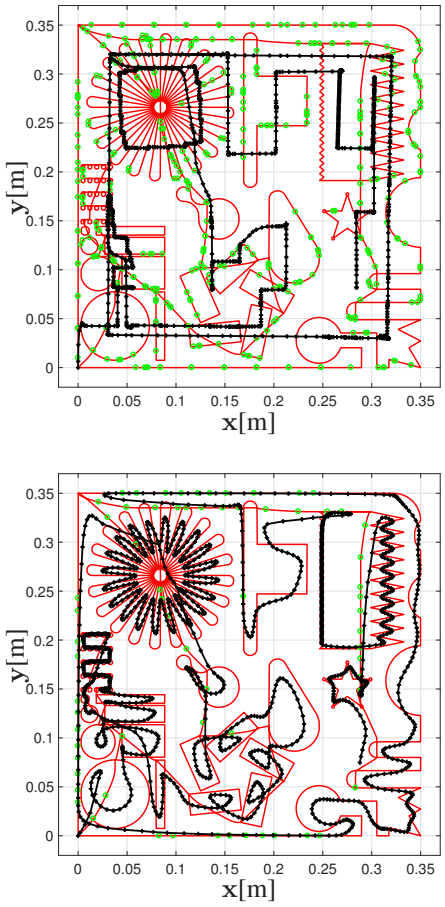

Fig. 14. Processed path (red) covering the desired path, slow stage motion (black), and points where the reference governor reduces the feed rate (green). Top: MPC-3-L. Bottom: MPC-3-H.

When the cost function is calibrated for a more aggressive tracking (H), the MPC processing time decreases but the mean acceleration, velocity and energy increase, while obviously the opposite occurs in the case of less aggressive tracking (L). Figure 12 shows how in the case of aggressive tracking (H) the slow stage stays away from the tracking error bounds, which results in shorter motion of the fast stage. On the other hand, Figure 13 shows that the velocity and acceleration of the slow stage become larger. Figure 13 also shows that in the case with short horizon ($N = 3$), the MPC uses less predictive information and hence has less capability of smoothing the slow stage motion, resulting in even larger acceleration and velocity. Since the work per unit mass is the product of velocity and acceleration, the previous consideration results in an increase in energy consumption for the designs with more aggressive tuning and shorter horizon. The quantitative values for all these effects are shown in Table I, in terms of mean and maximal acceleration, mean velocity and mean motion of the fast stage. In addition, Table I shows that when the MPC horizon becomes shorter, the processing time is in general only minimally affected, but the smoothness of the motion is significantly affected. Still, all the constraints are satisfied. The path produced by the MPC with short horizon

in the case of low (L) and high (H) aggressiveness are shown in Figure 14, where the first tends to make the slow stage move the least amount, until a sharp motion is required not to violate constraints, resulting in several sharp turns, while the second tends to remain close to the pattern shape, due to the high tracking weight. This also results in much more frequent feed rate reductions by SPRG in the first case than in the second. However, in all the cases MPC enforces all the constraints and hence processes the pattern correctly. This is also the case for shorter prediction horizon, even though for $N < 3$ the difference between MPC and SPRG is small.

B. Experimental validation

We have tested the algorithm in our experimental laboratory two-stage machine. The results are consistent with those obtained in simulation due to the high precision of the components and to the high fidelity of the simulation model. Note that in all the experiments we have added an initial 2s stop period to make the results easier to see, which can be noted for instance in the initial spikes in velocity and accelerations that are not easily visible in the simulation plots in Section V-A, due to being at the left border of the figures. The results obtained for the best calibration with $N = 20$ are reported in Figures 15–18. These results are consistent with those obtained in simulation in terms of both, completion time and constraints satisfaction. This is expected due to both, the precision of the model and inner-loop servo controllers, and the feedback action of MPC that rejects model errors and disturbances. The only differences can be noted between Figure 9 and Figure 17, where the acceleration in Figure 17 shows slightly larger spikes than reported in Figure 9. This happens because, despite the velocity and acceleration are available to the control loop, the acquisition board only allows logging of the position signal. Hence, velocities and accelerations shown in the figures are obtained by low pass filtered numerical differentiation, similar to [14]. This amplifies the noise, which despite being small because the encoders are high precision, is still present.

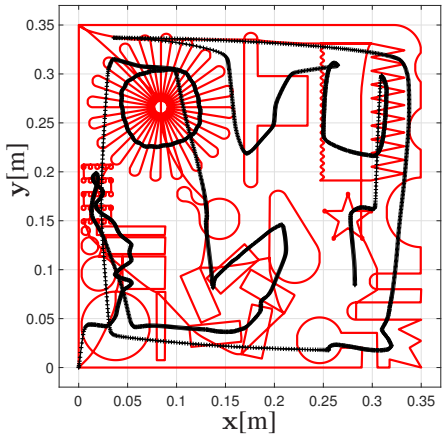

Fig. 15. Experimental results, MPC-20-B. Processed path (red) covering the desired path and slow stage motion (black).

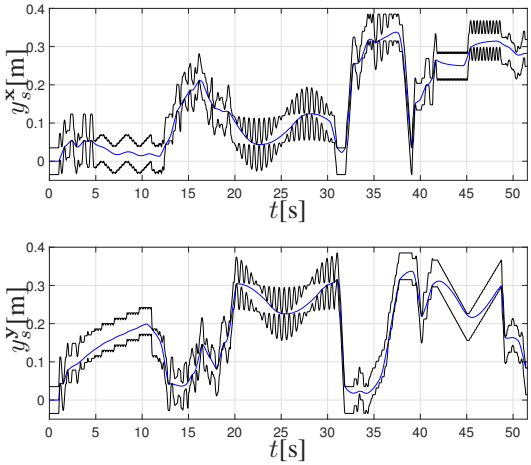

Fig. 16. Experimental results, MPC-20-B. Position of the slow stage $\mathbf{x}$, $\mathbf{y}$ axes (blue) and constraints (black) due fast stage range.

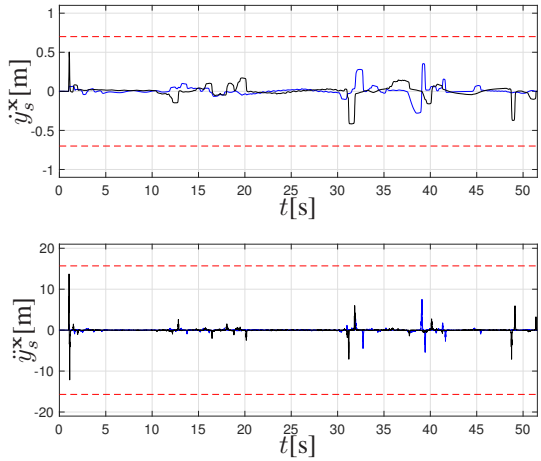

Fig. 17. Experimental results, MPC-20-B. Velocity and acceleration of the slow stage $\mathbf{x}$ axis (blue), $\mathbf{y}$ axis (black) and constraints (red).

We have also tested the other MPC calibrations reported in Table I, and all behaved as expected. In Figures 19–22 we report the results for the MPC with aggressive (H) tuning and short ($N = 3$) horizon, which is probably the most critical to run due to the high accelerations, and short horizon that allows fewer degrees of freedom to compensate for errors. Figures 19–22 show that the MPC works well also in this case and the results are consistent with the simulations, both in terms of processing time and constraint satisfaction.

Overall, the advantage of SPRG-MPC over LFS appears to be due to the capability to exploit the entire range of the operating constraints of the machine. SPRG-MPC actively enforces constraints during processing by slowing down the axis motion only when constraints would otherwise be violated. On the other hand LFS is tuned to avoid constraints, which results in slowed down motion in any condition, due to LFS being based on linear filtering as opposed to the nonlinearities

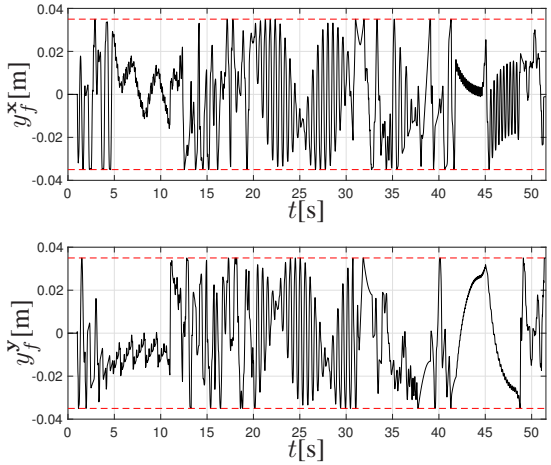

Fig. 18. Experimental results, MPC-20-B. Position of the fast stage for $\mathbf{x}$ and $\mathbf{y}$ axes, and range limits (red).

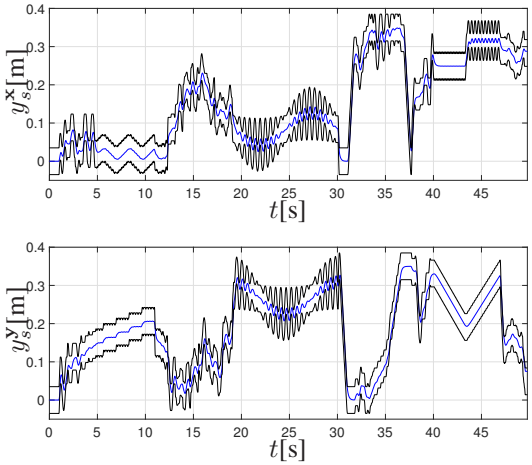

Fig. 20. Experimental results, MPC-3-H. Position of the slow stage $\mathbf{x}$, $\mathbf{y}$ axes (blue) and constraints (black) due to the reference and fast stage range.

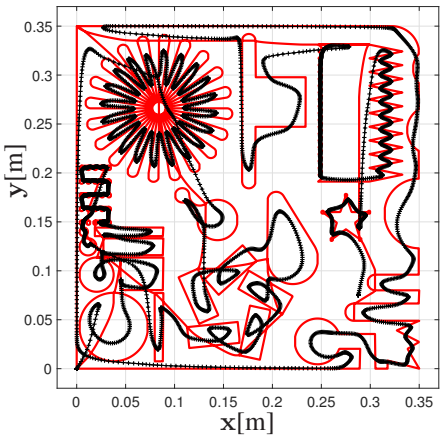

Fig. 19. Experimental results, MPC-3-H. Processed path (red) covering the desired path, slow stage motion (black).

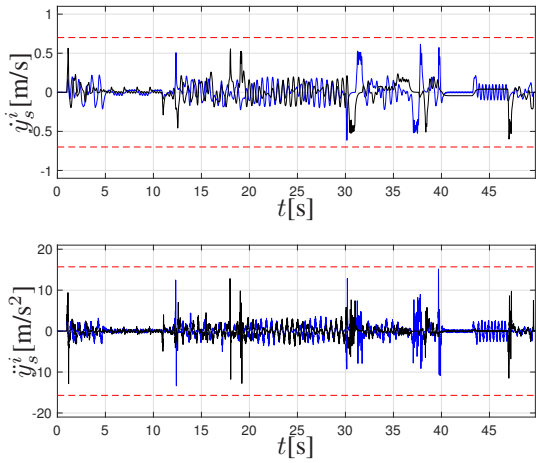

Fig. 21. Experimental results, MPC-3-H. Velocity and acceleration of the slow stage $\mathbf{x}$ axis (blue), $\mathbf{y}$ axis (black) and constraints (red).

induced by constraint enforcement in SPRG-MPC.

VI. Conclusions

We have proposed a method for trajectory generation and control for two-stage precision manufacturing machines where stage actuators with different dynamics and constraints must be coordinated to achieve fast and precise processing. We have developed an architecture based on a cascade of a spatial reference governor, that modulates the reference to achieve feasibility without deforming the physical shape of the pattern being processed, and MPC that seeks the optimal trade-off between processing time and motion smoothness. For the proposed method, we have proved recursive feasibility and finite time processing termination, and we have shown how to achieve real-time processing in any reasonably sized microprocessor. The approach was evaluated in simulation and on a laboratory experiment where the slow stage has been built with scaled dimensions with respect to real processing machines.

Besides developing a solution for a specific relevant application, this paper shows that MPC can be effective for applications in precision manufacturing. For many years such a domain was not a primary target for MPC due to relatively simple processing machine architectures, the limited computing capabilities and the slow optimization algorithms. However, the constant need for higher performance and quality, the increases in computing power, and the recent algorithmic and theoretical developments in MPC have induced transformations in the field that may position MPC to be very effective in manufacturing applications.

Appendix

Proof of Theorem 1: Constraint satisfaction and recursive feasibility are obtained by induction based on the fact that at

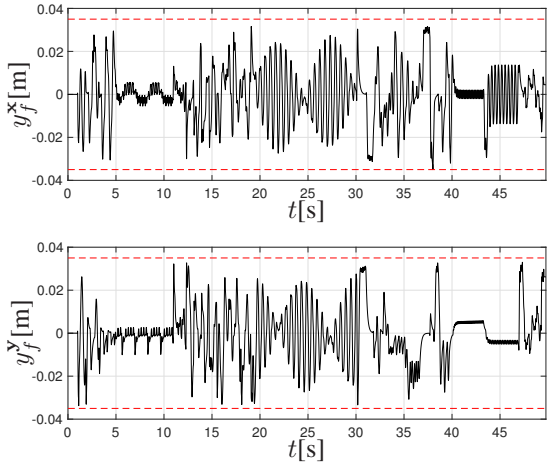

Fig. 22. Experimental results, MPC-3-H. Position of the fast stage for $\mathbf{x}$ and $\mathbf{y}$ axes, and range limits (red).

every $\tau \in \mathbb{Z}_{0+}$ (15) selects r based on (14), which exploits the (maximal) constraint admissible set. Due to Definition 1 if $(x^i(t), r^i(t)) \in \mathcal{O}^i_\infty$, $i \in \{\mathbf{x}, \mathbf{y}\}$, the constraints in (12) are satisfied at time t. For $t = 0$, $(x^i(0), r^i(0)) \in \mathcal{O}^i_\infty$ for $r^i(0) = q^i(0)$. For some $t \in \mathbb{Z}_{0+}$, let (14) have a feasible solution. By (15), this results in $r(t)$ such that $(x^i(t), r^i(t)) \in \mathcal{O}^i_\infty$. Since $\mathcal{O}^i_\infty$ is positive invariant for (12), i.e.,

$$r(\tau) = r(t),\ \forall \tau \geq t \implies (x^i(\tau), r^i(\tau)) \in \mathcal{O}^i_\infty,\ \forall \tau \geq t,$$

(14) is feasible for all $\tau \geq t$, at least by setting $r(\tau) = r(t)$, since $\min q^i = \max q^i = r^i$ and hence that (14c), (14d) are automatically enforced by enforcing (14b). Thus,

$$(x^i(\tau), r^i(\tau)) \in \mathcal{O}^i_\infty, \forall \tau \in \mathbb{Z}_{0+}.$$

For finite termination, since $(x^i_e(q(h)), q^i(h+1)) \in \text{int}(\mathcal{O}^i_\infty)$

$$\exists \bar{\rho} > 0 :\ (x^i_e(q(h)) \oplus \mathcal{B}(\bar{\rho}), q^i(h+1))) \subseteq \text{int}(\mathcal{O}^i_\infty).$$

Let $(x^i(t), r^i(t)) \in \mathcal{O}^i_\infty$, and $r^i(t) = q^i(\bar{h})$, and assume that for all $\tau \in \mathbb{Z}_{0+}$, $(x^i(t+\tau), q^i(\bar{h}+1)) \notin \mathcal{O}^i_\infty$. Thus, for some $\rho > 0$,

$$x^i(t+\tau) \notin x^i_e(q(h)) \oplus \mathcal{B}(\rho), \forall \tau \in \mathbb{Z}_{0+}.$$

However, since A^i is asymptotically stable, for any $\rho \geq 0$ there exists $\tau \in \mathbb{Z}_{0+}$ such that $\|x^i(t+\tau) - x^i_e\| \leq \rho$, which contradicts the previous statement. Then, necessarily

$$\exists \tau \in \mathbb{Z}_{0+} :\ x^i(t+\tau) \in \{x^i_e(q(h))\} \oplus \mathcal{B}(\bar{\rho}),$$

and hence $(x^i(t+\tau), q^i(\bar{h}+1)) \in \mathcal{O}^i_\infty$ and (14c), (14d) are satisfied because $(x^i(t+\tau), q^i(\bar{h})) \in \mathcal{O}^i_\infty$ and $(x^i(t+\tau), q^i(\bar{h}+1)) \in \mathcal{O}^i_\infty$. Thus, τ is a finite time for moving to a new point. By repeating such reasoning for every point in $\{q(h)\}_{h=0}^{\bar{h}}$, and since $\bar{h}$ is finite, the time to complete the processing of all points is finite. ■

Proof of Theorem 2: Let (16) be feasible at t and $U^*_t = [u^*_{0|t} \dots u^*_{N-1|t}]$ be the optimal input sequence. Since $x_{N|t} \in \mathcal{O}_\infty(r_{N|t})$, by Theorem 1 (14) is feasible at $t+1$ and

$$r_{N|t+1} = q(\mu_{N|t+1}),\ \mu_{N|t+1} = \kappa(x_{N|t}, \mu_{N|t}, \{q(h)\}_h).$$

Then, $\tilde{U}_{t+1} = [\tilde{u}_{0|t+1} \dots \tilde{u}_{N-1|t+1}]$ such that

$$\tilde{u}_{k|t+1} = u^*_{k+1|t},\ k \in \mathbb{Z}_{[0,N-2]}, \quad \tilde{u}_{N-1|t+1} = r_{N|t+1}$$

is a feasible input sequence, since the corresponding state trajectory $[\tilde{x}_{0|t+1} \dots \tilde{x}_{N|t+1}]$ is such that

$$\tilde{x}_{k|t+1} = x^*_{k+1|t},\ k \in \mathbb{Z}_{[0,N-1]}, \quad x^*_{N|t} \in \mathcal{O}_\infty(r_{N|t+1}),$$

then $\tilde{x}_{N|t+1} \in \mathcal{O}_\infty(r_{N|t+1})$, hence (16g) is satisfied, and satisfaction of (14c), (14d) implies satisfaction of (16e),(16f). Thus, $\tilde{U}_{t+1}$ is feasible for (16), and the reasoning can be repeated at any future time instant. ■

Proof of Theorem 3: Assume that the statement is not true. Then for some $\tau \in \mathbb{Z}_{0+}$, $r_{N|t} = r_{N|\tau}$, for all $t > \tau$. Because of the assumptions, since $r(t)$ is constant for any $t > \tau$,

$$\mathcal{V}(x(t+1)) \leq \mathcal{V}(x(t)) - L(x_{0|t}, r_{0|t}, u_{0|t}),$$

hence $\mathcal{V}$ is non-increasing and since $\mathcal{V}$ is bounded, it admits a limit $\bar{\mathcal{V}}$. Thus, $\lim_{t\to\infty} L(x_{0|t}, r_{0|t}, u_{0|t}) = 0$ and hence $\lim_{t\to\infty} \|x_{0|t} - x_e(r_{0|t})\| = 0$ and $\lim_{t\to\infty} u_{0|t} = r_{0|t}$. Thus,

$$\forall t_\rho > 0,\ \exists k_\rho \in \mathbb{R},\ t_\rho > \tau :\ \|x_{0|t_\rho} - x_e(r_{0|t_\rho})\| < \rho,$$

and by the reasoning in Theorem 1, $x(t_\rho) \in \mathcal{O}_\infty(q(h+1))$ which contradicts $r_{N|\tau} = r_{N|t}$ for all $t > \tau$. Considering also that (14c), (14d) are satisfied because $x(t_\rho) \in \mathcal{O}_\infty(q(\bar{h}))$ and $x(t_\rho) \in \mathcal{O}_\infty(q(\bar{h}+1))$, for every point $q(h)$, after a finite time the next point $q(h+1)$ is admissible for processing, and, by repeating the same reasoning, in finite time $q(\bar{h})$ is admissible, thus reaching completion of the processing path. ■

Proof of Corollary 1: The proof is similar to that of Theorem 3. Since U_{t-1} was feasible at time $t-1$ the first $N-1$ steps of $\bar{U}_t$ are such that the constraints in (16) are satisfied. Since $r_{N|t}$ is chosen such that $(x_{N|t-1}, r_{N|t}) \in \mathcal{O}_\infty$,

$$Ax_{N-1|t} + Br_{N|t} \in \mathcal{O}_\infty(r_{N|t}),$$

by the positive invariance of $\mathcal{O}_\infty$. Thus, $\bar{U}_t$ also satisfies (16g).

ACKNOWLEDGEMENTS

The authors thank Dr. V. Shilpiekandula and Dr. S. Haghighat for some of the early developments in this research, Dr. M. Brand for seminal work in developing the QP solver, and Dr. W. Yerazunis and J. Barnwell for their work in building the laboratory experimental setup.

REFERENCES

[1] M. Steinbuch and M. Norg, "Advanced motion control: An industrial perspective," *European J. Control*, vol. 4, no. 4, pp. 278–293, 1998.
[2] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[3] K. Erkorkmaz and Y. Altintas, "High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation," *Int. J. Machine Tools and Manufacture*, vol. 41, no. 9, pp. 1323–1345, 2001.
[4] C. Brecher, S. Lange, M. Merz, F. Niehaus, C. Wenzel, M. Winterschladen, and M. Weck, "NURBS based ultra-precision free-form machining," *CIRP Annals-Manufacturing Technology*, vol. 55, no. 1, pp. 547–550, 2006.
[5] S. Salcudean and C. An, "On the control of redundant coarse-fine manipulators," in *IEEE Conf. on Robotics and Automation*, 1989.
[6] S. Hara, T. Hara, L. Yi, and M. Tomizuka, "Novel reference signal generation for two-degree-of-freedom control for hard disk drives," *IEEE Trans. on Mechatronics*, vol. 5, no. 1, pp. 73–78, March 2000.
[7] Y. Michellod, P. Mullhaupt, and D. Gillet, "Strategy for the control of a dual-stage nano-positioning system with a single metrology," in *IEEE Conf. Robotics, Automation and Mechatronics*, 2006.
[8] S. J. Schroeck, W. C. Messner, and R. J. McNab, "On compensator design for linear time-invariant dual-input single-output systems," *IEEE Trans. Mechatronics*, vol. 6, no. 1, pp. 50–57, March 2001.
[9] S. Staroselsky and K. A. Stelson, "Two-stage actuation for improved accuracy of contouring," in *Proc. American Control Confernce*, 1988.
[10] S. Kwon and J. Cheong, "Robust minimum-time control with coarse/fine dual-stage mechanism," *Journal of Mechanical Science and Technology*, vol. 20, no. 11, pp. 1834–1847, 2006.
[11] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Nob Hill, Madison, Wisconsin, 2009.
[12] S. Di Cairano, "An industry perspective on MPC in large volumes applications: Potential Benefits and Open Challenges," in *Proc. 4th IFAC Nonlinear Model Predictive Control Conf.*, 2012, pp. 52–59.
[13] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V. Kolmanovsky, "The development of model predictive control in automotive industry: A survey," in *Proc. IEEE Conf. Control Applications*, 2012, pp. 295–302.
[14] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen, "Implementation of nonlinear model predictive path-following control for an industrial robot," *IEEE Trans. Control Systems Tech.*, vol. 25, no. 4, pp. 1505–1511, 2017.
[15] M. Böck and A. Kugi, "Real-time nonlinear model predictive path-following control of a laboratory tower crane," *IEEE Trans. Control Systems Tech.*, vol. 22, no. 4, pp. 1461–1473, 2014.
[16] D. Lam, C. Manzie, and M. C. Good, "Model predictive contouring control for biaxial systems," *IEEE Trans. Control Systems Technology*, vol. 21, no. 2, pp. 552–559, 2013.
[17] S. Di Cairano, A. Goldsmith, and S. Bortoff, "Model predictive control and spatial governor for multistage processing machines in precision manufacturing," in *Proc. 5th IFAC Nonlinear Model Predictive Control Conference*, 2015, pp. 398–403.
[18] D. Limon, I. Alvarado, T. Alamo, and E. Camacho, "MPC for tracking piecewise constant references for constrained linear systems," *Automatica*, vol. 44, no. 9, pp. 2382 – 2387, 2008.
[19] A. Ferramosca, D. Limon, I. Alvarado, T. Alamo, and E. Camacho, "MPC for tracking of constrained nonlinear systems," in *Proc. 48th IEEE Conf. on Decision and Control*, 2009, pp. 7978–7983.
[20] E. Garone, S. Di Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.
[21] D. Lam, C. Manzie, and M. C. Good, "Multi-axis model predictive contouring control," *Int. Journal of Control*, vol. 86, no. 8, pp. 1410–1424, 2013.
[22] T. Faulwasser and R. Findeisen, "Nonlinear model predictive control for constrained output path following," *IEEE Trans. Automatic Control*, vol. 61, no. 4, pp. 1026–1039, 2016.
[23] S. Di Cairano, M. Brand, and S. A. Bortoff, "Projection-free parallel quadratic programming for linear model predictive control," *Int. J. Control*, vol. 86, no. 8, pp. 1367–1385, 2013.
[24] S. Richter, C. N. Jones, and M. Morari, "Computational complexity certification for real-time mpc with input constraints based on the fast gradient method," *IEEE Trans. Automatic Control*, vol. 57, no. 6, pp. 1391–1403, 2012.
[25] P. Patrinos and A. Bemporad, "An accelerated dual gradient-projection algorithm for embedded linear model predictive control," *IEEE Trans. Automatic Control*, vol. 59, no. 1, pp. 18–33, 2014.

Stefano Di Cairano (SM'15) received the Master (Laurea), and the PhD in Information Engineering from the University of Siena, Italy, in 2004 and 2008. He was at the Technical University of Denmark and at the California Institute of Technology. From 2008 to 2011 he was with Powertrain Control R&A, Ford Research and Adv. Engineering, Dearborn, MI. Since 2011, he is with Mitsubishi Electric Research Laboratories, Cambridge, MA, where he is now a Senior Team Leader and a Senior Principal Researcher. His research is on optimization-based control strategies for complex mechatronic systems, in automotive, aerospace, factory automation, and transportation. His research interests include model predictive control, constrained control, optimization, estimation.

Dr. Di Cairano has authored/co-authored more than 100 peer reviewed papers in journals and 25 patents. He was the Chair of the IEEE CSS Technical Committee on Automotive Controls and a member of the CSS Conference Editorial Board. Currently, he is the Chair of IEEE Standing Committee on Standards, a Vice Chair of IFAC Technical Committee on Optimal Control, and an Associate Editor of the IEEE Trans. Control Systems Technology.

Abraham Goldsmith received a MS in electrical and computer engineering from Worcester Polytechnic Institute in 2008. Since then he has worked for Mitsubishi Electric Research Laboratories in Cambridge, MA, where he is currently a Principal Research Scientist. His research interests include computer architectures, the computational aspects of advanced control algorithms for embedded control systems, and design automation for embedded systems.

Uroš V. Kalabić (M16) received the M.S.E. and Ph.D. degrees in aerospace engineering from the University of Michigan, Ann Arbor, in 2011 and 2015. He has also received the B.A.Sc. degree in engineering science from the University of Toronto in 2010 and the M.S. degree in mathematics from the University of Michigan in 2014.

His job experience includes research internships at Ford Motor Company, in 2011 and 2012, and at Mitsubishi Electric Research Laboratories, in 2014. Since 2015, he has been a Research Scientist at Mitsubishi Electric Research Laboratories, Cambridge, Massachusetts. His research interests include the control of systems subject to state or output constraints and applications to aerospace and automotive systems.

Scott A. Bortoff (SM'11) received the B.S. and M.S. degrees from Syracuse University in 1985 and 1986, respectively, and the Ph.D. degree from the University of Illinois at Urbana-Champaign in 1992, all in Electrical Engineering. He has held positions of Assistant and Associate Professor of Electrical and Computer Engineering at the University of Toronto, from 1992-2000, various Group and Project Leaderships in the area of control systems at United Technologies Research Center from 2000-2009, and Group Manager of Mechatronics at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA from 2009-2015. He is currently a Distinguished Member of Research Staff at MERL. His research interests and responsibilities include system-level dynamic modeling and control of large-scale thermofluid and mechatronic systems, and shaping corporate research strategy in the areas of modeling and systems control. He is currently Associate Editor of the IEEE Control System Magazine and serves the Modelica community in various editorial capacities.