ARCHIVES OF ELECTRICAL ENGINEERING VOL. 72(4), pp. 1005–1016 (2023)

DOI 10.24425/aee.2023.147423

Deployment of a predictive-like optimal control law on a servo drive system using linear programming approach

DARIUSZ HORLA[1] ✉, PIOTR PINCZEWSKI[2]

[1] *Institute of Robotics and Machine Intelligence*
Poznan University of Technology
Piotrowo 3a Str., 60-965 Poznań, Poland
[2] *IT.integro sp. z o.o.*
Ząbkowicka 12 Str., 60-166 Poznań, Poland
e-mail: ✉*dariusz.horla@put.poznan.pl, pinczewskipiotr@gmail.com*

(Received: 31.07.2023, revised: 27.08.2023)

Abstract: Current drive control systems tend to push control loops to the limits of their performance. One of the ways of doing so is to use advanced optimization algorithms, usually related to model-based off-line calculations, such as genetic algorithms, the particle swarm optimisation or the others. There is, however, a simpler way, namely to use predictive control formalism and by formulation of a simple linear programming problem which is easy to solve using powerful solvers, without excessive computational burden, what is a reliable solution, as whenever the optimization problem has a feasible solution, a global minimizer can be efficiently found. This approach has been deployed for a servo drive system operated by a real-time sampled-data controller, verified between model-in-the-loop and hardware-in-the-loop configurations, for a range of prediction horizons, as an attractive alternative to classical quadratic programming-related formulation of predictive control task.

Key words: constraints, linear programming, optimal control, servo drive

1. Introduction

Drive technology is a core component to obtain proper performance of various machines, hand-held equipment, robots, or motion control systems. However, there is much more to consider than meets the eye, apart from just an apt selection of the motor or its suitable design. As any

new deployment of a novel approach should predominantly focus on the specific application, still improved performance is a cornerstone and needs to present better control quality or reduced cost of control. This could account for better dynamics, increased precision and quality or better energy efficiency of the resultant system [1,4].

To achieve these goals, drive systems must be selected taking these general ideas. Market development and continuous technological advances are leading to increase in expectations of the control quality of drive systems. As a solution, currently the growing popularity of adaptive and predictive controllers is seen [1,2]. However, the necessary computational power to apply the latter is much higher than that of classical PID-type controllers, and, in addition, their implementation and tuning require more knowledge of the problem and more work, though the control quality is clearly better. The other solutions often use fuzzy logic controllers, artificial neural networks or advanced optimization techniques using models, leading to particle-swarm optimization [3] or genetic algorithms used [6].

The above-mentioned predictive control or, in other words, receding-horizon control has been successfully introduced in the industry, where optimization stage is performed at integer multiplicities of the sampling period [2]. The optimization procedures are either off-line, though rather for simpler configurations, where no constraints are taken into consideration, and lead to obtaining analytical formulae describing control laws, or the optimal solution is found based on an iterative optimization approach. In such a case, interior-point methods for constrained quadratic programming problems are used, leading to exhaustive iterative calculations to find the solution.

There is, however, a simpler way of generating optimal control signals, whenever there is a knowledge about the model of the system, either linear, or nonlinear, in the form of the use of iterative optimization procedures with lower computational burden connected. The current paper presents the use of linear programming (LP) to generate optimal control signals in a drive control task using the powerful MOSEK solver, leading to obtaining the optimal solution in a couple of milliseconds. As in the case of predictive control, a number of model future outputs is generated in a specific time horizon [9], what means that, as with predictive control, knowledge of the plant model and some computational power are required for correct operation of the control system. In addition, there is also a problem of initialization of the procedure [7]. This is due to the need of repeated generation of the control signal based on the current measurements obtained from the system via feedback, updating the knowledge of the controller concerning the behaviour of the control loop.

In standard, the minimised cost function used in predictive-type controllers is based on a sum of squared tracking errors in some horizon. As per quadratic forms present in the described cost function, in the simulation environment one gets a computationally simple procedure with global minimum found in a reasonable time. Using sum of absolute error values instead of sum of squared errors is not a common choice, but due to the L_1 penalty function, errors close to zero are better penalized as in the case of L_2 type one [8].

In order to test the approach, Matlab's simulation environment is selected, to perform model-in-the-loop simulations, leading eventually to hardware-in-the-loop tests using the Modular Servo System of Inteco. Control performance of the sampled-data controller is analysed here in terms of a linear penalty function and two types of constraints imposed on the control signal, namely amplitude and rate constraints, and a range of prediction horizons to fully mirror the quality of control. The laboratory stand offers an easy connection to the hardware via USB, thanks to

a toolbox provided by Inteco. Since the current paper is the follower of the research report [5] considering generalised predictive control, also verified on the same laboratory stand, some details connected to the description of the system are provided in a brief manner. Nevertheless, all the necessary formulae, derivations and calculations needed to define the optimisation problem are provided.

The novelty of the paper is related to using a standard linear programming approach to obtain an efficient and reliable model-based control solution, using powerful and reliable LP solvers, whereas the major contribution is in terms of analysis of the interplay between prediction horizons, control performance and various constraints imposed on the control signal.

In Section 2, a short description of the laboratory stand is given, accompanied with the derivation of the model of the plant. Section 3 delivers a frank introduction to the predictive-like performance index and outlines the optimisation approach adopted in the research. Section 4 presents the means of performing optimisation in real-time, and Section 5 presents the main results in the rotational velocity control task. Finally, the last Section summarizes the paper.

2. Modular servo system laboratory stand

The experimental setup comprises a real-time DAC card, to offer communication between the hardware and the control software, including the solver, and the servo system comprising the 12 V/77 W DC motor, a tachogenerator and a brass cylinder forming an inertia load (with connected moment of inertia J and damping coefficient c), as presented in Fig. 1. The output voltage of the tacho-generator is proportional to the angular velocity, forming the output signal of the closed-loop system, i.e., $y(t) = \omega(t) = \dot{\theta}(t)$.

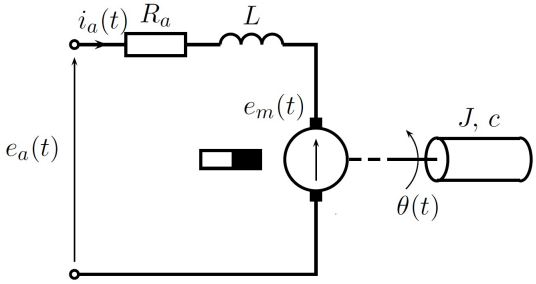


Fig. 1. Diagram of the experimental setup

The calculated control signal (calculated armature voltage) is fed to the servo system using an RT-DAC card and Simulink Coder to follow real-time requirements. The constrained control signal (armature voltage actually applied to the system) is limited to 12 V, and is presented throughout the paper in a dimensionless form as $|u(t)| \leq 1$ [10].

Let $\theta(s)$ denote the Laplace transform of the angular position $\theta(t)$ of the shaft, and $V(s)$ the transform of the calculated armature voltage $v(t) = e_a(t)$, K and T denote gain and time constant

of the following closed-loop transfer function:

$$G(s) = \frac{\theta(s)}{V(s)} = \frac{K}{s\,(Ts+1)}, \tag{1}$$

resulting from the derivations presented in [5]. Having transformed (1) into its state-space representation with the state vector of the form $\boldsymbol{x}(t) = [x_1(t), x_2(t)]^T = [\theta(t), \omega(t)]^T$ one gets

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= \frac{-1}{T}x_2(t) + \frac{K}{T}u(t), \end{aligned} \tag{2}$$

which correspond in a standard state-space notation to:

$$\begin{aligned} \boldsymbol{A} &= \begin{bmatrix} 0 & 1 \\ 0 & \dfrac{-1}{T} \end{bmatrix}, \\ \boldsymbol{x} &= \begin{bmatrix} 0 \\ \dfrac{K}{T} \end{bmatrix}, \\ \boldsymbol{c}^T &= \begin{bmatrix} 1, & 0 \end{bmatrix}^T, \\ d &= 0. \end{aligned} \tag{3}$$

When deriving the system of equations of state for the linear model, many simplifications should be made, one of which is the assumption the system being linear, what gives rise to the need to remove the static nonlinearity from the system, related to friction, which causes that for near-zero values of the control voltage supplying the system. Since a nonlinear model capturing the above problem can be written using the following formulae:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= c\,(u(t) - g\,(x_2(t)))\,, \end{aligned} \tag{4}$$

with $g(x_2)$ as an inverse static characteristic of the loop $u = f(\omega)$, and f is some nonlinear function, removal of the nonlinearity leads to the equations connected to (1).

In the case of a predictive control formalism, it is necessary to obtain a discrete-time counterpart of the continuous-time model of the system (2) subject to a sampling period T_s, to calculate samples of future outputs. The discrete-time model is obtained in an armature voltage-rotational velocity control loop, i.e., for

$$\dot{x}(t) = \frac{-1}{T}x(t) + \frac{K}{T}u(t), \tag{5}$$

which corresponds to first-order dynamics. Using the step-invariant transform one obtains:

$$\frac{Y(z)}{U(z)} = \frac{B_d z^{-1}}{1 - A_d z^{-1}}\,, \tag{6}$$

$$A_d = e^{-\frac{T_s}{T}}, \tag{7}$$

$$B_d = K\left(1 - e^{-\frac{T_s}{T}}\right), \tag{8}$$

eventually leading to a discrete-time state-space equations model

$$y_k = A_d y_{k-1} + B_d u_{k-1}\,, \tag{9}$$

$$x_{k+1} = e^{-\frac{T_s}{T}} x_k + K\left(1 - e^{-\frac{T_s}{T}}\right) u_k = A_d x_k + B_d u_k\,, \tag{10}$$

on the basis of knowledge of (6), with lower index k referring to a sample number of a specified continuous-time signal, i.e., $x_k = x(kT_s)$, etc.

3. Predictive-type performance index reformulation as a linear programming problem

In order to verify the performance of the closed-loop system for a given reference profile $r(t)$, a predictive-like performance index has been introduced in the form of the following sum of errors in a horizon of N_y samples, i.e.,

$$J = \sum_{i=1}^{N_y} |e_{k+i}|, \tag{11}$$

with the error samples defined as

$$e_{k+i} = r_{k+i} - y_{k+i} = r_{k+i} - (A_d x_{k+i-1} + B_d u_{k+i-1})\,. \tag{12}$$

Now, in order to formulate the optimization problem, the minimization task of the above-mentioned performance index refers to solving the task

$$\min_{\boldsymbol{u}} \sum_{i=1}^{N_y} |e_{k+i}|, \tag{13}$$

which corresponds, in turn, to the following linear programming problem solved at every sampling instant

$$\begin{aligned} &\min_{\boldsymbol{u},\boldsymbol{d}} \sum_{i=1}^{N_y} d_i \\ &\text{s.t.} - d_i \le e_{k+i} \le d_i \quad (i = 1, \ldots, N_y), \\ &\boldsymbol{d} \ge \boldsymbol{0}, \end{aligned} \tag{14}$$

with the vector of dummy variables $\boldsymbol{d}$ ensuring the absolute values to be minimized,

$$\boldsymbol{d} = \left[d_1,\ d_2,\ \ldots,\ d_{N_y-1},\ d_{N_y}\right]^T. \tag{15}$$

Now, the evolution of the state at the i-th step can be presented as (for a general SISO case):

$$\boldsymbol{x}_{k+i} = \boldsymbol{A}_d^i \boldsymbol{x}_k + \boldsymbol{A}_d^{i-1}\boldsymbol{B}_d u_k + \boldsymbol{A}_d^{i-2}\boldsymbol{B}_d u_{k+1} + \ldots + \boldsymbol{A}_d\boldsymbol{B}_d u_{k+i-2} + \boldsymbol{B}_d u_{k+i-1}\,, \tag{16}$$

or, simpler, in a matrix form

$$\boldsymbol{x}_{k+i} = \boldsymbol{F}(i)x_k + \boldsymbol{G}^T(i)\boldsymbol{u}_{k,i}\,, \tag{17}$$

with:

$$\boldsymbol{F}(i) = \boldsymbol{A}_d^i\,, \tag{18}$$

$$\boldsymbol{G}^T(i) = \left[\boldsymbol{A}_d^{i-1},\ \boldsymbol{A}_d^{i-2},\ \ldots,\ \boldsymbol{A}_d,\ \boldsymbol{I}\right]\boldsymbol{B}_d\,, \tag{19}$$

$$\boldsymbol{u}_{k,i} = \left[u_k,\ u_{k+1},\ \ldots,\ u_{k+i-2},\ u_{k+i-1}\right]^T. \tag{20}$$

In addition to the LP problem statement (14), the authors have decided to include information about potential constraints imposed on the control signal directly to the LP problem, given as either standard symmetrical amplitude cut-off constraints of the form

$$-\alpha \le u_{k+j} \le \alpha, \tag{21}$$

with α being the cut-off amplitude, enabling proper tracking, i.e., satisfying

$$\alpha \ge \left|\frac{1 - A_d}{B_d}\right|, \tag{22}$$

which in the referred case corresponds simply to $\alpha \ge 1$.

As an extension, enabling shaping the dynamics of the closed-loop system, the other constraint type has also been included, related to rate constraints, in the form

$$-\beta \le u_{k+j} - u_{k+j-1} \le \beta. \tag{23}$$

This type of constraint imposed alone always enables asymptotic tracking, provided that there are no simultaneous amplitude constraints disabling proper tracking. As per the defined range for amplitude constraints in the case of this deployment, it is obvious that $\beta \ge 2$ does not actually affect the performance of the system, as the amplitude between extreme admissible values of the control signal is actually 2.

4. Implementation issues of the predictive-type controller

In order to deploy the control strategy, a special solver interface had to be created, which does not take advantage of a well-developed MathWorks libraries performing optimization task, as they do not have their C source files available [11]. This is the reason, why the solver has not been created from the scratch, but the authors have applied for MOSEK student version solver, which commands can be easily handled by DLL files created by Simulink Coder, and thus – can work in the considered real-time sampled-data control system [12].

The considered controller has been implemented as a C-mex S-function, and subsequently compiled to mex64 (Matlab Executable) file using the template of the S-function available with the Matlab distribution. The issues which had to be, however, tackled out have been related to writing down all the relations between reference signal samples and predictions of future outputs over a selected range of prediction horizons, using the formulation as in Section 3. As per a dominating time constant T, identified at the level of approx. 1.1 s in the armature voltage-rotational velocity control loop, the sampling period for control purposes has been selected as 100 ms.

5. Simulation and experimental campaign

The experimental campaign has been performed for a 10 second-long reference signal providing expected changes of the rotational velocity of the shaft, with the step change at the half of the experiment. In such a configuration, and for the 100 ms sampling period, the predictive-like control law has been tested based on current measurements of the rotational velocity value. On the basis of this value, at each sampling instant a prediction of the output signal has been calculated, leading to solving a linear programming problem. The solution (20), being a vector of control actions has been used in a receding horizon style, i.e., to implement its first component and re-do the optimisation at the next sampling period.

On the basis of initial identification campaign, it has been found that $K = 225$ and $T = 1.1$ s. In addition, at all times it has been assumed that the number of control samples calculated corresponds to the actual prediction horizon N_y of the output signal.

At first, model-in-the-loop simulations have been performed over a grid of 19×19 configurations of prediction horizon values versus amplitude constraint imposed on calculated control signals. The values of the performance index used to assess control quality (11) are presented in Fig. 2. In addition, in that figure, three points are marked and selected on the basis of the performance index values, related to the worst, medium and the best control loop performance. It can be identified that either for tight constraints, or for short prediction horizon, control performance deteriorates.

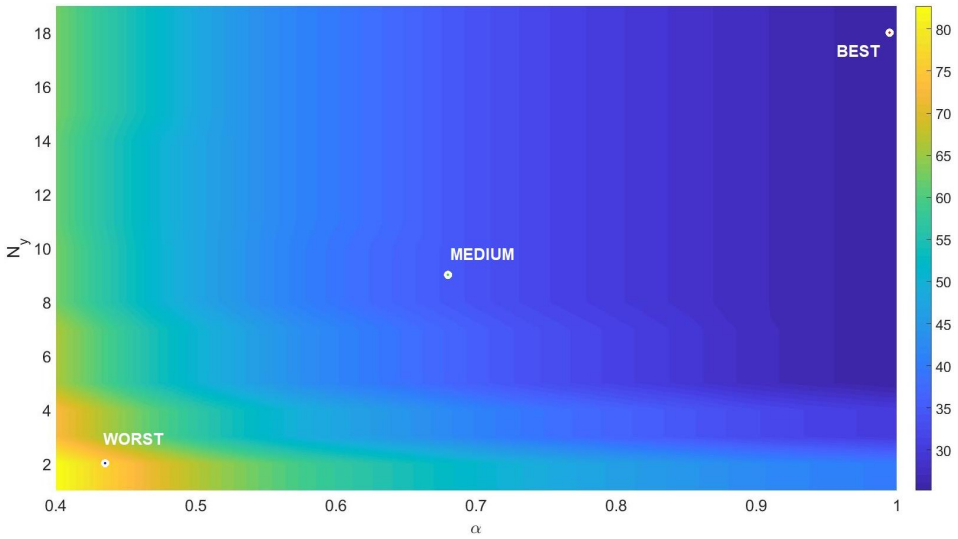
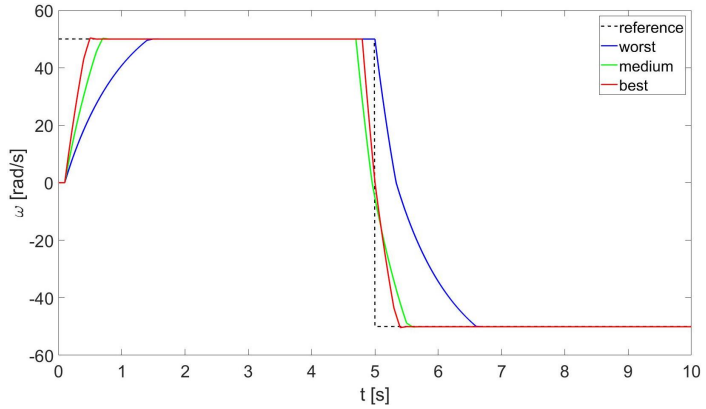


Fig. 2. Performance index vs. prediction horizon and amplitude constraint (model-in-the-loop tests)

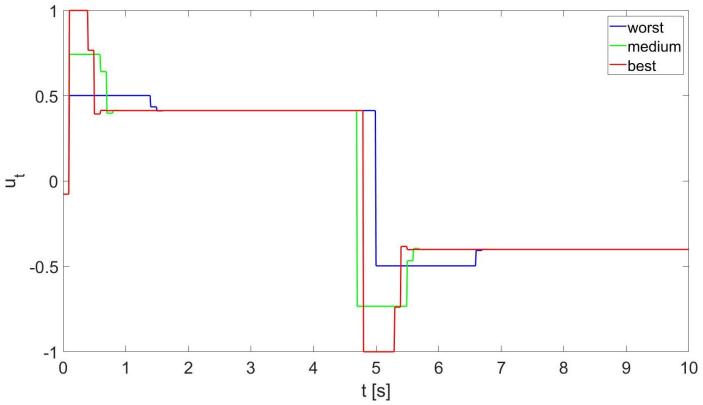
Selection of the prediction horizon at the level of $N_y = 19$ samples gives a 1.9 s-ahead prediction, enabling the controller to foresee the changes in the reference signal known a priori, and to take all the necessary steps to alter the values of the control signal.

In all the considered cases, the control amplitude constraint has been set at greater than the value of the cut-off level (with value of approx. 0.4) enabling proper asymptotic tracking of the rotational velocity.

As already remarked, for the triplet of points, performance of the system has been evaluated on the basis of plots presented in Fig. 3, where one can see the evolution of the output signals as functions of control actions applied to the system. It should be noted that as per optimization procedures running, the generated control signals in a specific horizon have been equal to constrained control signals, since constraints have been taken into account while carrying on optimization routines.



(a)



(b)

Fig. 3. (a) tracking performance; (b) control signal for model-in-the-loop tests and amplitude-constrained system

Obviously, the control signal with highest peak value results in the best transition property between positive and negative rotational velocities, taking full advantage of the prediction mechanism available in the system. This can be observed clearly as per full knowledge of the dynamic

and static properties of the linear model of the servo drive, and gives a good insight into the performance of the control loop in the real-world experiment.

As in the amplitude-constrained case, a similar campaign has been carried out in simulation-in-the-loop fashion and MOSEK solver used for rate constraints. Figure 4 presents performance index values, whereas Fig. 5 – tracking performance and control signals for selected configurations of the prediction horizon and rate constraints.

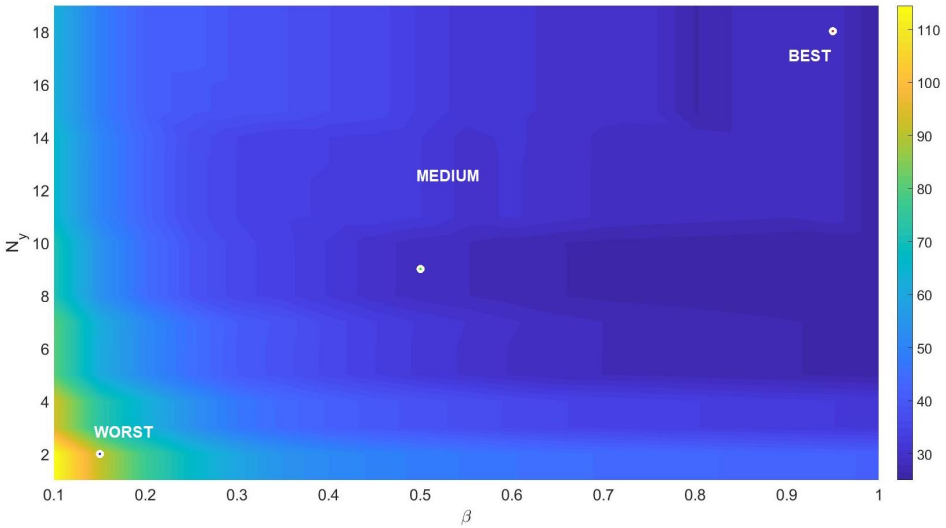
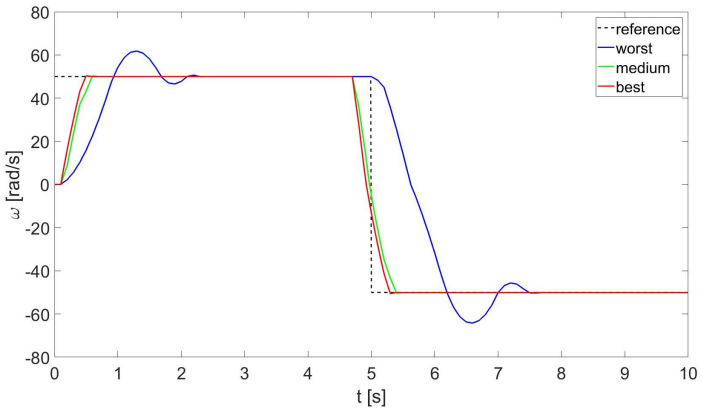
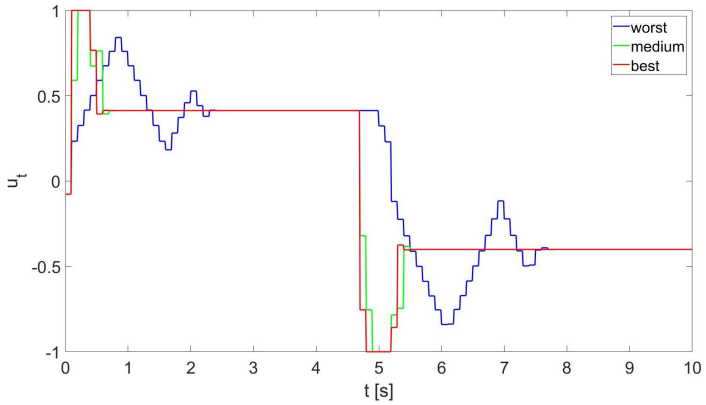


Fig. 4. Performance index vs. prediction horizon and rate constraint (model-in-the-loop tests)

Since the profile of the control signals presented in Fig. 5(b) depicts the fact that rate constraints are active actually twice during the evolution of each simulation, i.e., shortly after reference velocity changes, it can be clearly seen that the value of rate constraints modifies the agility of



(a)



(b)

Fig. 5. (a) tracking performance; (b) control signal for model-in-the-loop tests and rate-constrained system

the control process. Whenever the rate constraint value is large (far higher than 0), the prediction strategy works fine, offering foreseeing feature of the reference primitive changes. In the opposite case, the control signal is unable to exert sufficient changes at the output of the control loop.

In order to finally verify the performance of the real-world system, an experimental campaign for a grid of 11×11 points over a 10-second experiment horizon has been carried out, with the results concerning 121 values of the performance index have been presented in Fig. 6, originating from single runs of the experiment. It is to be noted that sudden changes (appearing as local minima or local maxima on the performance index surface) result from a tachogenerator output

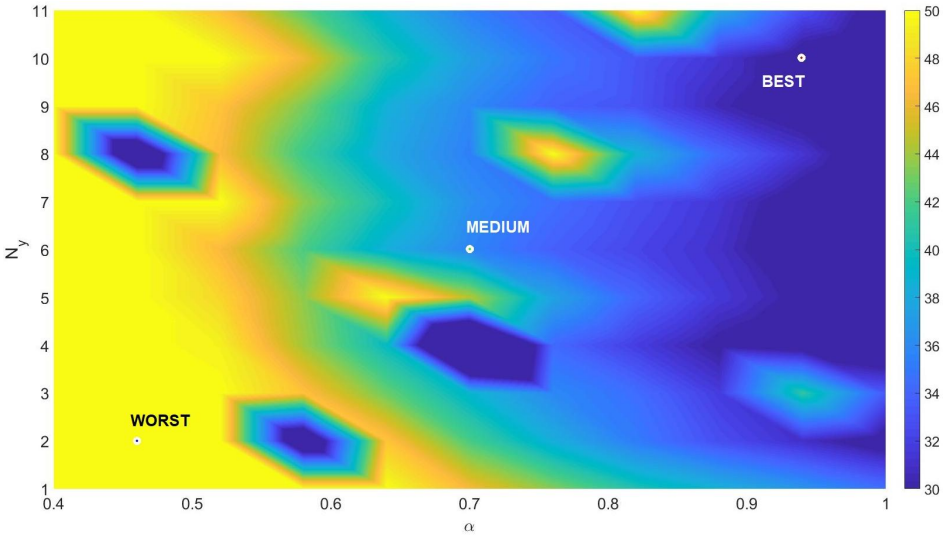
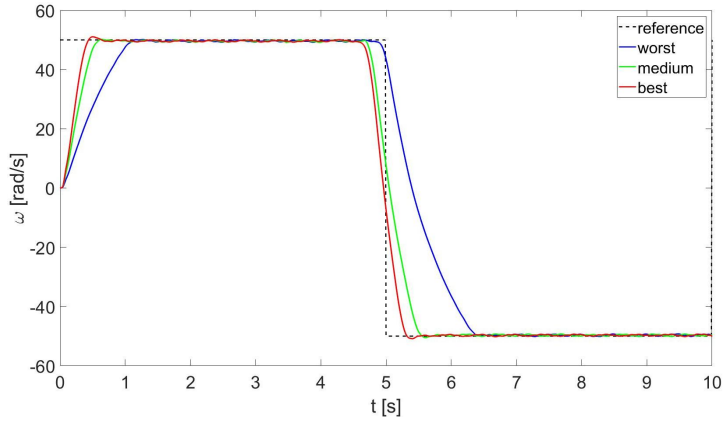


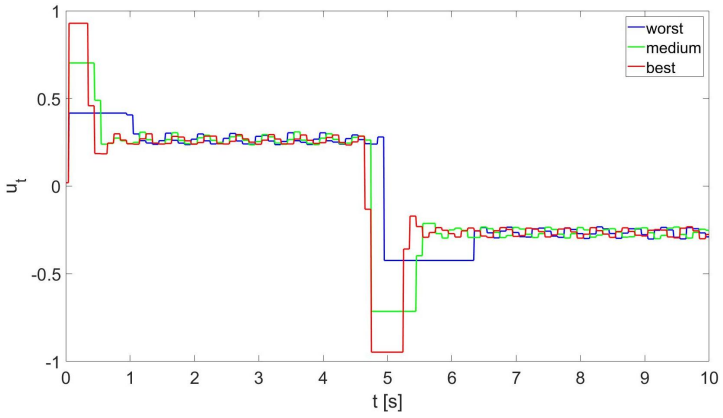
Fig. 6. Performance index vs. prediction horizon and rate constraint (hardware-in-the-loop experiments)

signal used to assess the value of the current rotational velocity. Due to the noise corrupting this signal, meaning – due to imperfect measurements, the shape of the surface has been slightly distorted, thought it still mimics the actual performance of the system, as expected on the basis of simulation-in-the-loop stages.

Figure 7 depicts the tracking performance in the closed-loop system for a selection of input parameters of the control system/solver. As can be clearly seen from the profiles of the control signals, imperfect measurement of the rotational velocity results in oscillatory behaviour of the control signal. This is definitely not due to the case of violation of real-time requirements, since the optimization procedure finished usually after 0.004–0.005 s for current measurements fed to the solver, what has been a neglectful fraction of the dominating time constant in the rotational velocity control loop, as well a small fraction of the sampling period.



(a)



(b)

Fig. 7. (a) tracking performance; (b) control signal for hardware-in-the-loop experiments and amplitude-constrained system

6. Summary

The paper presents a predictive control-related strategy to ensure tracking of a rotational velocity profile in a closed-loop system, with the control signal generated by means of optimization, based on MOSEK solver [13] applied to a linear programming problem. Simulation- and hardware-in-the-loop campaigns resulted in comparable results, enabling the authors to pursue further research on the topic in next stages, taking various configurations of prediction horizons, performance indices using penalty functions or weighed functions at future stages of this research.

The prepared software to interface MOSEK resulted in the use of C-mex S-function template to handle the solver properly, and to enable expected deployment quality, and reliability of the solution.

The presented optimization-based solution to tracking problem offers a computationally light alternative to large computational off-line burden connected to modern genetic algorithm-related of particle swarm optimization-based solutions, leading to full tractability of the obtained results, and taking all the advantage of simplicity of the linear programming problem. What it more, it offers an additional possibility to include constraints related to, e.g., expected interplay between specific samples, adding penalties, etc., leaving the solution to the LP solver.

Acknowledgements

This research work has been funded by Poznan University of Technology under grant no. 214/SBAD/0241.

References

[1] Burns R.S., *Advanced Control Engineering*, Butterworth Heinemann (2001).

[2] Camacho E.F., Bordons C., *Model Predictive Control*, Springer-Verlag (1999).

[3] Gad A.G., *Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review*, Archives of Computational Methods in Engineering, vol. 29, pp. 2531–2561 (2022), DOI: 10.1007/s11831-021-09694-4.

[4] Goodwin G.C., Graebe S.F., Salgado M.E., *Control System Design*, Prentice-Hall (2000).

[5] Horla D., *Experimental Results on Actuator/Sensor Failures in Adaptive GPC Position Control*, Actuators, vol. 10, no. 3, pp. 1–18 (2021), DOI: 10.3390/act10030043.

[6] Katoch S., Chauhan S.S., Kumar V., *A Review on Genetic Algorithm: Past, Present, and Future*, Multimedia Tools and Applications, vol. 80, pp. 8091–8126 (2021), DOI: 10.1007/s11042-020-10139-6.

[7] Ławryńczuk M., Marusak P.M., Chaber P., Seredyński D., *Initialisation of Optimisation Solvers for Nonlinear Model Predictive Control: Classical vs. Hybrid Methods*, Energies, vol. 15, 2483 (2022), DOI: 10.3390/en15072483.

[8] Ławryńczuk M., Nebeluk R., *Computationally Efficient Nonlinear Model Predictive Control Using the L1 Cost-function*, Sensors, vol. 21, 5835 (2021), DOI: 10.3390/s21175835.

[9] Maciejowski J., *Predictive Control with Constraints*, Prentice Hall (2002).

[10] http://www.inteco.com.pl/products/modular-servo/servo/, accessed June 2023.

[11] http://www.mathworks.com/products/optimization.html, accessed June 2023.

[12] https://www.mathworks.com/products/simulink-coder.html, accessed June 2023.

[13] https://docs.mosek.com/9.0/rmosek/index.html, accessed June 2023.