https://doi.org/10.31436/iiumej.v22i1.1538

IMPROVED CONSTRAINT HANDLING APPROACH FOR PREDICTIVE FUNCTIONAL CONTROL USING AN IMPLIED CLOSED-LOOP PREDICTION

MUHAMMAD ABDULLAH[1], JOHN ANTHONY ROSSITER[2], AND ALIA FARHANA ABDUL GHAFFAR[1*]

[1]Department of Mechanical Engineering, International Islamic University Malaysia, Jalan Gombak, 53100 Kuala Lumpur, Malaysia
[2]Department of Automatic Control and Systems Engineering, The University of Sheffield, S1 3JD, UK

[*]Corresponding author: aliafarhana@iium.edu.my

(Received: 6th July 2020; Accepted: 24th September 2020; Published on-line: 4th January 2021)

ABSTRACT: Predictive Functional Control is a simple alternative to the traditional PID controller which has the capability to handle process constraints more systematically. Nevertheless, the most basic form of PFC has suffered from ill-posed prediction due to its simplicity in formulation and assumption of constant future input dynamics. Although some constraints can be satisfied, nevertheless the performance may be very conservative due to this issue. The main objective of this paper is to improve the constrained performance of a PFC controller with a minimum modification of the existing formulation. Specifically, a novel constraint handling approach for PFC is proposed based on an implied closed-loop prediction. Instead of assuming a constant input as deployed in the conventional open-loop prediction, the implied closed-loop input dynamics are utilised to detect future constraint violations. In addition, a future perturbation is introduced into the prediction structure as an extra degree of freedom for satisfying the constraints. Two simulation results confirm that the proposed approach gives far less conservative constraint handling and thus better control performance compared to the nominal PFC. Furthermore, this novel implementation also alleviates the well-known tuning difficulties and prediction inconsistency issues that are associated with conventional PFC when handling constraints.

ABSTRAK: Kawalan Kefungsian Ramalan adalah alternatif mudah kepada kawalan tradisional PID yang mempunyai kekangan keupayaan bagi mengawal proses secara lebih tersusun. Namun, keadaan paling asas pada kesan PFC adalah daripada ramalan tak terajurapi yang disebabkan oleh formula ringkas dan anggapan dinamik input yang sama bagi masa depan. Walau kekangan ini dapat diatasi, namun prestasi akan berubah secara konservatif disebabkan oleh isu ini. Objektif utama kajian ini adalah bagi membaiki kekangan prestasi kawalan PFC dengan modifikasi minimum formula yang ada. Secara spesifik, pendekatan nobel kawalan PFC dicadangkan berdasarkan ramalan lingkaran-tertutup. Selain anggapan input tetap seperti yang dilakukan pada ramalan lingkaran-terbuka yang konservatif, dinamik input yang dibuat pada lingkaran-tertutup telah digunakan bagi mengesan kekangan masa depan yang bertentangan. Tambahan, gangguan yang bakal berlaku pada masa depan telah diperkenalkan ke dalam struktur ramalan sebagai tambahan darjah pada kebebasan bagi mengatasi kekangan. Dua dapatan simulasi kajian menyetujui pendekatan yang dicadangkan dan menyebabkan sangat kurang kekangan pengendalian pada sistem konservatif, oleh itu kawalan yang lebih bagus pada prestasi berbanding pada PFC nominal. Selain itu, pendekatan nobel ini juga

menghilangkan kesukaran pelarasan yang dikenali ramai dan ramalan isu tidak konsisten yang terdapat pada PFC konvensional apabila mengendali kekangan.

KEYWORDS: *predictive functional control (PFC); constraints handling; implied closed-loop prediction; constrained predictive controller*

1. INTRODUCTION

Advances in the industrial revolution triggered the need for advanced control methods that can work within a constrained environment. Generally, constraints can be classified into two categories: i) a hard constraint refers to a physical limitation of process hardware which typically is in terms of input and output rates and ii) a soft constraint refers to a state or output limit which may be violated to some extent when required or perhaps when unavoidable. Satisfying all of these constraints can provide several benefits such as lower maintenance costs, maximisation of profits, and a safer control environment [1]. Nevertheless, dealing with constrained systems poses several challenges. For example, one often needs to deploy a classical controller such as Proportional Integral Derivative (PID) and ensuring this control loop does not violate the limits may not be straightforward or systematic in general. The more systematic, but far more expensive alternative, is to implement Model Predictive Control (MPC); this approach utilises predictions explicitly and optimises the expected behaviour by minimising a quadratic cost function subjected to predictions satisfying process constraints [2, 3]. However, conventional MPC solutions [4] are expensive and computationally demanding and hence may not be viable for applications where the costs and complexity need to be similar to those of PID.

An alternative prediction-based method which is computationally simple and relatively inexpensive is Predictive Functional Control (PFC) [5-7]. Whereas a conventional MPC algorithm uses optimisation of a quadratic cost function and the entire prediction trajectory, the most basic PFC utilises the prediction at just one point in the future, which is far simpler to compute, and avoids optimisation by forcing its future prediction to match the desired target trajectory (often first order dynamics) at a specific coincidence horizon. Again, for simplicity, a well-known assumption is that the predicted future input dynamics are taken to be constant, as this simplifies the control law definition into the solution of a single degree of freedom (d.o.f) equality expression. Thus, programming the algorithm will be simple and it can be used on a low cost processor including a Programmable Logic Controller (PLC) [5,8]. Of course, as a consequence, PFC performance is suboptimal in general (e.g. measured against typical MPC objective functions), but because it provides straightforward tuning, implementation, and constraint handling, it can be effective for a number of Single Input Single Output (SISO) processes when compared to both PID and MPC considering its low computation demand and simple coding requirement [6,9]. Indeed, these features explain the widespread acceptance and successful implementation of PFC in many real industrial applications ranging from aerospace, automotive, and chemical processes [5,10].

The current implementation of PFC in handling input and rate constraints is by simply applying a clipping or saturation method. If the current manipulated input is violating the limits, then a maximum or minimum value will be sent to the plant. This simple framework can alleviate issues such as integral wind-up, due to its use of predictions, although it may still be suboptimal [5]. As for state and output constraints, the traditional PFC approach uses multiple controllers in parallel whereby the primary regulator is tuned to track the target trajectory while the other one is tuned to satisfy the limits [11], as shown in Fig. 1. A real-time prediction is made based on the primary input by a supervisor and if it violates the limits, then the input will be switched. Although this simple concept often works and

provides a fast control solution, it can nevertheless be considered obsolete with modern computing facilities and moreover lacks sufficient rigour. Besides, it is easy to formulate scenarios where this approach fails or leads to significant performance degradation [12] and thus improvements are needed.

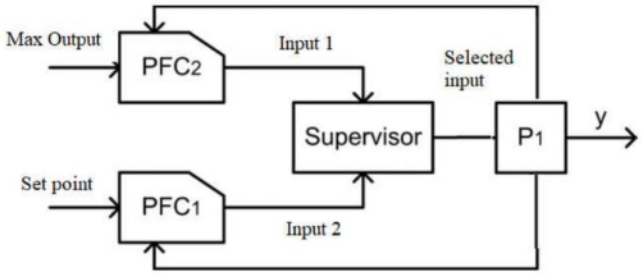

Fig. 1: Schematics of traditional PFC when handling output constraint.

A core aspect of efficient and accurate constraint handling is to ensure that the optimised predictions and the expected closed-loop behaviour are consistent [1,3, 12-13]. When there is inconsistency, ensuring that the model predictions meet constraints may be a poor reflection of whether the resulting closed-loop responses satisfy constraints, leading to either unpredicted constraint violations or the controlled response becomes very conservative in satisfying the constraints. In fact, the core weakness of the traditional PFC is its conventional assumption of its future input predictions being constant; in reality, this assumption is often inconsistent with the resulting closed-loop input to a significant degree [12]. Hence, even though PFC can check the expected constraint violations over a long prediction horizon, yet due to the prediction mismatch, the control law will produce a very conservative solution.

In previous work [12], an alternative parameterisation of the d.o.f. had been introduced within the PFC framework in order to improve the predictions and closed-loop behaviour's consistency, thus resulting in more accurate constraint handling and systematic tuning. Specifically, a Laguerre based parameterisation of the predicted future input replaced the conventional constant input assumption of PFC. This modification showed improvements in the prediction consistency with low order models and thus enabled less conservative constraint handling [14]. However, the use of a simple first order Laguerre function is still somewhat limited, and for higher-order systems, significant prediction inconsistency may still exist. While it is possible to increase the order of the Laguerre polynomial [15], this is not straightforward in general [16] and not in line with the simplicity concept which is an essential facet of PFC.

In summary, the specific objective of this new work is to propose a novel constraint handling concept for PFC, which can be applied to higher order systems and which minimises the prediction inconsistency, thus ensuring that the inequalities used for constraint handling are as accurate as possible and reducing conservatism and risk. In real applications, this is very beneficial as the system can be pushed to work nearer to its high profit boundary region without causing unexpected violations and thus fatigue damage. A core research gap in PFC is that there are no attempts in deploying the well-known concept from dual-mode MPC [3] whereby the predictions are made up by assuming a known feedback control law is in place for the asymptotic part of the predictions. In the case of PFC, a nominal control law is available for the constraint-free case and thus a simple proposal is to exploit this control law within the prediction formulation used for constraint handling. Nevertheless, the nominal PFC control law also needs to be modified further by

introducing a new slack variable or degree of freedom (d.o.f.) in the formulation to be utilised for handling the constraints. Thus, this paper will make two main contributions: (i) first, it will demonstrate how an effective d.o.f. can be incorporated alongside a suitable dual-mode formulation and (ii) second, it will show how the constraint handling can be implemented.

2. BACKGROUND ON PREDICTIVE FUNCTIONAL CONTROL

A brief review of the key assumptions, notations, and principles of conventional Predictive Functional Control laws together with a constraint handling method as proposed in [12] is described in this section.

2.1 Target Trajectory and Control Law Definition

A core principle of PFC is, at sample k, a desired target trajectory $r(k+i|k)$ is defined, where $i = 1,2,\ldots$ for future samples $k+i$ based on a desired steady-state target R. The target trajectory is defined as a *first-order-response* from the current process output $y_p(k)$ to R, hence:

$$r(k+i|k) = \left(1-\lambda^i\right)R + \lambda^n y_p(k), \quad i = 1,2,\ldots \tag{1}$$

where, λ is the implied desirable pole position. For industrial users, one can use a desired settling time, T_s and determine λ from the relationship [5]:

$$\lambda = e^{\frac{-3T}{T_s}} \tag{2}$$

with the sampling period, T. The second core principle in PFC is to force the system prediction, that is $y_p(k+n|k)$, to match the target n samples into the future, where n is representing the coincidence horizon (second tuning parameter) [5]. Hence, the control law can be summarised as:

$$r(k+n|k) = (1-\lambda^n)R + \lambda^n y_p(k) = y_p(k+n|k) \tag{3}$$

This is shown in Fig. 2 for clarity. The two *tuning parameters* or user choices are the desired closed-loop pole, λ, and the coincidence horizon, n.

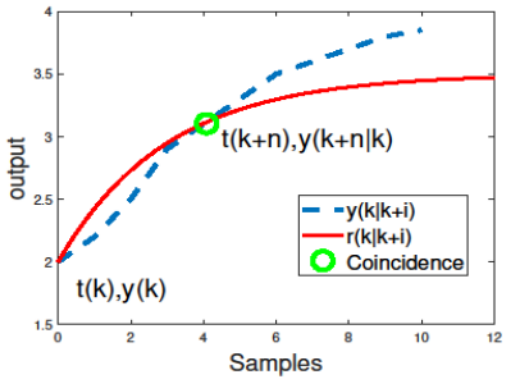

Fig. 2: PFC control law definition.

Remark 2.1: *For clarity of formulation, this work will not discuss the PFC framework that is related to a delay control problem and non-constant targets because these issues require more complex algebra and moreover have no effect on the core concepts proposed in this paper. Details are available in the references* [2,5].

https://doi.org/10.31436/iiumej.v22i1.1538

2.2 Prediction Structure and Degrees of Freedom

For solving the control law in Eq. (3), it is necessary to define the output prediction $y_p(k+n|k)$ and its dependence on the control signal $u(k|k)$. Prediction is standard in the literature [1] so just a brief illustration is given here. It is noted that this formulation ensures unbiased prediction in the steady-state and therefore offset free tracking.

Without loss of generality, the formulation in this paper utilises a state space model as it is easier to demonstrate the proposed control law in the next section. The one-step ahead output prediction for a state-space model with states, inputs and output $x(k), u(k), y_m(k)$ can be formed as:

$$x(k+1) = Ax(k) + Bu(k)\,;\;\; y_m(k+1) = Cx(k+1) \tag{4}$$

Note the use of subscript m to denote model. Typically, the real process output y_p differs from the model y_m, so process predictions are estimated using the measured signal $d(k)$ given in the structure illustrated in Fig. 3. Hence:

$$d(k) = y_p(k) - y_m(k) \quad \rightarrow \;\; y_p(k+i|k) = y_m(k+i|k) + d(k) \tag{5}$$

Consequently, the predicted plant output with the assumed predicted constant future input $(u(k) = u(k+1|k) = u(k+2|k), \ldots)$ can be written as:

$$y_p(k+i|k) = CA^i x_k + \left[CB, CAB, .. CA^{i-1}B\right] \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{L} u(k) + d(k) \tag{6}$$

Remark 2.2: *For constraint handling it is convenient to stack the output predictions into matrix form over some prediction horizon and hence one can define:*

$$\underrightarrow{y_p}(k+n|k) = Fx(k) + Hu(k) + Ld(k) \tag{7}$$

$$\underrightarrow{y_p}(k+n|k) = \begin{bmatrix} y_p(k+1|k) \\ y_p(k+2|k) \\ \vdots \\ y_p(k+n|k) \end{bmatrix};\; F = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^n \end{bmatrix};\; H = \begin{bmatrix} CB & 0 & 0 & \cdots \\ CAB & CB & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ CA^{n-1}B & CA^{n-2}B & CA^{n-3}B & \ldots \end{bmatrix}$$

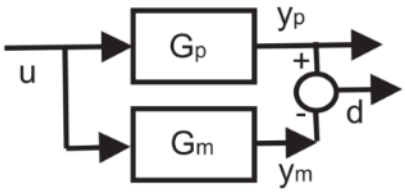

Fig. 3: Structure of an Independent Model (IM).

2.3 The PFC Control Law

The conventional PFC control law is defined by substituting the predictions in Eq. (6) into the control law definition Eq. (3). It is understood that one must first choose the desired closed-loop pole, λ ($0 < \lambda < 1$), and the coincidence horizon, n. In fact, the choice of n is less simple in general [13], but that discussion is outside the focus of this paper. Defining the n^{th} row of matrices F, H to be F_n, H_n respectively, the control law Eq. (3) is defined by solving the following for u_k (this form would be used in real implementation):

$$H_n Lu(k) + F_n x(k) + d(k) = (1 - \lambda^n)R + \lambda^n y_p(k) \quad (8)$$

Equation (8) can be solved to determine u_k but, in order to facilitate some loop analysis (nominal case), this can also be expressed by substituting $y_p(k) = y_m(k) + d(k) = Cx(k) + d(k)$. Hence, the control law Eq. (3) is presented as:

$$(1 - \lambda^n)\big(R - d(k)\big) = H_n Lu(k) + F_n x(k) - \lambda^n Cx(k) \quad (9)$$

Thus, the required control input can be represented as an augmented state feedback:

$$u(k) = \underbrace{\begin{bmatrix} \frac{-(F_n - C\lambda^n)}{H_n L} & \frac{(1-\lambda^n)}{H_n L} \end{bmatrix}}_{k} \underbrace{\begin{bmatrix} x(k) \\ \big(R - d(k)\big) \end{bmatrix}}_{\bar{x}(k)} \quad (10)$$

where K is the state feedback gain and $\bar{x}_k$ is the augmented state.

Remark 2.3: *The main reason why PFC is attractive is due to its simplicity in computing the control law of Eq. (10). Since $H_n L$ and F_n are only needed for a single horizon, n, the required computation is relatively straightforward and indeed can be calculated off-line* [2, 5].

2.4 Conventional Constraint Handling with Open-loop Predictions

A good constraint handling approach for PFC should be systematic and also simple enough to avoid the usage of common optimisers deployed in more expensive MPC algorithms and reference governor strategies [11, 17]. Let the rate, input and output constraints at every sample be defined respectively as:

$$\underline{\Delta u} \le \Delta u(k) \le \overline{\Delta u} \;;\; \underline{u} \le u(k) \le \overline{u} \;;\; \underline{y_p} \le y_p(k) \le \overline{y_p} \quad (11)$$

As discussed before, the classical PFC algorithm deploys a saturation method to deal with input and rate constraints [1,5], but that can be severely suboptimal in practice due to the difference between the input prediction and the implied closed-loop evolution. This inaccuracy is even more notable when output/state constraints are introduced in the optimised open-loop predictions, [14], as detailed next. The constraint handling procedure of PFC, as proposed by [12], is outlined as:

(1) Given a suitably long validation horizon, n_i (horizon used to check a future limit violation), the whole set of future predictions as formulated in Eq. (7) is computed.

(2) Group all the constraints (input, rate, and output) and the system predictions into a single set of linear inequalities as:

$$Mu(k) \le \boldsymbol{f}(K) \quad (12)$$

$$M = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ HL \\ -HL \end{bmatrix} ; \boldsymbol{f}(k) = \begin{bmatrix} \overline{u} \\ -\underline{u} \\ \overline{\Delta u} \\ -\underline{\Delta u} \\ L\overline{y_p} \\ -L\underline{y_p} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ u(k-1) \\ -u(k-1) \\ Fx(k) + Ld(k) \\ -Fx(k) - Ld(k) \end{bmatrix}$$

where $\boldsymbol{f}(k)$ depends on the states and the limits. It is advisable that the output predictions horizon, $\underrightarrow{y_p}(k+n)$, and the row dimension of H, need to be long enough in order to capture all the important dynamics [17].

(3) It is noted that the predictions will satisfy constraints if the selected input *u(k)* satisfies Eq. (12), that is, inequalities $Mu(k) \leq \boldsymbol{f}(k)$ can be used explicitly to determine this condition.

Next, a modern PFC constraint handling algorithm is given. This concept utilises a simple "*for*" loop to choose the $u(k)$, that is nearest to the unconstrained solution shown in Eq. (10) but does not violate the limits in Eq. (12).

Algorithm 2.1. *At each time step or sample:*

(1) Compute the unconstrained input $u(k)$ *as defined in Eq. (9).*
(2) Define the vector $\boldsymbol{f}(k)$*, from Eq. (13) where vector M is fixed.*
(3) A simple for *loop is utilised to inspect each row of M, here:*
(a) The i^{th} *constraint; represented by the* i^{th} *row of* $Mu(k) \leq \boldsymbol{f}(k)$*, is checked using* $a_i = M_i u(k) - \boldsymbol{f}_i(k)$.
(b) If $a_i > 0$*, then* $u(k) = \boldsymbol{f}_i(k)/M_i$*; else the original value of* $u(k)$ *is retained.*

It has been proven in [14] that for the nominal case where $d(k)$ is assumed constant and also for stable open-loop systems, a recursive feasibility property for Algorithm 2.1 is guaranteed where $u(k)$ converges to a feasible value nearest to the unconstrained choice.

Remark 2.4: *Since Algorithm 2.1 utilises a straightforward* for-loop*, the programming, coding, and computation becomes far simpler and faster compared to more demanding MPC approaches based on optimisation of a quadratic program. In addition, the algorithm is more systematic and faster than most of the conventional PID methods. However, it should be noted that the usage of PFC is often limited to a single-input-single-output (SISO) process and is rarely used for multi-input, multi-output (MIMO) systems [5].*

2.5 Limitations of Conventional PFC Constraint Handling

The main reason behind the popularity of PFC in industrial processes is due to its transparent tuning parameters. Here, one only needs to select the desired closed loop time constant, which is equivalent to λ. Then, the different choices of coincidence horizon, n, can be quickly explored using a computer while displaying the associated responses. Thus, a user can then select a suitable coincidence horizon, n, for their system. Nevertheless, there are two major trade-offs that need to be considered [8, 13, 15]:

(1) Smaller values of coincidence horizon, n, drive the output response closer to the target trajectory where the effect of tuning parameter, λ, becomes more significant. However, smaller, n, typically leads to poorer consistency between the predicted and actual responses, and thus inaccurate constraint handling and potentially less desirable behaviour.
(2) Conversely, larger, n, improves the prediction consistency, yet the closed-loop behaviour then approaches open-loop dynamics which decreases the significance and efficacy of λ as a tuning parameter.

Prediction mismatch is potentially catastrophic for reliable constraint handling as predictions satisfying constraints need not imply that the resulting closed-loop responses will satisfy constraints, and vice versa. Consequently, the constraint handling algorithm may be either highly conservative [12, 14] and/or overly aggressive.

https://doi.org/10.31436/iiumej.v22i1.1538

Remark 2.5: *It is well noted that one of the weaknesses of conventional PFC algorithm is its prediction mismatch [1] between its optimised predictions and the closed-loop behaviour that results. This inconsistency arises mainly due to the assumed constant future input prediction. Consistency between predictions and actual behaviour is needed to ensure good behaviour, especially for cases dealing with higher order systems.*

3. PFC CONSTRAINT HANDLING BASED ON AN IMPLIED CLOSED-LOOP PREDICTION

The main weakness of conventional PFC, as indeed with many conventional MPC algorithms, is its assumption of a constant future input. This assumption is typically inconsistent with the actual closed-loop behaviour, and thus incorrect constraint handling decisions are made [12]. To some extent, dual-mode approaches [18] can overcome this issue by ensuring the predictions are as close as possible to the actual closed-loop behaviour, but of course these are computationally demanding. The main contribution here is to exploit a similar concept, that is, to use implied closed-loop predictions for constraint handling, but in a simplified manner to maintain the core selling benefits of PFC. The proposal is novel in that dual-mode approaches still use *open-loop* predictions in transients, but with a long-term input trajectory that emulates the implied closed-loop after transients. Conversely, we will predict the expected closed loop behaviour explicitly for the entire horizon.

In summary, the expected benefits of this proposal are:

- More accurate predictions when checking output/state constraints implies less conservative decision making.
- A smaller coincidence horizon (n) can be selected without worrying about the prediction inconsistency which will retain more tuning capability.
- The proposed approach only requires a simple modification in the offline computations required for constraint handling.

3.1 Closed-loop Prediction PFC (CL-PFC) with Constant Input Perturbation

In order to use closed-loop predictions for constraint handling, an extra d.o.f. is required in the predictions. The proposal here is to add a constant perturbation to the implied closed-loop input [1] as this structure has been found to be effective elsewhere. Hence, based on the control law of Eq. (10), the closed-loop prediction model combines the open-loop model and control law, with a constant input perturbation, $c(k)$, as the new d.o.f. Thus:

$$\{\overline{x}(k+1|k) = \overline{A}\overline{x}(k) + \overline{B}u(k); u(k) = K\overline{x}(k) + c(k); c(k+i|k) = c(k), i > 0\} \quad (13)$$

$\overline{A}, \overline{B}$ are the augmented model with respect to $\bar{x}_k$.The implied closed-loop predictions take the following form (for clarity of presentation, we define $\phi = [A + \overline{B}K]$):

$$\begin{bmatrix} y_p(k+1|k) \\ y_p(k+2|k) \\ \vdots \\ y_p(k+n|k) \end{bmatrix} = \underbrace{\begin{bmatrix} C\phi \\ C\phi^2 \\ \vdots \\ C\phi^n \end{bmatrix}}_{P_c} \overline{x}(k) + \underbrace{\begin{bmatrix} C\overline{B} & 0 & 0 & \cdots \\ C\phi\overline{B} & C\overline{B} & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ C\phi^{n-1}\overline{B} & C\phi^{n-2}\overline{B} & C\phi^{n-3}\overline{B} & \ldots \end{bmatrix}}_{H_c} Lc(k) \quad (14)$$

$$\begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+n|k) \end{bmatrix} = \underbrace{\begin{bmatrix} -K \\ -K\phi \\ \vdots \\ -K\phi^{n-1} \end{bmatrix}}_{P_{cu}} \overline{x}(k) + \underbrace{\begin{bmatrix} I & 0 & 0 & \cdots \\ -K\overline{B} & C\overline{B} & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ -K\phi^{n-2}\overline{B} & C\phi^{n-3}\overline{B} & C\phi^{n-4}\overline{B} & \ldots \end{bmatrix}}_{H_{cu}} Lc(k) \quad (15)$$

https://doi.org/10.31436/iiumej.v22i1.1538

$$\begin{bmatrix} \Delta u(k|k) \\ \Delta u(k+1|k) \\ \vdots \\ \Delta u(k+n|k) \end{bmatrix} = P_{cDu}\bar{x}(k) + H_{cDu}c(k) + Q_{cDu}u(k) \quad (16$$

Remark 3.1: *For the unconstrained case, the value of perturbation as in Eq. (13) will be* $c(k) = 0$ *since the term* $u(k) = K\bar{x}_k$ *is derived from the original control law. Thus, a non-zero* $c(k)$ *is required solely for constraint handling and the underlying loop tuning is based on the original PFC law.*

3.2 Constraint Handling

For constraint handling, a similar procedure as discussed in Section 2.4, will be employed. The prime conceptual improvement here is the d.o.f., which is now expressed in terms of perturbation $c(k)$ instead of the input $u(k)$ and one needs to consider the entire input prediction and not just the first value. Thus:

$$\overline{M}c_k \leq \overline{\boldsymbol{f}}_k, \quad (17)$$

$$\overline{M} = \begin{bmatrix} H_{cu}L \\ -H_{cu}L \\ H_{cDu}L \\ -H_{cDu}L \\ H_cL \\ -H_cL \end{bmatrix}; \overline{\boldsymbol{f}}(k) = \begin{bmatrix} L\overline{u} \\ -L\underline{u} \\ L\overline{\Delta u} \\ -L\underline{\Delta u} \\ L\left(\overline{y_p} - d(k)\right) \\ -L\left(\underline{y_P} - d(k)\right) \end{bmatrix} - \begin{bmatrix} P_{cu}\bar{x}_k \\ -P_{cu}\bar{x}_k \\ P_{cu}\bar{x}_k + Q_{cDu}u(k-1) \\ -P_{cu}\bar{x}_k + Q_{cDu}u(k-1) \\ P_c\bar{x}_k \\ -P_c\bar{x}_k \end{bmatrix}$$

In a similar manner to Algorithm 2.1, the goal is to select the d.o.f. *c(k)* that is closest to its unconstrained value while satisfying the constraints in Eq. (11) (equivalently inequalities (17)).

Algorithm 3.1. *At each sample:*

(1) Set $c(k) = 0$.

(2) Define vector $\overline{\boldsymbol{f}}_k$ *as in Eq. (17) with the assumption that* $\overline{M}$ *does not change.*

(3) A simple loop is utilised to inspect each row of $\overline{M}$*, where:*

(a) The i^{th} *constraint represented by the* i^{th} *row of* $\overline{M}c(k) \leq \overline{\boldsymbol{f}}(\boldsymbol{k})$ *is checked with* $a_i = \overline{M}_i c(k) - \overline{\boldsymbol{f}}_i(k)$.

(b) If $a_i > 0$*, then* $c(k) = \overline{\boldsymbol{f}}_i(k)/\overline{M}_i$*, else use the existing* $c(k)$.

(4) Define the input from $u(k) = -K\bar{x}_k + c(k)$.

One can argue that better prediction consistency and more accurate constraint handling is obvious (and will be demonstrated in the examples section). Nevertheless, it is also important to establish that the proposed algorithm does indeed retain feasibility, that is the recursive feasibility property in the nominal case and moreover subject to changes in the target *R*.

Theorem 3.1: *When there is no change in* $d(k)$ *and also in the nominal case, Algorithm 3.1 can guarantee recursive feasibility with any desired target, R.*

Proof: Assume at sample $k-1$, the solution is feasible, then with no change in *R*, the choice $c(k) = c(k-1)$ must retain feasibility as the implied predictions at the previous and current

https://doi.org/10.31436/iiumej.v22i1.1538

sample would then be identical. Hence, it remains only to show that $c(k)$ can be chosen to retain feasibility, irrespective of any changes in R. Consider then the nominal control law given in Eq. (10) and augment this with the proposed perturbation term $c(k)$ as in Eq. (13), hence:

$$u(k) = \left[\frac{-(F_n - C\lambda^n)}{H_n L} \quad \frac{(1-\lambda^n)}{H_n L} \quad 1\right] \begin{bmatrix} x(k) \\ R - d(k) \\ c(k) \end{bmatrix} \tag{18}$$

It is clear that the terms R and $c(k)$ both act as scaled additive terms to $u(k)$, and thus any change in R can be countered by a suitable scaled change in $c(k)$ so that the predictions at sample k match those from sample $k-1$. In other words, recursive feasibility is guaranteed!

Corollary 3.2: *In practice, Algorithm 2.1 uses the smallest perturbation $c(k)$ possible so that performance is as close as possible to that desired. This also means that, if it is feasible, then $c(k)$ is driven to zero and offset free tracking will be ensured.*

The reader will note that we have not attempted to guarantee recursive feasibility subject to parameter uncertainty and disturbance changes as the literature on that area is far more complicated than would fit with the PFC philosophy: that is, simple coding and implementation. Of course, in practice, all MPC algorithms have some inherent degree of robustness and thus this lack of a guarantee is rarely a problem except for exceptional and challenging cases where one is unlikely to use PFC anyway.

4. METHODOLOGY FOR RESULT ANALYSIS

In this paper, the developed formulation will be used to set up the control algorithm for an arbitrary SISO system with constraints. Instead of quantitative analysis, this paper adopts a qualitative analysis since the main objective is to show the improvement between the proposed PFC with a conventional PFC. The control performance can be evaluated by observing its convergence speed to the desired setpoint within the constrained environment. Besides, for clarity of presentation, the obtained simulation results will not be compared with other types of controllers as that is not the main contribution of this paper and moreover, those comparisons are already well known in the existing literature. Indeed PFC performance cannot be contrasted with more comprehensive MPC alternatives as those deploy more complicated algorithms and are far more costly to implement and maintain. Interested readers can refer to these references on the comparison between MPC, PID and PFC [19, 20].

5. NUMERICAL EXAMPLES

This section provides two numerical examples with the characteristics of typical industrial processes to highlight the effectiveness of the proposed Algorithm 3.1 compared to the traditional PFC algorithm such as Algorithm 2.1. For clarity, the following comparisons are made:

(1) Investigation of the prediction behaviour of the two algorithms and their associated consistency with the resulting closed-loop dynamics.
(2) Comparison of the closed-loop controller performance when dealing with output constraints.
(3) Demonstration of the ability of the proposed controller to handle multiple constraints and uncertainties.

In order to emphasise a variety of characteristics, the examples use two second order dynamics systems as follows:

(1) A second order over-damped system model (pole at 0.4 and another pole at 0.8), with constraints of $\underline{u} = 0.12$, $\underline{\Delta u} = 0.05$, and $\overline{y} = 1.5$:

$$G_1 = \frac{z^{-1} + 0.3z^{-2}}{1 - 1.2z^{-1} + 0.32z^{-2}};\ n = 2\ ;\lambda = 0.7 \tag{19}$$

and to include the model uncertainty in the analysis, the real process representation is given by:

$$G_{1,p} = \frac{z^{-1} + 0.28z^{-2}}{1 - 1.21z^{-1} + 0.3z^{-2}} \tag{20}$$

(2) A second order non-minimum phase model (zero at 1.6 and poles at 0.6 and 0.85, respectively), with constraints of $\underline{u} = -3$, $\underline{\Delta u} = -1$, and $\overline{y} = 1.5$:

$$G_2 = \frac{0.1z^{-1} - 0.16z^{-2}}{1 - 1.45z^{-1} + 0.51z^{-2}};\ n = 5\ ;\lambda = 0.7 \tag{21}$$

and its real process representation is given by:

$$G_{2,p} = \frac{0.11z^{-1} - 0.15z^{-2}}{1 - 1.44z^{-1} + 0.5z^{-2}} \tag{22}$$

The reader is reminded that λ is selected based on the user preference, which is associated to the desired closed-loop time response (CLTR). The selection range is between $0 < \lambda < 1$. In principle, a smaller λ gives faster convergence and vice versa. Normally, the coincidence horizon, n, is taken to be at the low end of the values which gives good performance as this ensures that λ is also an effective tuning parameter.

5.1 Effect of Prediction Consistency when Identifying Future Constraint Violations

In this subsection, only the output constraint will be considered to highlight the performance improvement where the output limit for both examples is set to $\overline{y} = 0.8$. In addition, the plant transfer function is set to be equal as the model $G_p = G$. Notably, and as expected, the unconstrained control law for both PFC and CL-PFC for examples 1 and 2 are the same as shown in Figs. 3 and 4, respectively. Since PFC is a discrete controller, the results are plotted in terms of sampling instant instead of time.

However, in the presence of output constraints, their performances may differ since the adjustment is linked to different inequalities shown in Eq. (12) and Eq. (17), respectively. The first example is based on open-loop predictions, while the second example is based on implied closed-loop predictions. In this section, we present the *optimised* predictions for each algorithm, with and without constraint handling, and demonstrate the differences in the resulting decision making. The core observations are as follows:

- The unconstrained predictions at the first sample (same for both approaches) violate output constraints for large prediction horizons (refer to Figs. 5 and 6).
- Based on Fig. 7 and Fig. 8, Algorithm 2.1 (PFC, red dotted line) reduces the choice of constant $u(k)$ substantially to ensure that the maximum value of $y_p(k+i|k)$ satisfies the constraints, but the consequence is a small initial value of $u(k)$ and a much slower responding output prediction for both examples 1 and 2 respectively.

https://doi.org/10.31436/iiumej.v22i1.1538

- Conversely, Algorithm 3.1 (CL-PFC, blue dashed line) recognises that future inputs will change, thus the predictions used for constraint handling can retain the fast transients while not exceeding the constraints.

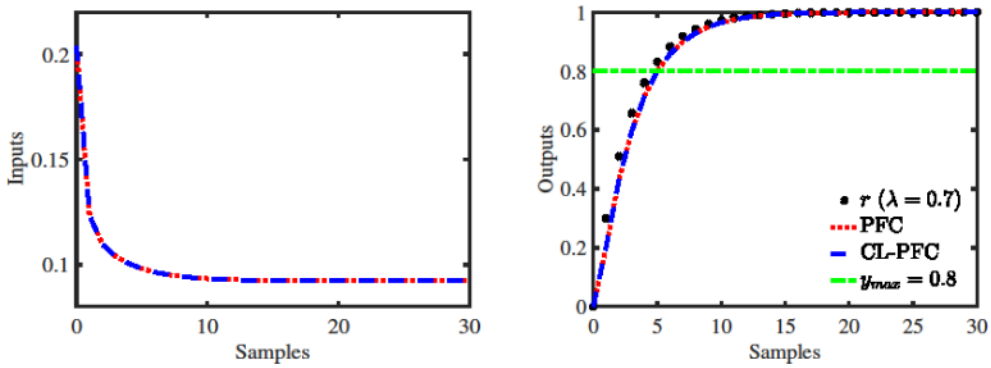

Fig. 3: Unconstrained closed-loop input and output of PFC and CP-PFC for example G_1.

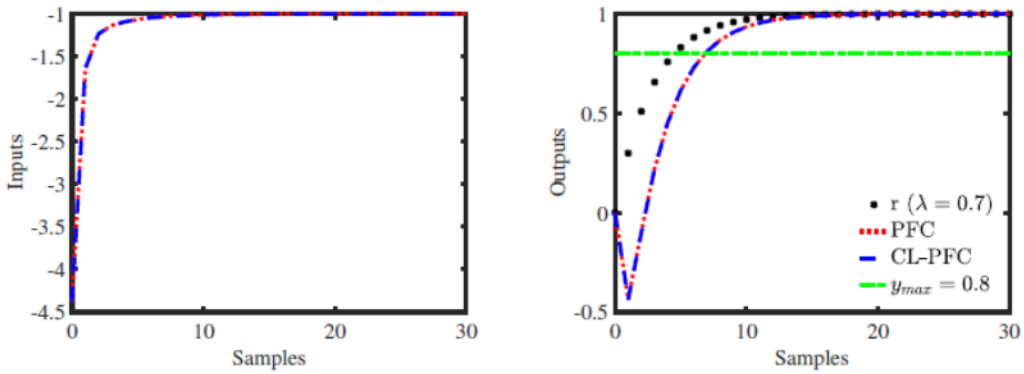

Fig. 4: Unconstrained closed-loop input and output of PFC and CP-PFC for example G_2.

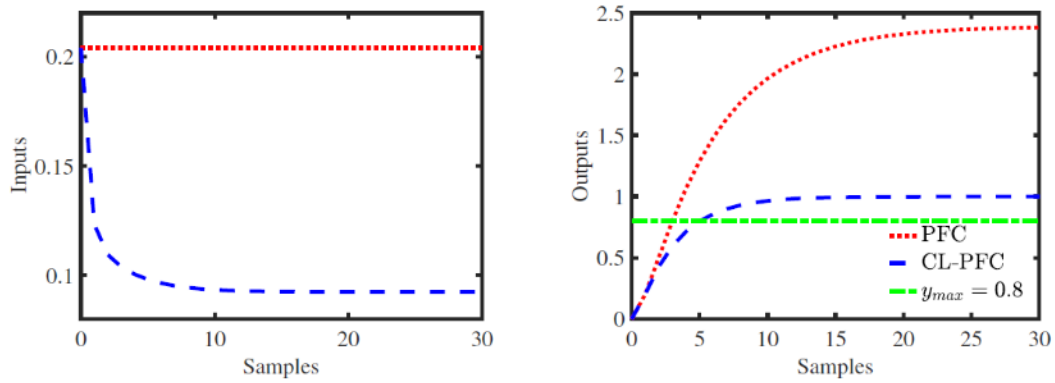

Fig. 5: Input and output predictions at the first sample of PFC and CP-PFC for example G_1.

https://doi.org/10.31436/iiumej.v22i1.1538

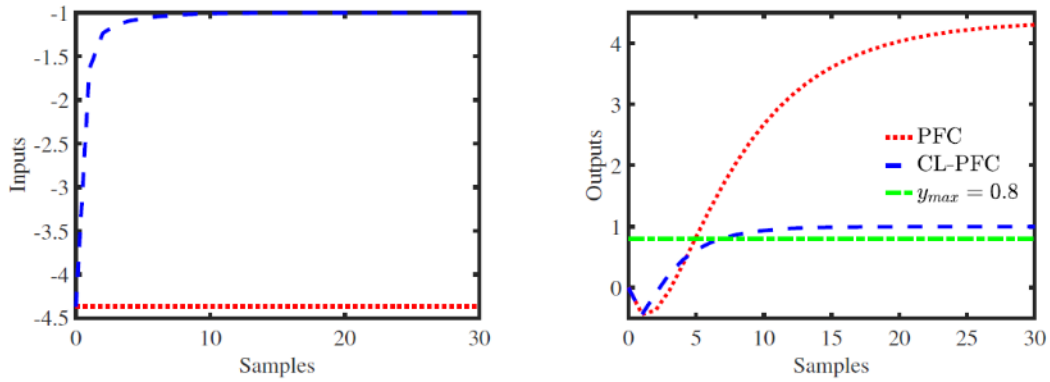

Fig. 6: Input and output predictions at the first sample of PFC and CP-PFC for example G_2.

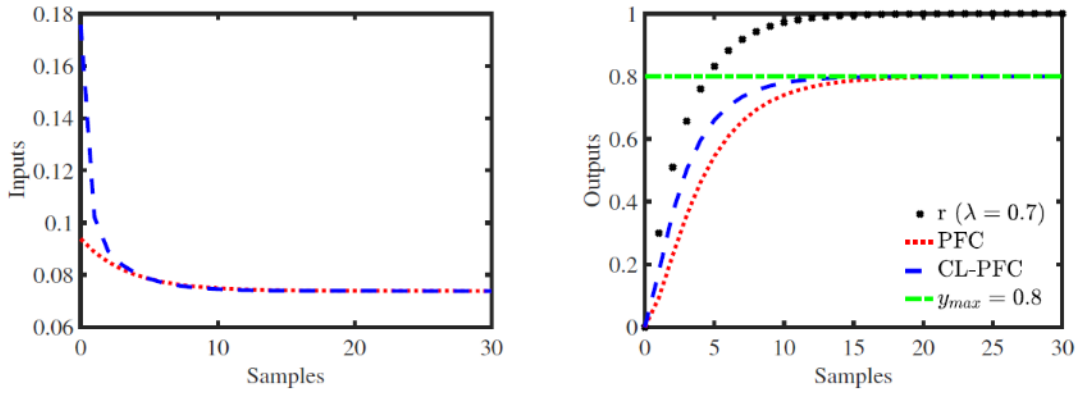

Fig. 7: Constrained input and output behaviour of PFC and CL-PFC for example G_1.

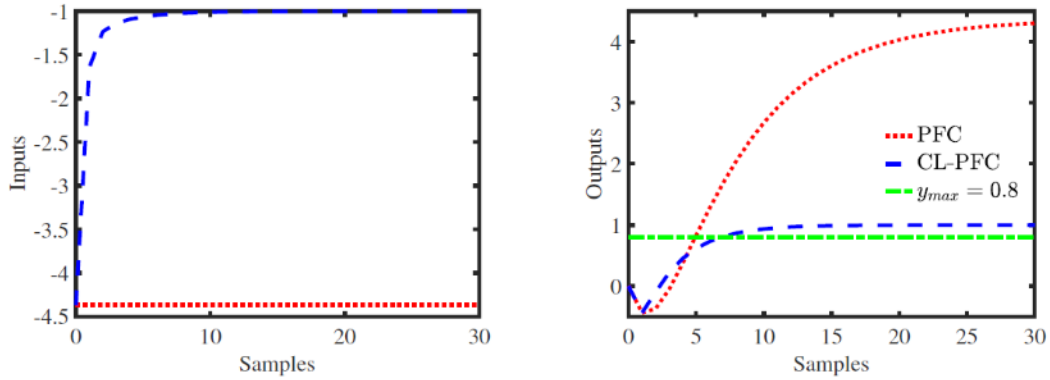

Fig. 8: Constrained input and output behaviour of PFC and CL-PFC for example G_2.

In summary, due to significant prediction mismatch in the open-loop predictions and the corresponding closed-loop behaviour in Algorithm 2.1, the constraint handling is highly conservative/suboptimal. Conversely, the proposed Algorithm 3.1 removes this mismatch thus ensuring effective and accurate constraint handling, therefore, no more loss of performance than necessary.

5.2 Handling Multiple Constraints and Uncertainties

In this section, the CL-PFC algorithm is tested with multiple constraints that include input, input rate, and output constraints to demonstrate that Algorithm 3.1 deals effectively

with all of these simultaneously. Moreover, some parameter uncertainties are introduced to demonstrate inherent robustness as discussed before in the introduction part of Section 4. In addition, to demonstrate the recursive feasibility properties more strongly, the examples include a switch from a feasible to an infeasible R. Figures 9 and 10 show the constrained closed-loop performance of CL-PFC for both example 1 and 2, respectively. It is noted that all the constraints are satisfied without a conflict and good performance is retained even in the presence of uncertainty while the system converges to the closest feasible output.

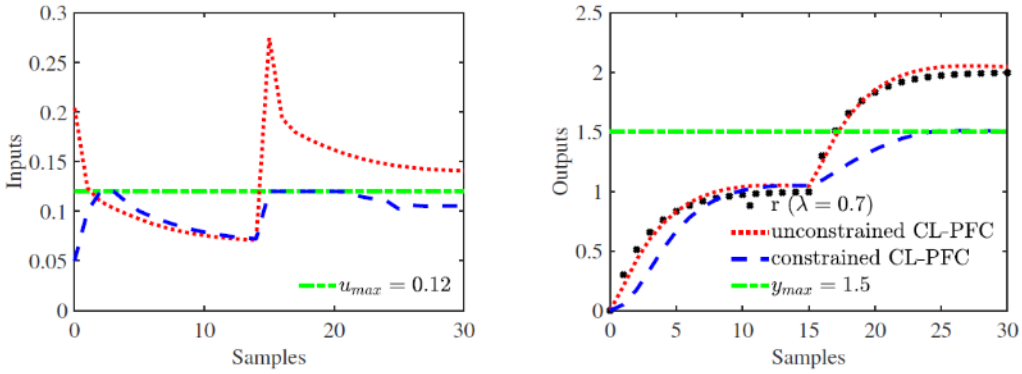

Fig. 9: Constrained and unconstrained performance for example G_1.

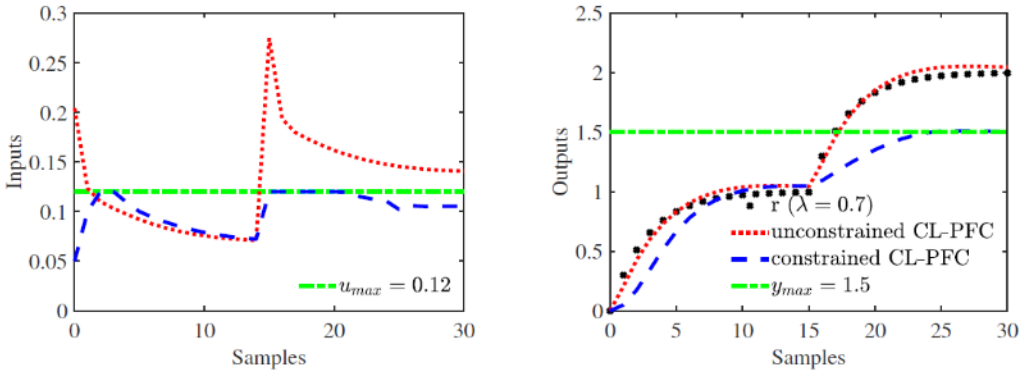

Fig. 10: Constrained and unconstrained performance for example G_2.

5.3 Possible Application and Practical Significance

As discussed in Section 1, PFC is a simple and practical controller that can be implemented in any processor with a few lines of coding. The main attraction of PFC is that it can provide a satisfactory control performance when handling constraints with minimum computation and transparent tuning parameter of the desired closed-loop time constant. It should be noted that this work does not claim that PFC is better than MPC and in fact, it can never be, due to the simplification of the algorithm. For low order processes and SISO systems, PFC can work well whereas the implementation of MPC will be very expensive for a simple application.

With the proposed framework, the user can get a less conservative control performance compared to a traditional PFC. For example, if this controller is implemented in a car, the car will slow down at the right time before cornering. Conversely, with a traditional PFC, the car will slow down long before the corner which will make it unnecessarily conservative/slow. Given the nominal performance is the same as conventional PFC, the authors expect that application of CL-PFC to real systems will demonstrate similar benefits [3, 17] in the constrained case; this constitutes future work.

6. CONCLUSIONS

This work presents a novel constraint handling approach for PFC that employs some of the recent ideas from a more conventional MPC. Specifically, by utilising the concept of an implied closed-loop prediction one can make the constraint handling decisions more accurate. With this modification, the constraint handling performance becomes more consistent and reliable when compared with the nominal PFC, as demonstrated in the numerical example simulations. With this finding, a simple system can work nearer to its constraints limits, but without constraint violation or sacrificing the desired dynamic performance. It is also worth highlighting that the required modifications for deploying this method are simple, thus in-line with the key requirements for PFC. In a real application, this benefit may be translated in terms of more production profit and a safer working environment as well as simple coding and maintenance.

Nevertheless, it should be noted that the constrained control performance is not claimed to be optimal. The underlying algorithm is still based on a simplified and far cheaper version of conventional MPC so the performance is not expected to be comparable. A few aspects which can be addressed next may include the rigorous extension of this approach to handle more challenging open-loop dynamics such as unstable, marginally stable, and oscillating systems. In addition, a more systematic and robust tuning procedure in selecting a suitable coincidence horizon in the presence of parameter uncertainty, noise and disturbances also needs to be investigated. A systematic sensitivity analysis that describes the relationship between the PFC tuning parameter and those uncertainties may provide useful information to a user before deploying this controller.

ACKNOWLEDGEMENTS

The author would like to acknowledge the International Islamic University Malaysia Research Acculturation Grant Scheme 2018 (IRAGS18-013-0014) and Ministry of Higher Education Malaysia for funding this work.

REFERENCES

[1] Rossiter, J. A. (2018). A first course in predictive control. CRC press.
[2] Haber, R., Bars, R., & Schmitz, U. (2012). Predictive control in process engineering: From the basics to the applications. John Wiley & Sons.
[3] Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. Automatica, 36(6), pp 789-814.
[4] Clarke, D. W., & Mohtadi, C. (1989). Properties of generalized predictive control. Automatica, 25(6), pp 859-875.
[5] Richalet, J., & O'Donovan, D. (2009). Predictive functional control: principles and industrial applications. Springer Science & Business Media.
[6] Richalet, J., & O'Donovan, D. (2011). Elementary predictive functional control: A tutorial. In 2011 International Symposium on Advanced Control of Industrial Processes (ADCONIP): May 2011; pp 306-313.
[7] Richalet, J., Rault, A., Testud, J. L., & Papon, J. (1978). Model predictive heuristic control. Automatica (Journal of IFAC), 14(5), pp 413-428.
[8] Khadir, M., & Ringwood, J. (2008). Extension of first order predictive functional controllers to handle higher order internal models. International Journal of Applied Mathematics and Computer Science, 18(2), pp 229-239.

[9] Haber, R., Rossiter, J. A., & Zabet, K. (2016). An alternative for PID control: predictive functional control-a tutorial. In 2016 American Control Conference (ACC): July 2016; pp. 6935-6940
[10] Richalet, J., Lavielle, G., Mallet, J., & Dreyfus, G. (2005). La commande prédictive. Eyrolles.
[11] Fiani, P., & Richalet, J. (1991). Handling input and state constraints in predictive functional control. In Proceedings of the 30th IEEE Conference on Decision and Control: December 1991; pp. 985-990.
[12] Abdullah, M., & Rossiter, J. A. (2019). Using Laguerre functions to improve the tuning and performance of predictive functional control. International Journal of Control, pp 1-13.
[13] Rossiter, J. A., & Haber, R. (2015). The effect of coincidence horizon on predictive functional control. Processes, 3(1), pp 25-45.
[14] Abdullah, M., Rossiter, J. A., & Haber, R. (2017). Development of constrained predictive functional control using Laguerre function based prediction. IFAC-PapersOnLine, 50(1), pp 10705-10710.
[15] Khan, B., & Rossiter, J. A. (2013). Alternative parameterisation within predictive control: a systematic selection. International Journal of Control, 86(8), pp 1397-1409.
[16] Muehlebach, M., & D'Andrea, R. (2017). Basis functions design for the approximation of constrained linear quadratic regulator problems encountered in model predictive control. In 2017 IEEE 56th Annual Conference on Decision and Control (CDC): December 2017; pp. 6189-6196.
[17] Gilbert, E. G., & Tan, K. T. (1991). Linear systems with state and control constraints: The theory and application of maximal output admissible sets. IEEE Transactions on Automatic control, 36(9), pp 1008-1020.
[18] Scokaert, P. O., & Rawlings, J. B. (1998). Constrained linear quadratic regulation. IEEE Transactions on automatic control, 43(8), pp 1163-1169.
[19] Valencia-Palomo, G., & Rossiter, J. A. (2012). Comparison between an auto-tuned PI controller, a predictive controller and a predictive functional controller in elementary dynamic systems. online publication, research gate.
[20] Haber, R., Rossiter, J. A., & Zabet, K. (2016). An alternative for PID control: predictive functional control-a tutorial. In 2016 American Control Conference (ACC): July 2016; pp. 6935-6940.