Embedded Subspace Predictive Control, a Model Based Design Approach

Rafael B. C. Lima and Péricles R. Barros

***Abstract*—This paper will present a model-based methodology for implementing an embedded, data-driven, subspace predictive controller. The control algorithm captures, in real-time, the characteristics of the plant and self-tunes through process modifications. A data-driven subspace approach is used, in which predictors are calculated directly from input and output data projections. Coding and performance aspects are discussed and tested in Model-In-the-Loop, Software-In-the-Loop and real application.**

***Index Terms*—Subspace Predictive Controller, Embedded Systems, Model Based Design.**

I. Introdução

A crescente demanda por produtos cada vez mais complexos vem motivando as equipes de desenvolvimento a explorarem diferentes metodologias de projeto em busca de maior produtividade. No contexto de sistemas embarcados, o projeto baseado em modelo (MBD - *Model Based Design*) está emergindo como uma solução eficiente para o fluxo de desenvolvimento de uma solução completa [1], [2]. A evolução dos *firmwares*, em especial, vem se tornando um diferencial para o sucesso dos produtos no mercado.

Controladores PID (Proporcional Integral e Derivativo) são, sem dúvida, os mais utilizados em ambientes industriais [3]. Essa dominância do mercado se justifica pela sua robustez e, principalmente, pela sua simplicidade. Porém, tal simplicidade traz desvantagens quando os sistemas a serem controlados são complexos e demandam critérios de controle exigentes. Devido aos recentes avanços nos ambientes de desenvolvimento, o projeto de sistemas de controle complexos pode ser implementado utilizando as facilidades do MBD e da geração automática de código para plataformas microcontroladas comerciais. Essa abordagem possibilita ao engenheiro de controle o foco no projeto dos controladores em si, uma vez que os detalhes de codificação, específicos para a plataforma, são abstraídos pelos pacotes de geração automática de código.

Controladores preditivos têm sido utilizados em aplicações industriais complexas por mais de três décadas. Várias formulações foram desenvolvidas tais como: DMC (*Dynamic Matrix Control*) [4], PFC (*Predictive Functional Control*) [5], GPC (*Generalized Predictive Control*) [6], (*State-space Predictive Control*) [7] etc. Em geral o termo controle preditivo é designado para estratégias de controle que utilizem um modelo do processo para prever suas saídas futuras e gerar uma entrada de controle que minimize uma certa função de custo. Uma revisão detalhada sobre os métodos MPC (*Model Predictive Control*) pode ser encontrada em [8].

Os objetivos básicos de métodos de controle MPC podem ser resumidos do seguinte modo:

- Prevenir violações nas restrições de entradas e saídas;
- Conduzir as saídas para suas referência especificadas;
- Prevenir desgaste excessivo dos atuadores.

Para realizar esses objetivos, a grande maioria dos métodos MPC se baseia nas seguintes ideias:

- Uso explícito de modelos para predição das saídas do sistema em instantes futuros;
- Lei de controle dependente das predições;
- Cálculo da sequência de controle por meio da minimização de uma função de custo;
- A utilização de um horizonte de predição móvel que é deslocado a cada instante de tempo.

Determinar um modelo dinâmico do processo é o primeiro passo em qualquer estratégia MPC [3]. Na maioria das aplicações existe uma etapa de modelagem para obtenção de modelos paramétricos, muitas vezes o passo mais demorado e oneroso no projeto de controle.

Nesse artigo será apresentada uma metodologia baseada em modelos para implementação de uma variação de controladores preditivos em sistemas embarcados. É apresentado um algoritmo de controle que capture, em tempo real, as características da planta e se auto-sintonize mediante mudanças no processo. Para atingir tais objetivos, será utilizada uma abordagem por subespaços voltada a dados, na qual os preditores são determinados diretamente a partir dos dados de operação de entrada e saída por meio de projeções, eliminando a necessidade da escolha de modelos paramétricos para a planta. Aspectos de codificação e desempenho serão discutidos em virtude das limitações intrínsecas de *hardwares* microcontrolados. Mais detalhes de métodos SPC (*Subspace Predictive Control*) podem ser encontrados em [9], [10], [11] e [12].

O trabalho está organizado conforme segue: na seção II são apresentados os conceitos de desenvolvimento baseado em modelo (MBD), na seção III é demonstrada a formulação matemática do algoritmo de controle preditivo proposto, na seção IV é descrita a planta termoelétrica na qual o sistema será validado, seguido pela análise dos resultados na seção V e, por fim, as conclusões na seção VI.

II. Projeto Baseado em Modelo

A metodologia MBD é uma abordagem centrada em modelos para o desenvolvimento de sistemas de controle, processamento de sinais, comunicações entre outros sistemas

R. B. C. Lima, Department of Electrical Engineering, Federal University of Campina Grande, Brazil, e-mail: rafael.lima@dee.ufcg.edu.br.

P. R. Barros, Department of Electrical Engineering, Federal University of Campina Grande, Brazil, e-mail: prbarros@dee.ufcg.edu.br.

dinâmicos [1]. Ao invés de partir de protótipos físicos ou especificações textuais, o MBD foca em um modelo que inclui todos os componentes relevantes do sistema: algoritmos, lógica de controle, componentes físicos e propriedade intelectual. Depois de desenvolvido, o modelo torna-se fonte de várias saídas, incluindo relatórios, código C e código HDL. O projeto baseado em modelo permite o desenvolvimento em nível de sistema ou componente, simulações, geração automática de código e testes contínuos. Um fluxo de trabalho típico está representado na Fig. 1.

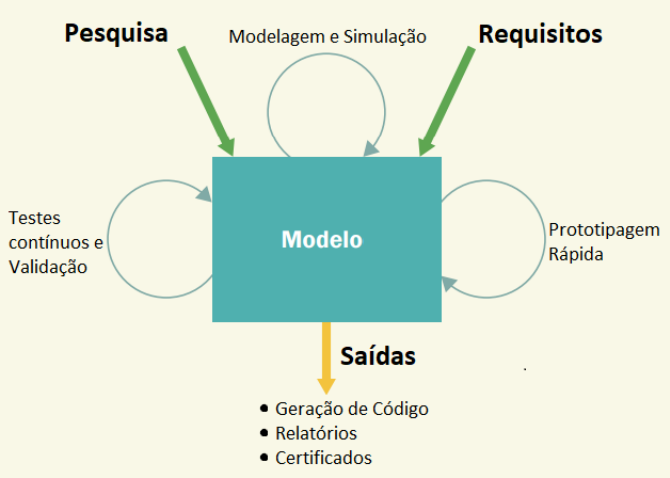

Fig. 1. Fluxo de trabalho no MBD.

A. Testes Estruturados

Uma das grandes vantagens no MBD é que os testes podem começar ainda na fase de requisitos, possibilitando simular as especificações e verificar se os critérios são atendidos. Como resultado, os defeitos são localizados e solucionados nas fases iniciais do projeto, reduzindo o custo total do desenvolvimento [13], [14]. No processo de desenvolvimento com MBD podem ser aplicados os testes em *Model-In-the-Loop* (MIL), *Software-In-the-Loop* (SIL), *Processor-In-the-Loop* (PIL) e *Hardware-In-the-Loop* (HIL).

- MIL - consiste em realizar simulações dos modelos da planta e do controlador em um mesmo ambiente computacional. Essas simulações são realizadas em linguagem nativa da ferramenta utilizada no desenvolvimento dos modelos, permitindo um completo domínio sobre todo o sistema simulado;
- SIL - as simulações dos modelos da planta e do controlador ainda são realizadas no mesmo ambiente computacional. Porém, a linguagem utilizada já é a nativa do dispositivo alvo, nesse caso C/C++;
- PIL - o modelo do controlador, convertido em código objeto (C/C++), passa a ser executado pelo microcontrolador, enquanto a planta continua a ser simulada em linguagem nativa da ferramenta utilizada para o desenvolvimento dos modelos. Uma camada de código é adicionada ao *firmware* para permitir o envio/recebimento de dados entre a ferramenta de simulação e o microcontrolador;
- HIL - nesse caso um computador de tempo real executa o código da planta e fornece interfaces de entrada e saída similares ao sistema real, enquanto o algoritmo de controle é executado no microcontrolador;
- Sistema real - por fim, os testes são realizados com o sistema embarcado interligado à planta, possibilitando verificar o funcionamento da lógica de controle no microcontrolador em conjunto como as interfaces de entrada/saída.

O objetivo desses testes é a verificação e validação dos requisitos do sistema em diferentes níveis do desenvolvimento onde, à medida que o andamento da solução avança, o funcionamento e o desempenho do sistema são comparados em cada fase de testes.

B. Geração Automática de Código

A geração automática do código executável, a partir do modelo simulado, é um dos grandes atrativos para a adoção da metodologia MBD na implementação de sistemas complexos. Um dos propósitos é a redução do tempo de desenvolvimento em relação à codificação manual, bem como, a mitigação de erros provenientes dela. A utilização de uma camada a mais de abstração na codificação provavelmente ocasionará a inclusão de linhas de código desnecessárias. Para homogenizar a comparação entre os cenários propostos, optou-se por não modificar manualmente os códigos gerados.

Na elaboração da solução apresentada neste trabalho, os modelos foram implementados no ambiente do Matlab®/Simulink® e convertidos automaticamente em código executável para o microcontrolador *Arduino Mega 2560 R3* a partir do pacote *Simulink® Support Package for Arduino® Hardware*.

III. Controlador Preditivo por Subespaços

Os controladores preditivos por subespaços são uma das estratégias de controle mais inovadoras da indústria na última década [15]. O SPC foi introduzido pela primeira vez em [16] e é baseado na combinação dos preditores dos algoritmos de identificações por subsespaços e métodos de controle preditivo baseados em modelos. No SPC, as matrizes dos preditores são obtidas diretamente de dados experimentais de entrada e saída, o que elimina a identificação de um modelo paramétrico intermediário. Suas principais vantagens em aplicações práticas estão no fato de não serem necessárias informações a priori, ser um método não iterativo assim como não ser necessário resolver equações diofantinas [10].

A. Definição dos Preditores

Assumindo que o sistema modelado tem características lineares, invariantes ao deslocamento e é multivariável, existe uma representação determinística em espaço de estados, dada por:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t), \end{aligned} \tag{1}$$

onde $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$, $x(t) \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, n é a ordem do modelo, m a quantidade de entradas e p a quantidade de saídas.

A (1) pode ser reescrita de forma matricial

$$Y_f = \Gamma_i X_f + T_{0|i-1} U_f, \tag{2}$$

onde U_f e Y_f são respectivamente as matrizes de Hankel dos dados de entradas e saídas futuras, X_f é a sequência de estados futuros, Γ_i a matriz estendida de observabilidade e $T_{0|i-1}$ uma matriz Toeplitz dos coeficientes da resposta ao impulso do sistema.

As matrizes $W_p \triangleq \binom{U_p}{Y_p}$ e X_f compartilham o mesmo espaço linha, onde U_p e Y_p são as matrizes de Hankel dos dados de entradas e saídas passadas. Portanto um preditor linear das saídas futuras $\hat{Y_f}$ pode ser definido como:

$$\hat{Y_f} = L_w W_p + L_u U_f = \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix}. \tag{3}$$

O preditor das saídas é a solução do problema de minimização

$$\min_{L_w, L_u} \left\| Y_f - \begin{pmatrix} L_w & L_u \end{pmatrix} \begin{pmatrix} W_p \\ U_f \end{pmatrix} \right\|_F^2, \tag{4}$$

que por sua vez é determinado pela projeção ortogonal do espaço linha de Y_f no espaço linha de $\begin{pmatrix} W_p \\ U_f \end{pmatrix}$, dado por:

$$\hat{Y_f} = Y_f / \begin{pmatrix} W_p \\ U_f \end{pmatrix} = Y_f \begin{pmatrix} W_p \\ U_f \end{pmatrix}^{\dagger} \begin{pmatrix} W_p \\ U_f \end{pmatrix}, \tag{5}$$

onde o operador $X^{\dagger}$ indica a pseudoinversa de X. Substituindo (3) em (5) os coeficientes do preditor $\begin{pmatrix} L_w & L_u \end{pmatrix}$ são calculados como:

$$\begin{pmatrix} L_w & L_u \end{pmatrix} = Y_f \begin{pmatrix} W_p \\ U_f \end{pmatrix}^{\dagger}. \tag{6}$$

A implementação numérica de (6) pode ser realizada de uma forma bastante eficiente por meio da decomposição QR [17]

$$\begin{bmatrix} W_p \\ U_f \\ Y_f \end{bmatrix} = \begin{bmatrix} R_{11} & 0 & 0 \\ R_{21} & R_{22} & 0 \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} Q_1^T \\ Q_2^T \\ Q_3^T \end{bmatrix}, \tag{7}$$

$$\begin{pmatrix} L_w & L_u \end{pmatrix} = \begin{pmatrix} R_{31} & R_{32} \end{pmatrix} \begin{pmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{pmatrix}^{\dagger}, \tag{8}$$

portanto, quando o número de colunas das matrizes de dados de Hankel $j \to \infty$, são obtidos os resultados da Tabela I.

TABELA I
RELAÇÕES IMPORTANTES.

$L_w W_p = \Gamma_i X_f$
$L_u = T_{0\|i-1}$

B. Lei de Controle

Definição do problema: Dada uma sequência de referência futura $r(t)$ onde $t \in \{1, .., N\}$ e medições das entradas $u(t)$ e saídas passadas $y(t)$ para $t \in \{-N, .., 0\}$, encontrar a sequência de controle futura $u_f = (u(1), .., u(N))$ tal que a função de custo J seja minimizada

$$J = \sum_{t=1}^{N} (\hat{y}(t) - r(t))^T (\hat{y}(t) - r(t)) + u(t)^T \lambda I u(t), \tag{9}$$

onde $\hat{y}(t)$ é o preditor da saída e $\lambda \in (0, \infty)$ é o termo que pondera a penalização da energia fornecida ao atuador em detrimento ao rastreio do sinal de referência. $0 < \lambda < 1$ prioriza o rastreamento e $\lambda > 1$ minimiza a energia do sinal de controle.

Definindo os vetores

$$u_f = \begin{pmatrix} u(1) \\ u(2) \\ .. \\ u(N) \end{pmatrix}, \hat{y}_f = \begin{pmatrix} \hat{y}(1) \\ \hat{y}(2) \\ .. \\ \hat{y}(N) \end{pmatrix}, \tag{10}$$

$$r_f = \begin{pmatrix} r(1) \\ r(2) \\ .. \\ r(N) \end{pmatrix}, y_p = \begin{pmatrix} y(-N+1) \\ y(-N+2) \\ .. \\ y(0) \end{pmatrix}, \tag{11}$$

$$u_p = \begin{pmatrix} u(-N+1) \\ u(-N+2) \\ .. \\ u(0) \end{pmatrix}, w_p = \begin{pmatrix} y_p \\ u_p \end{pmatrix}, \tag{12}$$

a função de custo pode ser reescrita como:

$$J = (\hat{y}_f - r_f)^T (\hat{y}_f - r_f) + u_f^T \lambda I u_f. \tag{13}$$

Substituindo (3) em (13)

$$J = (L_w w_p + L_u u_f - r_f)^T (L_w w_p + L_u u_f - r_f) + u_f^T \lambda I u_f. \tag{14}$$

O valor mínimo de J é obtido quando a derivada do traço de J em relação a u_f é zero

$$\frac{dTrace(J)}{du_f} = 0, \tag{15}$$

portanto

$$u_f = (\lambda I + L_u^T L_u)^{-1} L_u^T (r_f - L_w w_p). \tag{16}$$

Nota-se que a etapa de identificação da planta se resume à montagem das matrizes de Hankel dos sinais de entrada e saída e uma simples decomposição QR, dada por (8). É importante salientar que existem inúmeras ferramentas robustas para cálculo e atualização da decomposição QR. Essa abordagem, no entanto, não garante erro zero em regime estacionário a menos que exista um integrador no processo controlado [18].

C. Inclusão da Ação Integral

Uma das abordagens para incluir a ação integral no algoritmo de controle é re-escrever a representação em espaço de estados com um novo mapeamento $\xi(t) = x(t) - x(t-1)$

$$\xi(t+1) = A\xi(t) + B\Delta u(t)\Delta y(t) = C\xi(t) + D\Delta u(t), \quad (17)$$

onde $\Delta u(t)$ e $\Delta y(t)$ são, respectivamente, as variações do sinal de controle e da saída no instante t. De forma similar a (3), o preditor da variação da saída é dado por

$$\Delta\hat{y}_f = L_w\Delta W_p + L_u\Delta u_f, \quad (18)$$

em que

$$\Delta u_f = \begin{pmatrix} \Delta u(1) \\ \Delta u(2) \\ .. \\ \Delta u(N) \end{pmatrix}, \Delta y_f = \begin{pmatrix} \Delta y(1) \\ \Delta y(2) \\ .. \\ \Delta y(N) \end{pmatrix}, \quad (19)$$

$$\Delta w_p = \begin{pmatrix} \Delta y_p \\ \Delta u_p \end{pmatrix}. \quad (20)$$

É mostrado em [9] que (18) pode ser reescrita como

$$\hat{y}_f = \frac{1}{\Delta}L_w\Delta W_p + \frac{1}{\Delta}L_u\Delta u_f, \quad (21)$$

$$\hat{y}_f = \mathbf{y}(t) + \Pi L_w\Delta W_p + \Pi L_u\Delta u_f, \quad (22)$$

$$\hat{y}_f = \mathbf{y}(t) + L_{wI}\Delta W_p + L_{uI}\Delta u_f, \quad (23)$$

$$\hat{y}_f = F + L_{uI}\Delta u_f, \quad (24)$$

onde

$$\mathbf{y}(t) = \begin{pmatrix} y(t) \\ y(t) \\ .. \\ y(t) \end{pmatrix}, \quad \Pi = \begin{pmatrix} I & 0 & 0 & 0 \\ I & I & 0 & 0 \\ .. & .. & .. & 0 \\ I & I & .. & I \end{pmatrix}, \quad (25)$$

$$L_{wI} = \Pi L_w, \qquad L_{uI} = \Pi L_u, \quad (26)$$

e F é a resposta livre da saída

$$F = \mathbf{y}(t) + L_{wI}\Delta W_p. \quad (27)$$

Reescrevendo a função de custo, J se torna

$$J = (r_f - F - L_{uI}\Delta u_f)^T(r_f - F - L_{uI}\Delta u_f) + \Delta u_f^T\lambda I\Delta u_f. \quad (28)$$

Derivando $\frac{dTrace(J)}{d\Delta u_f} = 0$, resulta na seguinte lei de controle

$$\Delta u_f = (L_{uI}^T L_{uI} + \lambda I)^{-1}L_{uI}^T(r_f - F). \quad (29)$$

Essa abordagem gera um controlador preditivo *online*, no qual os preditores são recalculados a cada instante de tempo com a chegada de novas informações das entradas e saídas. Dessa forma é necessário calcular, a cada novo instante, uma decomposição QR (7), assim como uma inversa de matriz (29). Caso a planta seja invariante ao tempo pode-se utilizar uma abordagem alternativa em que é realizado um experimento em malha aberta inicial, no qual será baseado o cálculo dos preditores somente uma vez. Assim, parte da Lei de controle em (29) se torna constante, dependendo apenas do parâmetro de ajuste λ

$$\Lambda = (L_{uI}^T L_{uI} + \lambda I)^{-1}L_{uI}^T, \quad (30)$$

$$\Delta u_f = \Lambda(r_f - F), \quad (31)$$

nessa forma alternativa Λ é fixo e somente o termo $(r_f - F)$ é atualizado a cada instante de amostragem. Tornando o controlador menos adaptável a variações da planta, porém mais rápido em termos computacionais. Mais detalhes sobre o desenvolvimento da lei de controle podem ser encontrados em [9].

Algoritmo 1: **Algoritmo de controle preditivo por subespaços**

- Calcule os preditores L_u e L_w por meio de (8);
- Calcule os preditores diferenciais L_{uI} e L_{wI} por meio de (26);
- Calcule F por meio de (27);
- Feche a malha e aplique somente a primeira entrada da lei de controle (29) a cada intervalo de amostragem.

D. Saturação da Entrada de Controle

A obtenção da lei de controle com ação integral, dada por (29), não leva em consideração os valores máximos e mínimos aplicáveis na entrada de controle em uma situação real. Tal saturação pode levar a uma degradação de desempenho devido a erros do tipo *windup* ou estouro de integrador.

Uma alternativa para solucionar esse problema seria formular a lei de controle como um problema de otimização e adicionar as limitações do sinal de controle como restrições. Neste trabalho optou-se por seguir a estratégia do Algoritmo 2 e evitar o aumento da complexidade numérica gerado pela inclusão de um problema de otimização adicional.

Algoritmo 2: ***Anti-Windup***

- Caso $u \geq u_{MAX}$ e $\Delta u_f(1) > 0$, aplicar $\Delta u_f(1) = 0$;
- Caso $u_{MIN} < u < u_{MAX}$, aplicar $\Delta u_f(1)$ calculado;
- Caso $u \leq u_{MIN}$ e $\Delta u_f(1) < 0$, aplicar $\Delta u_f(1) = 0$.

IV. Planta Piloto Termoelétrica

O controlador preditivo embarcado será validado em uma planta térmica didática projetada pelos próprios autores em [19] e [20]. A planta é centrada em um módulo Peltier que possibilita o aquecimento ou resfriamento de uma massa metálica. Uma das faces da pastilha Peltier é acoplada a um dissipador de calor e um ventilador, com o objetivo de melhorar a troca de calor com o ambiente. A outra face é acoplada termicamente a uma peça cilíndrica de alumínio sólido, conforme a Fig. 2. Nesta peça de alumínio, está fixado um sensor de temperatura do tipo LM35.

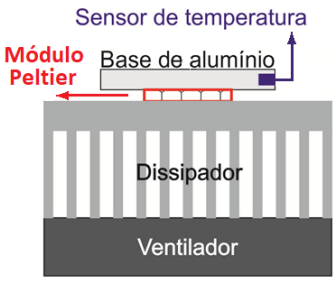

Fig. 2. Módulo de temperatura.

A planta piloto é composta pelo módulo Peltier de temperatura, circuitos de acionamento em ponte H, um sensor de

temperatura LM35, um amplificador para o sensor, uma fonte de alimentação ATX, um micro controlador *Arduino Mega 2560 R3* e um PC rodando o Matlab®/Simulink®, conforme ilustrado na Fig. 3.

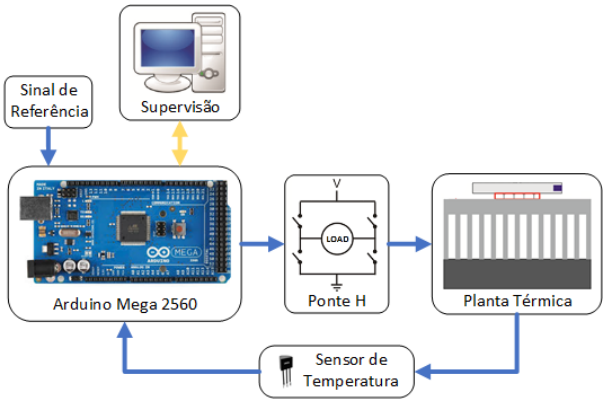

Fig. 3. Diagrama da planta.

Para o intuito de simulações, o comportamento dinâmico da planta será representado, pela função de transferência de primeira ordem, indicada em (32), que relaciona a tensão média aplicada na pastilha Peltier e a temperatura na massa metálica acoplada em sua face.

$$G(s) = \frac{0,42}{53s + 1}. \tag{32}$$

Esse modelo foi obtido experimentalmente analisando a resposta ao degrau da planta em torno do ponto de operação.

V. Análise dos Resultados

O fluxo de trabalho escolhido para o desenvolvimento da solução proposta envolve testes de *Model-in the-Loop* (MIL), *Processor-in the-Loop* (PIL) e no Sistema Real. Cada um desses é utilizado para validar um aspecto diferente do projeto. Os diagramas de testes são ilustrados na Fig. 4.

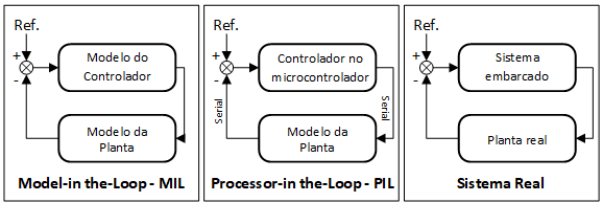

Fig. 4. Diagrama de testes.

Para avaliar o desempenho da saída em malha fechada é utilizado o índice IAE (*Integrated Absolute Error*), dado por:

$$IAE = \sum_{t=0}^{\infty} |r(t) - y(t)|, \tag{33}$$

onde $r(t)$ representa o sinal de referência e $y(t)$ a saída do processo.

Para avaliar a entrada de controle, $u(t)$, é utilizado o índice TV (*Total Variation*) definido pela soma de todas as variações do sinal de controle,

$$TV = \sum_{t=0}^{\infty} |u(t+1) - u(t)|, \tag{34}$$

quanto menor o TV mais suave é o sinal de controle.

A. Processor-in the-Loop (PIL)

Invertendo o fluxo usual de trabalho, primeiramente foram executados testes de PIL para investigar os limites numéricos da implementação do controlador no microcontrolador. Conforme (8), (16) e (29) a complexidade numérica do algoritmo de controle cresce proporcionalmente aos horizontes de controle e predição. Assumindo ambos os parâmetros iguais (N_h), foram realizados testes iniciais de desempenho para determinar os valores máximos de N_h que ainda possibilitariam que o algoritmo de controle fosse executado dentro da janela de tempo disponível.

É ilustrada na Fig. 5 a variação do consumo de memória RAM do microcontrolador em função do aumento dos horizontes de controle e predição. Observando o gráfico, concluise que o limite máximo é $N_h = 19$, no qual $99,1\%$ (8118 bytes) da memória SRAM é utilizada. O número de instruções do código gerado não sofre variação significativa em função do aumento de N_h, ocupando aproximadamente $11,8\%$ (29,2 kbytes) da memória *flash* do dispositivo. Os percentuais de ocupação das memórias de dados e instruções foram extraídos do log do pacote de geração automática de código, cada vez que o projeto foi recompilado.

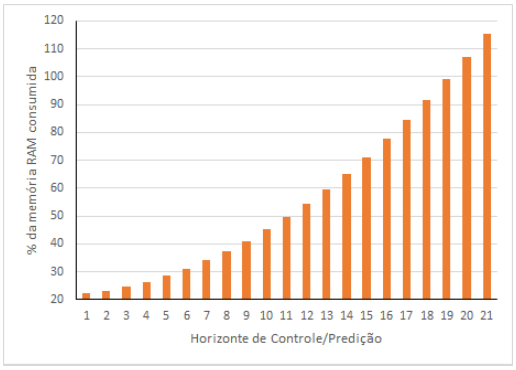

Fig. 5. Comparação da consumo de memória SRAM variando os horizontes de controle/predição.

É ilustrado na Fig. 6 o tempo de execução do algoritmo de controle em função do aumento do horizonte N_h. Caso utilize-se o limite máximo da memória RAM ($N_h = 19$), o tempo de execução do algoritmo será de, aproximadamente, $T = 0,18$ s. Fator limitante da amostragem do laço de controle.

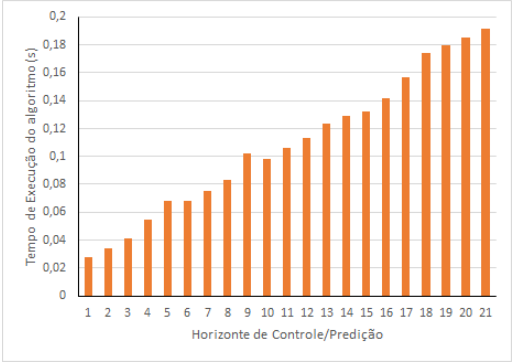

Fig. 6. Comparação do tempo de execução do algoritmo de controle variando os horizontes de controle/predição.

B. *Model-in the-Loop (MIL)*

Nessa seção, os modelos do controlador, dado por (29), e da planta (32), são utilizados para comparação de diferentes sintonias ao variar o parâmetro λ.

É utilizado um sinal de teste em malha aberta para o cálculo dos preditores. Por simplicidade é aplicado um pulso na entrada, com amplitude unitária e duração de 200 s. Os sinais, amostrados com 2 s, são ilustrados na Fig. 7.

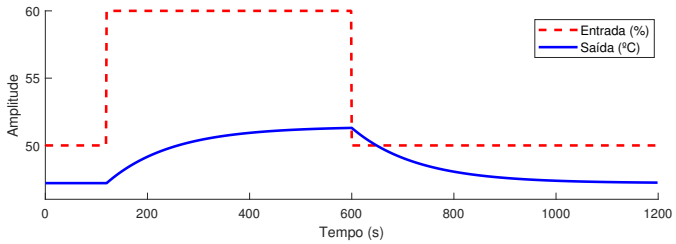

Fig. 7. Gráfico dos sinais de teste em malha aberta.

Uma das vantagens do MBD é a possibilidade de gerar inúmeros cenários simulados. Dentre eles, foram escolhidos 8 sintonias representativas, variando o parâmetro $\lambda = \{4, 20, 50, 100, 300, 500, 1000\}$. O horizonte foi mantido fixo em $N_h = 19$.

São ilustradas na Fig. 8 as curvas das saídas para uma mudança de referência em $t = 50$ s e uma perturbação de carga em $t = 800$ s. Quanto menor o λ mais rápido o sistema responde, porém aumentam as características oscilatórias, ocasionando *overshoot*.

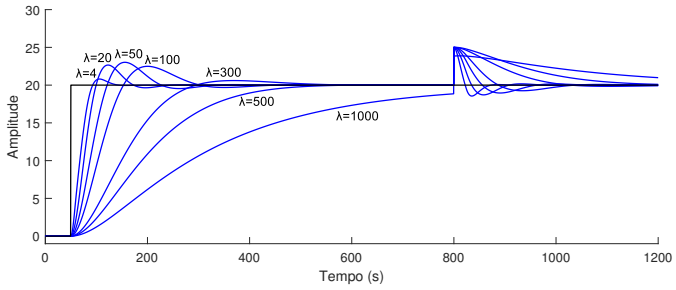

Fig. 8. Gráfico dos sinais de saída para diferentes valores de λ.

São ilustradas na Fig. 9 as curvas dos sinais de controle. Observa-se que quanto maior o λ mais suave é a resposta do controlador, minimizando o desgaste do atuador. Salientando que o sinal de controle é limitado, por questões físicas, na faixa de $0 - 100$, .

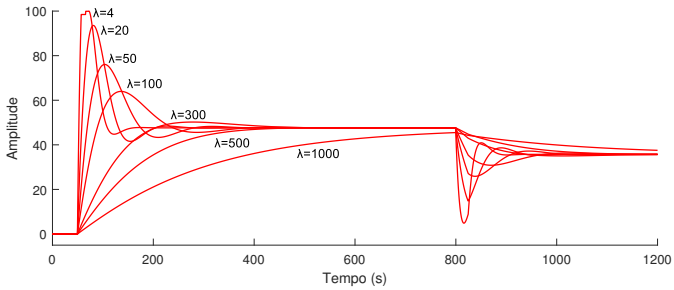

Fig. 9. Gráfico dos sinais de controle para diferentes valores de λ.

O comportamento observado nos gráficos é confirmado pela Tabela II que indica menores valores de IAE para controladores com menor λ, porém quanto menor o λ maior o TV, caracterizando um maior esforço de controle.

TABELA II
COMPARATIVO ENTRE AS VÁRIAS SINTONIAS SIMULADAS.

λ	IAE	TV
4	1010	242
20	1624	213
50	2411	149
100	3227	107
300	5418	66
500	7812	59
1000	13366	53

C. *Sistema Real*

Nessa seção foram realizados testes no sistema real, composto da planta física e do algoritmo de controle implementado no microcontrolador.

Assumindo que não existe conhecimento prévio sobre a planta, os preditores são calculados diretamente através dos dados de operação de entrada e saída. Por simplicidade foi aplicado um sinal de teste pulsado, em malha aberta, ilustrado na Fig. 10.

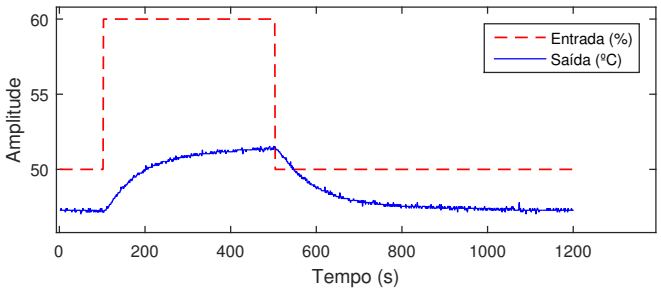

Fig. 10. Gráfico dos sinais de teste em malha aberta.

Baseado nos teste de MIL são aplicados 7 cenários diferentes variando o parâmetro de sintonia $\lambda = \{4, 20, 50, 100, 300, 500, 1000\}$. Para todos os casos o horizonte é mantido fixo em $N_h = 19$.

São ilustradas nas Fig. 11, 12, 13, 14, 15, 16 e 17 as curvas relevantes para uma mudança de referência em $t = 10$ s e uma perturbação de carga em $t = 800$ s. Tal qual o exemplo simulado, quanto menor o λ mais agressiva a resposta do controlador, resultando em uma resposta mais rápida à mudança de referência assim como rejeição a perturbações.

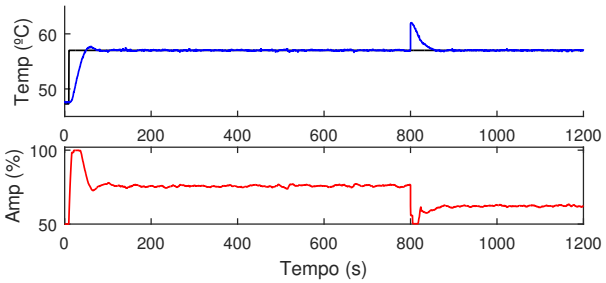

Fig. 11. Gráfico dos sinais para $\lambda = 4$.

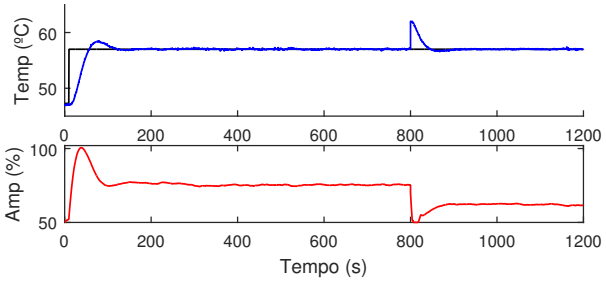

Fig. 12. Gráfico dos sinais para $\lambda = 20$.

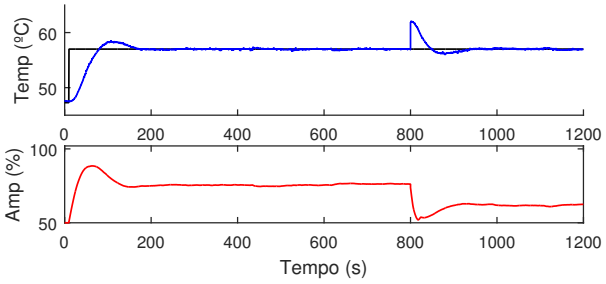

Fig. 13. Gráfico dos sinais para $\lambda = 50$.

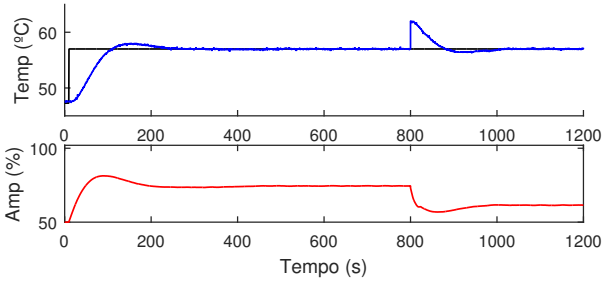

Fig. 14. Gráfico dos sinais para $\lambda = 100$.

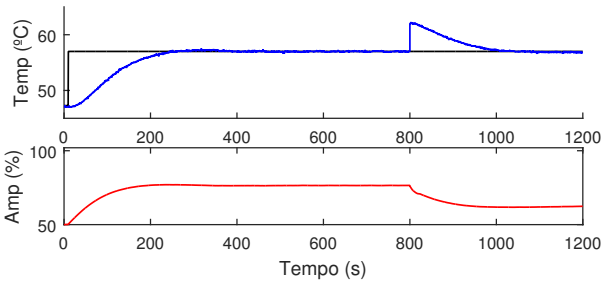

Fig. 15. Gráfico dos sinais para $\lambda = 300$.

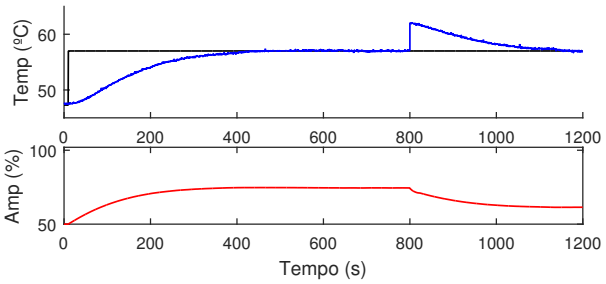

Fig. 16. Gráfico dos sinais para $\lambda = 500$.

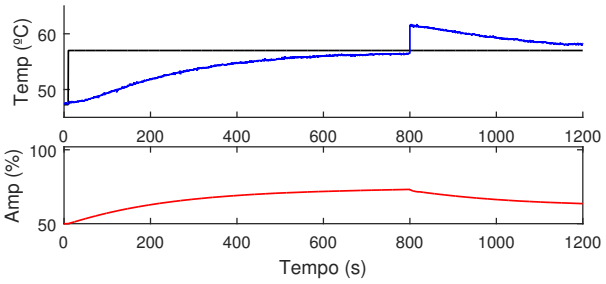

Fig. 17. Gráfico dos sinais para $\lambda = 1000$.

Inspecionando a função de custo (9) fica claro que existe um compromisso de sintonia entre erro de rastreamento e esforço de controle. Valores de λ muito pequenos acarretam na saturação do sinal de controle, conforme a Fig. 11. Por sua vez, valores muito grandes de λ ocasionam respostas excessivamente lentas conforme a Fig. 17.

TABELA III
COMPARATIVO ENTRE AS VÁRIAS SINTONIAS APLICADAS NA PLANTA PILOTO.

λ	IAE	TV
4	788	361
20	1015	197
50	1371	123
100	1734	77
300	3115	47
500	4377	39
1000	7157	32

De forma semelhante aos resultados simulados, ao observar a Tabela III fica claro que os valores do IAE, ocasionados

por desvios da referência, e os valores de TV, que refletem o esforço de controle, estão diretamente relacionados à escolha do parâmetro de sintonia λ.

VI. Conclusões

A evolução recente dos ambientes de desenvolvimento modernos está revolucionando a concepção de soluções embarcadas. Nesse artigo, Foi apresentada a implementação de um complexo algoritmo de controle em um sistema embarcado de 8 bits com recursos reduzidos. Sua idealização não demanda profundo conhecimento da linguagem nativa de programação da plataforma microcontrolada, em virtude da geração automática de código que o desenvolvimento baseado em modelo (MBD) proporciona.

O algoritmo de controle preditivo por subespaços prediz o comportamento da planta, em tempo real, a partir de projeções dos próprios dados de entrada e saída do processo. De forma que a principal escolha do projetista se resume ao parâmetro, λ, que pondera esforço de controle e erro de rastreamento. Testes de *Processor-In-the-Loop* (PIL) foram realizados para definição dos limites de hardware da plataforma microcontrolada assim como testes de *Model-In-the-Loop* (MIL) foram utilizados para gerar diferentes cenários de sintonia antes da implantação no sistema real.

Referências

[1] R. Aarenstrup, "Managing model-based design," *MathWorks Inc*, 2015.

[2] D. O'Sullivan, J. Sorensen, and A. Murray, "Model-based design streamlines embedded motor control system development," *Technical article, Analog Devices Inc.*, 2015.

[3] M. Johnson and M. Moradi, *PID Control: New Identification and Design Methods*. Springer-Verlag London, 2005.

[4] C. Cutler and B. Ramaker, "Dynamic matrix control - a computer control algorithm," *Proceedings of the joint American control conference*, pp. WP5–B, 1980.

[5] J. Richalet and D. O'Donovan, *Predictive Functional Control: Principles and Industrial Applications*, ser. Advances in Industrial Control. Springer London, 2009.

[6] D. Clarke, C. Mohtadi, and P. Tuffs, "Generalized predictive control part i. the basic algorithm," *Automatica*, vol. 23, no. 2, p. 137/148, 1987.

[7] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB*, ser. Advances in Industrial Control. Springer London, 2009.

[8] C. F. Camacho and C. Bordons, *Model Predictive Control*. Springer, 2013.

[9] B. Huang and R. Kadali, *Dynamic Modeling, Predictive Control and Performance Monitoring*. Springer, 2008.

[10] R. Kadali, B. Huang, and A. Rossiter, "A data driven subspace approach to predictive controller design," *Control engineering practice, vol. 11*, p. 261/278, 2003.

[11] S. Jing, R. Errouissi, A. Al-Durra, and I. Boiko, "A data-driven subspace predictive controller design for artificial gas-lift process," in *2015 IEEE Conference on Control Applications (CCA)*, Sept 2015, pp. 1179–1184.

[12] V. Vajpayee, S. Mukhopadhyay, and A. P. Tiwari, "Data-driven subspace predictive control of a nuclear reactor," *IEEE Transactions on Nuclear Science*, vol. 65, no. 2, pp. 666–679, Feb 2018.

[13] J. Lin, "Measuring return on investment of modelbased design," *MathWorks Inc*, 2011.

[14] C. Dias, R. Lima, and P. Barros, "Projeto baseado em modelo no desenvolvimento de um controlador de velocidade para motor BLDC," in *14º Simpósio Brasileiro de Automação Inteligente*, Brazil, 2019, pp. 275–281.

[15] S. Sedghizadeh and S. Beheshti, "Data-driven subspace predictive control: Stability and horizon tuning," *Journal of the Franklin Institute*, vol. 355, no. 15, pp. 7509 – 7547, 2018.

[16] W. Favoreel, B. D. Moor, and M. Gevers, "Spc: Subspace predictive control," *IFAC Proceedings Volumes*, vol. 32, no. 2, pp. 4004 – 4009, 1999, 14th IFAC World Congress 1999, Beijing, China, 5-9 July.

[17] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[18] K. J. Astrom and B. Wittenmark, *Computer-controlled Systems: Theory and Design (3Nd Ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2011.

[19] R. Lima, C. Dias, and P. Barros, "Desenvolvimento de uma planta didatica termica multivariavel baseada em modulos peltier," *XLII Congresso Brasileiro de Educacao em Engenharia (COBENGE)*, 2014.

[20] R. Lima, "Aplicação de algoritmo de controle preditivo por subespaços em uma planta piloto termoelétrica," *13th IEEE/IAS International Conference on Industry Applications*, 2018.

Rafael B. C. Lima Formado em Engenharia Elétrica com Ênfase em controle e automação (2010), Mestrado (2012) e Doutorado (2016) pela pela Universidade Federal de Campina Grande. Atua desde 2017 como professor Adjunto na UFCG desenvolvendo atividades principalmente nas áreas de Sistemas embarcados, Identificação e Controle de sistemas dinâmicos.

Péricles R. Barros Possui graduação em Engenharia Elétrica pela Universidade Federal da Paraíba (1979), especialização em Engenharia Elétrica pela Philips International Institute Eindhoven (1981), mestrado em Engenharia Elétrica pela Universidade Federal da Paraíba (1985), doutorado em Electrical And Computer Engineering - Control Syst pela The University of Newcastle Australia (1990), pós-doutorado pela Lund University(1997) e pós-doutorado pela Uppsala Universitet (1996). Atualmente é professor titular da Universidade Federal de Campina Grande. Tem experiência na área de Engenharia Elétrica, com ênfase em Eletrônica Industrial, Sistemas e Controles Eletrônicos. Atuando principalmente nos seguintes temas: Controle Adaptativo, Identificação Robusta, Método das Médias, Convergência de Controladores Adaptativos.