A LOW COST HARDWARE-IN-THE-LOOP APPROACH FOR PROTOTYPING

Daniel A. Nesvera, Gabriel J. C. Silva, Alex S. Schirmer, Carlos H. Barriquello, César Abascal, José E. Baggio, Rodrigo Deutsch, Társis B. Silva
Universidade Federal de Santa Maria – UFSM
Departamento de Eletrônica e Computação (DELC), Campus Sede – Santa Maria - RS
daniel.nesvera@ecomp.ufsm.br, gabriel.silva@ecomp.ufsm.br

Abstract*. A low-cost, high fidelity hardware-in-the-loop simulation using commercial off-the-shelf hardware is presented. This approach allows for fast, reliable and distibuted prototying ideal to be used in the academic environment, where costs and time are limited. Simulations were done in MATLAB/Simulink connected to a low-cost, open-source Arduino platform for better flexibility and benefit-cost ratio but can be extended to other software and/or hardware platforms.*

Palavras-chave: *Hardware-in-the-loop, Simulação, Arduino*

1. INTRODUÇÃO

As simulações por *hardware-in-the-loop* (HIL ou HITL) têm sua origem na indústria aeronáutica e são utilizadas em sistemas de alto custo, risco e competitividade, permitindo simulações com grande fidelidade executadas de forma extremamente ágil e relativamente barata.

Na plataforma HIL um modelo do sistema é obtido e simulado, permitindo amplo controle sobre suas características. Esta simulação é acoplada ao hardware a ser testado, transmitindo estímulos virtualmente idênticos ao sistema real que permitem uma análise da resposta do sistema.

Soluções comerciais são amplamente disponíveis mas possuem custo relativamente elevado e plataforma fechada, dificultando a utilização destes sistemas em ambiente de prototipagem acadêmica.

2. IMPLEMENTAÇÃO E VALIDAÇÃO

O sistema proposto utiliza uma placa de desenvolvimento *Arduino UNO,* que contém um microcontrolador *ATmega328P* e um computador com os softwares *MATLAB/Simulink.* A validação do sistema HIL foi baseada no controle de velocidade de um motor. A simulação da planta foi retirada de um exemplo disponibilizado pela *University of Michigan* [1]. A modelagem matemática do motor em corrente contínua (CC) é descrita pelo espaço de estados apresentado em Eq. (1) e Eq. (2).

$$\dot{X} = \begin{bmatrix} -b/J & K/J \\ -K/L & -R/L \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} \cdot [V] \quad (1)$$

$$Y = [1 \quad 0] \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (2)$$

Da Eq. (1), tem-se que b é o coeficiente de amortecimento, J é a inércia, K é a força contra eletromotriz (FCEM), R é a resistência e L a indutância. Na Eq. (2), V é a tensão no motor.

Em testes com o sistema real, o motor é controlado por um algoritmo de controle proporcional-integral-derivativo (PID), executado pelo microcontrolador e ilustrado na Fig. 1. Esta apresenta um bloco que reflete o funcionamento do controlador do motor e tem como entradas a velocidade desejada e a velociade atual do motor. O controlador PID

é executado por esse bloco, cuja saída é um valor de tensão que alimenta o bloco do motor de corrente contínua. Pertubações podem ser aplicadas sobre o motor.

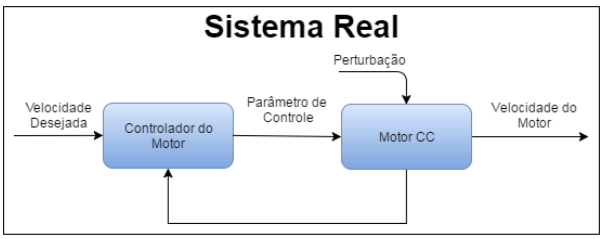

Figura 1. Diagrama de blocos do sistema real.

No teste com HIL a planta virtual é modelada por simulação com base na planta real do sistema, exigindo comunicação do microcontrolador à simulação e necessitando modificações no sistema. O microcontrolador deixa de receber sinais da planta pelas suas portas de propósito geral (*general-purpose input/output*, ou GPIO) e passa a comunicar-se com a simulação via serial, além de receber novos códigos para executar funções de codificação e decodificação dos dados recebidos. A simulação no *MATLAB/Simulink* necessita de um bloco que liga as saídas do controlador à entrada do motor e as velocidades atual e desejada do motor à entrada do controlador. Figura 2 apresenta o diagrama de blocos do teste por HIL. A parte superior da figura apresenta o ambiente de simulação onde a planta do sistema (motor) foi modelada matematicamente. Um bloco refere-se aos parametros iniciais da simulação e o bloco de registro de dados gera gráficos para serem analisado após a simulação. A parte inferior na consiste no controlador do sistema, também simulado no software.

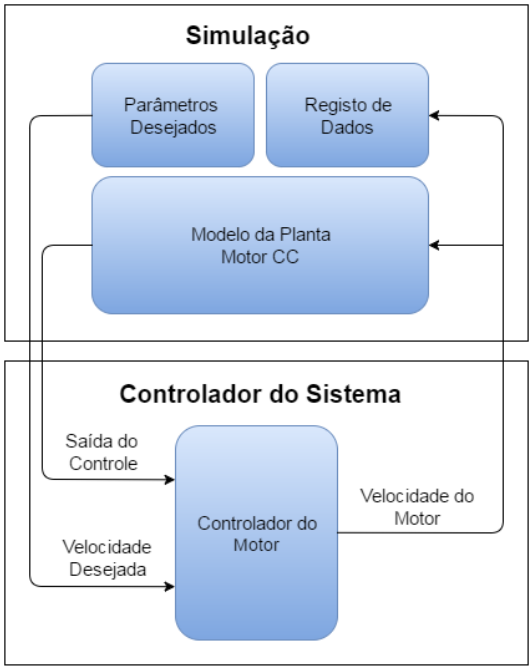

Figura 2. Diagrama de blocos do sistema HIL completo

A comunicação entre a placa de desenvolvimento e o computador é realizada por comunicação serial assíncrona, também denominada de *universal asynchronous receiver/transmitter* (UART). Não foi necessário o uso de placas que convertam a comunicação serial para USB visto que a plataforma *Arduino UNO* já possui hardware de conversão. O algoritmo de controle foi baseado em uma implementação de controle PID discreto fornecida pela empresa *Atmel* [2], adaptado para a implementação neste sistema. As constantes proporcional, integral e derivativa foram obtidas heuristicamente. O bloco que liga as entradas e saídas da simulação no software *MATLAB/Simulink* foi desenvolvido utilizando a linguagem de programação do próprio *MATLAB*. Neste, faz-se necessária a criação da comunicação serial, configuração de suas características e execução de codificadores e decodificadores semelhantes ao do microcontrolador. Os blocos utilizados no *MATLAB/Simulink* para a simulação do sistema HIL são apresentados na Fig. 3.

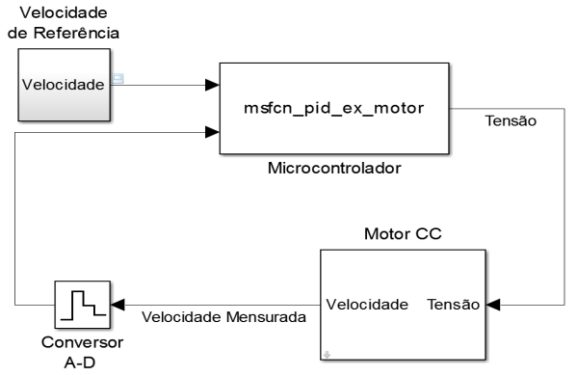

Figura 3. Diagrama de blocos do *MATLAB/Simulink*

Para a realização dos testes, o diagrama apresentado na Fig. 3 é simulado no software Simulink. Inicialmente é configurado um valor de amplitude do degrau referente a velocidade de entrada. O bloco "Microcontrolador" reúne a velocidade desejada e a velocidade mensurada, envia para o microcontrolador e espera o retorno desse que será utilizado na saída do bloco na forma de tensão. O bloco "Motor CC" utiliza a tensão e a função de transferência modelada para gerar uma velocidade. A velocidade do motor é mensurada e convertida para um sinal digital que realimenta o bloco do microcontrolador.

O sistema implementado apresentou a resposta dada na Fig. 4, onde a curva amarela representa a velocidade desejada e a curva vermelha representa a resposta da velocidade do motor com o controle PID implementado no microcontrolador. A curva apresentada foi adquirida após várias simulações que foram utilizadas para definir os valores das constantes do controle PID.

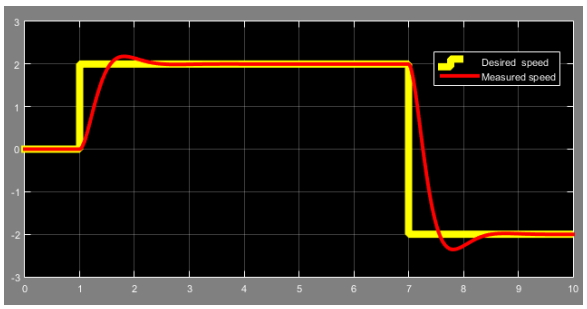

Figura 4. Curvas de resultado obtidos na simulação do sistema

3. CONCLUSÃO

O sistema simulado para validação foi altamente simplificado e apresenta baixa complexidade a fim de facilitar o desenvolvimento da plataforma. A partir desta pode-se aumentar gradativamente a complexidade dos sistemas simulados de forma a aproximá-los ao sistema real com mais fidelidade, garantindo resultados altamente satisfatórios.

Desenvolvimentos posteriores podem ser aplicados à quadrirotores, permitindo o desenvolvimento de aspectos do controle sem colocar em risco a integridade física do veículo aéreo, estruturas físicas ou seres vivos que possam estar em rota de uma possível colisão em caso de falha do sistema durante a prototipagem. O software comercial *X-Plane*, desenvolvido pela empresa *Laminar Research* e certificado pela *Federal Aviation Administration* (FAA), apresenta alta configurabilidade e desempenho, permitindo modelagem e simulação de aeronaves. Soluções para comunicação entre este e *MATLAB/Simulink* já existem, tais como apresentadas em *National Aeronautics and Space Administration* [3], e podem ser utilizadas para a complementação do sistema HIL aqui descrito.

Agradecimentos

Aos orientadores Carlos Henrique Barriquello e José Eduardo Baggio, sempre presentes nos momentos de dificuldade, dispostos a persistir conosco solucionando os problemas mais variados das maneiras mais criativas e inovadoras. Aos colegas de laboratório, cujo trabalho contribui enormemente no crescimento pessoal de todos.

REFERÊNCIAS

[1] University of Michigan, "DC Motor Speed: System Modeling", http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling.

[2] Atmel Corporation, “AVR221: Discrete PID controller”, 2006, http://www.atmel.com/images/doc2558.pdf.

[3] National Aeronautics and Space Administration, “X-Plane Connect”, 2015, GitHub repository, https://github.com/nasa/XPlaneConnect.