Algoritmos com Ações PID Embarcados em CLPs para Controle de Velocidade de Alimentadores de Esteiras de Aço de um Virador de Vagões

José Pinheiro de Moura [*] **João Viana da Fonseca Neto** [**]
Bianca Fontenele Lemos Veras [***]

[*] *Universidade Estadual do Maranhão, São Luís, MA, Brasil*
josepinheiro@professor.uema.br
[**] *Universidade Federal do Maranhão, São Luís, MA, Brasil*
joao.fonseca@ufma.br
[***] *Universidade Federal do Maranhão, São Luís, MA, Brasil*
bianca.lemos@discente.ufma.br

Abstract: Controllers with Proportional, Integral and Derivative (PID) actions are widely publicized and implemented in academia/industry. However, embedded applications in Programmable Logic Controllers (PLCs) in industrial processes is still a novelty. In this article, we present the development and simulated results of a PID controller algorithm with null Integral (I) and Derivative (D) actions to be embedded in PLCs, with application in the operational speed control of steel belt feeder of a car dumper, conceived in the CODESYS virtual simulator and embedded in a virtual PLC, also in the CODESYS simulation software, still in this context, the PID controllers algorithm is tested and validated in the MATLAB software.

Resumo:
Controladores com ações Proporcionais, Integrais e Derivativas (PID) são amplamente divulgados e executados na academia/indústria. Entretanto, aplicações embarcadas em Controladores Lógicos Programáveis (CLPs) em processos industriais ainda são novidades. Neste artigo, apresenta-se o desenvolvimento e resultados simulados de um algoritmo de controladores PID com as ações Integrais (I) e Derivativas (D) nulas para ser embarcado em CLPs, com aplicação no controle operacional de velocidade de Alimentador de esteira de aço (AL) de um Virador de Vagões (VVs). A simulação do sistema de controle foi realizada na plataforma de desenvolvimento CODESYS e o controlador embarcado em um CLP virtual contido no ambiente de simulação do CODESYS. Neste contexto, o algoritmo de controladores PID é testado e validado no software MATLAB.

Keywords: PID, PLC, Industrial processes, Virtual simulator CODESYS, software MATLAB.
Palavras-chaves: PID, CLP, Processos industriais, Simulador virtual CODESYS, *Software* MATLAB.

1. INTRODUÇÃO

Projetos de controle, em muitos casos, são problemas complexos que requerem considerações de problemas, como redução da carga de perturbação, rastreamento do ponto de ajuste, robustez em relação às variações na dinâmica operacional de processos e efeitos de ruídos de medição (Garpinger et al., 2014). O controlador com ações Proporcionais, Integrais e Derivativas (PID) pode ser uma alternativa para resolver, em parte, o problema de controle em prcessos industriais.

A sintonia de controladores PID não é uma tarefa trivial, na prática, muitas vezes, são sintonizados manualmente por meio de procedimentos de tentativa e erro. Em muitas situações os controladores PID tem a parte da ação derivativa nula, o que facilita a sintonia, mas que deve ser cuidadosamente cuidadosamente analisado, para que o desempenho e a eficiência do sistema de controle não sejam comprometidos.

Apesar de Controladores PID serem amplamente divulgados no meio acadêmico e em alguns setores da indústria a aplicação desse tipo de controlador embarcado em Controlador Lógico Programável (CLP) ainda é uma novidade em controle operacionais de equipamentos de mineração/portuários, como Viradores de Vagões (VVs) por exemplo. O CLP é um equipamento eletrônico especializado que desempenha funções de controle e monitoramento de máquinas e processos industriais de diversos tipos e níveis de complexidade, por meio de programas com linguagens específicas desenvolvidas pelo usuário e possui grande abrangência no setor industrial (Guo and Pecen, 2009).

As linguagens de comunicação para CLPs são normatizadas pelo padrão inernacional *International Electrotechnical Commission* (IEC) 61131-3. A parte 3 da norma IEC define cinco linguagens de programação, que são: i) *Structured Text* (ST) - Texto Estruturado; ii) *Instruction List* (IL) - Lista de Instruções; iii) *Ladder* (LD) - Linguagem *Ladder*; iv) *Function Block Diagram* (FBD) - Diagrama de Bloco e v) *Sequential Flow Chart* (SFC) - Diagrama de Fluxo ou Graficet.

As linguagens ST e IL são ditas textuais por conterem instruções na forma de texto. As linguagens LD e FBD são ditas gráficas por possuírem representação na forma de símbolos. Já a linguagem SFC é, normalmente, tida como linguagem gráfica,

porém permite também programações textuais. É comum em alguns ambientes de programação que atendem à IEC 61131-3, como o *software* de simulação CODESYS, a presença de uma sexta linguagem de programação conhecida como *Continuous Function Chart* (CFC), que não faz parte das definições da norma (Hanssen, 2015).

O *software* de simulação CODESYS é desenvolvido e comercializado pela *Smart Software Solutions* (3S), a versão 1.0 foi lançada em 1994 e atualmente encontra-se na versão 3.5 (Korobichuk et al., 2017). Sua interface de desenvolvimento é gratuita e não é orientado à um tipo específico de *hardware* de controlador programável ou sistema embarcado, inclusive não dependende de fabricantes específicos de *hardware*. É amplamente utilizado na indústria para diversos tipos de automações, variando de lógicas simples à robótica, controle de movimento e *Computer Numeric Control* (CNC) (Souza and Pereira).

Muitas instituições de ensino em todo o mundo estão trabalhando em aplicativos de ensino à distância. Nesse campo, os laboratórios remotos estão possibilitando o uso intensivo das instalações das universidades, ao mesmo tempo em que auxiliam no trabalho de professores e alunos.

Neste trabalho, apresenta-se um algoritmo projetado para ser utilizado no controle de velocidade de Alimentadores (ALs) de VVs por meio de lógicas desenvolvidas no *software* de simulaçao CodeSys e embarcada em um CLP virtual através do *software* de simulações CodeSys.

2. FORMULAÇÃO MATEMÁTICA

Nesta Seção, apresenta-se a formulação matemática de um sistema de primeira ordem de acordo com a norma ISA, compatível com a formulação usada em CLP

A expressão geral de uma função de transferência de primeira ordem é dada por

$$\frac{Y(s)}{X(s)} = \frac{K}{\tau s + 1}. \tag{1}$$

Sendo $Y(s)$ a variável de saída, $X(s)$ é a variável de entrada, K é o ganho do processo e τ é a constante de tempo de primeira ordem. A constante de tempo τ caracteriza o tempo de resposta de um sistema. Quanto maior a constante de tempo τ, a resposta do sistema é mais lenta.

O ganho do processo K é a razão entre a variação da variável de saída $Y(s)$ em relação à variável de entrada $X(s)$, que é dada por

$$K = \frac{\Delta Y}{\Delta X}. \tag{2}$$

2.1 Diagrama de Blocos

A representação em diagramas de blocos, é uma forma mais clara para se entender uma planta.

Na Figura 1 está a reperesentação em diagrama de blocos de uma planta em Malha Aberta (MA).

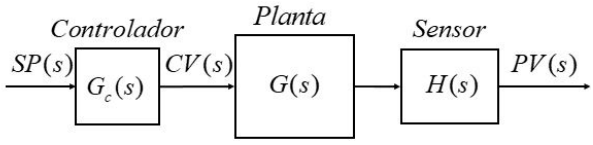

Figura 1. Diagrama de blocos da planta em MA.

Na Figura 2 está a reperesentação em diagrama de blocos de uma planta em Malha Fechada (MF).

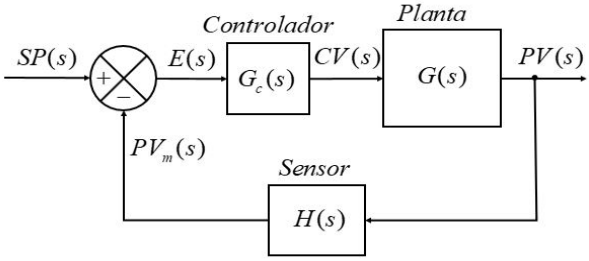

Figura 2. Diagrama de blocos da planta em MF.

Na Figura 3 está a reperesentação em diagrama de blocos de uma planta em MF de primeira ordem com ganho K_p.

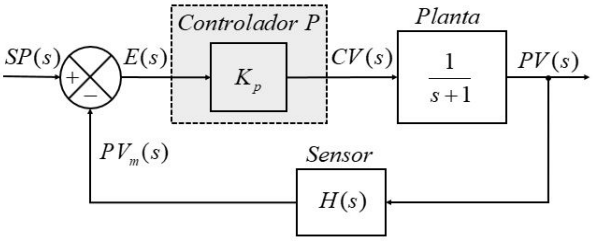

Figura 3. Diagrama de blocos da planta em MF com o ganho K_p.

3. PLANTA INDUSTRIAL

A planta usada para avaliar e validar os experimentos simulados no *MATLAB* e CODESYS é o processo operacional de controle de velocidade de AL de VV, automatizado com lógicas desenvolvidas em CLP.

O modelo matemático da planta em termo de FT e a equação do erro são dados a seguir:

$$G_{VV}^{AL}(s) = \frac{0,438}{s + 0,438}. \tag{3}$$

Onde, $G_{VV}^{AL}(s)$ é a FT, que representa a planta (velocidade do AL do VV.

$$SP(s) - PV(s) = E(s). \tag{4}$$

Onde, o $SP(s)$ é o valor de referência, $PV)$ é o valor númério da velocidade do AL e $E(s)$ é o erro entre $SP(s)$ e $PV(s)$.

Na Figura 4 está a representação em diagrama de blocos de uma planta em MF de primeira ordem com o ganho K_p associado com as Eqs. (3) e (4).

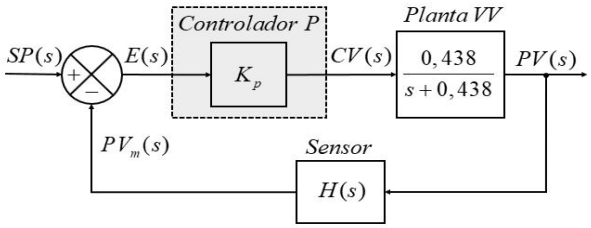

Figura 4. Diagrama de blocos da planta em MF com o ganho K_p.

3.1 Processo de Descarregamento de Vagões

O descaregamento de vagões é feito por um equipamento denominado de viradores de vagões. Estes, têm movimentos de rotatação em torno de seu próprio eixo e pode atingir até 180°. Na extremidade de cada vagão, os engates são móveis, permitindo o giro dos vagões durante o descarregamento sem necessidade de desacoplagem dos mesmos, conforme ilustrado na Figura 5. Ao girar, a carga dos vagões é transferida para o silo de abastecimento, que é esvazido por meio da rotação do Aimentador com velocidade de até 4,5 m/s, conforme esquemático apresentado na Figura 6.

Figura 5. Virador de vagões.

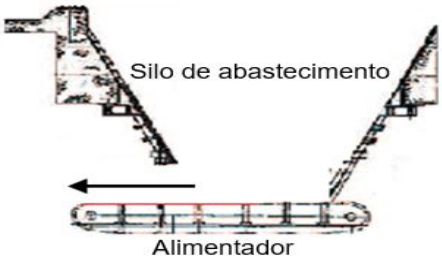

Figura 6. Silo de abastecimento.

3.2 Controladores Lógicos Programáveis - CLP

Os CLPs são computadores que executam rotinas cíclicas de operação, apresentadas em uma forma especial de programação de acordo com o padrão internacional (*International Electrotechnical Commission* - IEC), com o propósito de controlar processos industriais (Otto and Hellmann, 2009). Um CLP segue padrões similares às arquiteturas dos computadores digitais, sendo composto por uma unidade central de processamento (*Central Processing Unit* - CPU), um banco de memórias, um barramento para interligação de elementos e inteface para sinais de entradas e saídas (Silveira and Santos, 1998).

O princípio fundamental de funcionamento de um CLP é a execução contínua por parte da CPU de um programa que realiza ciclicamente as ações de entradas e saídas por meio de instruções de lógicas de programação desenvolvidas. Na Figura 7, ilustra-se o ciclo de processamento de um CLP. O projeto de controle de máquina é uma área única da engenharia que requer o conhecimento de certas técnicas de diagramação específicas e exclusivas. Muitos símbolos de componentes e formatos de *layout* são diferentes. Os símbolos básicos são usados com base na lógica booleana.

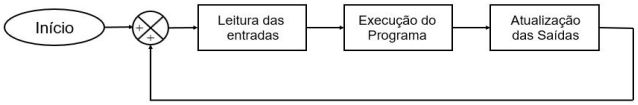

Figura 7. Ciclo de processamento do CLP.

Os CLPs, geralmente, possuem uma porta de expansão (um soquete de interconexão) que permitirá a adição de unidades especializadas, como contadores de alta velocidade e unidades de entrada e saída analógicas (contínua) ou discretas (digitais) adicionais (Hackworth and Hackworth, 2004).

3.3 Norma IEC 61131

Para atender às demandas da comunidade industrial internacional, foi formado um grupo de trabalho dentro da *International Electrotechnical Commission* (IEC) para avaliar o projeto completo de CLPs, incluindo instalação de *hardware*, testes, documentação, programação e comunicação (Otto and Hellmann, 2009). A IEC 61131 é dividida em 8 partes, que são apresentadas a seguir:

(1) A primeira parte da norma, a IEC 61131-1 define as informações gerais dos controladores programáveis (CP), delimitando e identificando as principais características relevantes para a seleção e aplicação de CPs, e também para qualquer equipamento ou acessório ligado à CPU.
(2) A parte 2 (IEC61131-2) da norma estabelece requisitos funcionais de segurança de manuseio, proteções e recomendações contra interferências eletromagnéticas e requisitos construtivos elétricos, mecânicos e ambientais.
(3) A norma IEC 61131, em sua parte 3 (IEC 61131-3), trata-se das linguagens de programação para CLP. Com o intuito de:
   - Fornecer metodologias de construção de lógicas de programação de forma estruturada e modular (Tiegelkamp and John, 2010), (John and Tiegelkamp, 2010), permitindo a quebra dos programas em partes gerenciáveis;
   - Definir 5 linguagens de programação, cada uma com suas características, de forma a cobrir a maioria das necessidades de controle atuais. As 5 lingaugens de programação para CLP, definida pela IEC 61131-3, são:
     · *Structured Text* (ST) - Texto Estruturado;
     · *Instruction List* (IL) - Lista de Instruções;
     · *Ladder* (LD) - Linguagem *Ladder*;
     · *Function Block Diagram* (FBD) - Diagrama de bloco;
     · *Sequential Flow Chart* (SFC) - Diagrama de Fluxo ou Grafcet.

- Permitir o uso de outras linguagens de programação, desde que obedeçam as mesmas formas de chamadas e trocas de dados (Visual Basic, Flow Chart, C++, etc);
- Possibilitar a abordagem e estruturação *top-down* e *bottom-up*, fundamentada em 3 princípios. Que são:
  - Modularização;
  - Estruturação;
  - Reutilização.
- Padronizar as principais definições
  - **Configurações (*Configurations*):** corresponde ao software necessário à um CLP ou conjunto de CLPs para que este(s) cumpra(m) suas funções de controle;
  - **Recursos (*Resources*):** qualquer elemento com capacidade de processamento dentro de uma configuração, capaz de executar programas;
  - **Tarefas (*Tasks*):** controla a execução de programas ou blocos funcionais de forma periódica ou por disparo por eventos (*triggers*);
  - **Unidade de Organização de Programas (*Program Organization Unit* - POU):** é a forma definida pela norma para se implementar o *software* do CLP por meio da associação de variáveis e instruções, utilizando as linguagens da norma IEC 61131-3 ou linguagens adicionais;
  - **Programas (*Programs*):** construído a partir de blocos funcionais e funções em qualquer das linguagens da norma;
  - **Blocos Funcionais (*Function Blocks*):** Partes de programas hierarquizados e estruturados de forma a serem parametrizáveis e reutilizáveis;
  - **Funções (*Functions*):** funções ou procedimentos, são elementos de programação que, diferente de blocos funcionais, não possuem persistência, gerando resultados a cada execução;
  - **Variáveis Globais e Locais (*Global and Local Variables*):**

(4) IEC 61131-4 trata da orientações para o usuário. Na quarta parte da IEC 61131 apresenta-se as orientações para que os usuários de CLPs possam comprá-los e instalá-los, isto é, indica a forma como seus usuários/fabricantes devem especificar *hardware* e *software* necessários ao projeto, assim como instalar, comissionar e certificar o sistema de automação instalado e formalizar a comunicação entre fornecedores e usuários finais.

(5) A parte 5 da IEC 61131 (61131-5) está relacionada com os aspectos de comunicação de um CP, como a definição do módulo de comunicação, seus blocos funcionais e mecanismos para conexão entre controladores e outros dispositivos de automação.

(6) IEC 61131–6: Comunicação via *Fieldbus*, a sexta parte da norma IEC 61131 especifica os requisitos para CPs e seus periféricos associados, tal como definido na Parte 1 da IEC 6131, que se destina a ser usado como o subsistema de lógica de um elétrico/eletrônico/programável eletrônico (E/E/PE) sistema relacionado com a segurança.

(7) IEC 61131-7, a sétima parte da norma IEC 61131 define e estrutura o uso de linguagem utilizada em programação difusa (*Fuzzy Control Language* - FCL).

(8) A IEC 61131-8, a oitava parte, trata-se da implementação das linguagens, essa parte da norma complementa a terceira parte, orientando os usuários envolvidos com programação, configuração, instalação e manutenção de CLPs, garantindo que a implementação de elementos comuns e linguagens de programação se dê de forma sistematizada.

4. ANÁLISE DAS AÇÕES PID

Para analisar os efeitos causados na FT pelas ações PID, usa-se um sistema de primeira ordem, dado na sua forma genérica por

$$G(s) = PV = \frac{1}{\tau s + 1}. \tag{5}$$

Sendo $G(s)$ a FT, PV a variável de processo (*process variable*) da planta de primeira ordem e τ é uma constante de tempo.

4.1 Análise da Ação P

A planta de primeira ordem em Malha Aberta (MA) com o ganho proporcional (K_p) é dada por

$$G^p_{MA}(s) = PV^p_{MA} = \frac{K_p}{\tau s + 1}. \tag{6}$$

A planta de primeira ordem em MF com o ganho proporcional (K_p) é dada por

$$G^p_{MF}(s) = PV^p_{MF} = \frac{K_p}{\tau s + 1 + K_p}. \tag{7}$$

Observa-se nas Eqs. (6) e (7) que a ação proporcional não influencia na ordem do polinômio característico, nem na ordem dos zeros da FT da planta em MA.

4.2 Análise da Ação PI

A planta em MA com o ganho K_p e o ganho integral K_i é dada por

$$G^{pi}_{MA}(s) = PV^{pi}_{MA} = \frac{K_p}{\tau_i \tau s^2} + \tau_i s. \tag{8}$$

Sendo $G^{pi}_{MA}(s)$ a planta em MA com os K_p e K_i.

A planta em MF com o ganho K_p e o ganho K_i é dada por

$$G^{pi}_{MF}(s) = PV^{pi}_{MF} = \frac{K_p}{\tau_i \tau s^2} + \tau_i s + K_p. \tag{9}$$

Sendo $G^{pi}_{MF}(s)$ a planta em MF com os K_p e K_i.

Observa-se nas Eqs. (8) e (9) que a ação integral influencia na ordem do polinômio característico, a ordem do sistema é acrescida de +1. Porém, a ordem dos zeros da FT da planta em MA continua inalterada.

4.3 Análise da Ação PD

A planta em MA com o ganho K_p e o ganho derivativo K_d é dada por

$$G^{pd}_{MA}(s) = PV^{pd}_{MA} = \frac{K_p \tau s}{\tau s + 1}. \tag{10}$$

Sendo $G^{pd}_{MA}(s)$ a planta em MA com os K_p e K_d.

A planta em MF com o ganho K_p e o ganho K_d é dada por

$$G_{MF}^{pd}(s) = PV_{MF}^{pd} = \frac{K_p \tau s}{(K_p \tau d + \tau)s + 1}. \quad (11)$$

Sendo $G_{MF}^{pd}(s)$ a planta em MF com os ganhos K_p e K_d. Observa-se nas Eqs. (10) e (11) que a ação derivativa não influencia na ordem do polinômio característico, mas a ordem dos zeros da FT da planta em MA é alterda em +1.

4.4 Funções de Transferência de Acordo com a Norma ISA

A FT genérica em MA no domínio do tempo é dada por

$$CV(t) = K_p E(t) + \frac{K_p}{\tau_i} \int_0^t E(t)dt + K_p \tau_d(t). \quad (12)$$

Sendo $CV(t)$ a variável controlada (*control variable*).

Aplicando a transformada de *Laplace* na Eq. (12), a FT genérica em MA no domínio da frequência é dada por:

$$CV(s) = K_p E(s) + \frac{K_p}{\tau_i} \int_0^t E(s)dt + K_p \tau_d(s). \quad (13)$$

Rearrumando a Eq. (13), tem-se:

$$\frac{CV(s)}{E(s)} = K_p + \frac{K_p}{\tau_i s} + K_p \tau_d s. \quad (14)$$

5. EXPERIMENTOS

Os experimentos são simulados em uma planta de primeira ordem, para o controle de velocidade de um AL de um VV com as ações integrais (I) e derivativas (D) nulas, apenas a ação proporcional (P) está ativada, para atuar no controle da velocidade do AL. A simulação é feita com algoritmos desenvolvidos no *softare* de simulação CODESYS, validados no *MATLAB* e embarcados em um CLP, por meio do *software* de simulação CODESYS nas linguagens *LADDER*, Texto estruturado e bloco de funções.

5.1 Código Desenvolvido no MATLAB

Os experimentos são, primeiramente, testados, avaliados e validados no *Software* MATLAB, conforme Figura abaixo.

```
clc
clear all

num1 = [0.438];
den1 = [1 0.438];
t = 0:0.01:200;
Kp1 = 1;
Ts1 = 1
Ts2 = 1/0.9
Ki1 = Kp1/Ts1
num1_Kp1 = [Kp1*Ts1*0.438];
den1_Kp1 = [1 Kp1*0.438];
num2_Kp1 = [Kp1*Ts2*0.438];
den2_Kp1 = [1 Kp1*0.438];
[y1,x1,t] = step(num1,den1,t);
[y1_Kp1,x1_Kp1,t] = step(num1_Kp1,den1_Kp1,t);

figure(1)
  plot(t,y1*2.25,'-r','linewidth',1.5)%,('-r','linewidth',1.5)
  hold on
  plot(t,y1_Kp1*4.5,'-k','linewidth',1.5)%,('-r','linewidth',1.5)
  hold on
  ylabel('Velocidade - (m/s)')
  xlabel('Tempo (s)')
  legend('SP=4,50 m/s','SP=2,25 m/s')
  axis([0 50 0 6])
grid
```

Figura 8. Algoritmo desenvolvido no MATLAB.

Para testar, avaliar e validar o modelo da planta, desenvolveu-se um simulador no *SIMULINK/MATLAB* em diagramas de blocos associados com o algoritmo desenvolvido no MATLAB, conforme Figura 9, associado com as Eqs. (3) e (4).

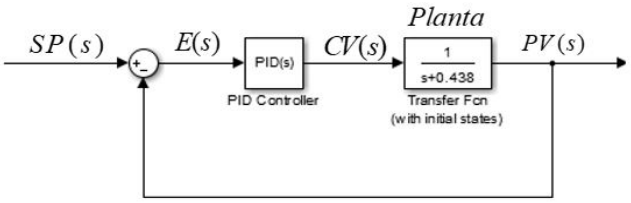

Figura 9. Diagrama de blocos desenvolvido no SIMULINK.

Os resultados da simulação no SIMULINK são apresentados na Figura 10

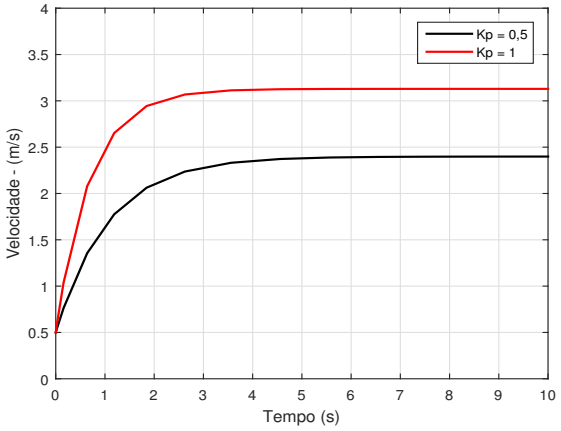

Figura 10. Resultados simulados no SIMULINK com $K_p = 0,5$ e $K_p = 1$, respectivamente.

5.2 Implementação do Código Desenvolvido no MATLAB no Simulador/Emulador CODESYS

O *software* de simulação CODESYS é um simuldor/emulador de código aberto. Na Figura 11, apresenta-se a árvore do projeto do sistema de primeira ordem para o controle de velocidade de AL de um VV.

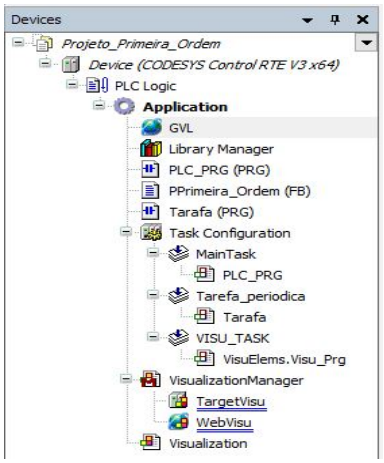

Figura 11. Árvore do projeto de controle PID de primeira ordem
Fonte: (Autor).

Na Figura 12, apresenta-se as variáveis globais do sistema de primeira ordem.

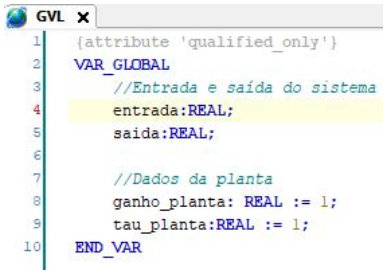

Figura 12. Variáveis globais.
Fonte: (Autor).

Na Figura 13 está ilustrada a lógica em *LADDER*, desenvolvida no *software* de simulação CODESYS.

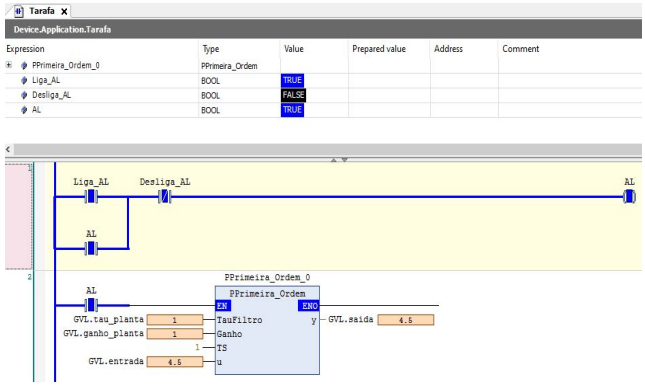

Figura 13. Lógica *LADDER* para ligar/desligaa AL e bloco de funções da FT de primeira ordem.

Na Figura 14, apresenta-se o comportamento da velocidade do AL, em primeira instância para o *setpoint* de 2,25 m/s e em segunda instância para o *setpoint* de 4,25 m/s.

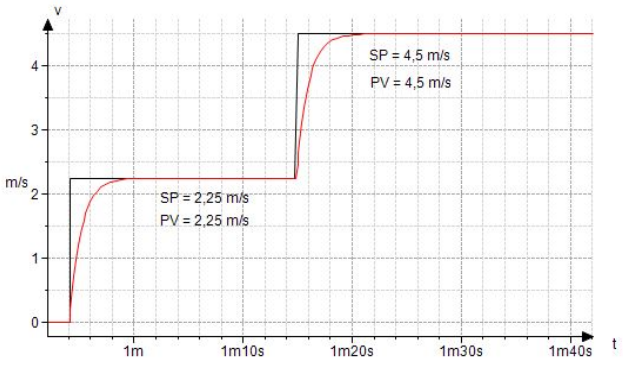

Figura 14. Resultado embarcado no CLP do *software* de simulação CODESYS, com $K_p = 1$ e $\tau s = 1$, respectivamente, associado com a Eq. (3).

5.3 Análise dos Resultados Simulados

Observa-se na Figura 10 que o desempenho do algoritmo no *software* MATLAB atigiu 33% e 67% do objetivo para $K_p = 0,5$ e $K_p = 1$, respectivamente. Na Figura 14, o algoritmo desenvolvido no *software* de simulação CODESYS apresentou bom desempenho de controle com: $SP = 2,25$ m/s e $SP = 4,5$ m/s e com $K_p = \tau = 1$, atingindo a saída PV desejada.

6. CONCLUSÃO

Sistemas de controle com ações PID mostram-se eficientes, quando são bem sintonizados porém, os métodos adotados na prática são, na mioria, métodos clássicos com um dos termo P, I, ou D nulo para facilitar a sintonia do controlador. Outro fato de se deve salientar, é a implementação de controladores PID em CLP, ainda é pouco divulgado nos níveis mais baixo do setor industrial de mineração/portuário. Portanto, neste trabalho, apresentou-se os resultados simulados, de um algoritmo de controle PID, desenvolvido e implentado no simulador/emulador CodeSys, apresentando desempenho satisfatório no controle de velocidade de AL de um VV.

AGRADECIMENTOS

Agradecemos ao Departamento de Engenharia de Computação da UEMA por tornar essa pesquisa possível e ao curso de Engenharia Elétrica da UFMA pela contribuição técnico/científicos para o desenvolvimento deste trabalho. Reconhecemos a CAPES por promover e apoiar os estudos avançados que contribuíram para a realiazação deste trabalho. Também, somos especialmente gratos à FAPEMA pelo incentivo à pesquisa de alto nível no Estado do Maranhão.

REFERÊNCIAS

Garpinger, O., Hägglund, T., and Åström, K.J. (2014). Performance and robustness trade-offs in pid control. *Journal of Process Control*, 24(5), 568–577.

Guo, L. and Pecen, R. (2009). Design projects in a programmable logic controller (plc) course in electrical engineering technology. *The technology interface journal*, 10(1).

Hackworth, J.R. and Hackworth, F.D. (2004). *Programmable logic controllers: programming methods and applications*. Pearson New Jersey.

Hanssen, D.H. (2015). *Programmable logic controllers: a practical approach to IEC 61131-3 using CODESYS*. John Wiley & Sons.

John, K. and Tiegelkamp, M. (2010). Concepts and programming languages, requirements for programming systems, decision-making aids.

Korobiichuk, I., Dobrzhansky, O., and Kachniarz, M. (2017). Remote control of nonlinear motion for mechatronic machine by means of codesys compatible industrial controller. *Tehnicki Vjesnik-Technical Gazette*, 24(6), 1661–1667.

Moura, J.P., Neto, J.V.F., Ferreira, E.F.M., and Araujo Filho, E.M. (2020). On the design and analysis of structured-ann for online pid-tuning to bulk resumption process in ore mining system. *Neurocomputing*.

Otto, A. and Hellmann, K. (2009). Iec 61131: A general overview and emerging trends. *IEEE Industrial Electronics Magazine*, 3(4), 27–31.

Silveira, P.R. and Santos, W.E. (1998). Automação-controle discreto-5ª edição. *São Paulo: Editora Érica*.

Souza, L.C. and Pereira, A.L.S. (????). Estudo e aplicação de linguagens de programação utilizando o software codesys. *Instituto federal de educação, ciência e tecnologia de goiás*.

Tiegelkamp, M. and John, K.H. (2010). *IEC 61131-3: Programming industrial automation systems*. Springer.