LUCAS EMANUEL SANTOS TORRES

MODELAGEM MATEMÁTICA DE UM MOTOR CC COM CONTROLADOR PD VIA ALGORITMOS GENÉTICOS APLICADO SOBRE UMA PLATAFORMA MÓVEL

LAVRAS - MG
2022

LUCAS EMANUEL SANTOS TORRES

MODELAGEM MATEMÁTICA DE UM MOTOR CC COM CONTROLADOR PD VIA ALGORITMOS GENÉTICOS APLICADO SOBRE UMA PLATAFORMA MÓVEL

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Lavras como parte das exigências do Curso de Engenharia de Controle e Automação, para a obtenção do título de Bacharel.

Prof. Dr. Leonardo Silveira Paiva
Orientador

LAVRAS - MG
2022

LUCAS EMANUEL SANTOS TORRES

MODELAGEM MATEMÁTICA DE UM MOTOR CC COM CONTROLADOR PD VIA ALGORITMOS GENÉTICOS APLICADO SOBRE UMA PLATAFORMA MÓVEL

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Lavras como parte das exigências do Curso de Engenharia de Controle e Automação, para a obtenção do título de Bacharel.

Prof (a). Me (a). Alessandra Rose Crosara Rios Campos
Prof. Dr. Dimitri Campos Viana
Prof. Dr. Leonardo Silveira Paiva

Prof. Dr. Leonardo Silveira Paiva
Orientador

LAVRAS - MG
2022

AGRADECIMENTOS

Agradeço toda minha família, em especial aos meus pais, Rosilene e Luciano, onde em minha mãe encontrei a dedicação, empenho, sabedoria e resiliência para alcançar meus objetivos, e em meu pai a força, sabedoria e minha vocação e paixão pela engenharia.

Aos meus irmãos de outra mãe que a República Mato Dentro me proporcionou, pelo apoio nos momentos difíceis, pelos valores e lições ensinadas, bem como pelas risadas e parceria ao longo dos anos.

Aos meus amigos, Rhuan Souza, Paulo Goneli e Luisa Bonfim, pela amizade, apoio e companheirismo.

À Universidade Federal de Lavras pela estrutura, conhecimento e oportunidades concedidas.

Ao prof. Leonardo Silveira Paiva pela paciência, pela orientação, por acreditar no meu potencial, pelo conhecimento e incentivo na realização deste trabalho.

Ao Centro Acadêmico de Engenharia de Controle e Automação por todo o conhecimento, oportunidades e desafios entregues.

Por fim, agradeço a Deus.

“Alguns homens veem as coisas como são, e dizem: ‘Por quê?’ Eu sonho com as coisas que nunca foram e digo ‘Por que não?’” (Geroge Bernard Shaw)

RESUMO

Embora a história aponte 1866 como efetivamente o ano de surgimento das máquinas elétricas e do motor elétrico, a humanidade sempre fez uso de fontes motoras para a realização dos mais distintos trabalhos, como trabalhos agrícolas e transporte de cargas. Ao longo da história, o tipo de fonte motora foi modificado à medida que a tecnologia foi avançando, como por exemplo: o uso de tração humana e animal na pré-história e o uso na máquina a vapor na Primeira Revolução Industrial. Embora a teoria de controle já fosse aplicada antes do surgimento do motor elétrico, foi de fato com advento dele que esse processo se mostrou mais eficiente, uma vez que, graças as suas caraterísticas, ele é mais fácil de controlar quando comparado com outros tipos de motores. Esse trabalho foi realizado com a finalidade de desenvolver a plataforma móvel que permita aplicar os resultados calculados, referentes a conceitos de modelagem e controle de posição do sistema. O primeiro passo consistiu em determinar uma função de transferência que descreva o comportamento de um motor elétrico de corrente contínua. Para tal, mesclou-se técnicas tradicionais em caixa preta com o conhecimento dos parâmetros que descrevem o comportamento de um motor CC, possibilitando uma modelagem caixa cinza. Posteriormente, um controlador definido pela sintonia de Ziegler-Nichols foi usado como ponto de partida para Algoritmos Genéticos. Esses AGs têm como objetivo melhorar a resposta ao degrau do sistema, reduzindo tempo de assentamento e sobressinal do mesmo. Com um controlador para posição definido, os dados foram atualizados no Arduino e aplicados ao robô móvel, visando verificar a integridade dos resultados, assim como avaliar se os ganhos do controlador projetado se comportam de acordo com o esperado. Como resultados, o modelo matemático projetado para o motor de corrente contínua contempla as características do modelo para o intervalo de trabalho nominal. Entretanto, em regimes de operação abaixo do nominal, o modelo não consegue atender completamente ao regime transiente do motor CC. Para o controle de posição, os Algoritmos Genéticos são uma ótima alternativa para o ajuste fino dos ganhos do controlador. Os AGs apresentam uma boa performance quando seus parâmetros de trabalho são bem definidos. A plataforma móvel utilizada para validação do controle projetado entrega um resultado satisfatório, conseguindo concluir bem o trajeto definido. Entretanto, devido ela ser composta por material de baixo preço, apresenta alguns erros associados.

Palavras-chave: Motor Elétrico, Modelagem Matemática, Controle, Algoritmos Genéticos, Robótica Móvel.

ABSTRACT

Although the human history points to 1866 as the year of the emergence of electric machines and electric engines, we always used of motor sources to carry out the most different work, such as agricultural works and cargo transport. Throughout history, the kind of this motor source has changed as technology has advanced, such as: the use of human or animal traction in prehistory and the use of the steam engine in the First Industrial Revolution. Although the control theory was already applied before the emergence of the electric engine, it was in fact with its advent that this process proved to be more efficient. Since, due to its characteristics, it is easier to control when compared to other types of engines. This work was carried out with the objective of developing a mobile platform that applies the calculated results, referring to concepts such as modeling and system control. The first step consisted in determining a transfer function that describes the behavior of a direct current motor. For this, traditional black box techniques were mixed with the knowledge of the parameters that describe the behavior of a DC motor, enabling a gray box modeling. Subsequently, a controller defined by Ziegler-Nichol's tuning was used as a starting point for Genetic Algorithms. These AGs aim to improve the step response of the system, reducing settling and overshoot time. With a controller for defined position, the data were updated in the Arduino and applied to the mobile robot, in order to verify the integrity of the results, as well as to assess whether the gains of the designed controller behave as expected. As a result, the mathematical model designed for the direct current motor includes the characteristics of the model for the rated working range. However, in sub-rated operating regimes, the model cannot fully meet the transient regime of the DC motor. For position control, Genetic Algorithms are a great alternative for fine-tuning controller gains. AGs present a good performance when their working parameters are well defined. The mobile platform used to validate the designed control delivers a satisfactory result, managing to complete the defined path well. However, because it is composed of low-priced material, it has some associated errors.

Keywords: Electric Engine, Mathematical Modeling, Control, Genetic Algorithms, Mobile Robotic.

SUMÁRIO

PRIMEIRA PARTE
INTRODUÇÃO GERAL

1 INTRODUÇÃO

O Trabalho de Conclusão de Curso visou controlar a posição de uma plataforma móvel sobre um trajeto predeterminado dentro de um ambiente controlado, através da definição e cálculo de um modelo matemático e aplicação de um controlador PD sobre cada motor, definido via Algoritmo Genético.

O Objetivo principal do trabalho foi o esclarecimento de conceitos como identificação de sistemas, modelagem matemática, controle PID e Algoritmos Genéticos aplicados a uma planta real – uma plataforma robótica móvel. Como objetivos específicos, tem-se:

- A utilização das técnicas clássicas de modelagem e identificação de sistemas para definir um modelo caixa cinza para o motor de corrente contínua;
- A definição de um controlador para a planta através das técnicas de sintonia de Ziegler-Nichols;
- O ajuste dos ganhos do controlador através da implementação de Algoritmos Genéticos;
- Construção de uma plataforma móvel e implementação do código que comanda a plataforma;
- Aplicação da plataforma móvel, sobre um caminho predeterminado, implementada com o controlador projetado.

Este trabalho foi desenvolvido no formato de artigo acadêmico, sendo desmembrado em duas distintas partes.

A primeira parte é composta por:

1) Uma breve introdução, contendo uma contextualização e objetivo do trabalho, assim como sua organização;
2) Uma revisão bibliográfica, contendo o esclarecimento e contextualização dos assuntos abordados no artigo;
3) E considerações finais, exibindo uma breve análise dos resultados encontrados com a realização deste trabalho.

A segunda parte é composta exatamente pelo artigo científico, que descreve e analisa todo o procedimento utilizado para a realização deste trabalho.

2 REFERENCIAL TEÓRICO

2.1 Modelagem e identificação de sistemas

Segundo Lathi (2007), um sistema é descrito como uma estrutura que processa um conjunto de sinais de entrada e retorna um conjunto de sinais de saída. Podendo apresentar sinais de distintas naturezas, como um impulso de tensão elétrica ou uma vibração que um fluido aplica sobre uma membrana. O sistema é um ambiente onde esses sinais são interpretados e retransmitidos como um sinal distinto. A fim de se conhecer mais sobre esses determinados sistemas a modelagem matemática e a identificação de sistemas se faz necessária.

A modelagem matemática é um campo que busca formas de desenvolver e implementar os modelos matemáticos de sistemas reais através de uma análise físico-matemática destes. Contudo, para esse tipo de análise, faz-se necessário um conhecimento aprofundado do sistema a ser modelado, a fim de se conhecer as nuances que esse sistema pode apresentar. Entre as diversas técnicas úteis para realização da modelagem matemática, uma das mais recorrentes é a modelagem caixa branca. A modelagem caixa branca baseia-se no conhecimento da natureza ou física que envolvem o processo a fim de se equacionar o sistema (AGUIRRE, 2007).

Funcionando como técnicas alternativas à modelagem matemática, a identificação de sistemas estuda técnicas que requerem pouco ou nenhum conhecimento prévio do sistema (AGUIRRE, 2007). Também conhecida como modelagem caixa preta ou modelagem empírica, esses métodos baseiam-se em analisar a resposta do sistema a um determinado sinal de entrada, comumente a resposta do sistema a uma entrada em degrau.

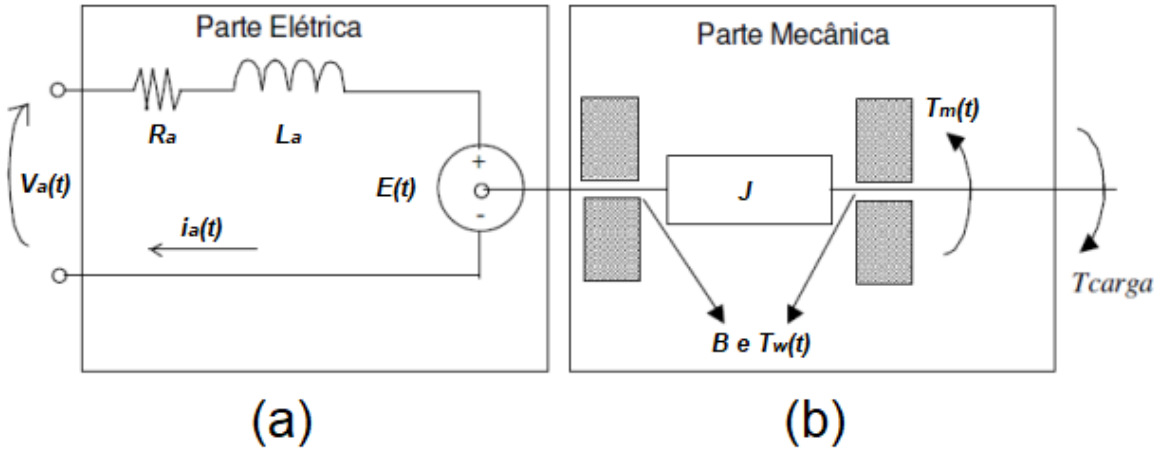
2.1.1 Modelagem matemática de um motor de corrente contínua

Hoje, o motor elétrico é uma ferramenta essencial no funcionamento da maioria dos equipamentos, sendo utilizado para operações simples, como em um ventilador; mas também para atividades de extrema complexidade e precisão, como para articular um manipulador que realiza cirurgias. E, desde o advento da primeira máquina elétrica em 1866, pelo cientista Werner Siemens (WOLFF, 2004), o motor elétrico é alvo de inúmeros estudos disponíveis na

literatura. Na busca de se conhecer as leis da física que regem o maquinário, a busca por um modelo que descreve o sistema do motor fez-se necessário.

Para Canal, Valdiero e Reimbold (2017), a modelagem matemática descreve um fenômeno escrito em função de relações matemáticas. E, na modelagem matemática de um motor de corrente contínua, devem ser considerados as características elétricas, mecânicas e eletromecânicas do motor. A Figura 01 ilustra o circuito equivalente resultante para um motor CC, com o circuito elétrico da armadura e diagrama de corpo livre do rotor separados em (a) e (b), respectivamente.

Figura 01 - Diagrama esquemático de um motor CC: (a) circuito elétrico de armadura; (b) diagrama de corpo livre do eixo do motor.



Fonte: adaptado de Da Silva (2012)

Ao analisar a Figura 01-a, o circuito elétrico do motor é descrito através de uma equação do somatório das tensões na malha, definida através da lei de Kirchhoff para circuitos elétricos, resultando em:

$$Ra.ia(t) + La.\frac{d\ ia(t)}{dt} + E(t) - Va(t) = 0 \qquad (01)$$

Na Equação 01 tem-se que: Ra representa a resistência de armadura do motor, ia representa a corrente elétrica da armadura, La é a indutância do enrolamento da armadura, E expressa a tensão de força contra eletromotriz gerada na armadura e Va representa a tensão aplicada na armadura do motor.

A partir das leis de Newton, o somatório de torques da parte mecânica do motor, expresso na Figura 01-b, pode ser escrita pela equação:

$$Tm(t) = Tw(t) + Tj(t) + Tl(t) \quad (02)$$

Sendo que, na Equação 02, Tm é o torque eletromagnético do motor. O torque de perdas é representado por Tw, Tj é o torque de inércia do motor e Tl representa o torque da carga aplicada no motor.

Segundo o princípio de d'Alembert, o conjugado de inércia do eixo é proporcional ao momento de inércia J e a aceleração angular do motor, representada pela derivada da velocidade angular ω (REGNER, 2019).

$$Tj(t) = J * \frac{d}{dt}\omega(t) \quad (03)$$

O conjugado de perdas Tw é representado pelo somatório entre o torque de atrito viscoso do motor e a contribuição de atrito do motor e da carga Tf. O toque de atrito viscoso do motor é denotado pelo coeficiente de atrito viscoso do motor Bv, que é linear e proporcional à velocidade angular do motor. Conforme descrito por Fonseca (2014), o atrito Tf possui caráter não linear e de intensidade muito inferior ao atrito viscoso, e consequentemente pode ser desprezado para modelagem do sistema.

$$Tw(t) = \omega(t) * Bv + Tf(t) \quad (04)$$

Para a modelagem matemática do motor, é considerado um motor livre, e consequentemente o toque de carga Tl é desprezado para o cálculo. Então, substituindo as Equações 03 e 04 na Equação 02, tem-se uma nova equação para descrever o somatório de torques do motor.

$$Tm(t) = \omega(t) * Bv + J * \frac{d}{dt}\omega(t) \quad (05)$$

A tensão de força contra eletromotriz E gerada internamente no motor é proporcional à velocidade angular ω e à constante de tensão de força contra eletromotriz Ke. Essa constante representa uma característica importante do motor, em que ela determina a velocidade do motor a um dado valor de tensão aplicada. Para tal, deve ser considerado o fluxo e a corrente de campo constantes e desprezado oscilações de fluxo decorrente de outros efeitos (STUART, 2013).

$$E(t) = Ke * \omega(t) \tag{06}$$

O Torque do motor Tm é oriunda da equação de força magnética (F = BLi) (STUART, 2013). Um condutor energizado e na presença de um fluxo de campo perpendicular à espira do condutor sofre a ação de uma força. Sendo o valor da força proporcional ao valor da corrente elétrica aplicada nesse condutor. Segundo Da Silva Viganó e Prado (2019), a contribuição de todos os condutores no campo magnético produz um vetor de força total que desencadeia o toque eletromagnético desenvolvido pelo motor, proporcional à corrente de armadura.

$$T(t) = Kt * Ia(t) \tag{07}$$

Na Equação 07, Kt representa a constante de torque do motor, denominado pelas características físicas do motor.

Considerando uma tensão constante aplicada ao circuito de armadura e o motor operando em regime permanente, as Equações 06 e 07 são utilizadas para um fator comum, obtendo-se outra relação útil (STUART, 2013):

$$Ke = Kt \tag{08}$$

As Equações 06 e 07 representam os comportamentos eletromecânicos do motor, e com as Equações 01 e 05 constituem o conjunto básico que descrevem o comportamento de um motor CC. Tomando a transformada de Laplace dessas, chega-se em:

$$Va(s) - E(s) = (Ra + La * s)Ia(s) \tag{09}$$

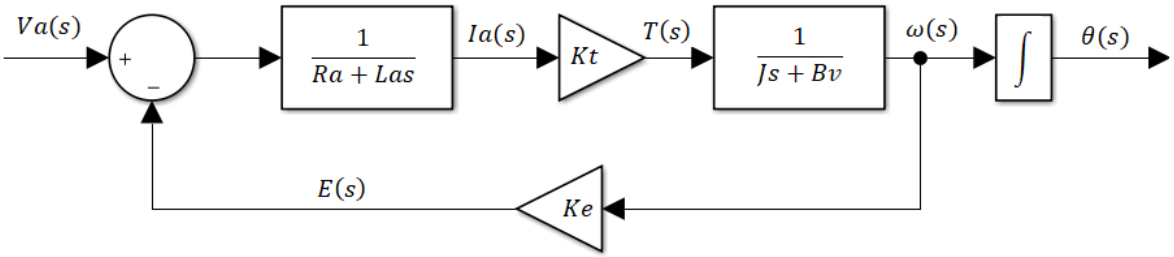
$$T(s) = (J * s + Bv)\omega(s) \tag{10}$$

$$E(s) = Ke * \omega(s) \tag{11}$$

$$T(s) = Kt * Ia(s) \tag{12}$$

Com as equações que descrevem o comportamento do motor CC, é possível representar o sistema em diagrama de blocos, através da Figura 02. O diagrama de blocos representa o sistema com um sinal de entrada, dado pela tensão aplicada na armadura do motor $Va(s)$, e com a saída sendo a velocidade angular $\omega(s)$ ou a posição angular $\theta(s)$, ou ambas.

Figura 02 – Diagrama de blocos de um motor de corrente contínua



Fonte: Do autor (2022)

A partir do diagrama de blocos, mostrado na Figura 02, e realizando as devidas manipulações matemáticas nas Equações 09, 10, 11 e 12, a função de transferência que descreve a relação entre a tensão de entrada $Va(s)$ e a velocidade angular $\omega(s)$ é definida como:

$$G(s) = \frac{\omega(s)}{Va(s)} = \frac{Kt}{(La*s+Ra)*(J*s+Bv)+Kt*Ke} \tag{13}$$

A partir da Equação 13, a função de transferência que descreve a relação entre a posição angular $\theta(s)$ e a tensão de armadura do motor $Va(s)$ é definida como:

$$H(s) = \frac{\theta(s)}{Va(s)} = \frac{Kt}{(La*s+Ra)*(J*s+Bv)*s+Kt*Ke*s} \tag{14}$$

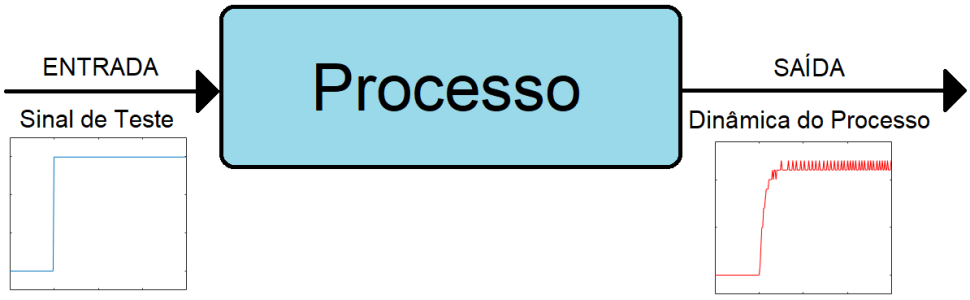
Contudo, a indutância de armadura La possui efeito muito menor comparado com ação mecânica, na maioria dos casos (REGNER, 2019). E consequentemente ela pode ser desprezada na Equação 13, caso necessário, tornando o motor tipicamente um sistema de segunda ordem descrito pela função de transferência:

$$H(s) = \frac{\theta(s)}{Va(s)} = \frac{Kt}{Ra*J*s^2 + (Ra*Bv + Kt*Ke)*s} \quad (15)$$

2.1.2 Métodos de modelagem matemática baseados na resposta ao degrau

Para projetar um sistema de controle, faz-se necessário realizar comparações entre os sinais de entrada e saída envolventes (COELHO e DOS SANTOS COELHO, 2004). Comumente, os sinais particulares de entrada são submetidos a comparações com as respostas dos sistemas. Alguns sinais de testes são tipicamente utilizados para determinar a dinâmica de um sistema, dentre os mais recorrentes tem-se: impulso, degrau e rampa. A Figura 03 ilustra os sinais de um processo para determinar a dinâmica desse sistema.

Figura 03 – Componentes de um sistema de controle



Fonte: Do autor (2022)

O sinal do tipo degrau, mais usual dentre os sinais de teste, consiste no chaveamento abrupto na magnitude desse sinal, pelo acréscimo ou decréscimo (COELHO e DOS SANTOS COELHO, 2004). O acionamento de um interruptor funciona como a aplicação de um sinal do tipo degrau em uma carga pois, em um dado intervalo de tempo o chaveamento do dispositivo altera abruptamente o fornecimento da tensão. No exemplo, deve-se considerar um sistema alimentado com tensão CC.

Alguns parâmetros dos modelos matemáticos podem ser determinados através da curva de reação do sistema, quando se é aplicado um sinal degrau na entrada. Segundo Brosilow e

Joseph (2002), uma simples análise gráfica da resposta é adequada para a caracterização da dinâmica do processo, além da definição dos parâmetros que definem a função de transferência.

Um modelo paramétrico da dinâmica de um sistema com reposta tipicamente de primeira ordem é da forma:

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau * s + 1} * e^{-\theta * s} \tag{16}$$

Através da Equação 16 tem-se que K representa o ganho do sistema, τ é a sua constante de tempo e θ expressa o tempo morto do sistema. Essas três variáveis parametrizam a função de transferência do processo, sendo fundamentais para a definição do modelo matemático. A Equação 16 também pode ser empregada para processos em que o tempo morto é nulo, ou seja, a resposta do sistema apresenta tempo de reação instantâneo ao se aplicar um sinal na entrada.

Para sistemas de segunda ordem a Equação 17 descreve o modelo paramétrico do processo:

$$\frac{Y(s)}{U(s)} = \frac{{\omega_n}^2}{s^2 + 2 * \zeta * \omega_n * s + {\omega_n}^2} \tag{17}$$

Na Equação 17 tem-se que ω_n representa a frequência natural do sistema e ζ é o fator de amortecimento.

A literatura está repleta de métodos que determinam a modelagem matemática de sistemas de primeira e segunda ordem, utilizam a resposta do processo ao degrau unitário. Para Coelho e Dos Santos Coelho (2004), entre os disponíveis, os métodos mais recorrentes foram apresentados por Ziegler/Nichols (1942), Sundaresan/Krishnaswamy (1977), Nishikawa (1984), Smith (1985), Mollenkamp (1988) e Haguglund (1991). Tais métodos são muito difundidos na literatura, e serão omitidos nesse momento. Para mais informações sobre eles, poderá, no entanto, consultar as seguintes referências bibliográficas: [Aguirre, 2007; Coelho e Dos Santos Coelho, 2004].

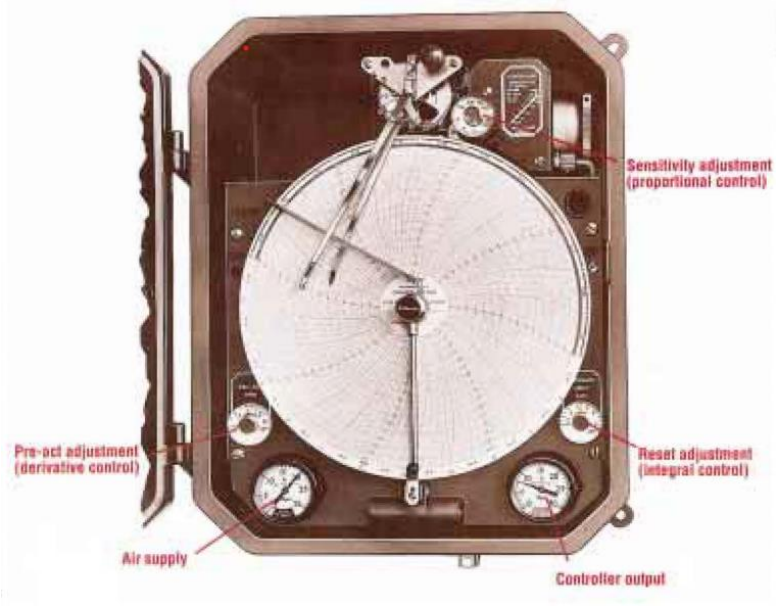
2.2 Controle PID

Segundo Paiva (2010), o controle automático surgiu com a necessidade de a humanidade substituir o homem por equipamentos capazes de realizar suas ações em algumas tarefas de demasiada periculosidade, reduzindo os custos de produção e melhorando a qualidade. Relatos indicam a existência de uma máquina autônoma já em 1500 A.C. Se tratava de um relógio de água autorregulado, encontrado nos restos das civilizações egípcias e mesopotâmicas (VILLAÇA e SILVEIRA, 2013).

Desde então, apareceram diversos modelos e formas de controle, entre elas o controlador PID, umas das mais conhecidas. Ogata (2011) completa que, o controle PID é um dos mais úteis, principalmente quando o modelo matemático do processo é desconhecido.

Como lembra Villaça e Silveira (2013), o primeiro controlador em três termos, proporcional, integral e derivativo (PID), foi introduzido em 1936 pela Taylor Instrument Company. Contudo, foram John G. Ziegler e Nathaniel B. Nichols, em 1942, responsáveis pelo método capaz de sintonizar cada um dos três termos e então popularizar o controlador PID.

Figura 04 – O Fulscore controller, primeiro controlador PID.



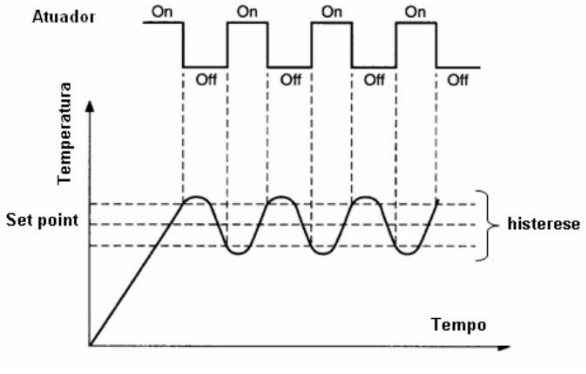
Fonte: Babb (1980)

Um controlador é uma função capaz de realizar algumas operações matemáticas a partir da comparação entre uma referência predeterminada com o sinal de saída do processo, calcular a diferença entre os sinais e determinar o erro, e através desse erro definir um sinal de controle

a fim de alcançar o valor objetivo (PAIVA, 2010). Chamadas de ações de controle, o entendimento dessas operações é essencial para a compreensão da sintonia de controlador PID. As quatro ações de controle são identificadas como: ação liga-desliga (on-off), ação proporcional, ação integral e ação derivativa.

A ação liga-desliga é aplicada em sistemas em que não há a necessidade de boa precisão ou desempenho, permitindo oscilação contínua da variável de controle na margem do valor objetivo. A saída do controlador assume apenas dois valores possíveis, ligado ou desligado. A Figura 05 exemplifica um sistema com esse tipo de ação. Um exemplo de aplicação recorrente é o controle de nível de água através de uma boia.

Figura 05 – Saída de um controlador liga-desliga.



Fonte: Ribeiro (2015)

A ação proporcional leva esse nome pois a posição do elemento final é proporcional à magnitude da diferença entre o setpoint e a medição. De caráter contínua, analógica e uniforme, a ação proporcional apresenta saída proporcional à amplitude do erro no controlador (RIBEIRO, 2005). Quando comparado à ação liga-desliga, o controle proporcional permite que uma válvula de controle assuma qualquer valor de abertura entre 0% e 100%. Segundo Paiva(2010), a ação proporcional é capaz de eliminar as oscilações decorrentes do controle liga-desliga, entretanto não ocorre a eliminação do erro em regime permanente do processo.

Segundo Ribeiro (2005), a ação integral é proporcional à integral do erro, no tempo. Em outras palavras, a ação integral é de caráter corretivo, proporcional à duração da existência do erro entre o setpoint e a medição. Atuando à medida que existe erro, a ação integral é um complemento para ação proporcional, capaz de eliminar o desvio permanente do sistema. De acordo com Ribeiro (2005), embora elimine o erro em regime permanente, a ação integral não consegue eliminar o pico do sinal (overshoot). E como dito por Paiva (2010), quando é

associada à ação proporcional, a ação integral se inicia após a proporcional, existindo um atraso entre ambas.

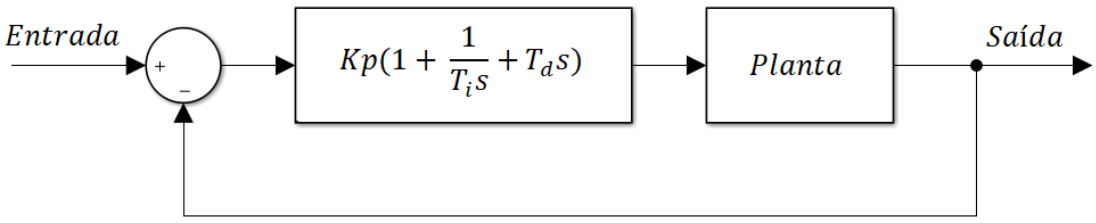
A ação derivativa é capaz de detectar a variação do erro entre o setpoint e a medição e então fornecer uma saída proporcional a essa variação, ou seja, a ação é proporcional a derivada desse desvio entre o setpoint e a medição, no tempo (RIBEIRO, 2005). Segundo Paiva (2010), a ação derivativa não se prepara para o surgimento do erro, gera uma componente proporcional decorrente da variação desse erro e consequentemente atrasa a resposta da ação proporcional do controlador. Comumente empregada em conjunto com a ação proporcional, a ação derivativa modifica a intensidade da ação proporcional, variando a sensibilidade. Num controle PID, a ação derivativa é a primeira a atuar no processo.

O controlador PID é uma técnica de controle de processos que combina os três tipos de ação: proporcional, integral e derivativa. Esse controlador apresenta diversas técnicas de sintonia descritas na literatura e com a utilização dessas regras, ajustes finos no controlador PID podem ser realizados em campo. O controlador PID é definido a partir da seguinte função de transferência:

$$G(s) = Kp * (1 + \frac{1}{Ti*s} + Td * s) \tag{18}$$

Na Equação 18, Kp representa o ganho proporcional, Ti é o tempo integral e Td expressa o tempo derivativo. A Figura 06 representa o diagrama de blocos de uma planta com o controlador PID.

Figura 06 – Diagrama de blocos de um processo com um controlador PID



Fonte: Do autor (2022)

Embora o controlador PID seja o mais recorrente, em algumas aplicações não há a necessidade de se ter as três ações de controle atuando. Para tal, o controlador PID é encontrado em outras três configurações possíveis: controlador P, controlador PI e controlador PD. Essas

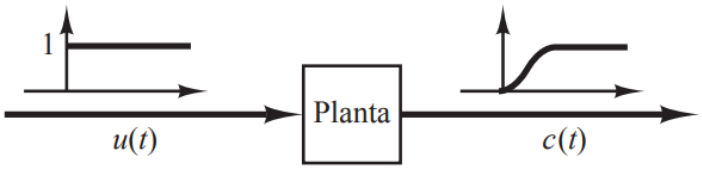
três variações são alcançadas através da Equação 18 em que: para um sistema sem a ação integral, o tempo integral tende ao infinito e para um sistema que não apresenta a ação derivativa, o tempo derivativo é nulo.

2.2.1 Método de sintonia de Ziegler e Nichols em malha fechada

As regras de sintonia de Ziegler e Nichols são muito úteis para sistemas em que a modelagem matemática é desconhecida, também podendo ser aplicadas para os processos que a modelagem é conhecida. Segundo Ogata (2011), a sintonia de Ziegler e Nichols do controlador PID baseia-se na análise da resposta experimental ao sinal degrau ou no cálculo do valor de Kp, quando apenas uma ação proporcional é empregada e que esse valor resulte em uma estabilidade marginal do processo.

Ziegler e Nichols descreveram dois métodos para realizar a sintonia de um controlador PID. O primeiro consiste em uma análise gráfica da curva de reação ao sinal degrau de um sistema em malha aberta, comumente empregada para plantas sem integradores ou par de polos conjugados. A Figura 07 exemplifica a que tipo de saída que pode ser aplicada esse primeiro método.

Figura 07 – Resposta ao degrau unitário de uma planta adequada para aplicar o primeiro método de Ziegler-Nichols.



Fonte: Ogata (2011)

O segundo método de sintonia, em malha fechada, baseia-se primordialmente em determinar o valor do Kcr e Pcr, ganho crítico e período crítico, respectivamente. O processo é submetido a um controlador puramente proporcional, considerando $T_i = \infty$ e $T_d = 0$. Em sequência, a malha de controle é colocada em oscilação. Segundo Ogata (2011), o ganho Kp é elevado de zero até o valor crítico Kcr, ponto em que a saída do sistema expressa uma oscilação sustentada pela primeira vez. O método só se aplica caso o sistema apresente um valor de Kcr em que as oscilações sejam sustentadas. De acordo com Senai (1999), com Kcr definido, é

possível determinar a frequência de oscilação e consequentemente o período crítico Pcr. Com as variáveis definidas experimentalmente, os parâmetros Kp, Ti e Td são escolhidos de acordo com a fórmula mostrada na Tabela 01.

Tabela 01 – Método de sintonia de Ziegler-Nichols para controlador PID baseado no ganho crítico Kcr e no período crítico Pcr

$Tipo\ de\ controlador$	K_p	T_i	T_d
P	$0{,}5K_{cr}$	∞	0
PI	$0{,}45K_{cr}$	$\frac{1}{1{,}2}P_{cr}$	0
PID	$0{,}6K_{cr}$	$0{,}5P_{cr}$	$0{,}125P_{cr}$

Fonte: Ogata (2011)

2.3 Algoritmos Genéticos

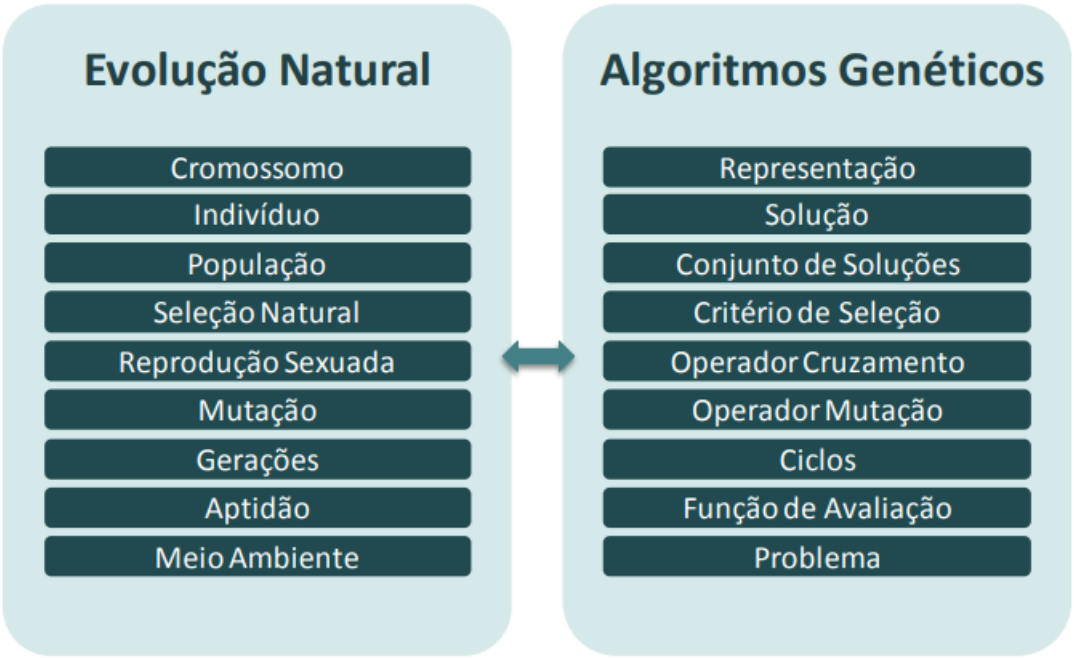
Através de um desejo humano de compreender e controlar o mundo, o termo ciência surgiu. E ao longo da história, o ser humano inspirou-se na natureza ao redor para construir gradualmente todo o conhecimento que o permite compreender inúmeros fenômenos do mundo, além de aperfeiçoar a tecnologia a qual está envolvido. Na mesma linha de raciocínio, o ramo da Inteligência Artificial (IA) e vida artificial se originou, cujo princípio está muito embasado nas teorias desenvolvidas por Charles Darwin na sua obra *A origem das espécies* de 1859.

O termo Algoritmo Genético (AG) foi introduzido entre os anos de 1960 e 1970 por John Holland, junto de colegas e alunos da Universidade de Michigan. O objetivo inicial de Holland não foi projetar algoritmos para resolver problemas específicos e sim estudar formalmente os AGs. Em sua obra *Adaptation in Natural and Artificial Systems* (1975), Holland apresentou o AG como uma abstração da evolução biológica e forneceu uma estrutura para adaptação do algoritmo (MITCHELL, 1998).

A Figura 08 mostra uma analogia entre alguns termos das ciências biológicas e termos de Inteligência Artificial e Algoritmos Genéticos. Segundo a teoria proposta por Charles Darwin, os indivíduos mais aptos geneticamente (com maior probabilidade de reprodução e longevidade) apresentam maior potencial de perpetuarem seus códigos genéticos. Esses códigos genéticos representam os cromossomos e determinam a identidade de cada indivíduo (MENDES, 2013). Analogamente, no ramo dos algoritmos genéticos, os cromossomos geralmente são representados como uma solução candidata para o problema (HOLLAND,

1992). E toda a estrutura dos algoritmos genéticos pode ser explicada a partir de conceitos biológicos.

Figura 08 – Analogia entre evolução natural e Algoritmos Genéticos



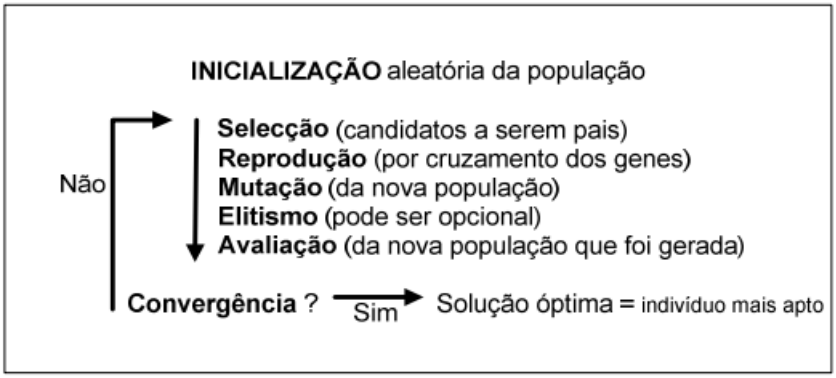
Fonte: Mendes (2013)

Com a popularização dos algoritmos genéticos, a utilização deles para projetos de otimização se destaca nos últimos anos. Projetos de otimização consiste em problemas de maximização ou minimização de uma função ou conjunto de funções em um definido espaço ou domínio. Contudo, a computação evolutiva e os algoritmos genéticos vieram para ampliar e aperfeiçoar esse contexto. Os métodos de otimização normalmente são classificados em programação linear e não-linear. Ainda, os métodos não-lineares são subdivididos em numéricos, determinísticos e estocásticos. Os métodos numéricos consistem em percorrer todo o espaço de soluções possíveis, possuindo baixa eficiência em casos que a região de pesquisa é elevada. Segundo Polak (1971), os métodos determinísticos conseguem encontrar uma solução através de um ponto de referência, para tal deve se definir um vetor de direções para que o algoritmo percorra no espaço, normalmente utilizando a derivada da função. Os métodos estocásticos são baseados em algoritmos de probabilidade; eles utilizam de avaliações da função objetivo e implementam parâmetros estocásticos que visam direcionar ao ponto ótimo (MARTÍNEZ, 2009).

2.3.1 Estrutura Básica de um Algoritmo Genético

Baseado no Darwinismo, Holland (1992) descreve um algoritmo genético como composto pelos seguintes operadores genéticos: inicialização, avaliação, seleção, cruzamento, mutação, atualização e finalização. O fluxograma da Figura 09 exemplifica a estrutura de um Algoritmo Genético.

Figura 09 – Estrutura básica de funcionamento de um Algoritmo Genético



Fonte: Paiva (2010)

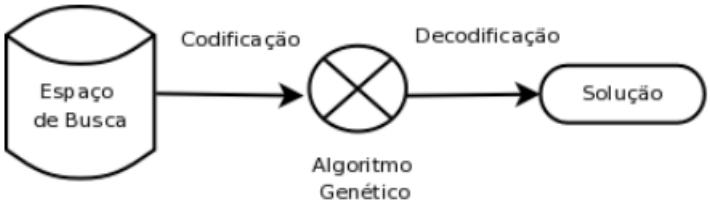
Antes de explorar os operadores genéticos, uma característica fundamental a ser definida na implementação de um algoritmo genético é a codificação ou representação genética. Essa codificação descreve o formato em que as possíveis soluções de um problema são expressas em um indivíduo (cromossomo). As respostas de um problema são possíveis apenas quando acontece uma correta codificação e manipulações de indivíduos (DAVIS, 1991). Existem diferentes tipos de codificação descritos na literatura, dentre elas, as mais difundidas são: a representação binária, a representação por inteiros e a representação por ponto flutuante. Mais conhecida, a representação binária é a clássica descrita nos trabalhos de Holland para algoritmos simples (HOLLAND, 1992). Análogo com os sistemas biológicos, os AGs também são representados como *genótipo* e *fenótipo*. Genótipo remete-se ao próprio gene do indivíduo em seu estado natural e Fenótipo são as características que o gene desencadeia no indivíduo. A Tabela 02 exemplifica alguns exemplos de genótipos e fenótipos. Ainda, a descodificação, processo que complementa a representação genética, é descrito como o processo de construção da solução a partir de um cromossomo (MENDES, 2013). A Figura 10 exemplifica o processo de codificação e decodificação de um indivíduo.

Tabela 02 – Exemplos de fenótipos e genótipos correspondentes em alguns tipos de problemas

Genótipo	*Fenótipo*	*Problema*
0010101001110101	10869	*otimização numérica*
CGDEHABF	*comece pela cidade C, depois passe pelas cidades G, D, E, H, A, B e termine em F*	*caixeiro viajante*
$C_1R_4C_2R_6C_4R_1$	*se condição 1 (C_1) execute regra 4 (R_4), se (C_2) execute (R_6), se (C_4) execute (R_1)*	*regras de aprendizado para agentes*

Fonte: Lucas (2002)

Figura 10 – Processo de Codificação e Decodificação em um Algoritmo Genético



Fonte: Gonçalves (2002)

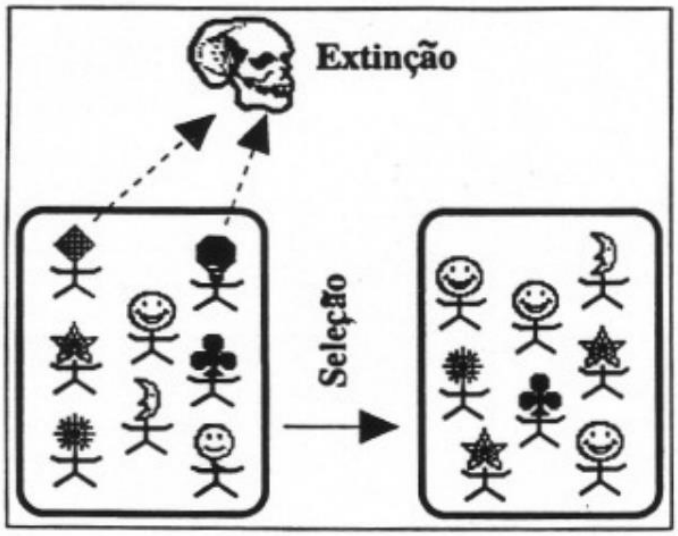
O primeiro operador genético descrito na obra *Adaptation in Natural and Artificial Systems* de 1975, de Holland, é a Inicialização (geração da população inicial). Uma população inicial pode ser gerada na maioria por duas formas distintas. Os indivíduos são gerados através de uma heurística ou eles são gerados randomicamente (SOARES, 2006). Como na teoria da seleção natural, Tanomaru (1995) diz que os indivíduos devem apresentar diferentes graus de adaptabilidade ao espaço que estão inseridos. Ainda, faz-se necessário que a população cubra o máximo possível da área no espaço de busca, não sendo demasiadamente pequena ou grande. Populações pequenas diminuem a probabilidade de se atingir uma solução ótima e populações grandes requerem um alto consumo de tempo computacional.

A avaliação é o operador genético ao qual todos os indivíduos de uma população são verificados segundo uma função objetivo. De acordo com Lucas (2002), esse é momento do AG e que mais depende do problema como um todo, pois o Algoritmo Genético deve ser capaz de responder quão boa uma solução é para o problema. A função objetivo atribui a cada indivíduo um valor que remete a quão bem adaptado esse cromossomo está no espaço (TANOMARU, 1995), ou seja, quanto maior essa medida, maiores são sua possibilidade em

sobreviver, reproduzir e passar o código genético para gerações subsequentes. Esse valor, em inglês, é conhecido como “fitness”.

A seleção é um operador genético que imita os processos de seleção natural e reprodução, em que os indivíduos mais aptos e com boas características perpetuam a espécie, enquanto aqueles com baixa adequabilidade apresentam maior chance de serem extintos. Segundo Lucas (2002), a seleção é um processo dirigido e cumulativo. Dirigido pois a seleção ocorre de forma determinística, ou próximo disso, visto que um indivíduo deve ser capaz para sobreviver em um dado ambiente. E cumulativo pois os benefícios de uma carga genética são mantidos entre as gerações subsequentes. A Figura 11 elucida o processo de seleção. Na literatura existem vários métodos para seleção, entretanto, os principais são: Seleção por Ranking, Seleção por Roleta, Seleção por Torneio, Seleção Uniforme e Elitismo.

Figura 11 – Processo de seleção para uma população de 8 indivíduos.

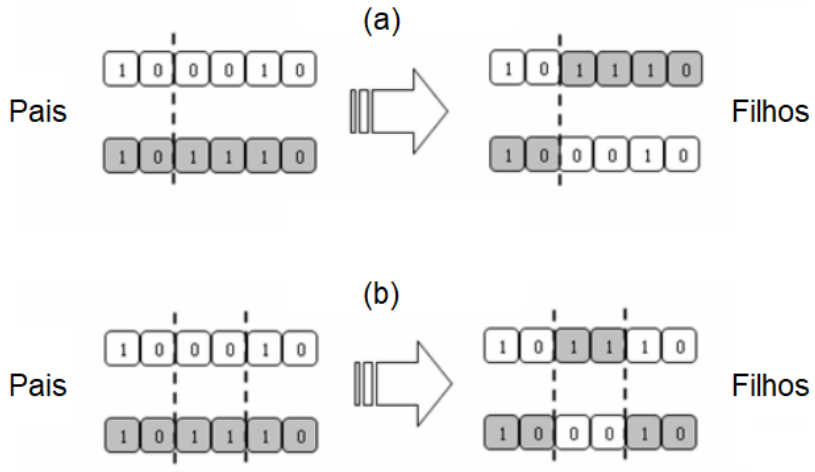


Fonte: Tanomaru (1995)

Reprodução, cruzamento ou recombinação é o operador genético que se comporta como um mecanismo de reprodução sexuada de seres biológicos, também conhecido como “crossover”. De acordo com Holland (1992), o cruzamento, na biologia, é um processo em que pares de cromossomos passam por uma recombinação de alelos, e em Algoritmos Genéticos acontece de forma análoga. O operador extrai os melhores genes dos diferentes indivíduos e os recombinam, a fim de tirar vantagem daquele material genético presente na população. Os dois métodos mais usuais para o cruzamento nos AGs são o cruzamento de um ponto e cruzamento de dois pontos. No primeiro, um ponto é escolhido randomicamente e a partir desse ponto

ocorre a permutação entre os pares; no segundo método, dois pontos de quebra são escolhidos ao acaso e a permutação dos genes ocorre entre esses dois pontos (PAIVA, 2010). A Figura 12 exibe um exemplo de cruzamento em um ponto (a) e dois pontos (b).

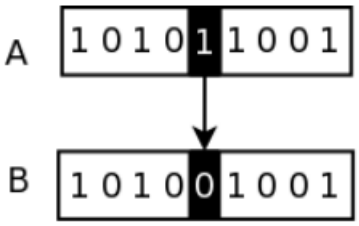
Figura 12 – Operador Genético de Reprodução: (a) cruzamento em um ponto; (b) cruzamento em dois pontos.



Fonte: adaptado de Soares (2006)

A mutação em Algoritmos Genéticos é descrita como um equivalente da busca aleatória (TANOMARU, 1995). Segundo Soares (2006), esse é o processo que assegura a variedade genética em uma população. A mutação consiste na modificação de um ou mais genes da estrutura de um cromossomo, escolhido ao acaso. Alguns dos métodos mais simples e usuais para a mutação em AGs são a mutação por troca (*swap mutation*), onde n pares de genes trocam de posição; a mutação *flip*, em que o gene recebe ao acaso um valor dentro do alfabeto válido; e a mutação *creep*, no qual um valor é somado ou subtraído no gene do indivíduo (LUCAS, 2002). Para populações com codificação binária, é comumente utilizado a inversão de um único gene do cromossomo. A Figura 13 ilustra a técnica.

Figura 13 – Operador de mutação para codificação binária



Fonte: Gonçalves (2002)

Após a aplicação dos operadores genéticos de reprodução e mutação, uma população de descendentes é gerada e consequentemente inseridos na população segundo a teoria dos AGs. Entretanto, essa inserção ocorre de diferentes maneiras. No formato mais tradicional, todos os indivíduos são substituídos por seus descendentes; nesse caso, a população de descendentes possui mesmo tamanho da população geradora. Esse modelo é conhecido como estratégia "virgula", e nela perde-se boas soluções encontradas na população anterior. Outra forma muito difundida é a estratégia "soma" ou estratégia do elitismo. Nela, apenas um percentual de descendentes é produzido, em relação aos pais. Essa estratégia visa garantir que os melhores indivíduos da população anterior permaneçam, substituindo os indivíduos com pior valor da função objetivo (operador avaliação) pelos novos membros. Segundo Paiva (2010), deve-se ficar atento à porcentagem de indivíduos substituídos, pois caso seja excessiva pode levar a uma convergência prematura do algoritmo genético. Uma convergência prematura pode levar o algoritmo genético a não encontrar a melhor solução possível para a função.

A partir da formação de uma nova população, o operador de atualização é aplicado novamente e verificado se o critério de parada é atingido. Caso não seja atingido, é reiniciado o ciclo na população; caso seja atingido, o algoritmo genético é finalizado. A finalização não é necessariamente um operador genético, apenas é composta por alguns testes para dar fim ao processo do AG. A finalização é atingida quando um número de gerações é criado, uma solução ótima é encontrada, dentre outros.

A estrutura de um Algoritmo Genético é muito explorada na literatura, e no atual trabalho ela foi apresentada de forma simplista. Para mais detalhes sobre a estrutura de um AG, consultar as seguintes referências: [Davis(1991), Holland (1992), Mitchell (1998)].

2.4 Robótica

Embora o robô seja uma invenção do século XX, a história mostra que a ideologia por trás da robótica surgiu muito antes. Talvez o relato histórico mais antigo encontrado é de um cachorro mecânico encontrado no Egito antigo e datada de 2000 A.C. Na mitologia grega, existem escrituras sobre a capacidade de Hefesto, deus da tecnologia, metal, artesões e ferreiros, em dar vida a seres de metais, como dito por Bulfinch (2002):

> Tinha o poder de dar movimento próprio às suas obras, de sorte que os trípodes (carros e mesas) podiam mover-se sozinhos para entrar ou sair

> do palácio celestial. Chegava a dotar de inteligência as servas de ouro que fazia para cuidar dele próprio. (BULFINCH, 2002, p. 10)

Figura 14 – Demonstração do Unimate ao público no programa Tonight Show

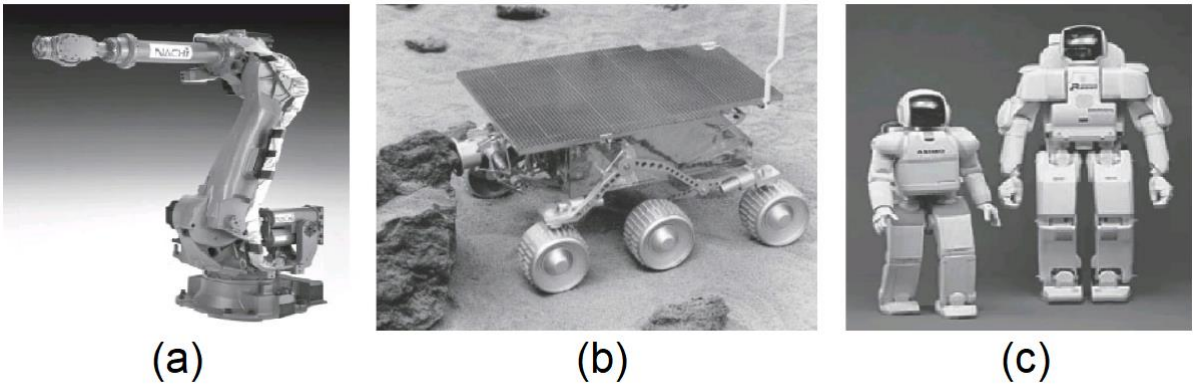
Fonte: Automate.org. (https://www.automate.org/a3-content/joseph-engelberger-unimate)

A obra fictícia e peça teatral *Rossum's Universal Robots* do autor Karel Capek, em 1922, foi uma das maiores contribuições para a popularização do termo "robô" e consequentemente habitar a consciência das pessoas em volta do mundo (CAPEK, 2004). Mas a migração entre a imaginação e a realidade exigiu anos de pesquisas, principalmente nos campos da matemática, física e eletrônica. Na literatura existem diferentes relatos da criação do primeiro robô, contudo, segundo Hockstein *et al.* (2007), foi com a introdução do Unimate pela General Motors, em 1958, que a utilização de robôs na industria disparou. Esse era um robô utilizado para auxiliar nas linhas de produção de automóveis, operando pela primeira vez em uma linha de montagem em 1961. A Figura 14 mostra a primeira aparição do Unimate em uma apresentação no programa Tonight Show. Nessa oportunidade o robô foi programado para realizar algumas manobras como jogar uma bola de golfe em um copo, servir cerveja e conduzir a banda do programa. Desde então, a robótica se tornou algo recorrente dentro do ambiente fabril, sendo aplicada para as mais distintas funções.

Os robôs são conhecidos como agentes ou estruturas físicas designados para a execução de uma determinada atividade em um ambiente físico. Segundo Rugel e Norvig (2013), duas partes fundamentais em uma estrutura robótica são os efetuadores (end effector) e os sensores. Os efetuadores são as estruturas responsáveis por exercer as forças físicas num dado ambiente, como garras, articulações, rodas e pernas dos robôs; os sensores são as estruturas que permitem o robô perceber o ambiente a sua volta, como câmeras, giroscópios, ultrassom entre outros

sensores. Com essa ideologia, existe uma vasta diversidade de robôs que operam as mais distintas atividades, em diferentes ambientes e situações.

Figura 15 – Categoria de Robôs: (a) manipulador robótico industrial; (b) Soujouner da Nasa, um robô móvel; (c) robôs humanoides da Honda, exemplo de manipuladores móveis



Fonte: adaptado de Rugel e Norvig (2013)

Normalmente, a maioria das estruturas robóticas são distribuídas entre três categorias principais (RUGEL e NORVIG, 2013). Os manipuladores, popularmente conhecidos como braços mecânicos, são estruturas fixas em seu ambiente de trabalho. É composto por várias articulações e um efetuador que permite que o manipulador coloque seu efetuador em qualquer posição dentro de seu local de trabalho. A Figura 15-a ilustra um modelo de manipulador muito empregado em linhas de montagem de automóveis. O segundo grupo são os robôs móveis, exemplificado pela Figura 15-b. Esse grupo consiste em robôs de deslocamento, ou seja, são estruturas voltadas para se locomover de um local a outro; e por consequência são constituídos de rodas, pernas ou estruturas similares. A terceira e última categoria consiste nos manipuladores móveis. Esse grupo combina a mobilidade de robôs moveis com a manipulações de um efetuador dos manipuladores. A Figura 15-c são dois modelos de robôs humanoides da empresa Honda que mesclam algumas características dos dois modelos anteriores.

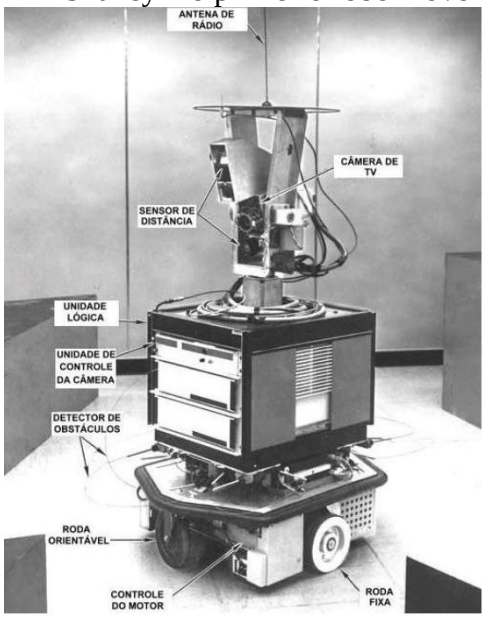
A robótica de manipuladores é muito difundida na literatura, e como não é o foco do trabalho, ela será omitida nesse momento. Para mais informações, características, e princípios matemáticos sobre manipuladores, consultar as seguintes referências: [Craic, 2005; Baturone, 2005; Russel e Norvig, 2013]

2.4.1 Robótica Móvel

Como um grande diferencial da robótica móvel para a robótica de manipuladores, o espaço de trabalhos desses robôs é demasiadamente grande, possibilitando que eles consigam ser empregados nas mais variadas funções e aplicações. Um robô móvel consiste em um conjunto de máquina acoplado a um sistema de transporte e manipulação de materiais (ROCHA, 2001), sendo ligado, comumente, através de uma rede sem fio a uma sede de computadores. De acordo com Baturone (2005), esse tipo de robô requer inteligência suficiente para reagir e tomar as decisões baseando-se em observar o ambiente em seu entorno, necessitando de um sistema de navegação automática, incluindo tarefas de planejamento, percepção e controle.

Surgiram com a necessidade de reduzir ou eliminar algumas interações humanas em ambientes perigosos - como o transporte de peças e componentes dentro de uma indústria que normalmente aconteciam por veículos guiados manualmente. O primeiro robô móvel reconhecido na literatura é datado de 1968. Desenvolvido no *Stanford Research Institute* (SRI) e conhecido como *Shakey*, ele conseguia se locomover e empurrar alguns blocos em uma superfície plana graças aos vários sensores equipados nele (PEREIRA, 2003). A Figura 16 é uma fotografia do robô em 1970. Nos anos seguintes, vários novos robôs começaram a aparecer, apresentando suas melhorias, como o *Stanford Cart* em 1977 e o *Hilare* em 1983.

Figura 16 – Shakey – o primeiro robô móvel registrado



Fonte: Pereira (2003)

Segundo Rocha (2001), as principais vantagens da robótica móvel são: aumento do nível de automação e flexibilidade no ambiente, otimização do fluxo de matérias e eliminação de risco humano em locais potencialmente perigosos. Contundo, existe uma problemática envolvendo o uso de baterias, já que o descarregamento dessa bateria pode resultar em problemas para o robô e para as pessoas e ambiente envolvidos. Graças a essas e outras características, o surgimento da robótica móvel trouxe consigo vários desafios tecnológicos, como o melhoramento na tecnologia de baterias, sistemas de navegação automatizados e controle eficiente de rotas (DA ROCHA, 1998 e ROCHA, 2001).

3 CONSIDERAÇÕES FINAIS

A realização deste trabalho comprovou a veracidade dos conceitos de modelagem matemática e controle para a definição de um controlador PID aplicado a um motor de corrente contínua, e a eficiência de algoritmos genéticos para o ajuste dos ganhos do controlador afim de melhorar a curva de resposta do sistema. A aplicação de um controle de posição definido em uma plataforma robótica móvel se mostrou eficiente, uma vez que a plataforma consegue realizar os movimentos desejados mesmo considerando que cada motor é tratado separadamente pelo microcontrolador e que a plataforma foi desenvolvida com material de baixo custo. Por fim, foi possível demostrar a importância em se conhecer a modelagem matemática de um sistema, assim como definir um controlador perante essa modelagem; e que a aplicação desses conceitos em sistemas reais ajuda a compreensão e uma melhor operação desses processos.

4 REFERÊNCIAS BIBLIOGRÁFICAS

AGUIRRE, Luis Antonio. **Introdução à Identificação de Sistemas- Técnicas lineares e não-lineares aplicadas a sistemas reais**. 3. ed. Belo Horizonte, MG: Editora da UFMG, 2007.

BABB, Michael. **Pneumatic Instruments Gave Birth to Automatic Control**. Control Engineering Magazine: Reference Guide to PID Tuning - Part 3. Cahners Publication, 1980.

BATURONE, Aníbal Ollero. **Robótica: Manipuladores y Robots Móviles**. Marcombo, 2005.

BROSILOW, Coleman; JOSEPH, Babu. **Techniques of model-based control**. Upper Saddle River, NJ/EUA: Prentice Hall Professional, 2002.

BULFINCH, Thomas. **O Livro de Ouro da Mitologia: Histórias de Deuses e Heróis**. Tradução de David Jardim Júnior. 26. ed. Rio de Janeiro: Ediouro Publicações, 2002.

CANAL, Ivan Paulo; VALDIERO, Antonio Carlos; REIMBOLD, Manuel Pérez. Modelagem Matemática de Motor de Corrente Contínua e Análise Dinâmica. Congresso Nacional de Matemática Aplicada e Computacional (CNMAC), Gramado, RS, 2016. In: **Proceeding Series of the Brazilian Society of Applied and Computational Mathematics**, v. 5, n. 1, 2017.

CAPEK, Karel. **RUR (Rossum's Universal Robots)**. Traduzido por Paul Selver e Nigel Playfair. Penguin, 2004.

COELHO, Antonio Augusto Rodrigues; DOS SANTOS COELHO, Leandro. **Identificação de Sistemas Lineares**. 1. ed. Florianópolis, SC: Editora da UFSC, 2004.

CRAIG, John J. **Introduction to Robotics: Mechanics and Control**. 3th ed. Person Education, 2005.

DA ROCHA, Rui Paulo Pinto. **Desenvolvimento de um Sistema de Gestão de AGVs**, 1998. Tese de Mestrado. Faculdade de Engenharia da Universidade do Porto.

DA SILVA, Carolina Jaqueline Nascimento. **Caracterização de um Conjunto Didático para Ensaio de Motor de Corrente Contínua**. 2012.

DA SILVA VIGANÓ, Ian; PRADO, Márcia Lissandra Machado. **Projeto e Implementação de Controladores Robustos para um Motor de Corrente Contínua (C.C.)**, 2019.

DAVIS, Lawrence. **Handbook of genetic algorithms**. Van Nostrand Reinhold, New York, 1991.

FONSECA, Henrique Moura. **Modelagem e Controle da Posição Angular de um Motor CC Acoplado a uma Haste Flexível**. 2014.

GONÇALVES, André Ricardo. Algoritmos Genéticos. **FEEC, Unicamp**. Campinas, SP, 2002.

HOCKSTEIN, Neil G. et al. A History of Robots: From Science Fiction to Surgical Robotics. **Journal of Robotic Surgery**, v. 1, n. 2, p 113-118, 2007.

HOLLAND, John H. **Adaptation in Natural and Artificial Systems: An Introduction Analysis with Applications to Biology, Control, and Artificial Intelligence**. London, England: MIT Press, 1992.

LATHI, Bhagwandas Pannalal. **Sinais e Sistemas Lineares**. Tradução de Gustavo Guimarães Parma. 2. ed. Porto Alegre, RS: Bookman, 2007.

LUCAS, Diogo C. Algoritmos Genéticos: uma Introdução. **Apostila referente a disciplina de Inteligência Computacional**, Universidade Federal do Rio Grande do Sul, Brasil, 2002.

MARTÍNEZ, Luiz Carlos Castilho. **Otimização dos Circuitos de Refrigerante nos Trocadores de Calor de Sistemas de Refrigeração por Compressão de Vapor**. 2009. Tese de Doutorado. Pontifícia Universidade Católica do Rio de Janeiro. PUC-Rio.

MENDES, Bianca Gonçalves. **Otimização da Localização de Poços de Petróleo com Completação Seca Utilizando Algoritmos Genéticos**. 2013. Tese de Doutorado. Pontifícia Universidade Católica do Rio de Janeiro. PUC-Rio.

MITCHELL, Melanie. **An Introduction to Genetic Algorithms**. 1st ed. London, England: MIT press, 1998.

OGATA, Katsuhiko. **Engenharia de Controle Moderno**. Tradução de Heloísa Coimbra de Souza. 5th ed. São Paulo, SP: Person Education, 2011.

PAIVA, Leonardo Silveira. **Aplicação de Algoritmos Genéticos para Sintonia de Controladores**. 2010. Tese de Doutorado. Instituto Politécnico do Porto. Instituto Superior de Engenharia do Porto.

PEREIRA, Jonas et al. **Avaliação e Correção do Modelo Cinemático de Robôs Móveis visando a Redução de Erros no Seguimento de Trajetórias**. 2003.

POLAK, Elijah. **Computational Methods in Optimization: a Unified Approach**. London: London Academic press, 1971.

REGNER, Rosângela Rommel. **Modelagem Matemática da Dinâmica da Roda de Tração de um Veículo com Acionamento Elétrico**. 2019.

RIBEIRO, Marco Antônio. Controle e Automação. 1. ed. **Tek Treinamento e Consultoria**: Salvador, BA, 2005.

ROCHA, Rui Paulo. **Estado da Arte da Robótica Móvel em Portugal**. Lisboa, Portugal: Departamento de Engenharia Eletrotécnica da Universidade de Coimbra (FCTUC), 2001

RUGEL, Stuart; NORVIG, Peter. **Inteligência Artificial**. Tradução de Regina Célia Simille de Macedo. 3. ed. Editora Campus, 2013.

SENAI. Fundamentos de Controle de Processo. Manual de Instrumentação elaborado para o **Programa de Certificação de Pessoal e Manutenção** coordenado por Evandro de Figueiredo Neto e Robson Santos Cardoso. ES Brasil, 1999.

SOARES, Márcio Morelli. **Análise do Uso de Algoritmos Genéticos na Otimização do Planejamento Mestre da Produção**. 2006. Tese de Doutorado. Pontifícia Universidade Católica do Paraná. PUC-PR.

STUART, Sam. **Dc Motors, Speed Controls, Servo Systems: An Engineering Handbook**. 3rd ed. Elsevier, 2013.

TANOMARU, Julio. Motivação, Fundamentos e Aplicações de Algoritmos Genéticos. In: **II Congresso Brasileiro de Redes Neurais**. 1995. p. 373-403.

VILLAÇA, Marco Valério Miorim; SILVEIRA, Jony Laureano. Uma Breve História do Controle Automático. **Revista Ilha Digital**, v. 4, p. 3-12, 2013.

WOLFF, Joca. **O Motor Elétrico: Uma História de Energia, Inteligência e Trabalho**. Editora UNERJ, Jaraguá do Sul, 2004.

SEGUNDA PARTE

ARTIGO

ARTIGO – MODELAGEM MATEMÁTICA DE UM MOTOR CC COM CONTROLADOR PD VIA ALGORITMOS GENÉTICOS APLICADO SOBRE UMA PLATAFORMA MÓVEL

Modelagem Matemática de um Motor CC com Controlador PD via Algoritmos Genéticos aplicado sobre uma Plataforma Móvel

Lucas Emanuel Santos Torres*

**lucas.torres@estudante.ufla.br*

Universidade Federal de Lavras - Campus Universitário, CEP 37200-900 • Lavras/MG

Resumo: Desde os primórdios, a humanidade sempre utilizava de fontes motoras para a realização de trabalhos. Surgindo da necessidade de facilitar essas atividades, essas fontes motoras utilizavam de tração humana, tração animal, força da água, vento e vapor, para diferentes épocas da humanidade. E foi efetivamente com o surgimento das máquinas elétricas em 1866 que a situação se transformou completamente, possibilitando ao homem um grande avanço tecnológico. E com a modernização dos motores, assim como dos sistemas em que eles estão inseridos, houve a necessidade em se aplicar a teoria de controle, a fim de conseguir precisão e exatidão no movimento. Neste trabalho, foi desenvolvida uma plataforma robótica móvel equipada com um microcontrolador Arduino que se desloca através de um circuito predeterminado em um ambiente controlado. O robô móvel foi construído com o intuído de aplicar e validar um controlador de posição projetado para o motor de corrente contínua inserido na plataforma. Inicialmente, a metodologia do trabalho teve como base, a definição de uma modelagem matemática para o motor elétrico de corrente contínua, utilizando das técnicas clássicas descritas na literatura. Foi definido um controlador PD inicial pelas técnicas de sintonia de Ziegler-Nichols. Os ganhos do controlador foram submetidos aos Algoritmos Genéticos, visando realizar o ajuste fino para o controle do motor. Esse controlador foi repassado ao robô visando avaliar o resultado do controle de posição calculado. Como resultados, a modelagem matemática definida responde bem a curva de reação do motor CC para valores de tensão nominal. Os Algoritmos Genéticos mostram-se uma ótima alternativa para o controle de posição, entregando um ajuste fino satisfatório para os ganhos do controlador. E, a plataforma desenvolvida responde bem ao controle projetado; contudo, devido ao baixo custos dos materiais utilizados, ela apresenta alguns erros associados.

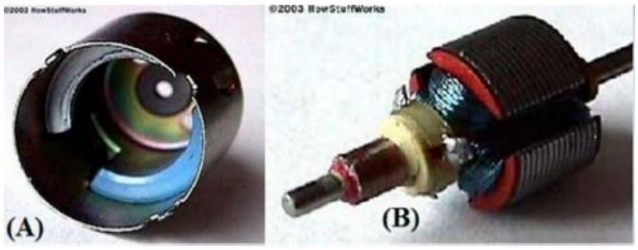
***Palavras chaves**:* Motor de Corrente Contínua, Modelagem Matemática, Controle PD, Algoritmos Genéticos.

1. INTRODUÇÃO

O trabalho de William Gilbert, através da obra *De Magnete, Magneticisque Corporius et do Magno Magnete Tellure – Physiologia Nova* de 1600, é conhecido por disseminar os seus experimentos e conhecimentos no campo da eletricidade e magnetismo (GILBERT, 1958), sendo considerado uma das mentes mais importantes para a história das máquinas elétricas. Contudo, foram três séculos de experimentos e descobertas até efetivamente o surgimento da primeira máquina elétrica, em 1866, através do cientista berlinense Werner Siemens (WOLFF, 2004). O equipamento funcionava tanto como um gerador de eletricidade, como também era capaz de operar como um motor, e já apresentava todas as principais características dos motores elétricos atuais.

Responsáveis por dar vida a uma enorme quantidade de aparelhos e dispositivos atualmente, os motores elétricos são componentes com a função de produzir movimento através da rotação de seu próprio eixo quando submetido à energia elétrica, contínua ou alternada. Estas estruturas são constituídas basicamente por rotor e estator, como mostrado na Figura 01. O corpo do rotor é feito em material ferromagnético, atrelado a eletroímãs ou imãs permanentes; enquanto o estator é composto por bobina de cobre enroladas e adequadamente dispostas em torno do seu corpo, que também é um material ferromagnético.

Figura 01 – Composição de um motor de corrente contínua: (a) Estator; (b) Rotor.



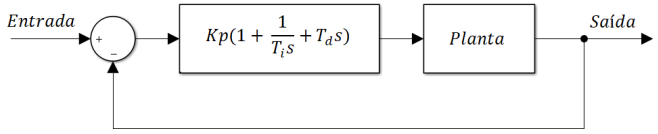
Fonte: adaptado de Da Silva (2012)

Com a aplicação do motor elétrico em sistemas cada dia mais complexos, como cirurgias robóticas e agricultura de precisão, conhecer e controlar o comportamento dinâmico dele se faz plenamente necessário. Sendo um desafio constante que envolve problemas relacionados à engenharia, sistemas mecânicos, circuitos elétricos, teoria de controle e outras áreas, o controle é uma ferramenta fundamental afim de se assegurar um bom desempenho do motor (CANAL, VALDIERO e REIMBOLD, 2017).

Difundida em 1936 e desenvolvido pela *Taylor Instrument Company*, o controlador PID é a estratégia de controle mais conhecida e aplicada atualmente, devido a sua facilidade em se utilizar em plantas que não se conhece sua dinâmica [OGATA, 2011; VILLAÇA e SILVEIRA, 2013]. Contudo, quando se conhece a mecânica do sistema, é possível retirar o melhor potencial dessa técnica de controle. O controlador PID

é uma técnica de controle que combina três ações de controle: proporcional, integral e derivativa. A ação proporcional é responsável por aumentar a velocidade da resposta, a ação integral elimina o erro do sistema em regime permanente, e a componente derivativa tem como objetivo evitar que o erro se torne maior, realizando uma ação de antecipação [PAIVA, 2010; RIBEIRO, 2005]. A Figura 02 demostra a aplicação do controlador PID sobre um sistema.

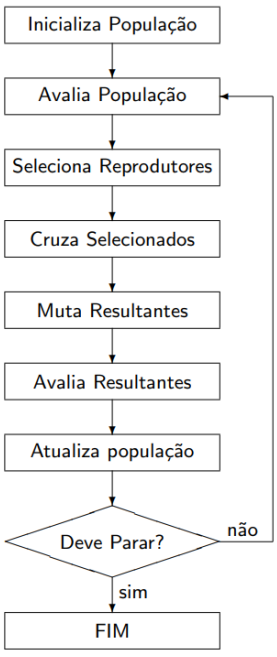
Figura 02 – Diagrama de blocos de um processo com um controlador PID



Fonte: Do autor (2022)

A abordagem em teoria de controle é algo muito difundido na literatura. Embora existam diferentes técnicas para definição de um controlador, o ajuste dele deve ocorrer, em suma, diretamente na planta; isso dificulta o ajuste fino do controle (OGATA, 2011). Como uma alternativa ao ajuste fino dos ganhos de um controlador, a otimização computacional através de algoritmos genéticos (AGs) é um conceito que aplica diferentes técnicas de otimização, que ajudam na busca de uma solução ótima para uma função objetivo. Ao serem implementados para o controlador do projeto, os algoritmos genéticos oferecem uma vasta combinação de possibilidades, explorando todo o espaço de operação de uma função, sendo capazes de limitar os intervalos de possibilidade para cada ganho ou mesmo encontrar uma solução (MITCHELL, 1998).

Figura 03 – Fluxograma de um Algoritmo Genético



Fonte: Lucas (2002)

Baseado no Darwinismo, o funcionamento lógico de um AG é descrito como: inicialmente uma população é gerada, com um número de representantes pré-definida. Cada indivíduo apresenta um sequenciamento genético, também conhecido como cromossomo. Todas as características do indivíduo estão associadas a esses cromossomos, como aptidão genética, capacidade de passar o seu sequenciamento a um novo individuo, entre outras (HOLLAND, 1992). A cada nova geração, alguns membros da população são selecionados (dirigido e cumulativo), passam por um processo de crossover e mutação afim de produzir uma nova geração e assegurando a variabilidade genética [LUCAS, 2002; SOARES, 2006; TANOMARU, 1995]. O algoritmo genético identifica os melhores indivíduos para essa nova geração e descarta os demais. Esse ciclo se repete até o momento em que o critério de parada é alcançado. Podendo ser através de uma solução ótima atingida ou número máximo de gerações alcançada. Cada uma dessas etapas é representada por um operador genético, como mostrado no fluxograma da Figura 03.

Geralmente, para um controle satisfatório, a modelagem matemática é uma peça fundamental. É uma área de grande perpetuação para as engenharias, uma vez que, com os parâmetros calculados ou identificados, torna-se possível uma boa representação matemática da planta (LATHI, 2007). A modelagem matemática pode ser alcançada de diferentes formas. Quando se conhece as leis físicas e químicas que regem o sistema, esse tipo de modelagem é chamada de caixa branca. Segundo Aguirre (2007), na modelagem caixa branca o sistema é descrito por uma equação que descreve todas, ou quase todas, características e parâmetros do sistema a ser modelado . Quando são desconhecidas as características da planta, a alternativa é a modelagem caixa preta. Esse tipo de modelagem parte de técnicas que analisam a curva de reação do sistema quando aplicado um sinal de entrada conhecido (COELHO e DOS SANTOS COELHO, 2004). Na modelagem caixa cinza, mescla das duas anteriores, entende-se que apenas parte dos parâmetros do sistema são conhecidos, sendo eles utilizados para auxiliar as técnicas empregadas na caixa preta.

2. MATERIAIS E MÉTODOS

A seção está dividida em três partes distintas: a plataforma móvel e as peças para construção dela; os softwares utilizados durante o processo, para o funcionamento do robô móvel, definição da modelagem matemática e controlador; e o fluxograma que descreve cada etapa de desenvolvimento.

2.1. PLATAFORMA MÓVEL

A construção da plataforma móvel veio com a necessidade em validar os resultados encontrados, como a modelagem do motor CC e a os ganhos do controlador PD projetados. Para tal, a escolha da estrutura deu-se em modificar um kit pronto, a fim de melhorar alguns pontos negativos desse. A estrutura utilizada para realizar as modificações foi um kit educacional composto por um chassi em acrílico, um Arduino UNO, dois motores CC (3~6V) com caixa de redução, duas rodas em borracha, uma roda boba e um suporte para pilhas modelo AA.

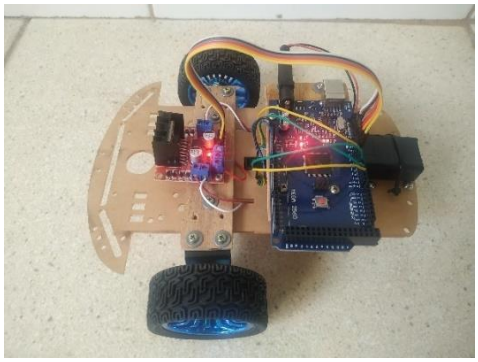
A primeira modificação foi na troca do microcontrolador que acompanha o kit didático. O kit baseia-se na utilização do

microcontrolador Uno, entretanto a placa foi substituída por um Arduino Mega, que apresenta as mesmas características do Uno, entretanto disponibiliza maior números de portas de entrada/saída. Essa modificação ocorreu decorrente a uma necessidade futura em expandir as funcionalidades da plataforma, com a adição de novos sensores que permitam a plataforma ter maior interpretação do ambiente ao seu redor.

A modificação mais importante foi a substituição do par de motores do kit, motivada pela ausência de sensores de velocidade (encoder) nesses motores. Embora o kit permitisse a adição desses módulos, a escolha do motor modelo JGA25-370 e encoder magnético hall 370 se justifica pois ambos são acoplados, o que reduz consideravelmente o erro de medição quando comparado ao modelo apropriado para o kit.

Como o chassi da plataforma não estava preparado para a adição do par de motores JGA25-370, foi preciso realizar a adição de uma superfície em madeira de 128mm x 32mm. Essa chapa foi presa entre os motores e o chassi e visou garantir sustentação da plataforma. A Figura 04 é uma fotografia do robô móvel utilizado.

Figura 04 – Fotografia da plataforma móvel utilizada



Fonte: Do autor (2022)

2.1.1. MICROCONTROLADOR ARDUINO MEGA 2560 REV3

Esse modelo de Arduino é baseado no microcontrolador ATmega2560. Possui cinquenta e quatro pinos entrada/saída digitais, das quais quinze podem ser usados como saída PWM. Também, possui cristal oscilador de 16 MHz, dezesseis entradas/saídas analógicas, 4 portas seriais de hardware UARTs, conexão USB, conector de alimentação, conexão ICSP e um botão de reset da placa (ARDUINO, 2020). A Figura 05 mostra uma placa, produzida pela Arduino SRL.

Figura 05 – Placa Arduino modelo MEGA 2560 Rev3

Fonte: Arduino (2020)

No projeto, tem como função realizar o controle de todo o robô móvel, recebendo informações dos sensores encoder e alimentando o par de motores. Assim, toda ação que ocorre no sistema passa por ele. Baseado na linguagem de programação C/C++, o Arduino é um microcontrolador prático e simples, sendo muito implementado em projetos pequenos e com baixos custos. Para o projeto, o Arduino foi alimentado com uma bateria de 9V através do conector de alimentação.

A principal diferença com a sua versão menor e mais famosa, o Arduino UNO, é a quantidade de portas e periféricos, que lhe permite comandar sistemas maiores e mais complexos. Vale destacar que o Arduino Mega também possui maior memória flash, RAM e ROM, e embora o microcontrolador ATmega2560 seja mais robusto que o presente no Arduino UNO, a capacidade de processamento entre ambos é similar.

2.1.2. MOTOR CC 6V MODELO JGA25-370

Com o intuito de movimentar o robô, um par de motores é utilizado acoplado ao chassi. Com faixa de operação entre 2V e 6V e corrente contínua, esse motor também consegue operar com tensão abaixo de 1V em condições de carga mínima. Seu toque e velocidade de saída variam de acordo com a tensão aplicada. Possui velocidade de 5980 RPM nominal, porém ele é acoplado a uma caixa de redução de 1:46. Essa caixa de engrenagens metálicas visa reduzir a velocidade final e aumentar o torque desenvolvido.

Figura 06 – Motor JGA25-370 com caixa de redução

Fonte: Seeed Technology Co. (https://www.seeedstudio.com/JGA25-370-Geared-Motor-p-4119.html)

A 6V e sem carga, ele entrega 130 RPM com corrente de 100 mA. Já com carga e mesmos 6V, ele opera com 100 RPM, corrente de 0.45A e torque de 1 N.cm. Possui corrente de stall de 1.8A e torque de stall de 3.6 N.cm. Quando acoplado ao sensor encoder e a caixa de redução, o motor tem um peso de 95g. Esse elevado peso é compensado pela robustez apresentada pelo conjunto, que consegue suportar toda a estrutura facilmente.

O controle e alimentação dos motores é realizado a partir da ponte H de modelo L298N. Como o par de motores requer uma corrente elevada para o projeto, a alimentação desses foi separada dos demais componentes do robô móvel. A ponte H foi exclusivamente alimentada com duas baterias 18650

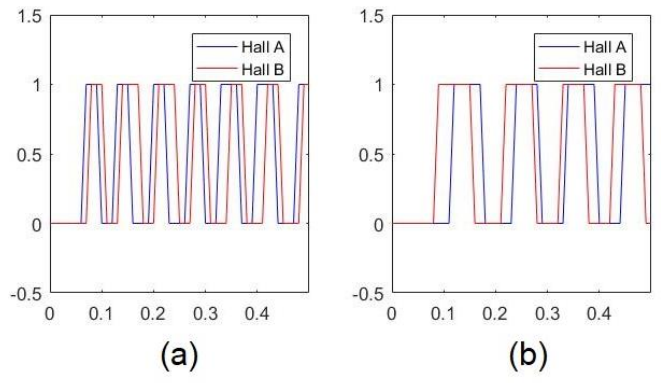
Samsung 22000mAh 3.7V, a fim de atender a demanda desses dois motores.

2.1.3. ENCODER MAGNÉTICO HALL 370

Esse sistema gera pulsos digitais de acordo com a rotação do eixo do motor, sendo que esses dados são utilizados para definir a velocidade angular ou deslocamento angular atual do motor. Cada motor está acoplado a um encoder magnético de efeito Hall. Esse sensor é composto por um disco magnético multipolar acoplado ao eixo principal do motor e dois sensores de efeito Hall (A e B). Possui faixa de operação entre 3.3V e 5V, e entrega resolução de onze pulsos por volta nominal do motor. Considerando a caixa de redução de 1:46 acoplada ao motor, o encoder capta quinhentos e seis pulsos digitais por volta na saída do motor.

Ao girar, o eixo no motor CC também gira o disco magnético multipolar acoplado a ele. Esse movimento faz com que os polos magnéticos do disco passam a frente de ambos os sensores Hall. A cada passagem de um polo positivo, o sensor Hall emite um pulso digital, também conhecido como "tick". Como o disco magnético possui vinte e dois polos alternados, cada sensor Hall emite onze pulsos a cada volta completa do eixo. Conhecido como encoder de quadratura, esses sensores Hall (A e B) estão defasados de 90º e geram saídas de onda quadrada fora de fase em 90º. O offset de onda é desejável para determinar o sentido de giro do eixo, uma vez que o motor gira em uma direção, a onda A precederá a onda B e na direção oposta, a onda B precederá a onda A. A Figura 07 ilustra a diferença entre as ondas A e B ao inverter o sentido de giro do motor.

Figura 07 – Resposta dos sensores Hall: (a) sentido horário; (b) sentido anti-horário



Fonte: Do autor (2022)

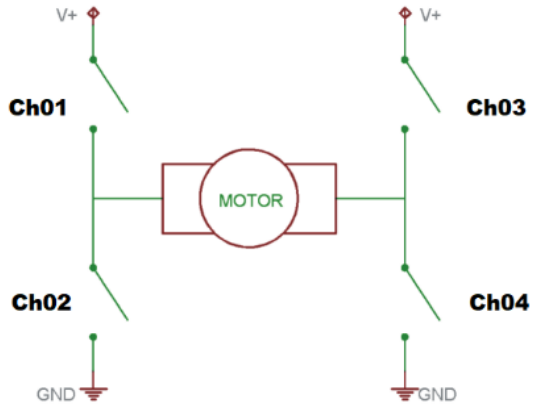
Para o microcontrolador Arduino esses eventos representados pelos pulsos digitais são tratados como interrupções externas, e consequentemente deve-se conhecer a distribuição dos pinos no microcontrolador para que esse consiga interpretar o sinal corretamente.

2.1.4. MÓDULO DRIVER PONTE H DUBLA BASEADO NO CHIP L298N

O módulo permite o controle de dois motores CC separadamente, em ambas as direções e em velocidades específicas ao variar o valor da tensão aplicada. Com tensão de operação de 4~35V, esse módulo tem como principal objetivo isolar eletricamente o microcontrolador dos motores CC, assim como permitir que o microcontrolador comande atuadores com tensão de trabalho superiores à sua própria.

A ponte H é um circuito empregado para inverter o sentido de direção da corrente em um motor CC. Em seu modelo simplificado, sua composição pode ser descrita pela presença de quatro chaves mecânicas entre o motor e a fonte de alimentação, mostrado através da Figura 08. A ponte H leva esse nome graças à semelhança do circuito com a letra H do alfabeto romano.

Figura 08 – Modelo simplificado do circuito de uma ponte H



Fonte: adaptado de Patsko (2006)

O fechamento das chaves Ch01 e Ch04 fará com que a corrente elétrica circule em um sentido no motor. Com a abertura dessas chaves e o fechamento das chaves Ch02 e Ch03, o sentido da corrente é invertida e o motor passa a girar em sentido contrário. Nesse tipo de sistema deve existir um sistema de segurança a fim de garantir que o chaveamento aconteça como esperado.

No caso do módulo, não existem chaves mecânicas. Esse chaveamento é realizado por transistores e diodo, contudo o sistema apresenta princípio de funcionamento análogo ao descrito acima. O driver também é capaz de controlar a velocidade de um motor utilizando um sinal PWM para regular o nível de tensão aplicada. O sinal PWM utiliza a relação do duty cycle do circuito para controlar o valor de alimentação, oscilando entre 0% e 100%.

Na interação entre o Arduino e o driver L298N, o sinal de PWM é tratado por um único byte. Ou seja, o sinal 255 enviado pelo Arduino é convertido para um sinal PWM de 100% no driver, assim como o sinal 0 é convertido pela placa para um sinal PWM de 0%.

2.1.5. CHASSI E RODAS

Confeccionado em acrílico, o chassi do robô móvel possui 21cm de comprimento e 15,1cm de largura. Ele constitui a estrutura do robô e possui todos os demais dispositivos acoplados a ele. Como o chassi original não foi preparado para suportar os dois motores utilizados, a adição de uma pequena barra em madeira alinhada em 180º com a disposição dos motores se fez necessária. Com 12,8cm de comprimento e 3,2cm de largura, essa barra em madeira foi acoplada juntamente ao chassi e ao par de motores, visando garantir sustentação para a estrutura e fixação para os motores.

Duas rodas em borracha foram acopladas aos motores do sistema. Com 65mm de diâmetro e 27mm de largura, o material dos pneus garante uma boa aderência do veículo ao solo. Uma terceira roda se faz necessária na extremidade traseira. Para tal, uma roda móvel em nylon, chamada de "roda boba", foi utilizada. Essa roda móvel não possui mecanismo de movimentação, servindo apenas para garantir a sustentação da estrutura.

2.2. SOFTWARES

2.2.1. ARDUINO IDE

De código aberto, essa IDE é um software fornecido pela própria empresa desenvolvedora das placas Arduino. Sendo um software com funcionamento muito simples, a escrita do código e o upload para a placa é algo facilitado. Escrito em função da linguagem de programação C/C++, o software é um ambiente de desenvolvimento integrado que fornece aos usuários inúmeros códigos simples para que possam aplicar em uma vasta gama de sensores e atuadores. Esse software é capaz de reconhecer todas as placas Arduino disponíveis no mercado e consequentemente realizar a comunicação entre microcontrolador e máquina sem que haja intervenção do usuário.

Além de uma boa quantidade de bibliotecas, o software também fornece suporte a correção de erros, adição de novas bibliotecas, monitor serial e documentação de sintaxe.

2.2.2. SOFTWARE PROCESSING

O Processing é um software de código aberto, funcionando como um complemento para a IDE do Arduino. Embora a IDE do Arduino seja mais difundida, ela surgiu de uma implementação dentro do software do Processing, e por isso ambos apresentam muitas particularidades em comum. Para o projeto, sua funcionalidade consiste em realizar comunicação serial e troca de informações com a interface do Arduino, receber os dados predefinidos e escrevê-los em um documento no formato de um banco de dados.

Desde sua criação em 2001, o software é muito utilizado para aprendizagem e prototipagem. O software possui escrita principalmente em C/C++, com um bom enfoque em bibliotecas voltadas a programação de interfaces e conteúdos visuais. Devido ao fato de sua plataforma ser de caráter gratuito e de código aberto, o Processing é uma alternativa a algumas ferramentas existentes no mercado.

2.2.3. MATLAB

O MatLab é um software interativo com alta performance voltada para computação numérica de análise e visualização de dados. Apresenta uma infinidade de aplicações para as mais diferentes áreas. Entre algumas de suas capacidades tem-se o cálculo de matrizes, processamento de sinais, análise numérica e projeção de gráficos. Possui uma interface de fácil acesso e manuseio com uma linguagem de programação que mescla entre Python, C/C++, Fortran, Java e outras (MATHWORKS, 1994).

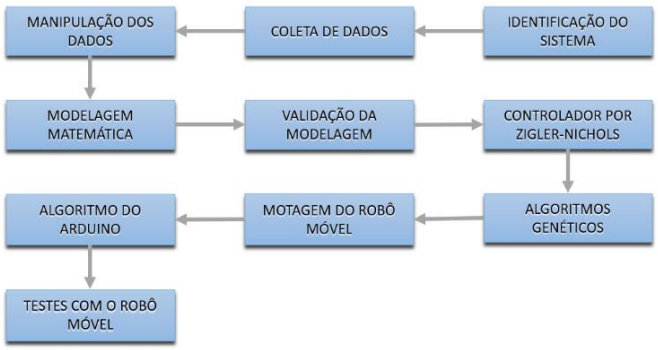
O software também conta com uma vasta quantidade de ferramentas, chamadas de "toolbox", que visam simplificar algumas aplicações, trabalhando com uma interface mais detalhada e específica de acordo com a ferramenta. O Simulink, ferramenta mais popular, é uma plataforma voltada para a simulação de sistema e prototipagem, ideal para esse tipo de aplicação. Possuindo uma interface simples e completa, o Simulink permite trabalhar com distintos tipos de sistemas.

Dentro da pesquisa, o MatLab foi utilizado para análise dos dados do motor, modelagem matemática do motor e desenvolvimento de um controlador. Também, o software aplicou algoritmos genéticos no controlador visando realizar um ajuste fino de seus ganhos.

2.3. FLUXOGRAMA

Essa seção detalhará toda a metodologia de análise utilizada, a fim de se chegar a uma solução para o problema. Para tal, as etapas do processo são descritas no fluxograma da Figura 09. Cada bloco do fluxograma é detalhado nas seções a seguir.

Figura 09 – Fluxograma de todas as etapas de desenvolvimento



Fonte: Do autor (2022)

2.3.1. IDENTIFICAÇÃO DO SISTEMA

Para desenvolver a plataforma móvel, o conjunto motor CC e encoder compunha o principal e mais importante sistema do robô. Já que, para que o robô móvel consiga se locomover conhecendo com demasiada precisão sua posição, o encoder se faz muito necessário. Para tal, todo o estudo de modelagem do sistema se deu através do comportamento do motor CC perante a aplicação de um sinal de tensão em seus terminais.

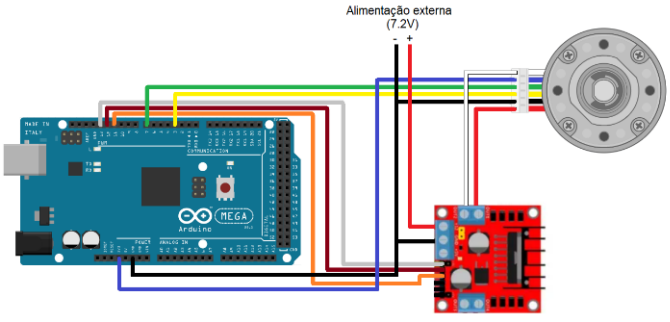
Como a utilização de motores de corrente contínua é algo comum, a literatura apresenta recorrentes trabalhos que aprofundam sobre as características e modelagem matemática desse. É possível encontrar testes que permitem uma aproximação dos parâmetros que descrevem o motor CC, a fim de se realizar uma modelagem caixa branca. Como fazer ensaios com eixo travado para cálculo da resistência de armadura Ra (DA SILVA, 2012), realizar o run down test para determinar o momento de inércia J (FONSECA, 2014) ou realizar uma regressão com dados de torque mecânico e velocidade angular do motor para definir o atrito viscoso do motor Bv (MORAIS, 2015). Contudo, no projeto optou-se em trabalhar com as técnicas para modelagem caixa preta. Para mais informações sobre os testes para definição dos parâmetros do motor, consultar os seguintes trabalhos: [Ortega e Capacho, 2009; Da Silva, 2012; Fonseca, 2014; Morais, 2015; Pinto, 2017]

Conhecendo a equação que rege a modelagem matemática do motor de corrente contínua, o trabalho baseou-se em mesclar os conceitos da modelagem caixa branca com a modelagem caixa preta. Em vista de que, a escolha das técnicas tradicionais de sistemas de primeira e segunda ordem só se mostrou possível após conhecer as características do motor.

2.3.2. COLETA DE DADOS DO MOTOR

Para a aquisição dos dados, o circuito mostrado na Figura 10 foi montado.

Figura 10 – Circuito esquemático do motor CC com encoder



Fonte: Do autor (2022)

O motor CC é ligado no módulo ponte H L298N através dos condutores de alimentação do motor. O módulo L298N é alimentado por duas baterias de 3,7V em série, totalizando 7,2V na entrada da ponte H. Como o módulo L298N apresenta uma queda de tensão, o motor é alimentado com aproximadamente 6V. Três condutores são ligados nas portas ENA, IN1 e IN2 do módulo L298N, considerando o motor no lado A da ponte H, e em três portas digitais. As portas IN1 e IN2 são usadas para acionar e comandar o sentido de giro do eixo do motor, a porta ENA é utilizada para comandar o sinal PWM no motor. Os fios de alimentação do encoder são ligados no Arduino, através das portas +3,3V e GROUND respectivamente. As portas digitais 07 e 03 do Arduino recebem as ondas quadradas do encoder, o microcontrolador interpreta os sinais e determina a posição atual do eixo do motor.

Para facilitar a compreensão e visando distinguir os motores, o motor CC do lado esquerdo, em relação a frente do robô móvel, foi chamado de motor B, enquanto o motor CC do lado direito foi chamado de motor A.

Com o circuito na Figura 10 montado, a IDE do Arduino escreve um código no microcontrolador. Esse código consiste em aplicar um sinal degrau de tensão no motor após dois segundos de inicialização do código. Em comunicação serial com a IDE do Arduino, o software Processing lê as três informações que são enviadas pelo microcontrolador e as escreve em banco de dados. Os dados lidos representam o tempo em ms, valor do degrau aplicado e posição angular atual do eixo do motor, respectivamente. Para esse algoritmo, considerou um período de amostragem de 10ms no microprocessador.

2.3.3. MANIPULAÇÃO DOS DADOS

Com os dados brutos obtidos não é possível realizar uma modelagem a partir das técnicas tradicionais. Os dados de posição, dados em pulsos, não podem ser utilizados, pois eles apresentam um integrador puro na função de transferência, ou seja, a amplitude tende a crescer ao infinito à medida que um degrau continua sendo aplicado a sua entrada. Conhecendo o intervalo de tempo, dado pelo tempo de amostragem da coleta, e o deslocamento angular, os dados são convertidos em pulsos/segundo através do conceito de velocidade média. A Equação 01 descreve a conversão dos dados em posição [pulsos] para velocidade [pulsos/segundos]:

$$v(tm) = \frac{\Delta p}{\Delta t} = \frac{p(tm) - p(tm-1)}{t(tm) - t(tm-1)} \qquad (01)$$

Na Equação 01, v representa a velocidade, tm é o tempo de amostragem, p é a posição e t expressa o tempo, dado em segundos.

2.3.4. MODELAGEM DO SISTEMA COM TÉCNICAS DE PRIMEIRA E SEGUNDA ORDEM

Após a manipulação dos dados, alguns métodos de modelagem para sistemas de primeira e segunda ordem foram utilizados. Para tal, o software MatLab se fez necessário afim de se conseguir resultados mais plausíveis. Foram utilizadas sete distintas técnicas para sistemas de primeira ordem e três técnicas para sistemas de segunda ordem, visando ter-se um

amplo campo de possibilidades e tentando encontrar a melhor resposta em malha aberta para ambos os sistemas (motor A e motor B). Como técnicas para sistemas de primeira ordem, tem-se: método de Nishikawa, método de um tau, método de quatro taus, método de Ziegler-Nichols, método de Smith, método de Haglund e o método de Sundaresan e Krishnaswami. Já para sistemas de segunda ordem, foram utilizados os métodos: de Sundaresan, de Mollenkamp e de Smith.

Para definir o melhor modelo para cada motor de corrente contínua, dentre os calculados, o melhor encontrado foi definido através das técnicas de erro quadrático mínimo. Os modelos foram avaliados através de três métricas:

Erro quadrático médio (MSE): entrega a média dos erros quadráticos. Apresenta maior dificuldade em analisar a resposta, uma vez que quanto maior o valor da resposta maior será a diferença entre os dois vetores. Deve-se atentar ao resultado, já que não se tendo um valor de referência pode se tornar complicado a compreensão da resposta.

Erro quadrático médio da raiz normalizada (NRMSE): essa técnica entrega um percentual de precisão entre um vetor de dados e um vetor de referência, através do cálculo de raiz normalizada.

Erro quadrático médio normalizado (NMSE): também entregando um percentual entre um vetor de dados e um vetor de referência, consiste em um complemento para a primeira técnica.

As técnicas foram empregadas já nos modelos com saída expressando a posição de ambos os motores. Para tal, cada modelo foi alterado, adicionado um integrador puro em cada um. Devido a posição ser descrita como a integral da velocidade em função do tempo, para um sistema no domínio contínuo da frequência, essa integral é expressa como um integrador puro.

2.3.5. VALIDAÇÃO DA MODELAGEM MATEMÁTICA

Para a validação da melhor modelagem matemática calculada, definida na seção anterior, o motor CC foi submetido a diferentes amplitudes de tensão. Essa variação de tensão é possível pois o módulo L298H envia um sinal PWM de tensão elétrica ao motor. A aquisição dos dados deu-se da mesma forma como descrito na seção 2.2.2 - Coleta de dados do motor.

Os dados também foram submetidos a uma manipulação, como descrito na seção 2.3.3 – Manipulação dos dados. Após, eles foram comparados, considerando as métricas utilizadas para definição do modelo e analisando o regime transiente da curva de resposta para cada situação, sobrepondo os dados coletados com a curva prevista pela modelagem. Então posteriormente é validado a modelagem.

2.3.6. DEFINIÇÃO DO CONTROLADOR PID POR SINTONIA DE ZIEGLER-NICHOLS

A definição do controlador PID através das técnicas de sintonia de Ziegler-Nichols faz-se necessária como um ponto de partida para os algoritmos genéticos, que serão citados futuramente.

Sabendo que a planta a ser controlada apresenta um integrador puro, devido as características da função de transferência de um motor CC, foi utilizado a segunda técnica das sintonias de Ziegler-Nichols. A primeira técnica de sintonia não foi possível pois curva de reação da posição do motor apresenta comportamento que não consegue ser avaliado pela técnica em malha aberta.

Então, o sistema foi submetido a um controlador considerando $T_i = \infty$ e $T_d = 0$. A partir disso, é definido a equação característica de malha fechada do sistema para definir Kcr e Pcr. O Kp é dado através do Kcr do sistema, que por sua vez é descrito como o valor do ganho para que o sistema apresente oscilações sustentadas. Sendo o Pcr o período dessas oscilações sustentadas. Para definir o ganho crítico é utilizado o Critério de Estabilidade de Routh; enquanto, o período das oscilações é definido pela substituição $s = j\omega$ na equação característica da função de transferência do motor.

Conhecendo Kcr e Pcr, o controlador PID é definido através da tabela de sintonia de Ziegler-Nichols para malha fechada. Ainda, para determinar os ganhos do controlador, é utilizado as seguintes relações: $Ki = Kp/Ti$ e $Kd = Kp * Td$.

2.3.7. AJUSTE DO CONTROLADOR PID UTILIZANDO AG'S

Para esse trabalho, o desenvolvimento e implementação dos algoritmos genéticos ocorre através da ferramenta MATLAB (versão R2016a) e as funções que disponibiliza para tal. A plataforma oferece inúmeras aplicações especializadas e voltadas para a configuração dos AGs, além de apresentar uma vasta documentação sobre o assunto, contendo exemplos e explicações para todo o funcionamento do algoritmo genético dentro da ferramenta. Motivado por isso, os códigos, em suma maioria, utilizados durante os procedimentos serão omitidos neste trabalho. Para mais informações consultar Genetic Algorithm Options disponível em: [Mathworks, 1994].

A implementação e configuração dos Algoritmos Genéticos visa minimizar a função objetivo, e para isso são definidos alguns parâmetros, sendo eles:

PopulationSize: define o tamanho da população;

PopIniRange: define os limites superiores e inferiores dos dados, nesse projeto os limites para os ganhos proporcional e derivativo, podendo ser expandido para Lower bound (lb) e Upper bound (up) para sistemas com mais parâmetros;

InitialPopulation: especifica a população inicial. Para o trabalho, foi utilizado os valores encontrados através da sintonia de Ziegler-Nichols;

Generations: especifica o número de máximo de gerações;

Elitecount: define o número de melhores indivíduos que passaram para a próxima geração. No âmbito deste trabalho, foi definido a regra de 5% de indivíduos elites;

StallGenLimit: define o critério de parada caso aconteça de várias gerações sem que haja melhoria na função objetivo;

CrossoverFcn: especifica a função de cruzamento. Para o trabalho, foi utilizado a função de um ponto;

SelectionFcn: especifica a função de seleção, sendo utilizada a função torneio para este trabalho.

Além dos parâmetros, também houve a necessidade de se definir os índices de desempenho para o AG. Um índice de desempenho é uma equação que visa otimizar o desempenho de um sistema de controle PID, ajustando os ganhos desse controlador. O algoritmo genético foi contemplado com 05 índices de desempenho (ID); sendo eles, Integral do Erro Absoluto (IAE), Integral do Erro Absoluto Ponderado pelo Tempo (ITAE), Integral do Erro Quadrático (ISE), Integral do Erro Quadrático Ponderado pelo Tempo (ITSE) e Média do Erro Quadrático (MSE).

A Integral do Erro Absoluto (IAE) é a integral das áreas acima e abaixo da referência, dada por:

$$I_{IAE} = \int_0^\infty |e(t)|dt \quad (02)$$

A Integral do Erro Absoluto Ponderado pelo Tempo (ITAE) é uma expansão do IAE, com acréscimo do vetor de tempo. Esse ID penaliza os erros que se mantêm ao longo do tempo, dado pela Equação 03:

$$I_{ITAE} = \int_0^\infty t|e(t)|dt \quad (03)$$

A Integral do Erro Quadrático (ISE) pondera os erros ao quadrado, dada pela Equação 04. Esse ID tende a entregar uma resposta rápida, com baixa estabilidade e várias oscilações.

$$I_{ISE} = \int_0^\infty |e(t)|^2 dt \quad (04)$$

Assim como ITAE, a Integral do Erro Quadrático Ponderado pelo Tempo (ITSE) é uma expansão do índice ISE que penaliza os erros que persistem no tempo. Dado por:

$$I_{ITSE} = \int_0^\infty t|e(t)|^2 dt \quad (05)$$

O critério Média do Erro Quadrático (MSE) avalia todos os desvios e variações em relação ao valor de referência. Apresentando boa resposta para função de avaliação pura ou híbrida, é dada pela Equação 06:

$$I_{MSE} = 1/N \sum_{i=1}^{N} e^2(i) \quad (06)$$

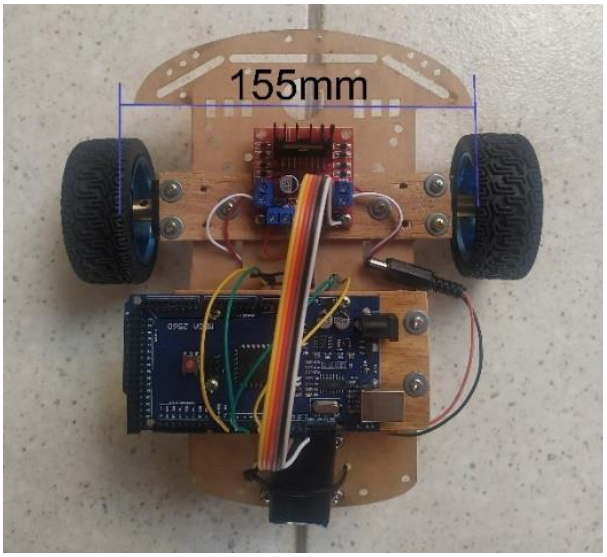
Antes de coletar resultados dos algoritmos para o modelo matemático, foi necessário definir dois parâmetros cruciais para o bom funcionamento do AG, tamanho da população e número de gerações. Segundo Paiva (2010), esses dois critérios influenciam diretamente no desempenho do algoritmo e devem ser bem avaliados. Para ambos, foi utilizado o critério em testar o AG para diferentes valores de população e geração, coletando o desempenho em cada passo. Para a população foi considerado o tamanho de 20 a 100 indivíduos, com o acréscimo de 20 a cada nova tentativa. Para gerações foi adotado o intervalo de 20 a 50 gerações, com o acréscimo de 10 a cada nova tentativa.

Com o pré-processo concluído, o AG foi aplicado considerando cada índice de desempenho e coletando a melhor solução. A escolha dos ganhos do controlador entre as 05 respostas deu-se de forma visual, ao se avaliar a melhor curva de saída e que melhor atenda ao problema.

2.3.8. MONTAGEM DA PLATAFORMA MÓVEL

Como a ideia da plataforma móvel é aplicar o controlador e validar todos os dados definidos, a plataforma não apresentou caráter de alta complexidade. Constituída de um chassi, dois motores com encoder, duas rodas, uma roda boba, um Arduino, um módulo de ponte H, duas pilhas de 3.7V, uma bateria de 9V e fios para ligação, todos os itens foram montados de forma a construir uma pequena plataforma móvel. Após a montagem, as rodas dos motores firam posicionadas de forma paralela com 155mm de distância entre elas, como mostrado na Figura 11:

Figura 11 – Distância entre as duas rodas dos motores



Fonte: Do autor (2022)

A parte elétrica do sistema, em suma, já estava montada, como é mostrado no circuito da Figura 10. Embora ela apresente apenas o circuito para um dos motores, o outro foi montado de forma a espelhar o circuito apresentado.

A parte frontal da plataforma ficou definida como sendo a parte com ambos os motores; e consequentemente a parte traseira definida como a parte da roda boba. A escolha dessa disposição é importante, uma vez que a roda boba não conseguiria afetar a posição da plataforma quando essa se locomove para frente. Por consequência, a plataforma foi desenvolvida de forma a não conseguir se locomover completamente para trás, já que a roda boba poderia desencadear uma mudança de direção.

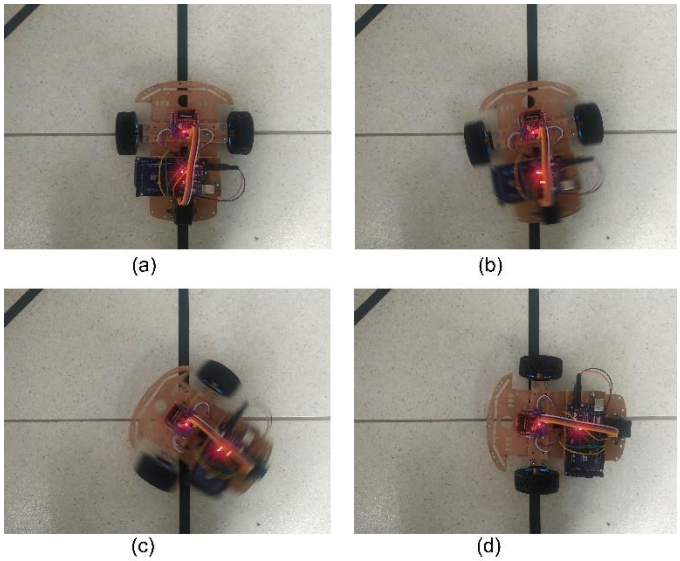
2.3.9. IMPLEMENTAÇÃO DO ALGORITMO DO ARDUINO

Antes de explicar os códigos aplicados, duas observações são necessárias: a primeira é a conversão dos pulsos lidos pelo encoder para uma distância em milímetros. Sabendo que o encoder captura onze pulsos por rotação nominal do eixo do motor, que o eixo está acoplado em uma redução de 1:46 e que a pneu do motor tem um diâmetro 65mm, a conversão acontece através da seguinte relação:

$$pulso = \frac{p_e * red}{\emptyset_r * \pi} * V_{desejado} = 2.4779 * V_{desejado} \quad (07)$$

Na relação, dada pela Equação 07, tem-se que p_e representa a quantidade de pulsos que o encoder contabiliza por volta, red é a redução para a saída, $\emptyset_r$ é o diâmetro do pneu e $V_{desejado}$ expressa o valor desejado em milímetros para conversão.

Figura 12 – Conversão da plataforma móvel à esquerda em 90°, em quatro fotografias: (a) ponto inicial; (b) início do movimento; (c) final do movimento; (d) ponto de parada.



Fonte: Do autor (2022)

A segunda observação diz a relação de pulsos para a rotação da plataforma móvel. Para o atual trabalho, qualquer tipo de mudança de direção ocorreu ao rotacionar a plataforma sobre o próprio eixo entre os motores, como mostrado na Figura 12.

Para o robô rotacionar a esquerda, em relação a sua frente, significa que o motor B, a esquerda da frente da plataforma, foi programado para andar para trás, enquanto o motor A, a direita, foi programado para andar para frente. Ambos programados para percorrer a mesma distância simultaneamente, o que desencadeia na rotação sobre o próprio eixo. Para o cálculo da quantidade de pulsos necessária para essa rotação, a relação da Equação 07 foi modificada para uma nova relação.

$$pulso = \frac{p_e * red * \emptyset_d}{\emptyset_r * 360°} * A_{desejado} = 3.3517 * A_{desejado} \quad (08)$$

A relação da Equação 08 é uma modificação da Equação 07, e por consequência necessita dos mesmos parâmetros da sua genitora, com a adição de $\emptyset_d$ que representa a distância entre os dois pneus, como mostrado na Figura 11, ou diâmetro da rotação. $A_{desejado}$ é o ângulo de rotação desejado em graus.

Para ambas as relações, 07 e 08, aconteceu arredondamento para número inteiro da resposta quando necessário.

Os códigos utilizados no Arduino podem ser separados como três algoritmos: o algoritmo de cálculo do controlador PD, com função de nome “pidMotor”; o algoritmo de comando do motor, chamado pela função “setMotor”; e o algoritmo principal.

A função “pidMotor” tem como objetivo receber os parâmetros do motor, realizar os cálculos e atualizar esses parâmetros. Com esses novos parâmetros calcular o sinal PWM que deve ser aplicado no motor, assim como a direção do motor. A função “pidMotor” esta ilustrada na Figura 13.

Figura 13 – Função “pidMotor”

```
sketch_Projeto_teste §

void pidMotor(long currT, int pos, int target, long *prevT, float *eprev, float *pwr, int *dir){

  float kp = 22.1807;
  float kd = 0.8597;

  float deltaT = ((float)(currT-*prevT))/1.0e6;
  *prevT = currT;
  int e = pos-target;
  float dedt = (e-*eprev)/(deltaT);

  float u = kp*e + kd*dedt;

  *pwr = fabs(u);
  if(*pwr>255) {      *pwr = 255;}

  *dir = 1;
  if(u<0) {           *dir = -1;}

  *eprev = e;
}
```

Fonte: Do autor (2022)

O algoritmo recebe alguns parâmetros por referência, pois esses serão atualizados no interior da função; destaque para os parâmetros “pwr” e “dir” que representam sinal PWM e direção, respectivamente, que serão aplicados no motor. Os parâmetros “currT”, “pos” e “target” representam tempo em microssegundos de início do código, posição atual do motor e posição desejada respectivamente. Esses três são utilizados para a atualização dos parâmetros por referência. Com os

cálculos realizados, o algoritmo implementa o controlador PD na variável de saída "u" e como consequência determina os novos valores para "pwr" e "dir". O erro é armazenado em uma variável. A função foi implementada de forma a atender ambos os motores na mesma estrutura.

Para comandar o motor, a função "setMotor" recebe os valores do sinal PWM e direção da função anterior, e por consequência aciona o respectivo motor. A Figura 14 mostra o código utilizado para a função.

Figura 14 – Função "setMotor"

sketch_Programa_Principal §

```
void setMotor(int dir, int pwmVal, int ppwm, int in1, int in2) {
  analogWrite(ppwm, pwmVal);

  if(dir == 1){
    digitalWrite(in2,LOW);
    digitalWrite(in1,HIGH);
  }
  else if(dir == -1) {
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
  }
  else {
    digitalWrite(in1,LOW);
    digitalWrite(in2,LOW);
  }
}
```

Fonte: Do autor (2022)

A função recebe como parâmetros os valores do sinal PWM e direção de giro nas variáveis "pwmVal" e "dir", respectivamente. Os demais parâmetros passados no escopo da função são para definir qual o motor a ser aplicado os valores, uma vez que a função foi construída de forma genérica. A função envia o valor do PWM para o motor e segundo o valor de "dir" alimenta o motor no sentido desejado.

O algoritmo principal (além de conter as duas funções anteriores) inicia o Arduino, conta os pulsos dos encoder's e exprime a lógica por trás dos motores, com condições para garantir que os motores concluam um caminho já predefinido.

Para a contagem dos pulsos em cada encoder foi utilizado as funções contidas na Figura 15. Ambas as funções apresentam mesmo objetivo, sendo aplicadas cada uma em seu respectivo motor. Elas realizam a contagem dos pulsos à medida que o programa reconhece uma interrupção através da chamada do método "attachInterrupt" contido no setup do Arduino. Enquanto as funções da Figura 15 recebem um terminal do encoder, o outro terminal é passado para a chamada do "attachInterrupt"; contudo, esse último deve ser ligado a uma porta do Arduino que reconheça interrupções. Para o trabalho, foram ligados nas portas 2 e 3 para os motores B e A, respectivamente.

Figura 15 – Funções para contagem dos pulsos em cada encoder

sketch_Programa_Principal §

```
void readEncoderA() {
  int a = digitalRead(Encoder_A2);
  if(a>0) {   posA++;}
  else      {   posA--;}
}

void readEncoderB() {
  int b = digitalRead(Encoder_B2);
  if(b>0) {   posB++;}
  else      {   posB--;}
}
```

Fonte: Do autor (2022)

Como o código foi estruturado de forma que a plataforma móvel percorra um circuito predeterminado, o algoritmo foi introduzido em passos, ou seja, cada ação do motor é numerada como um passo. A lógica para a troca desse passo deu-se através da função switch/case do Arduino, em que cada movimento representa um passo. Ao fim do passo, o algoritmo entra em uma condição de espera e realiza a troca de passo. A condição para a troca de passo só é alcançada quando ambos os motores estão muito próximos do target.

2.3.10. TESTES COM A PLATAFORMA MÓVEL

Após a definição do algoritmo, ele foi implementado de forma que a plataforma realizasse um circuito definido. Para tal, a estrutura switch/case foi alterada a fim de conter todos os passos necessários. O novo código foi enviado ao microcontrolador e adicionado a alimentação através da bateria, para que ele não dependesse de cabos para o funcionamento.

Os testes constituíram em analisar as posições de parada da plataforma à medida que ela realizava um novo passo. Além de constatar que a distância era o que foi dimensionado no algoritmo.

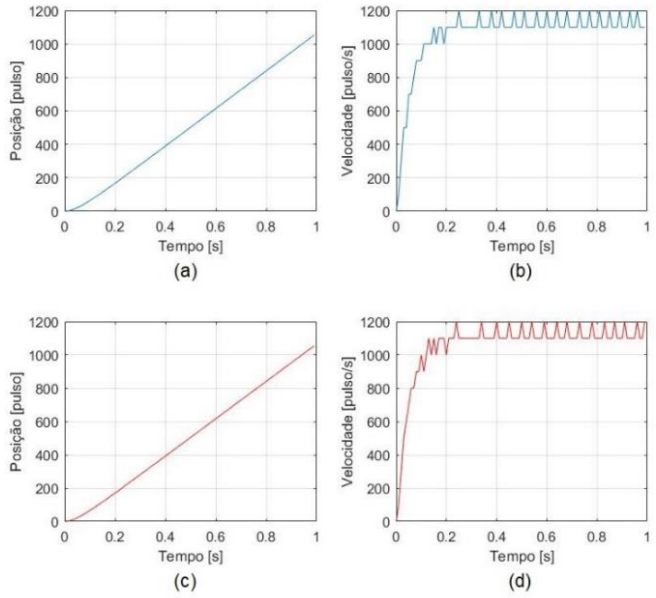
3. RESULTADOS E DISCUSSÕES

A aquisição dos dados deu-se de forma simples e conclusiva. Como o microcontrolador Arduino MEGA possui regime de trabalho assíncrono, o período de amostragem ficou definido em 10 ms e a saída exprimindo os pulsos lidos pelo par de encoder. A saída escolhida, dada em posição (pulso), demostra mais confiabilidade nos dados; uma vez que, caso o microcontrolador entregasse uma saída em velocidade (pulsos/segundo), o microcontrolador não conseguiria realizar a operação no período de amostragem definido.

O código coletou dados durante um intervalo de 90 segundos, a fim de verificar a estabilidade dos dados ao longo do tempo. Contudo, como não houve a necessidade da utilização de todos os dados, o intervalo de análise ficou definido entre os segundos 2 e 3 desses dados, o degrau foi aplicado exatamente no instante de 2 segundos da captura dos dados. A Figura 16

exibe os dados utilizados para a análise dos motores. Em (a) e (c) tem-se a posição, dada em pulsos, dos motores A e B, respectivamente; em (b) e (d) tem-se a velocidade, dada em pulsos/segundos, para os motores A e B, respectivamente.

Figura 16 – Resposta do motor CC a entrada ao degrau: (a) posição do motor A para um degrau de 5,97V; (b) velocidade do motor A para um degrau de 5,97V; (c) posição do motor B para um degrau de 5,96V; (d) velocidade do motor B para um degrau de 5,96V



Fonte: Do autor (2022)

Ainda na Figura 16, vê-se que ambos os motores se comportam de forma muito similar considerando o regime de operação escolhido, e, embora as unidades de medidas pareçam não ter significado físico, elas podem facilmente ser convertidas para posição angular e velocidade angular. Contudo optou-se em manter os dados brutos para a análise e modelagem matemática.

As técnicas clássicas para a modelagem matemática foram aplicadas sobre as curvas de velocidade de ambos os motores. Após, foi aplicado um integrador puro (1/s) sobre cada função de transferência visando modificar a saída dada em velocidade para uma resposta dada em posição. As Tabelas 01 e 02 exprimem o desempenho de cada modelagem considerando as métricas MSE, NRMSE e NMSE, para os motores A e B respectivamente. Foram consideradas as seguintes técnicas para sistemas de primeira ordem: o método de Nishikawa, o método de uma constante de tempo, o método de quatro constantes de tempo, o método de Ziegler-Nichols, o método de Smith, o método de Haglund, o método de Sundaresan e Krishnaswami. Já, para sistemas de segunda ordem foram avaliados os seguintes métodos: método de Sundaresan, método de Mollenkamp e método de Smith.

Tabela 01 – Desempenho dos modelos para o motor A

Métodos	*Ordem*	*MSE*	*NRMSE*	*NMSE*
Método de Nishikawa	1^a	27.9968	98.34%	99.02%
Método de 1 τ	1^a	5.9305	99.24%	99.35%
Método de 4 τ (98%)	1^a	5.9305	99.24%	99.35%
Método de Ziegler-Nichols	1^a	5.9305	99.24%	99.35%
Método de Smith	1^a	8.1684	99.10%	99.26%
Método de Haglund	1^a	5.9305	99.24%	99.35%
Método de Sundaresan e Krishnaswami	1^a	24.4583	98.45%	99.08%
Método de Sundaresan	2^a	5.7562	99.25%	99.41%
Método de Mollenkamp	2^a	5.0795	99.29%	99.49%
Método de Smith	2^a	57.9178	97.61%	98.86%

Fonte: Do autor (2022)

Tabela 02 – Desempenho dos modelos para o motor B

Métodos	*Ordem*	*MSE*	*NRMSE*	*NMSE*
Método de Nishikawa	1^a	7.0890	99.16%	99.30%
Método de 1 τ	1^a	94.0656	96.96%	98.35%
Método de 4 τ (98%)	1^a	9.1268	99.07%	99.20%
Método de Ziegler-Nichols	1^a	9.1268	99.07%	99.20%
Método de Smith	1^a	9.1268	99.07%	99.20%
Método de Haglund	1^a	9.1268	99.07%	99.20%
Método de Sundaresan e Krishnaswami	1^a	50.7171	97.76%	98.91%
Método de Sundaresan	2^a	46.4465	97.86%	98.95%
Método de Mollenkamp	2^a	8.3928	99.09%	99.25%
Método de Smith	2^a	95.7471	96.93%	98.34%

Fonte: Do autor (2022)

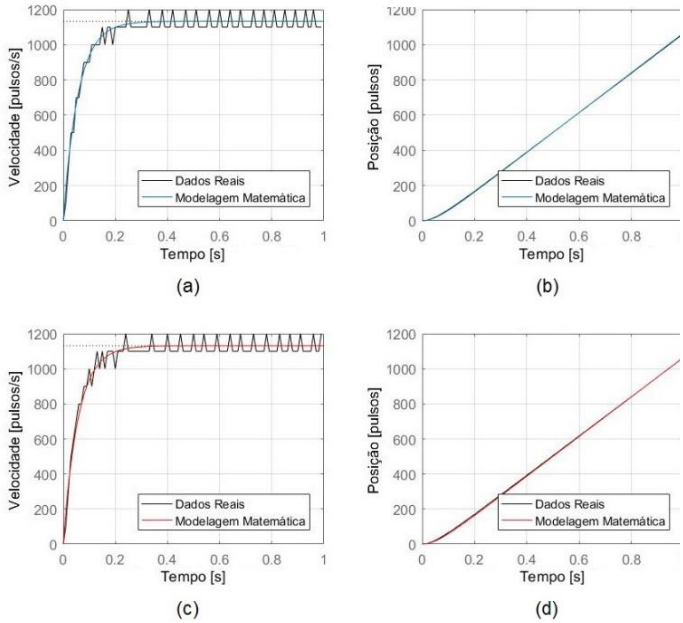
A escolha da melhor modelagem deu-se através das técnicas de erro médio quadrático, sendo que, para o motor A, a melhor modelagem foi deduzida através da técnica de Mollenkamp; e para o motor B, a modelagem com melhor desempenho foi feita pelo método de Nishikawa. Contudo, em B, devido ao modelo de Mollenkamp apresentar resultado satisfatório por ser de ordem 02, contempla melhor as características do motor CC, optou-se em utilizá-lo. A modelagem do motor A apresentou desempenho de 99.29% e 99.49% para NRMSE e NMSE, respectivamente; com MSE de 5.0795. Já o motor B apresentou desempenho de 99.09% e 99,25% para NRMSE e NMSE, respectivamente; com erro por MSE de 8.3929. As Equações 09 e 10 descrevem a relação posição, dada em pulsos, pela tensão aplicada nos motores A e B, respectivamente.

$$G_A(s) = \frac{\theta(s)}{Va(s)} = \frac{189.8}{4.705*10^{-6}*s^3+0.05715*s^2+s} \quad (09)$$

$$G_B(s) = \frac{\theta(s)}{Va(s)} = \frac{188.5}{4.705*10^{-6}*s^3+0.05715*s^2+s} \quad (10)$$

Ao analisar as Equações 09 e 10 percebe-se que as modelagens matemáticas definidas para cada um dos motores CC apresentam comportamento e caráter muito similar. Consequentemente, o seguimento do trabalho deu-se determinando apenas uma modelagem para o par de motores, dada pela modelagem encontrada para o motor A e definida pela Equação 09. Considerando a modelagem da Equação 09 aplicada aos dados coletados do motor B, tem-se 16.6985, 98.72% e 99.13% para MSE, NRMSE e NMSE, respectivamente.

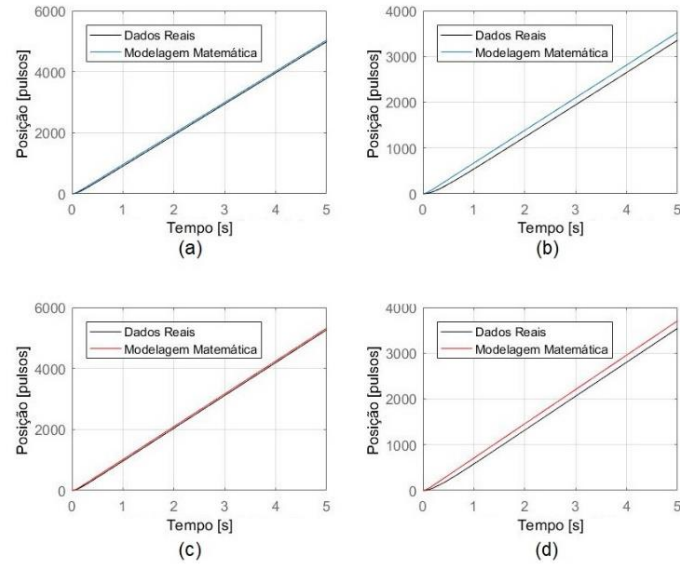
Figura 17 – Resultado da modelagem matemática: (a) velocidade do motor A para um degrau de 5,97V; (b) posição do motor A para um degrau de 5,97V; (c) velocidade do motor B para um degrau de 5,96V; (d) posição do motor B para um degrau de 5,96V



Fonte: Do autor (2022)

A Figura 17 compara os dados coletados com a resposta da modelagem em função da velocidade e da posição para ambos os motores. Em 24-a e 24-c tem-se a resposta dos motores A e B, respectivamente, em função da velocidade; enquanto, em 24-b e 24-d mostra a resposta em função da posição para ambos os motores.

Figura 18 – Validação da modelagem matemática: (a) posição do motor A para um degrau de 5,37V; (b) posição do motor A para um degrau de 3,76V; (c) posição do motor B para um degrau de 5,67V; (d) posição do motor B para um degrau de 3,95V



Fonte: Do autor (2022)

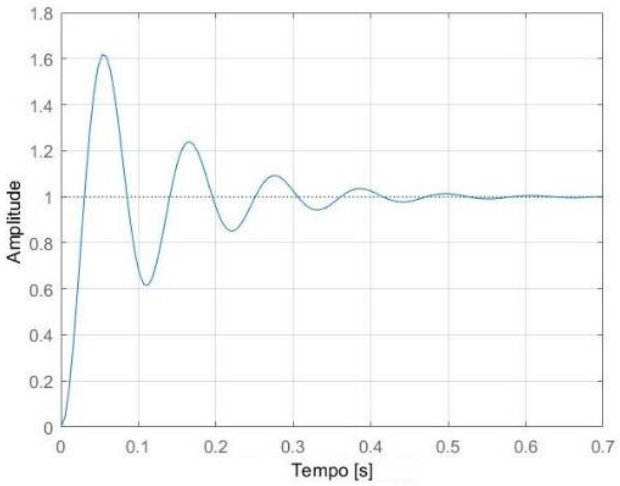
Para a validação da modelagem, os motores foram sujeitos a duas intensidades de tensão distintas, em cada. A Figura 18 mostra a comparação entre a saída do modelo calculada e os dados coletados. Ao analisar as curvas de reação, percebe-se que para tensões abaixo do nominal, os motores apresentam maior dificuldade na partida, demostrando um regime transiente mais lento. Também, nota-se que a curva do motor apresenta pequenas oscilações durante o regime transiente, decorrentes de perdas no motor e perdas por atrito.

As oscilações no regime transiente não são previstas pela modelagem matemática devido ao caráter simplista dessa (baixa ordem do sistema), que desconsidera algumas características do motor afim de reduzir a complexidade do sistema. Sobretudo, a modelagem se mostra satisfatória decorrente do resultado entregue.

Embora exista erro no regime transiente na curva de reação do motor CC, o erro não oscila quando a curva entra em regime permanente, se mantendo constante. Como o modelo utilizado consegue acompanhar a curva de reação do motor em regime permanente, o erro não cresce quando a curva de reação estabiliza.

Com o modelo matemático definido, faz-se necessário uma breve análise da resposta do sistema ao degrau em malha fechada. A Figura 19 exibe a resposta do sistema em malha fechada para um degrau unitário. Percebe-se que o sistema apresenta muitas oscilações no regime transitório, com um alto sobressinal. Destaca-se também que o tempo de assentamento é elevado, o que faz a necessidade de um ajuste fino através de um controlador.

Figura 19 – Resposta do processo em malha fechada para uma entrada do tipo degrau unitário



Fonte: Do autor (2022)

Ao se analisar os três polos encontrados no sistema em malha fechada, sendo eles $[-12129;\ -8{,}62\ \pm 57i]$, optou-se em trabalhar apenas com as ações proporcional e derivativa. Como a planta apresenta um integrador puro em malha aberta, não é interessante adicionar uma ação integral no controlador, pois

ela aumenta a ordem e tende a levar o sistema a instabilidade, graças a disposição dos polos da planta. Embora a ação integral reduza o tempo de assentamento, ela também eleva o sobressinal da curva de reação. O controlador PID é possível, contudo, o controlador PD já é suficiente para um bom controle da planta. A ação derivativa é uma alternativa viável pois ela pode ser implementada a fim de afetar ou cancelar alguns dos polos da planta, e por consequência manipular o comportamento da curva a disposição do usuário.

Com a modelagem definida, o controle se fez necessário para extrair um bom desempenho do motor de corrente contínua. Como o foco é o controle da posição do motor e o processo apresenta um integrador, de acordo com a Equação 09, foi adotado o segundo método das regras de sintonia do controlador PID de Ziegler-Nichols para malha fechada. Para tal, ajustando $T_i = \infty$ e $T_d = 0$, obtém-se a seguinte função de transferência para o processo em malha fechada:

$$\frac{\theta(s)}{Va(s)} = \frac{189.8*Kp}{4.705*10^{-6}*s^3+0.05715*s^2+s+Kp} \qquad (11)$$

Para o sistema em malha fechado da Equação 11, o ganho crítico encontrado foi de $Kcr = 63.9971$ com período de oscilação $Pcr = 0.0136$. Com base nesses valores, a Tabela 03 apresenta os valores de Kp, Ti e Td.

Tabela 03 – Parâmetros do controlador PID pelo método de Ziegler-Nichols

Controlador	*Kp*	*Ti*	*Td*
PID	38.3983	0.0068	0.0017

Fonte: Do autor (2022)

Usando algumas relações para definir os ganhos do controlador, a Tabela 04 exibe os valores dos ganhos Kp, Ki e Kd calculados para o controlador da planta.

Tabela 04 – Ganhos do controlador PID pelo método de Ziegler-Nichols

Controlador	*Kp*	*Ki*	*Kd*
PID	38.3983	5634.8	0.0654

Fonte: Do autor (2022)

Analisando os dados encontrados na Tabela 04, verifica-se que o ganho integral acaba por prejudicar o controle da planta, percebido pelo valor distante do ganho da ação. O polo da ação integral tende a anular o zero pela ação derivativa, além de afastar o par de polos conjugados. Isso leva a curva de reação a ter um menor tempo de assentamento, apresentando um alto sobressinal com diversas oscilações. Como o método de sintonia por Ziegler-Nichols não está adaptado para um controlador PD, a ação integral foi apenas desprezada, deixando os ganhos como apresentado na Tabela 05.

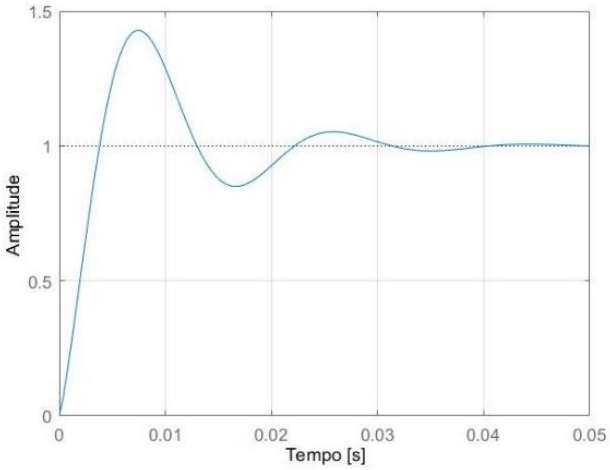
Tabela 05 – Ganhos do controlador PD pelo método de Ziegler-Nichols

Controlador	*Kp*	*Kd*
PD	38.3983	0.0654

Fonte: Do autor (2022)

Considerando o controlador definido pela Tabela 05, a resposta do sistema a uma entrada do tipo degrau unitário encontra-se ilustrada pela Figura 20:

Figura 20 – Resposta do sistema com o controlador PID para uma entrada em degrau unitário



Fonte: Do autor (2022)

Ao analisar o gráfico da Figura 20, tem-se que a planta com o controlador definido pelas técnicas de Ziegler-Nichols embora apresente um tempo de assentamento satisfatório, de $Ts = 0.029s$, a resposta apresenta muitas oscilações com um alto valor para o overshoot, $MP\% = 42{,}8\%$. Indicando que o método de Ziegler-Nichols não entrega uma resposta muito satisfatória ao modelo apresentado, necessitando que ocorra um ajuste fino. Ainda, ao ser aplicado à planta, esse controlador estabiliza a planta após várias oscilações, o que não é interessante. Contudo, os valores encontrados são utilizados como um ponto de partida para os AGs, a fim de se obter uma melhor sintonia do controlador.

Para definir os valores de tamanho da população e número de gerações foram feitos inúmeros testes, a fim de se ter um valor conclusivo para ambos. Contudo, o algoritmo apresentou rápida conversão para todos os valores e, em maioria, precisando de pouco menos que 20 interações, com população entre 20 e 40 indivíduos. Devido ao baixo impacto e o foco não estar na análise de performance do algoritmo, no recorrente trabalho esses dados não serão apresentados. Para análise de performance do algoritmo, assim como maior apresentação desses, consultar a referência: [Paiva, 2010]. Sendo, então, definido 60 indivíduos para a população e 40 interações. A escolha desses valores deu-se como medida de segurança, uma vez que a convergência da população ocorre muito próximo de 20 interações.

A Tabela 06 apresenta os parâmetros utilizados para a inicialização da população do algoritmo genético:

Tabela 06 – Valores de inicialização do algoritmo genético

Parâmetro	*Valor*
PopulationSize	60.0
Lower bound	[0.0 0.0 0.0]
Upper bound	[50.0 0.0 10.0]
InitialPopulation	[38.3983 0.0 0.0654]
Generations	40.0
Elitecount	3.0
ScallGenLimit	20.0
CrossoverFcn	*@crossoversinglepoint*
SelectionFcn	*@selectiontournament*

Fonte: Do autor (2022)

Ao analisar a Tabela 06, tem-se que o intervalo do parâmetro PopIniRange, Lower bound e Upper bound, para a ação integral é nulo. Embora o algoritmo utilize as três ações, a ação integral foi desconsiderada, visto que, experimentalmente, para o controle da planta ela não se faz necessária. Já os intervalos para as ações proporcional e derivativa foram definidos como margens para os valores encontrados pela técnica de sintonia de Ziegler-Nichols. O parâmetro ScallGenLimit foi definido como 20 gerações, 1/2 do total de gerações, e em todas as análises ele foi o critério de parada do algoritmo, uma vez que a convergência do algoritmo foi rápida e eficiente.

A Tabela 07 registra os valores obtidos para cada parâmetro, em função de cada índice de desempenho atribuído.

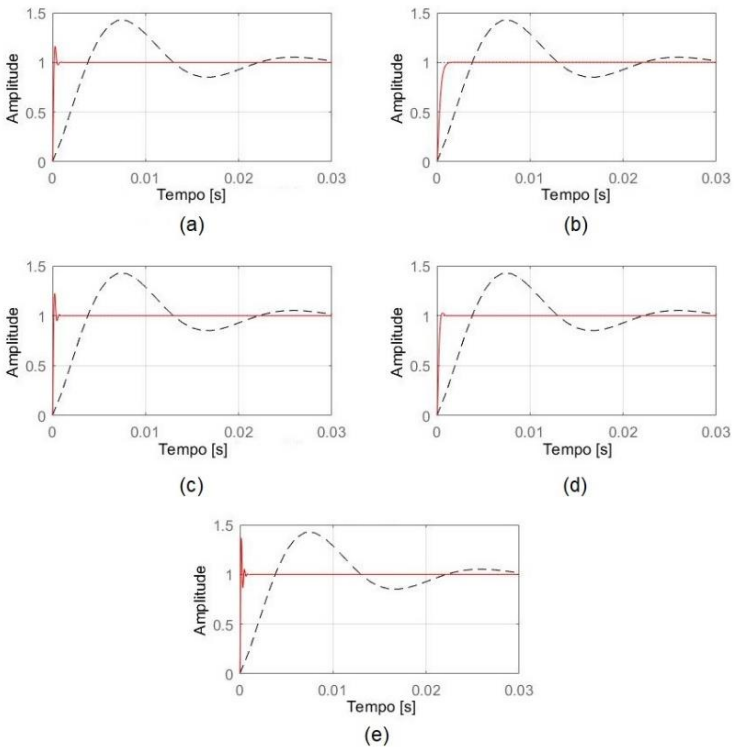
Tabela 07 – Especificações e parâmetros das respostas do sistema

		Ganhos do Controlador		Desempenho do Sistema		
		Kp	*Kd*	*MP%*	*Ts*	*Tr*
Ziegler-Nichols		38.3983	0.0654	42.8%	0.0296	0.0029
AGs	IAE	49.9973	3.6377	16.2%	0.0007	0.0001
	ITAE	22.1807	0.8597	0.2%	0.0010	0.0006
	ISE	3.8486	4.9399	22.2%	0.0007	0.0001
	ITSE	25.9907	1.5996	2.7%	0.0007	0.0003
	MSE	8.0883	9.9780	36.8%	0.0006	0.0001

Fonte: Do autor (2022)

Na Tabela 07, tem-se que MP% é o sobressinal da resposta em percentual, Ts é o tempo de assentamento ou estabilização, em segundos; e Tr expressa o tempo de subida, também em segundos. Ainda na Tabela 07, percebe-se que cada índice de desempenho apresentou uma resposta diferenciada, contudo com desempenho do sistema melhor e mais rápido. Todos os cinco modelos apresentaram um tempo de resposta muito menor, com os valores de sobressinal inferiores ao calculado pela técnica de Ziegler-Nichols, exceto o MSE que apresentou um sobressinal próximo ao encontrado pela técnica calculada. Sendo assim, para o processo em estudo, os índices de desempenho aplicados para a sintonia do controlador PD apresentaram resultados melhores que o encontrado através da técnica de sintonia do controlador de Ziegler-Nichols. A Figura 21 expressa a resposta do sistema a uma entrada do tipo degrau unitário com controlador definido pela técnica de Ziegler Nichols versus a resposta do sistema ao degrau para cada índice de desempenho.

Figura 21 – Desempenho do controlador PD por Ziegler-Nichols (linha pontilhada) versus Desempenho do controlador PD para cada índice de desempenho com AG (linha contínua): (a) IAE; (b) ITAE; (c) ISE; (d) ITSE; (e) MSE.



Fonte: Do autor (2022)

Para aplicar na planta, os ganhos do controlador PD selecionados foram aqueles encontrados através do índice de desempenho ITAE. Embora esse método apresente uma resposta mais lenta entre as demais encontradas via algoritmos genéticos, ela possui menor sobressinal, além de não apresentar oscilações visíveis na resposta. Para esse sistema em estudo, o índice de desempenho ITAE projetou o melhor controlador entre os cinco obtidos nos algoritmos genéticos.

Analisando o controlador escolhido, percebe-se que ele movimentou o par de polos conjugados da planta em malha fechada. O controlador PD adicionou um zero de forma a praticamente anular um desses polos do sistema, o que impactou na curva de resposta. Essas características levou a curva de reação a ter baixo sobressinal (MP% = 0,2) com um baixo tempo de assentamento (Ts = 1 ms), se mostrando ideal para o controle da planta.

Definido o controlador, montada a plataforma móvel e implementado os algoritmos, a trajetória escolhida para avaliar o desempenho do controle no sistema é descrito na Tabela 08, contendo cada passo que o algoritmo deve realizar.

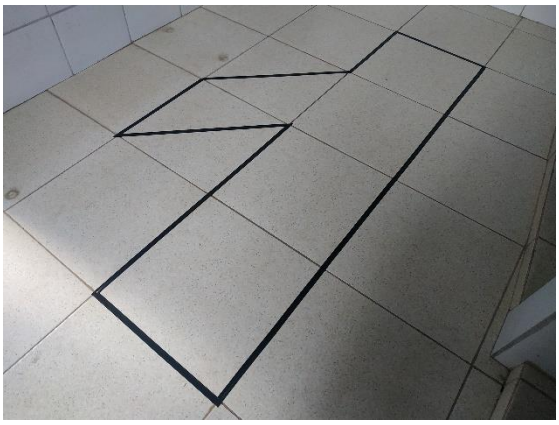
Tabela 08 – Passos para a plataforma móvel realizar o circuito

Número do Passo	Ação	Target do Motor A	Target do Motor B
Passo 01	Andar 2000mm.	4956	4956
Passo 02	Girar a esquerda em 90°	302	−302
Passo 03	Andar 500mm.	1239	1239
Passo 04	Girar a esquerda em 90°	302	−302
Passo 05	Andar 500mm.	1239	1239
Passo 06	Girar a direita em 45°.	−151	151
Passo 07	Andar 707mm.	1752	1752
Passo 08	Girar a esquerda em 45°.	151	−151
Passo 09	Andar 500mm.	1239	1239
Passo 10	Girar a esquerda em 135°.	453	−453
Passo 11	Andar 707mm.	1752	1752
Passo 12	Girar a direita em 135°.	−453	453
Passo 13	Andar 1000mm.	2478	2478
Passo 14	Girar a esquerda em 90°.	302	−302
Passo 15	Andar 500mm.	1239	1239
Passo 16	Girar a esquerda em 90°	302	−302

Fonte: Do autor (2022)

Para facilitar e compreender a posição do robô móvel, o ambiente foi demarcado com uma linha indicando o percurso ao qual a plataforma iria se locomover. A Figura 22 mostra o circuito determinado.

Figura 22 – Ambiente demarcado com o circuito ao qual o robô móvel iria percorrer



Fonte: Do autor (2022)

A plataforma realizou o circuito algumas vezes e analisou-se que no geral ela consegue concluir o trajeto bem, com alguns erros observados.

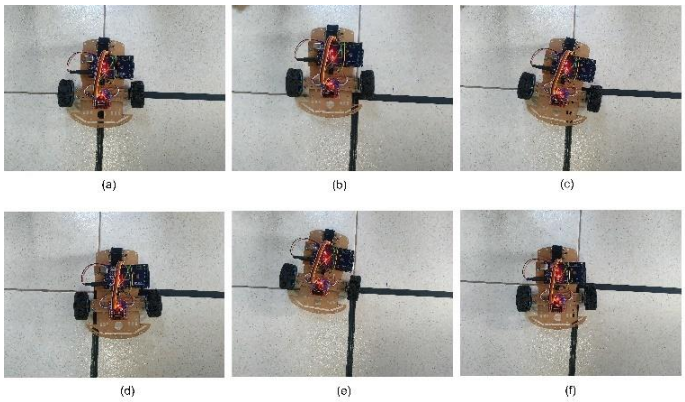
Considerando movimentos isolados, para deslocamentos em linha reta a plataforma apresenta erro mínimo, mesmo que com cada motor sendo tratado separadamente. Para distâncias pequenas o erro não é perceptível a olho nu, sendo observado apenas através da leitura do encoder. Por exemplo, para realizar o movimento de 500 pulsos, ambos os motores param com valores entre 498 e 502 pulsos. Cada pulso significa o deslocamento de aproximadamente 2,5 mm na roda da plataforma. Para distâncias altas o erro entre os dois motores se torna mais perceptível, uma vez que há um erro desencadeado pelo chip L298N da ponte H, que não consegue operar tão bem com ambos os motores simultâneos. Para movimentos longos o módulo oferece tensão diferente para cada motor, mesmo o sinal de controle sendo idêntico. Esse problema pode levar a plataforma a desviar para o lado do motor que recebe menor tensão elétrica.

Para o movimento de rotação isolado, a plataforma consegue realizar o deslocamento muito bem para altos ângulos. Como no movimento curto em linha reta, a rotação apresenta erro mínimo não visível no deslocamento da plataforma móvel. Esse erro é de aproximadamente 2 pulsos no deslocamento total

Outro problema encontrado pela plataforma foi o ambiente utilizado para testes. Esse ambiente não oferece boa aderência aos pneus da plataforma, o que aumenta o erro em cada passo decorrente de deslizamento. Embora esteja diretamente associado ao ponto de parada do passo, percebe-se deslizamentos dos pneus em outras situações.

A soma dos erros ligados a cada passo impacta diretamente na posição de parada da plataforma ao final do percurso, uma vez que o erro é claramente perceptível. A plataforma conseguiu chegar ao fim do percurso se posicionando próximo ao ponto esperado, como é mostrado na Figura 33.

Figura 23 – Parada da plataforma ao final do percurso. (a) ponto desejado para parada; (b) tentativa 01; (c) tentativa 02; (d) tentativa 03; (e) tentativa 04; (f) tentativa 05.



Fonte: Do autor (2022)

A Figura 23 mostra que para o trajeto definido, a plataforma apresenta boa precisão mesmo com os erros associados. Os erros levaram a plataforma a finalizar o trajeto com ângulo defasado e fora do ponto desejado, mostrado em 33-a. Contudo, em todas as tentativas, a plataforma finalizou dentro da linha demarcada sobre o ambiente para sinalizar o percurso. Esse tipo de precisão não seria possível se a planta operasse em malha aberta.

4. CONCLUSÃO

O presente trabalho teve o objetivo de construir uma plataforma móvel com o par de motores acionados por meio de um controlador PID. Com uma modelagem matemática calculada e um controle ajustado através de Algoritmos Genéticos, para assim, a plataforma ser capaz de se movimentar de forma autônoma sobre um cenário

predeterminado e garantindo que essa consiga alcançar o seu local de destino corretamente.

Considerando a modelagem, os modelos em caixa cinza entregam resultados satisfatórios, quando os motores elétricos de corrente contínua operam com valores de tensão elétrica próximas ao nominal. Sendo capaz de calcular e definir bem os parâmetros do motor para essa região de trabalho. Para baixas tensões de trabalho, os motores apresentam alguns comportamentos no regime transiente que a modelagem caixa cinza não consegue contemplar em sua totalidade, contudo elas ainda entregam um desempenho muito positivo.

Embora o ajuste dos controladores através de Algoritmos Genéticos tenha atingido um desempenho muito satisfatório, há uma necessidade em estudar melhor os parâmetros para eles a fim de concretizar o resultado. Os AGs apresentam uma relativa complexidade na definição de seus parâmetros, uma vez que uma má análise deles pode tornar inviável a utilização dos algoritmos.

A plataforma móvel entregou uma boa resposta com o controlador projetado, mostrando um desempenho muito satisfatório na sua locomoção. Sendo capaz de se locomover para frente e realizar rotações muito bem, considerando que cada motor é tratado separadamente. Contudo, na realização de trajetos, a somatória de erro associado a cada etapa e problemas como derrapagem leva a plataforma a um erro na posição final.

Este projeto é uma abordagem para aplicações de plataformas autônomas com controle em malha fechada, permitindo que ocorram avanços nos estudos a respeito deste assunto, podendo acontecer melhorias no trabalho, assim como possibilitando aplicações desse tipo de plataforma em ambientes fabris e/ou agrícolas.

4.1. PERSPECTIVAS FUTURAS

Como sugestões para trabalhos futuros, citam-se:

- Realização de uma modelagem matemática através da técnica caixa branca, assim como realizar uma modelagem de toda a estrutura motor, eixo e roda;
- Aprofundamento na definição de parâmetros para Algoritmos Genéticos e um estudo de desempenho dos parâmetros para ele;
- Desenvolvimento de uma plataforma móvel mais robusta, uma vez que por se tratar de um kit didático, algumas limitações, tanto em hardware quanto em software, foram encontrados;
- Possibilitar a comunicação sem fio com outros dispositivos e/ou veículos também inseridos em seu ambiente de trabalho; permitindo a definição de novas trajetórias facilmente, assim como conhecimento do ambiente em torno dele;
- Implementação de outros sensores na plataforma móvel, como acelerômetro, ultrassónico, outros. Para assim permitir uma maior compreensão e conhecimento do robô quanto ao ambiente em que está inserido.

6. REFERÊNCIAS BIBLIOGRÁFICAS

AGUIRRE, Luis Antonio. **Introdução à Identificação de Sistemas- Técnicas lineares e não-lineares aplicadas a sistemas reais**. 3. ed. Belo Horizonte, MG: Editora da UFMG, 2007.

ARDUINO. Arduino Mega 2560 Rev3. **Data Sheet**. Arduino SRL, p. 1-18, 2020. Disponível em: https:\\docs.arduino.cc/static/fb5ca0dab7208a15b958650cdeeb9194/A000067-datasheet.pdf. Acesso em: 15 dez. 2021

CANAL, Ivan Paulo; VALDIERO, Antonio Carlos; REIMBOLD, Manuel Pérez. Modelagem Matemática de Motor de Corrente Contínua e Análise Dinâmica. Congresso Nacional de Matemática Aplicada e Computacional (CNMAC), Gramado, RS, 2016. In: **Proceeding Series of the Brazilian Society of Applied and Computational Mathematics**, v. 5, n. 1, 2017.

COELHO, Antonio Augusto Rodrigues; DOS SANTOS COELHO, Leandro. **Identificação de Sistemas Lineares**. 1. ed. Florianópolis, SC: Editora da UFSC, 2004.

DA SILVA, Carolina Jaqueline Nascimento. **Caracterização de um Conjunto Didático para Ensaio de Motor de Corrente Contínua**. 2012.

FONSECA, Henrique Moura. **Modelagem e Controle da Posição Angular de um Motor CC Acoplado a uma Haste Flexível**. 2014.

GILBERT, William. **De magnete**. Translated by P. Fleury Mottelay. New York, Dover Publications Inc., 1958.

HOLLAND, John H. **Adaptation in Natural and Artificial Systems: An Introduction Analysis with Applications to Biology, Control, and Artificial Intelligence**. London, England: MIT Press, 1992.

LATHI, Bhagwandas Pannalal. **Sinais e Sistemas Lineares**. Tradução de Gustavo Guimarães Parma. 2. ed. Porto Alegre, RS: Bookman, 2007.

LUCAS, Diogo C. Algoritmos Genéticos: uma Introdução. **Apostila referente a disciplina de Inteligência Computacional**, Universidade Federal do Rio Grande do Sul, Brasil, 2002.

MATHWORKS. **Mathworks**, c1994. Produtos, MatLab. Disponível em: https://www.mathworks.com/products/matlab.html. Acessado em: 19 dez. 2021.

MATHWORKS. **Mathworks**, c1994. Genetic Algorithm Options. Disponível em: https://www.mathworks.com/help/gads/genetic-algorithm-options.html. Acessado em: 15 fev. 2022.

MITCHELL, Melanie. **An Introduction to Genetic Algorithms**. 1st ed. London, England: MIT press, 1998.

MORAIS, Filipe Alexandre. **Estudo em Vazio sobre Motores CC com Imã Permanente com Aplicações na Industria Automotiva**, 2015. Universidade Federal do Paraná.

OGATA, Katsuhiko. **Engenharia de Controle Moderno**. Tradução de Heloísa Coimbra de Souza. 5th ed. São Paulo, SP: Person Education, 2011.

ORTEGA, Manuel Guillermo Quijano; CAPACHO, Carlos Gerardo Hernández. **Obtención experimental de los parámetros del motor que se utilizará em el sistema de locomoción de uma esfera rodante**, 2009.

PAIVA, Leonardo Silveira. **Aplicação de Algoritmos Genéticos para Sintonia de Controladores**. 2010. Tese de Doutorado. Instituto Politécnico do Porto. Instituto Superior de Engenharia do Porto.

PATSKO, Luís Fernando. Tutorial Montagem da ponte H. **Maxwell Bohr-Instrumentação Enetrônica**, 2006. fotografia color.

PINTO, Mauricio da Silva. **Modelagem matemática da dinâmica de um sistema dosador de adubo à taxa variável acionando por um motor elétrico de corrente contínua**. 2017.

RIBEIRO, Marco Antônio. Controle e Automação. 1. ed. **Tek Treinamento e Consultoria**: Salvador, BA, 2005.

SOARES, Márcio Morelli. **Análise do Uso de Algoritmos Genéticos na Otimização do Planejamento Mestre da Produção**. 2006. Tese de Doutorado. Pontifícia Universidade Católica do Paraná. PUC-PR.

TANOMARU, Julio. Motivação, Fundamentos e Aplicações de Algoritmos Genéticos. In: **II Congresso Brasileiro de Redes Neurais**. 1995. p. 373-403.

VILLAÇA, Marco Valério Miorim; SILVEIRA, Jony Laureano. Uma Breve História do Controle Automático. **Revista Ilha Digital**, v. 4, p. 3-12, 2013.